



**Лабораторная работа № 13**  
**Анализатор сетевых протоколов Wireshark**

**NetCracker Technology**  
**2011**

## Теоретическая часть

### 1. Введение

Термин снифер происходит от английского «to sniff» – нюхать – и представляет собой программу или программно-аппаратное устройство, предназначенное для перехвата и последующего анализа, либо только анализа сетевого трафика, предназначенного для других узлов.

Перехват трафика может осуществляться:

- обычным «прослушиванием» сетевого интерфейса (метод эффективен при использовании в сегменте концентраторов (хабов) вместо коммутаторов (свитчей), в противном случае метод малоэффективен, поскольку на снифер попадают лишь отдельные фреймы);
- подключением снифера в разрыв канала;
- ответвлением (программным или аппаратным) трафика и направлением его копии на снифер;
- через анализ побочных электромагнитных излучений и восстановление таким образом прослушиваемого трафика;
- через атаку на канальном (MAC-spoofing) или сетевом уровне (IP-spoofing), приводящую к перенаправлению трафика жертвы или всего трафика сегмента на снифер с последующим возвращением трафика в надлежащий адрес.

В начале 1990-х широко применялся хакерами для захвата пользовательских логинов и паролей, которые в ряде сетевых протоколов передаются в незашифрованном или зашифрованном слабыми алгоритмами виде. Широкое распространение концентраторов позволяло захватывать трафик без больших усилий в больших сегментах сети практически без риска быть обнаруженным. Сниферы применяются как в благих, так и в деструктивных целях. Анализ прошедшего через снифер трафика позволяет:

- обнаружить паразитный, вирусный и закольцованный трафик, наличие которого увеличивает загрузку сетевого оборудования и каналов связи (сниферы здесь малоэффективны; как правило, для этих целей используют сбор разнообразной статистики серверами и активным сетевым оборудованием и её последующий анализ);
- выявить в сети вредоносное и несанкционированное ПО, например, сетевые сканеры, флудеры, троянские программы, клиенты пиринговых сетей и другие (это обычно делают при помощи специализированных сниферов – мониторов сетевой активности);
- перехватить любой незашифрованный (а порой и зашифрованный) пользовательский трафик с целью получения паролей и другой информации;
- локализовать неисправность сети или ошибку конфигурации сетевых агентов (для этой цели сниферы часто применяются системными администраторами).

Поскольку в «классическом» снифере анализ трафика происходит вручную, с применением лишь простейших средств автоматизации (анализ протоколов, восстановление TCP-потока), то он подходит для анализа лишь небольших его объёмов.

Как ни странно, в природе существует великое множество сниферов, поэтому их разделяют на категории:

- (HTTP Analyzer, IEWatch Professional, EffeTech HTTP Sniffer), перехватывают HTTP заголовки;
- (O&K Print Watch, PrintMonitor, Print Inspector), позволяют контролировать и управлять процессом печати в сети;
- (Wireshark, TracePlus32 Web Detective, CommView);
- (MSN Shiffer, ICQ Sniffer, AIM Sniff, IM-Sniffer), предоставляют перехваченную переписку в удобно читаемом виде;
- (Cain & Abel, Ace Password Sniffer), перехватывают и контролируют разнообразные пароли;
- (Kismet, airodump-ng, CommView for WiFi), перехватывают трафик беспроводных сетей даже без подключения к этим сетям;
- (Network Probe, Etherscan Analyzer).

## **2. Wireshark: основы**

Wireshark – это анализатор сетевого трафика. Его задача состоит в том, чтобы перехватывать сетевой трафик и отображать его в детальном виде. Анализатор сетевого трафика можно сравнить с измерительным устройством, которое используется для просмотра того, что происходит внутри сетевого кабеля, как например вольтметр используется электриками для того чтобы узнать что происходит внутри электропроводки (но, конечно, на более высоком уровне). В прошлом такие инструменты были очень дорогостоящими и проприетарными. Однако, с момента появления такого инструмента как Wireshark ситуация изменилась. Wireshark – это один из лучших анализаторов сетевого трафика, доступных на сегодняшний момент. Wireshark работает на основе библиотеки pcap. Библиотека Pcap (Packet Capture) позволяет создавать программы анализа сетевых данных, поступающих на сетевую карту компьютера. Разнообразные программы мониторинга и тестирования сети, сниферы используют эту библиотеку. Она написана для использования языка C/C++ так что другие языки, такие как Java, .NET и скриптовые языки использовать не рационально. Для Unix-подобных систем используют libpcap библиотеку, а для Microsoft Windows NT используют WinPcap библиотеку. Программное обеспечение сетевого мониторинга может использовать libpcap или WinPcap, чтобы захватить пакеты, путешествующие по сети и в более новых версиях для передачи пакетов в сети. Libpcap и WinPcap также поддерживают сохранение захваченных пакетов в файл и чтение файлов содержащих сохранённые пакеты. Программы написанные на основе libpcap или WinPcap могут захватить сетевой трафик, анализировать его. Файл захваченного трафика сохраняется в формате, понятном для приложений, использующих Pcap.

## 2.1 Для чего используется Wireshark?

- Системные администраторы используют его для решения проблем в сети.
- Аудиторы безопасности используют его для выявления проблем в сети.
- Разработчики используют его для отладки сетевых приложений.
- Обычные пользователи используют его для изучения внутреннего устройства сетевых протоколов.

## 2.2 Возможности Wireshark

- Работает на большинстве современных ОС (Microsoft Windows, Mac OS X, UNIX). Wireshark – продукт с открытым исходным кодом, распространяемый на основании лицензии GPL. Его можно использовать на любом количестве компьютеров, не опасаясь за ввод лицензионных ключей, продление лицензии и другие неприятные мероприятия. Поэтому сообществу очень легко добавлять в него поддержку новых протоколов в виде плагинов или напрямую вшить её в исходный код.
- Перехват трафика сетевого интерфейса в режиме реального времени. Wireshark может перехватывать трафик различных сетевых устройств, отображая его имя (включая беспроводные устройства). Поддерживаемость того или иного устройства зависит от многих факторов, например от операционной системы.
- Множество протокольных декодировщиков (TELNET, FTP, POP, RLOGIN, ICQ, SMB, MySQL, HTTP, NNTP, X11, NAPSTER, IRC, RIP, BGP, SOCKS 5, IMAP 4, VNC, LDAP, NFS, SNMP, MSN, YMSG и другие).
- Сохранение и открытие ранее сохраненного сетевого трафика.
- Импорт и экспорт файлов из других пакетных анализаторов. Wireshark может сохранять перехваченные пакеты в большое количество форматов других пакетных анализаторов, например: libpcap, tcpdump, Sun snoop, atmsnoop, Shomiti/Finisar Surveyor, Novell LANalyzer, Microsoft Network Monitor, AIX's iptrace.
- Позволяет фильтровать пакеты по множеству критерий.
- Позволяет искать пакеты по множеству критерий.
- Позволяет подсвечивать захваченные пакеты разных протоколов.
- Позволяет создавать разнообразную статистику.

Ниже перечислены некоторые вещи, которые Wireshark делать не умеет.

- Wireshark – это не система обнаружения вторжений. Он не предупредит о том, если кто-то делает странные вещи в сети. Однако если это происходит, Wireshark поможет понять что же на самом деле случилось.
- Wireshark не умеет генерировать сетевой трафик, он может лишь анализировать имеющийся. В целом, Wireshark никак не проявляет себя в сети, кроме как при резолвинге доменных имен, но и эту функцию можно отключить.

## 2.3 Интерфейс Wireshark

Интерфейс программы Wireshark представлен на рисунке 1.

