

Настройка параметров перехвата

В предыдущей главе процесс элементарного перехвата пакетов рассматривался с применением параметров настройки, устанавливаемых в диалоговом окне *Capture Interfaces* (Интерфейсы для перехвата). Но в Wireshark предоставляется немало других возможностей для настройки параметров перехвата, которые мы еще не рассматривали. Для доступа к этим параметрам выберите команду *Capture*⇒*Options* из главного меню.

В диалоговом окне *Capture Interfaces* имеется немало настраиваемых параметров, позволяющих сделать процесс перехвата пакетов более удобным. Оно разделено на три вкладки: *Input* (Ввод), *Output* (Вывод) и *Options* (Параметры). Рассмотрим их по очереди.

Вкладка Input

Основное назначение вкладки *Input* (рис. 4.10) – отображать все сетевые интерфейсы, доступные для перехвата пакетов, а также некоторые основные сведения о каждом из этих интерфейсов. К этим сведениям относится удобное имя сетевого интерфейса, предоставляемое на уровне операционной системы, график сетевого трафика, наглядно показывающий пропускную способность данного интерфейса, а также дополнительные параметры настройки конфигурации, в том числе состояние смешанного режима работы сетевого интерфейса и размер буфера. На правом краю рассматриваемой здесь вкладки, не показанном на рис. 4.10, находится также столбец для отображения фильтра, применяемого при перехвате. Подробнее об этом речь пойдет в разделе “Фильтры перехвата”.

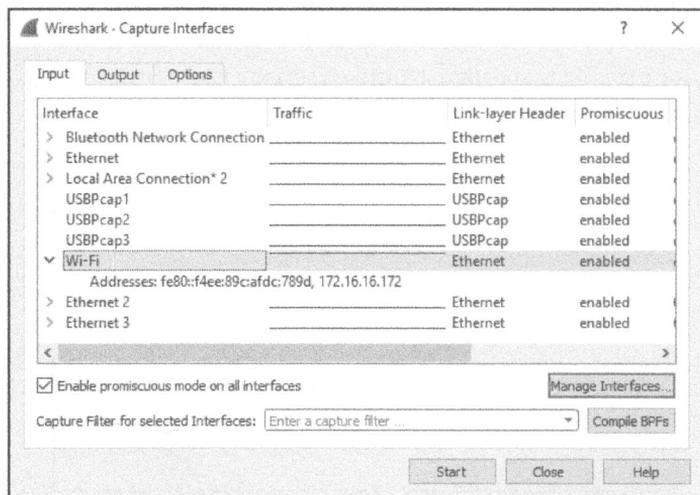


Рис. 4.10. Вкладка *Input* диалогового окна *Capture Interfaces*

На вкладке Input можно щелкнуть на большинстве параметров, чтобы сразу же отредактировать их. Так, если требуется отменить смешанный режим работы сетевого интерфейса, достаточно щелкнуть на его поле и сменить активное состояние этого режима на неактивное, выбрав соответствующий пункт из раскрывающегося меню.

Вкладка Output

На вкладке Output (рис. 4.11) можно организовать автоматическое сохранение перехваченных пакетов в файле вместо того, чтобы сначала перехватывать их, а затем сохранять в файле. Ведь манипулировать сохраненными пакетами намного легче. В частности, перехваченные пакеты можно сохранить в одном файле, в нескольких файлах и даже в кольцевом буфере, как поясняется далее, чтобы манипулировать целым рядом созданных файлов. Чтобы активизировать такой режим сохранения перехваченных пакетов, введите полный путь и имя файла в текстовом поле File. А для того чтобы выбрать каталог и предоставить имя файла, щелкните на кнопке Browse... (Просмотр).