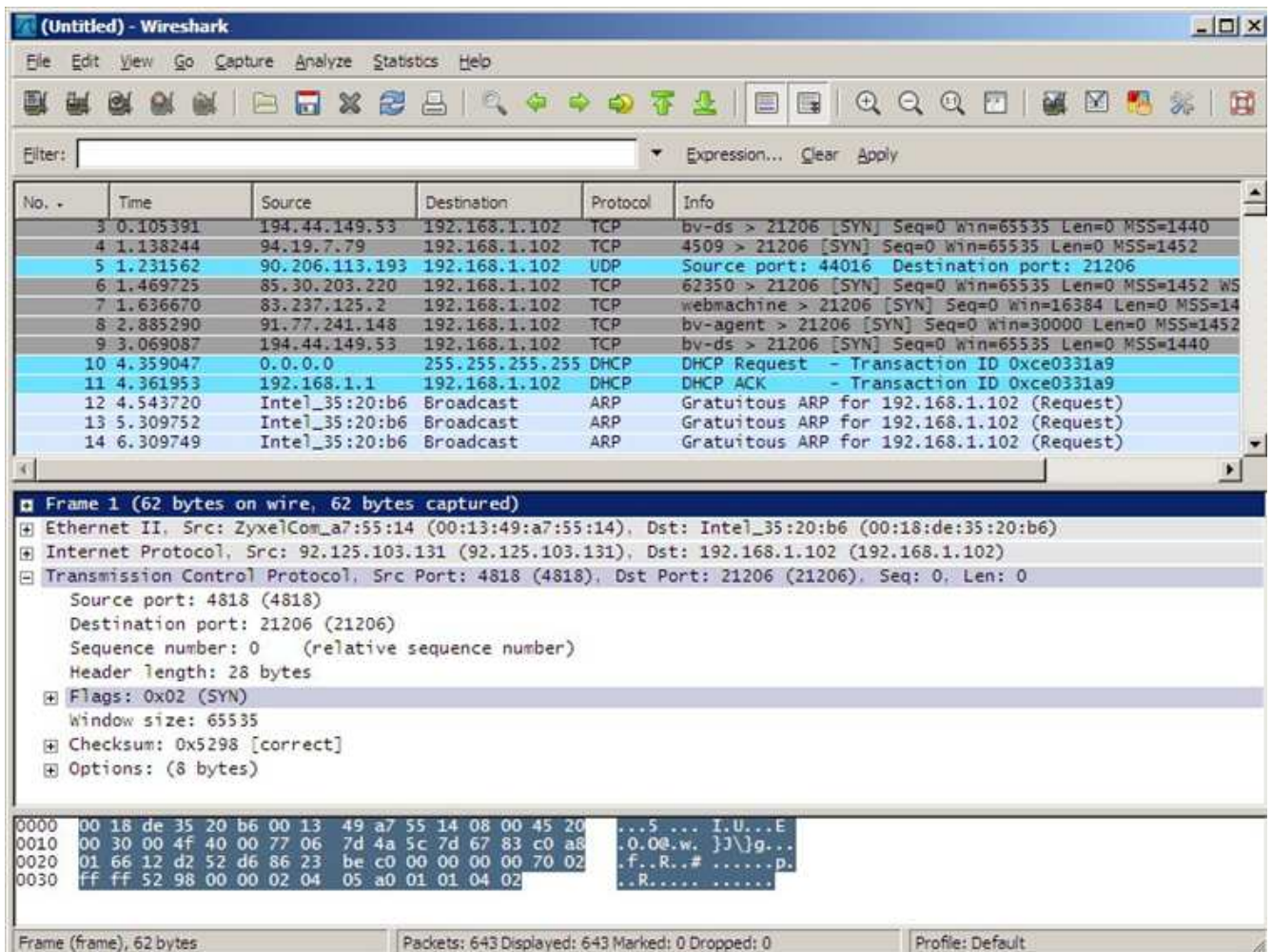


Рисунок 1 – Главное окно программы Wireshark

Рассмотрим интерфейс более подробно. Сверху находится стандартные для Windows приложения меню и тулбар, на них подробно останавливаться смысла не имеет. Далее следует фильтр, в нем можно задавать критерии фильтрации пакетов, подробнее описание работы с ним рассмотрим позже. Следом идет окошко со списком всех перехваченных пакетов. В нем доступна такая информация как: номер пакета, относительное время получения пакета (отсчет производится от первого пакета; параметры отображения времени можно изменить в настройках), IP адрес отправителя, IP адрес получателя, протокол, по которому пересылается пакет, а также дополнительная информация о нем. Как можно заметить, разные протоколы подсвечены разными цветами, что добавляет наглядности и упрощает анализ. Далее видно окно, в котором представлена детальная информация о пакете согласно сетевой модели OSI (подробнее см. Википедию). Ну, и самое нижнее окно показывает нам пакет в сыром HEX виде, то есть побайтово. Конфигурация интерфейса может быть легко изменена в меню View. Например, можно убрать окно побайтового представления пакета (оно же Packet Bytes в меню



View), так как в большинстве случаев (кроме анализа данных в пакете) оно не нужно и только дублирует информацию из окна детального описания.

## 2.4 Перехват трафика

Перехват трафика является одной из ключевых возможностей Wireshark. Движок Wireshark по перехвату предоставляет следующие возможности:

- перехват трафика различных видов сетевого оборудования (Ethernet, Token Ring, ATM и другие);
- прекращение перехвата на основе разных событий: размера перехваченных данных, продолжительность перехвата по времени, количество перехваченных пакетов;
- показ декодированных пакетов во время перехвата;
- фильтрация пакетов с целью уменьшить размер перехваченной информации;
- запись дампов в несколько файлов, если перехват продолжается долго. Движок не может выполнять следующие функции:
- перехват трафика с нескольких сетевых интерфейсов одновременно (однако, существует возможность запустить несколько копий Wireshark – каждая для своего интерфейса);
- прекращение перехвата в зависимости от перехваченной информации.

Чтобы начать перехват трафика нужно иметь права Администратора на данной системе и выбрать правильный сетевой интерфейс. Итак, начнем. Чтобы выбрать сетевой адаптер, с которого будет выполняться перехват нужно нажать на кнопку Interfaces на тулбаре, либо их меню Capture > Interfaces... (отмечены красным цветом на рисунке 2).

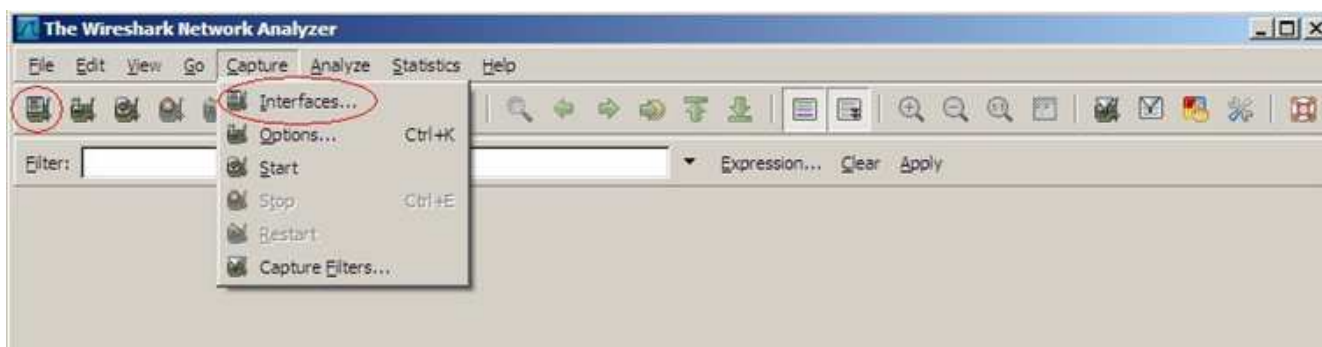


Рисунок 2 – Выбор интерфейса для перехвата

После нажатия на одну из этих кнопок появится окно со списком сетевых интерфейсов, доступных в системе (рисунок 3).



Рисунок 3 – Список сетевых интерфейсов

На этом списке можно увидеть такую информацию как название интерфейса, IP адрес интерфейса, сетевая активность интерфейса (представлена в виде общего количества пакетов с момента появления окна и количество пакетов в секунду). Также из этого окна можно посмотреть настройки перехвата (рисунок 4) и информацию об интерфейсе (рисунок 5).

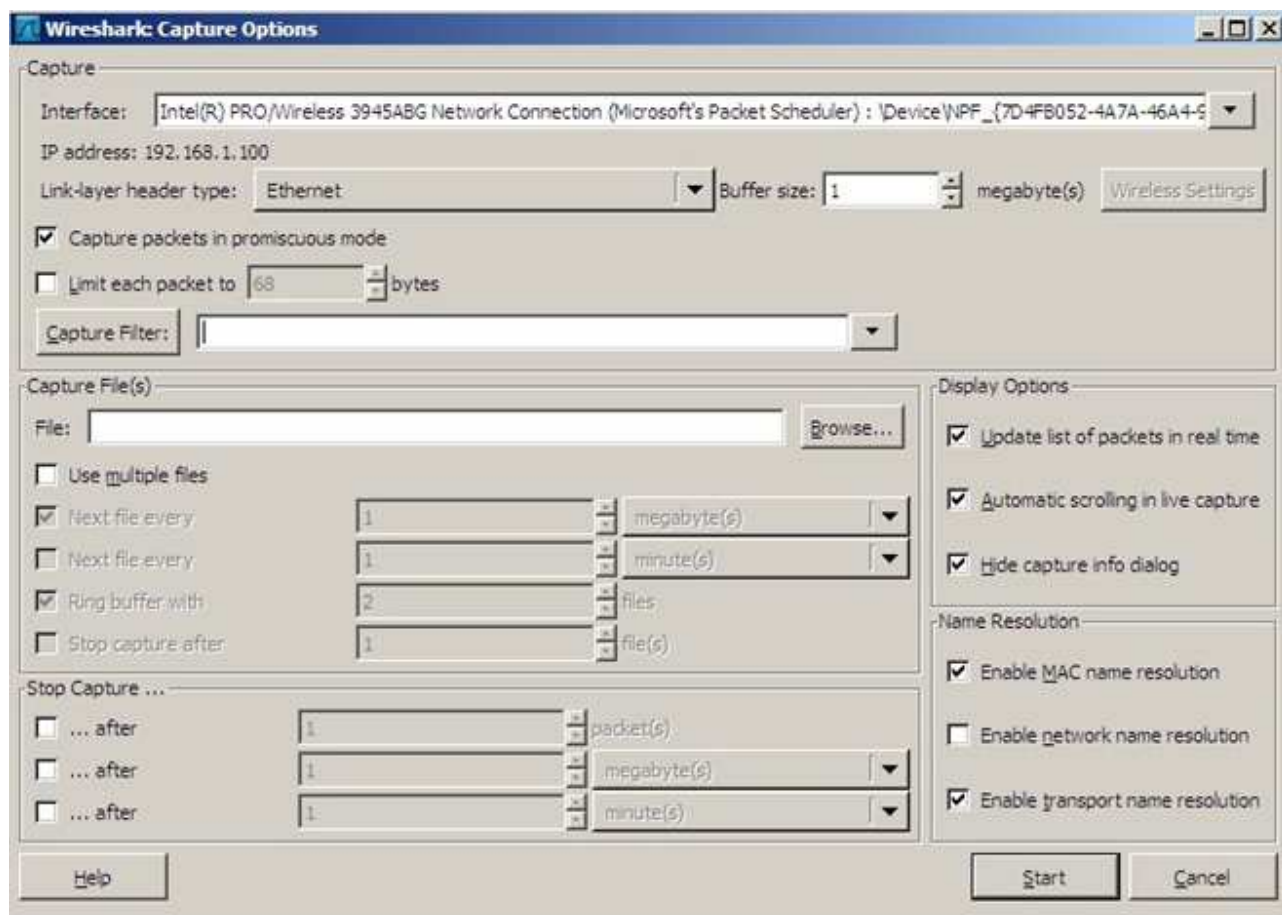


Рисунок 4 – Настройки перехвата

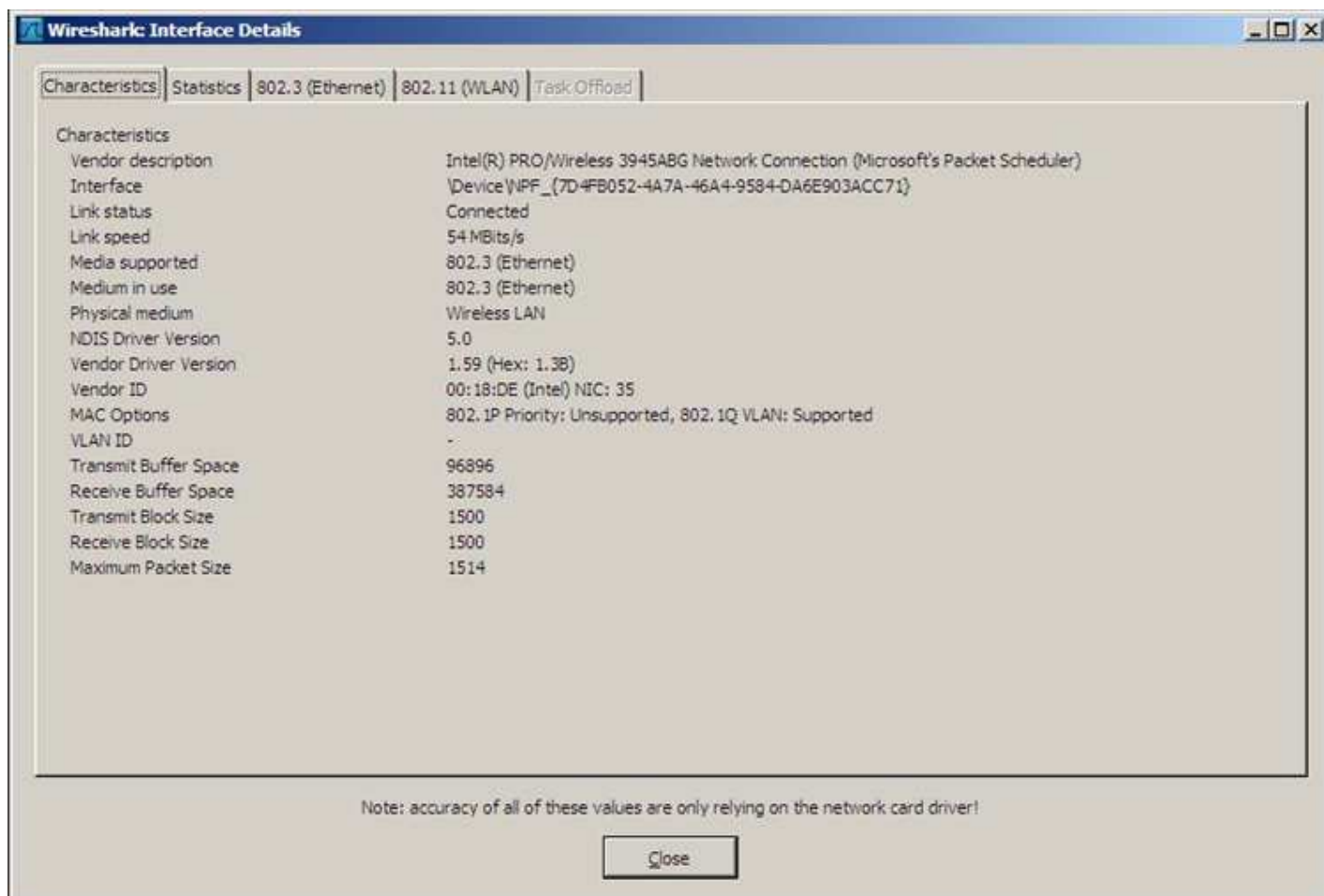


Рисунок 5 – Информация об интерфейсе

В настройках перехвата можно изменять такие параметры как фильтрация пакетов, запись дампа в несколько файлов, прекращение перехвата по разным критериям (количество пакетов, количество мегабайт, количество минут), опции показа пакетов, резолвинг имен. В большинстве случаев эти параметры можно оставить по умолчанию. Итак, всё готово к началу перехвата, осталось нажать кнопку Start.

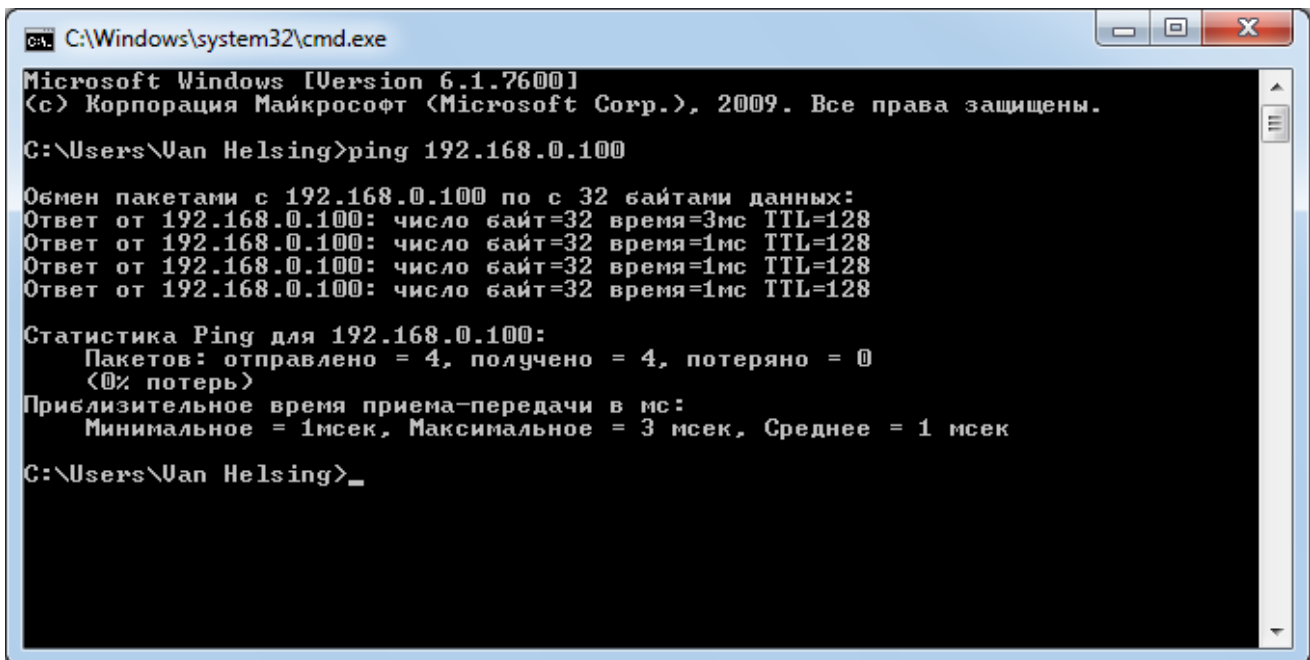


### 3. Практика перехвата

После нажатия на кнопку Start начался перехват пакетов. Если сетевая активность высокая, то можно сразу увидеть массу непонятных входящих или исходящих пакетов. Они нас пока мало волнуют, сейчас мы займемся изучением всем известной утилиты ping.

#### 3.1 Утилита ping

Нажмем Win+R и введем в строке выполнить cmd. Откроется консоль, введем там команду ping <IP адрес>, как показано на рисунке 6. IP адрес следует писать, исходя из конфигурации конкретной сети.

A screenshot of a Windows command prompt window. The title bar shows the path 'C:\Windows\system32\cmd.exe'. The window content displays the following text:

```
Microsoft Windows [Version 6.1.7600]
(c) Корпорация Майкрософт (Microsoft Corp.), 2009. Все права защищены.

C:\Users\Van Helsing>ping 192.168.0.100

Обмен пакетами с 192.168.0.100 по 32 байтами данных:
Ответ от 192.168.0.100: число байт=32 время=3мс TTL=128
Ответ от 192.168.0.100: число байт=32 время=1мс TTL=128
Ответ от 192.168.0.100: число байт=32 время=1мс TTL=128
Ответ от 192.168.0.100: число байт=32 время=1мс TTL=128

Статистика Ping для 192.168.0.100:
  Пакетов: отправлено = 4, получено = 4, потеряно = 0
  (0% потерь)
Приблизительное время приема-передачи в мс:
  Минимальное = 1мсек, Максимальное = 3 мсек, Среднее = 1 мсек

C:\Users\Van Helsing>
```

Рисунок 6 – Выполнение команды ping

Теперь, если опрос хоста прошел так же успешно, как показано на рисунке, откроем окно Wireshark, чтобы посмотреть на это более подробно. Там мы скорее всего увидим полный бардак и разбираться в этом нужно будет очень долго. Тут нам на помощь и придут фильтры! Утилита ping работает по протоколу ICMP, поэтому впишем название этого протокола в строку фильтра и нажмем Apply. Должно получиться нечто похожее на рисунок 7.

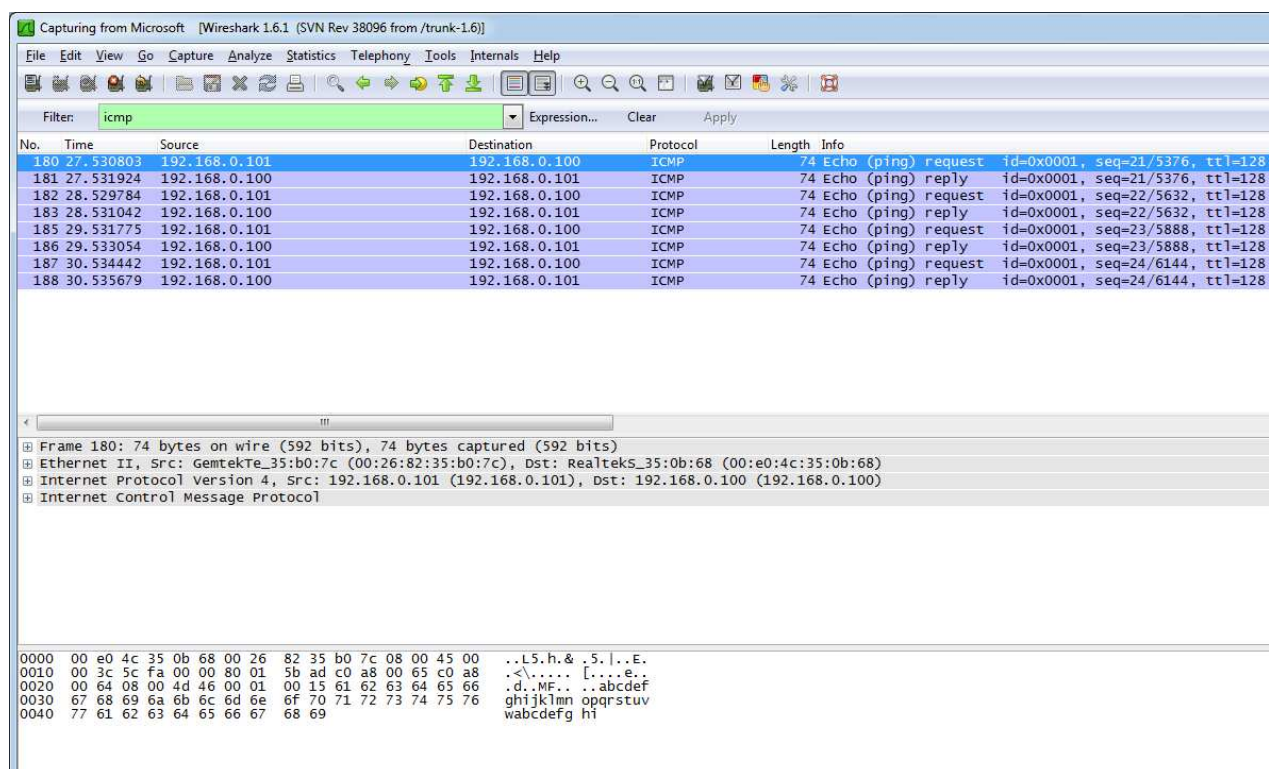


Рисунок 7 – Фильтрация по протоколу ICMP

Здесь мы можем наблюдать как происходит Echo Request и Echo Reply в протоколе ICMP изнутри: какие тестовые данные посылаются, какие флаги символизируют о том, что это именно Echo Request, и другую не менее важную информацию.

### 3.2 Перехват сообщений Интернет-мессенджеров

Для того, чтобы перехватить сообщения, передаваемые в QIP необходимо поставить фильтр aim на TCP-пакеты и отправить сообщение.

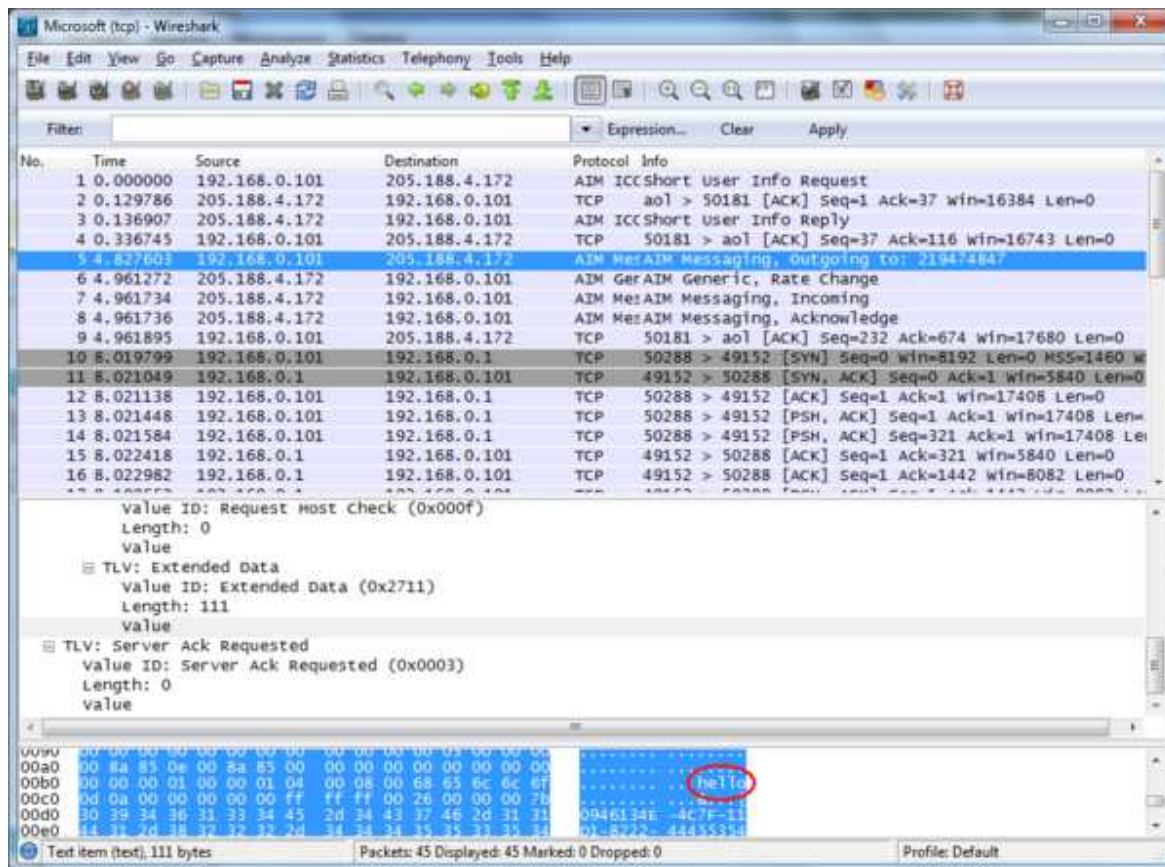


Рисунок 8 – Фильтрация по протоколу AIM сообщений посланных в QIP

Как видно из скриншота, по протоколу AOL данные передаются в незашифрованном виде. Однако прочитать удастся только сообщения набранные латинскими буквами.

Если попробовать проделать тоже самое в Skype, уже по одному списку пакетов видно, что так легко перехватить переписку, как в QIP, уже не получится. Данные передаются по HTTPS (HyperText Transfer Protocol Secure) и TLS (Transport Layer Security), то есть, в зашифрованном виде. Как результат, прочитать набранное сообщение не получится.

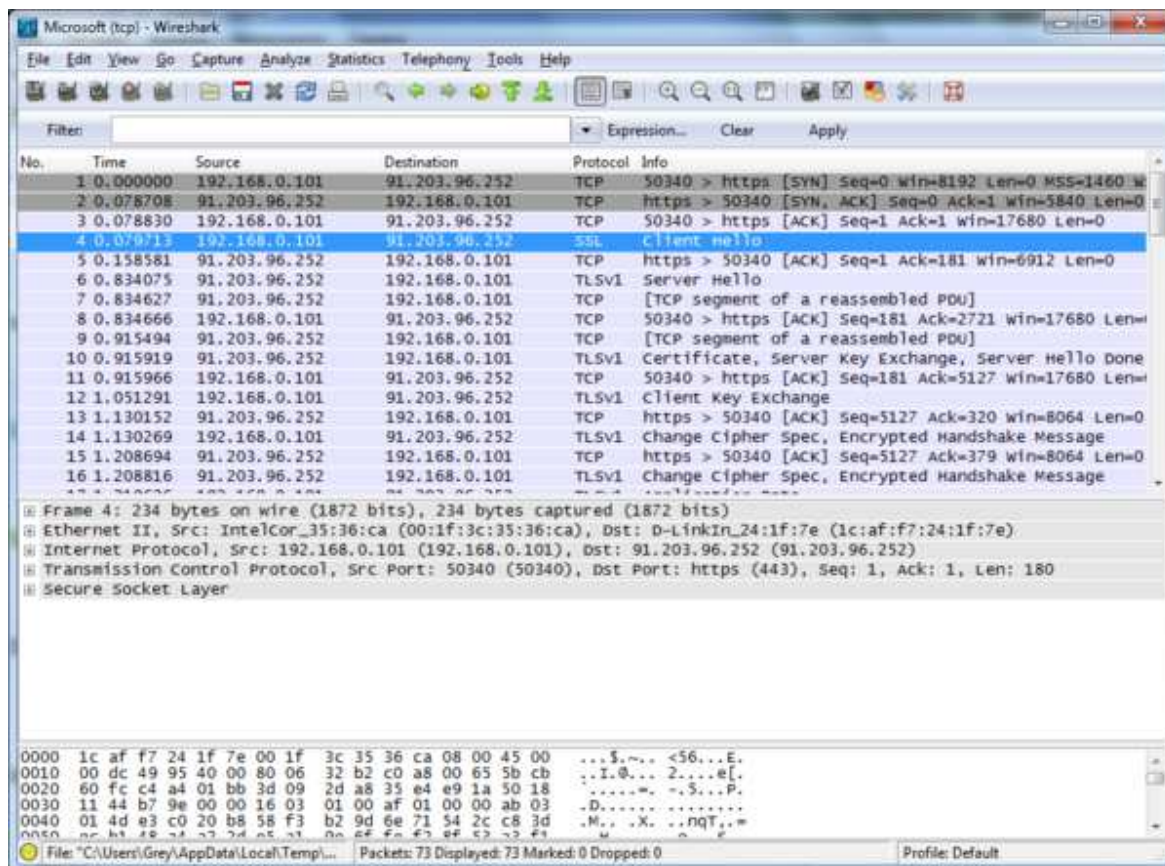


Рисунок 9 – Сообщения в зашифрованном виде посланные через Skype



### 3.3 Перехват FTP трафика

В этом пункте рассматривается перехват документа, передающегося по протоколу FTP без шифрования, и убедимся, что при использовании шифрования на основе TLS ничего полезного мы перехватить не сможем. В качестве FTP сервера используется сервер указанный преподавателем, в качестве клиента – любой браузер, например, Internet Explorer.

Запускаем захват пакетов в Wireshark и делаем фильтр по протоколу FTP для удобства (набираем «ftp or ftp-data» без кавычек). Набираем в строке адреса браузера адрес нашего FTP сервера: ftp://<IP адрес сервера> и жмем Enter. На сервере следует выбрать указанный преподавателем текстовый документ (например test.txt) и скачать его. Теперь посмотрим, что произошло в сниффере, и какие пакеты были перехвачены. На рисунке 10 можно видеть, что перехватить можно не только данные, которые передаются по протоколу, но и логин с паролем.

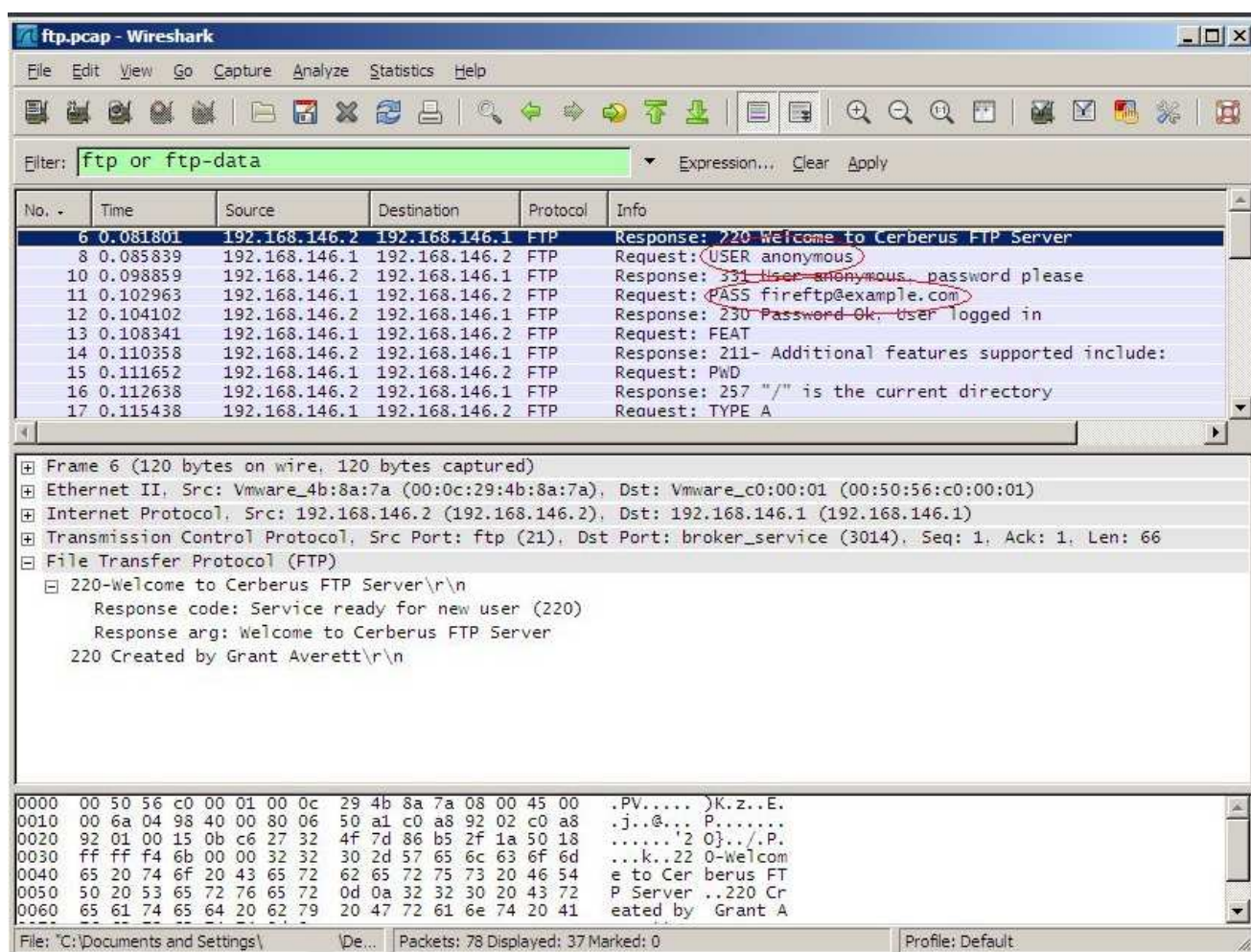


Рисунок 8 – Перехват логина и пароля

Теперь найдем в перехваченных пакетах содержание нашего документа. Несколько слов о процессе передачи файлов по протоколу FTP: в самом начале сервер посылает клиенту баннер приветствия (в данном случае это 220-Welcome to Cerberus FTP Server), пользователь проходит аутентификацию на сервере с



помощью команд USER и PASS, получает список директорий с помощью команды LIST и запрашивает нужный файл с помощью команды RETR. Команду RETR мы и будем искать в списке пакетов. Для этого нужно нажать Ctrl+F, выбрать в опциях поиска Find by String и Search in Packet Bytes, в строке поиска ввести RETR и нажать Enter. Будет найден пакет, в котором клиент посылает эту команду серверу, если файл существует, то сервер пришлет ответ 150 Opening data connection, а в следующем пакете и будет содержимое документа (рисунок 11).

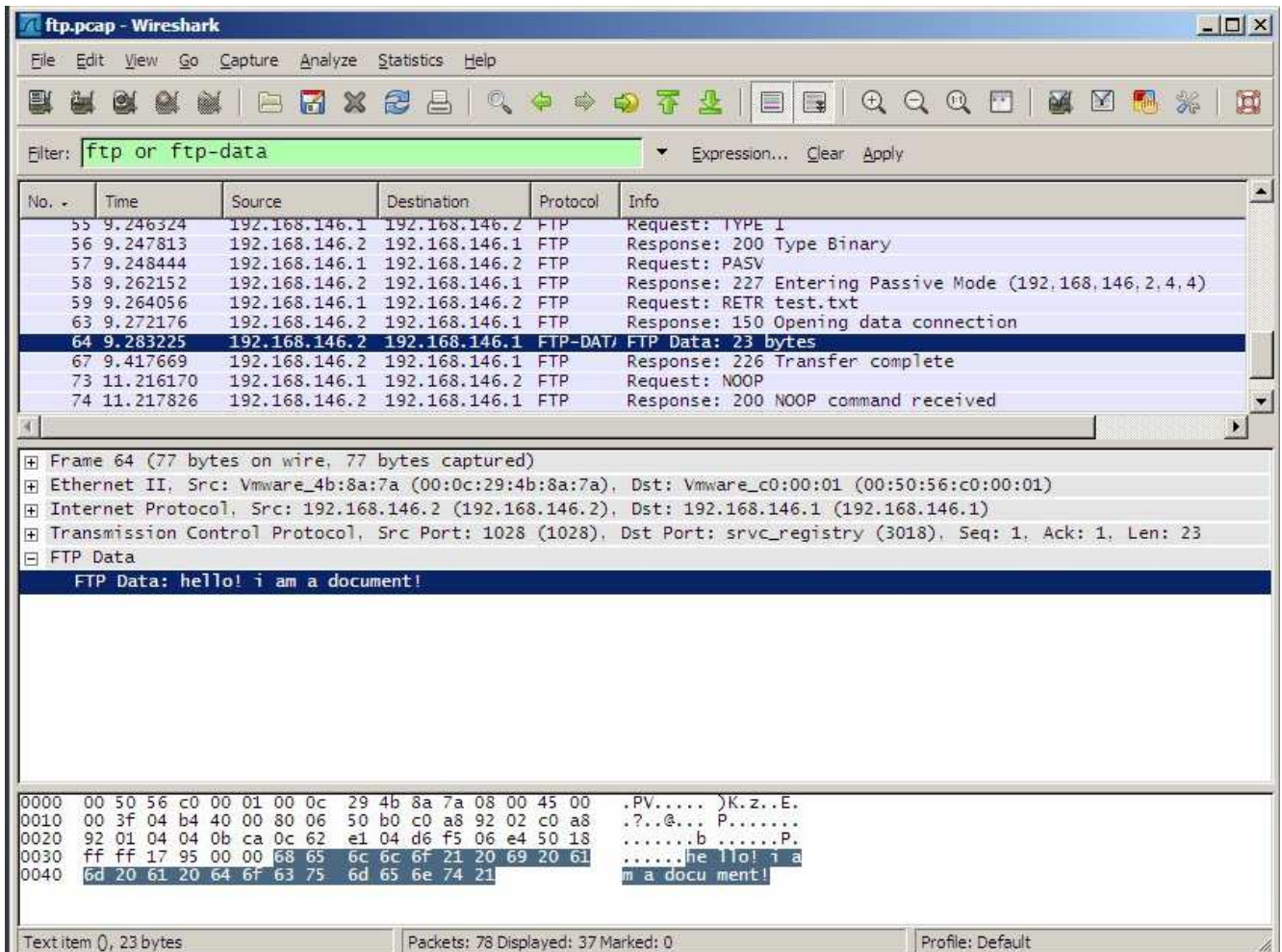


Рисунок 11 – Содержимое переданного документа

Теперь включим на сервере безопасную передачу данных на основе протокола TLS и повторим процедуру. Для установки безопасного соединения клиент использует команду AUTH TLS. При просмотре списка перехваченных пакетов становится ясно, что ничего полезного из них получить не получится.

### 3.4 Перехват документа, переданного по протоколу SMB

SMB (Server Message Block) – формат сообщений на основе протокола совместного использования файлов Microsoft/3Com, используемый для передачи файловых запросов (open – открыть, close – закрыть, read – прочитать, write – записать и т. п.) между клиентами и серверами. Откроем Wireshark, запустим перехват пакетов и фильтр по протоколу SMB. Затем зайдём на удаленный расшаренный ресурс (указанный преподавателем) и скачаем файл test.txt. Остановим перехват пакетов, нажав на кнопку «Stop the running live capture». Теперь посмотрим на то, что мы получили. В списке пакетов найдем запрос на передачу файла test.txt:

*Trans2 Request, FIND\_FIRST2, Pattern: \test.txt.*

Спустя несколько пакетов можно будет увидеть ответ:

*Read AndX Response, FID 0x0004, 23 bytes.*

В этом пакете и будет содержание документа.

## 4. Сбор статистики

Wireshark позволяет структурировать данные о перехваченных пакетах и представлять их в виде графиков и диаграмм.

Рассмотрим сбор статистики с сетевого интерфейса, через который работает http-сервер и torrent-клиент, а также другие клиенты, отсылающие значительно меньше пакетов, чем перечисленные.

### 4.1 График входящего/исходящего трафика

Для построения графика нужно выбрать пункт IO Graphs из меню Statistic.

Появится окно, показанное на рисунке 12. Wireshark позволяет построить до пяти графиков.

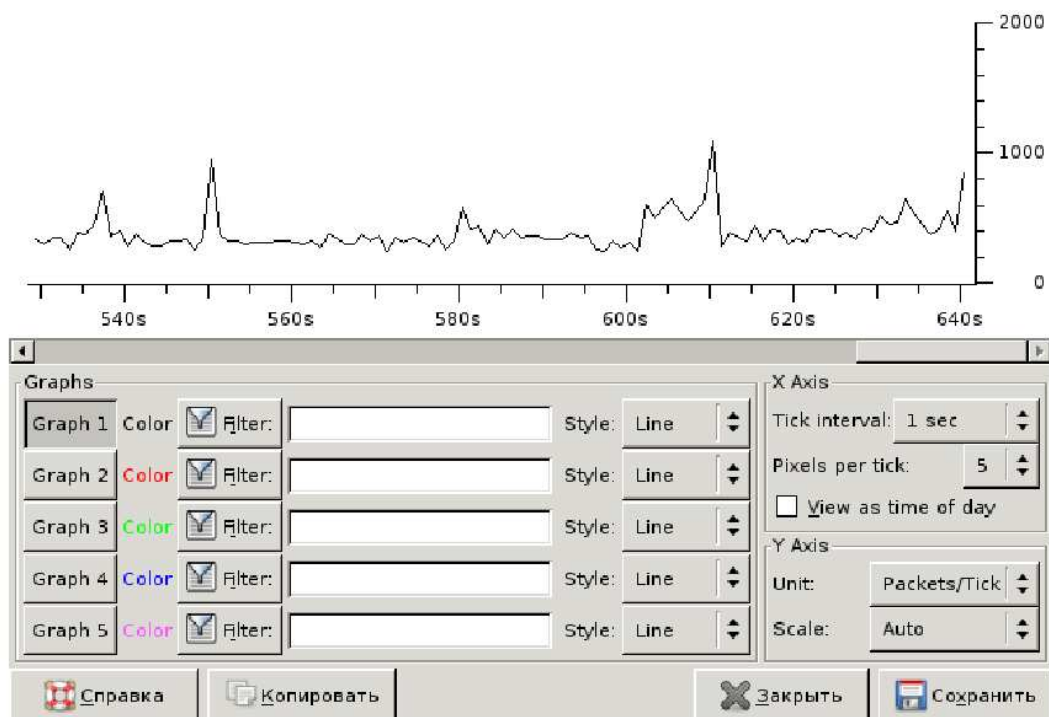


Рисунок 12 – Окно построения графика трафика

Построим графики входящего и исходящего трафика, передаваемого по протоколу bittorrent и http. Первый фильтр будет отображать входящий torrent-трафик, второй – исходящий, третий и четвертый – входящий и исходящий http-трафик (рис. 13.)

Исходя из графика можно заметить, что нагрузка на http-сервер очень мала по сравнению с нагрузкой на torrent-клиент. Такая статистика будет полезна при расчете конфигурации серверов и распределения нагрузки на них.

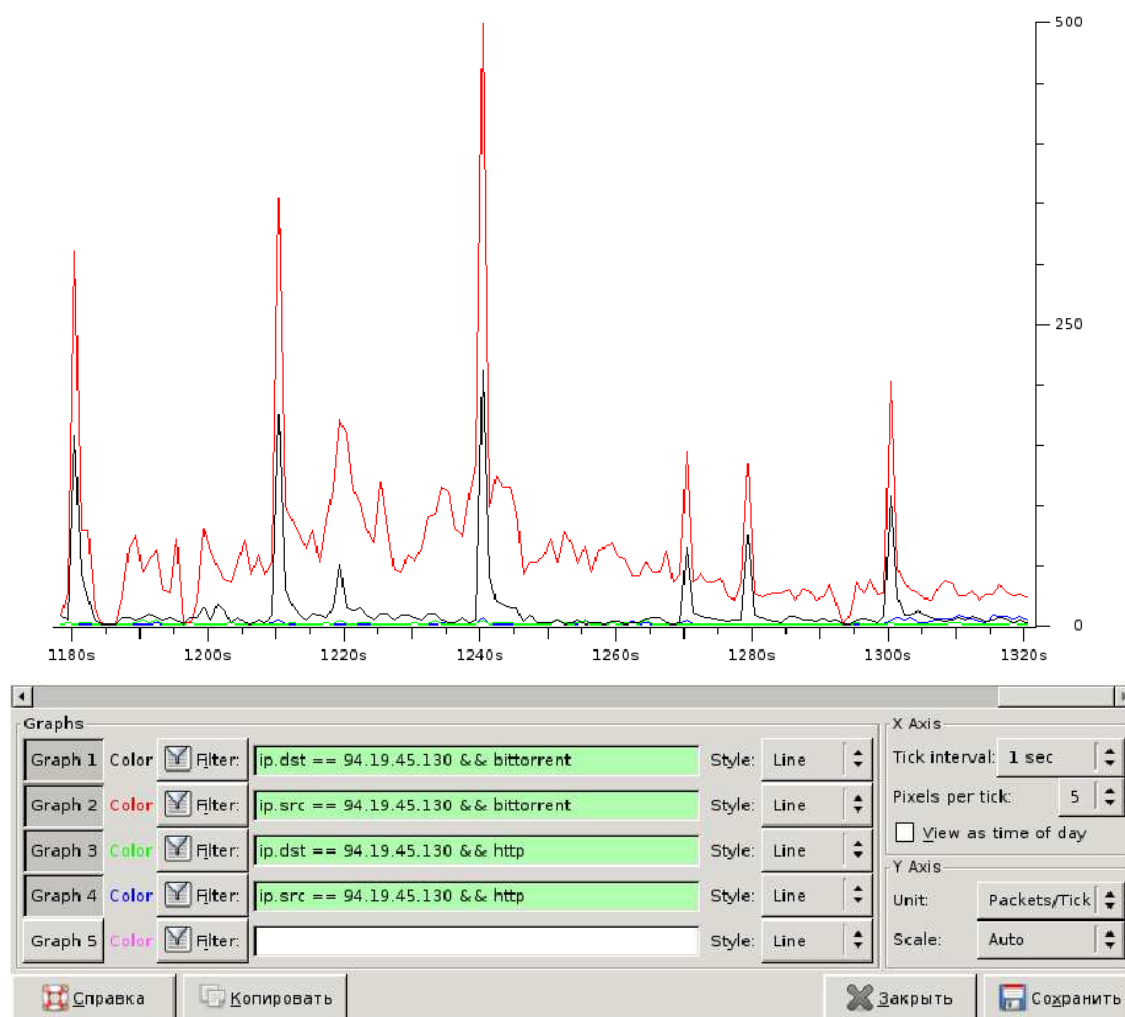


Рисунок 13 – Построенные графики http и bittorrent трафика.

## 5. Заключение

Проблема sniffing была актуальной раньше – в сетях основанных на концентраторах (хабах) – она остается актуальной и сейчас для сетей на коммутаторах (свитчах), благодаря такой технологии как ARP spoofing. Более того, сегодня семимильными шагами развивается технологии беспроводных сетей, где sniffing возможен даже в пассивном режиме.

Единственным решением, препятствующим sniffingu, является шифрование. Нельзя допускать использования фирменных небезопасных прикладных протоколов или унаследованных протоколов, передающих данные явным образом. Замена небезопасных протоколов (таких как telnet) на их надежные шифрованные аналоги (такие как SSH) представляется серьезным барьером от перехвата. Замена всех небезопасных протоколов в большинстве случаев маловероятна.

Вместо прекращения использования протоколов, передающих данные явным образом, остается только одна возможность - шифрование всего сетевого трафика на 3 уровне, используя IPSec. Осуществляя шифрование на 3 уровне, возможно

продолжать использовать небезопасные протоколы, поскольку все данные будут инкапсулированы IPSec и зашифрованы при передаче по сети. Таким образом, унаследованные приложения, которые используют старые протоколы, не пострадают. IPSec полностью прозрачен для приложений и пользователей. Это открытый стандарт, поддерживаемый многими вендорами, включая Microsoft и Cisco. Кроме того, многие реализации Unix поддерживают IPSec. Легкая настройка IPSec в Win2k/XP дополнительно увеличивает его доступность.

Осуществление технологии шифрования на 3 уровне, таких как IPSec решает проблему сниффинга полностью. Масштабируемость, распространенность, доступность IPSec выделяет его как прагматическое решение проблемы перехвата сетевого трафика.



## Порядок выполнения работы

1. Изучить теоретическую и практическую часть.
2. Установив фильтр по протоколу icmp и выполнив команду ping указав IP-адрес компьютера соседа провести анализ отправленных и полученных пакетов (пункт 3.1).
3. Установив фильтр по протоколу aim и запустив Интернет-месседжер QIP перехватить сообщения переписки. (пункт 3.2).

Внимание! Отправляемые и принимаемые сообщения должны быть набраны латинскими символами.

4. Установив фильтр ftp or ftp-data провести перехват FTP трафика. (пункт 3.3).
5. Научится производить сбор статистики и структурировать данные о перехваченных пакетах на примере данных передаваемых http-сервером и torrent-клиентом (пункт 4).
6. Ключевые моменты практической работы зафиксировать скриншотами.
7. Оформить отчёт. Содержание отчёта смотри ниже.
8. Защитить отчёт.

## Содержание отчёта.

Отчёт готовится в электронном виде и распечатывается. Отчёт содержит

1. Скриншоты перехваченных пакетов и их описание.
2. Перечень фильтров применяемых в работе и их назначение.
3. Графики статистики передаваемых пакетов.
4. Выводы по работе.

## Контрольные вопросы

1. Каково назначение анализаторов сетевых протоколов?
2. Способы перехвата трафика.
3. Что позволяет установить анализ прошедшего через сниффер трафика?
4. Приведите примеры снифферов указав их области применения.
5. Назовите основные возможности Wireshark.
6. Опишите методику перехвата пакетов.
7. Что такое фильтры пакетов и зачем они применяются?

## Литература

1. <http://forum.antichat.ru/printthread.php?t=97460&pp=40> - Wireshark - практика перехвата
2. <http://wiki.auditory.ru/Участник:Marina.Maximova/3И/Wireshark> - WireShark
3. [tessie.mitht.rssi.ru/it/pdf/net/lab2.pdf](http://tessie.mitht.rssi.ru/it/pdf/net/lab2.pdf) - Анализ сетевого трафика с помощью программы «Wireshark».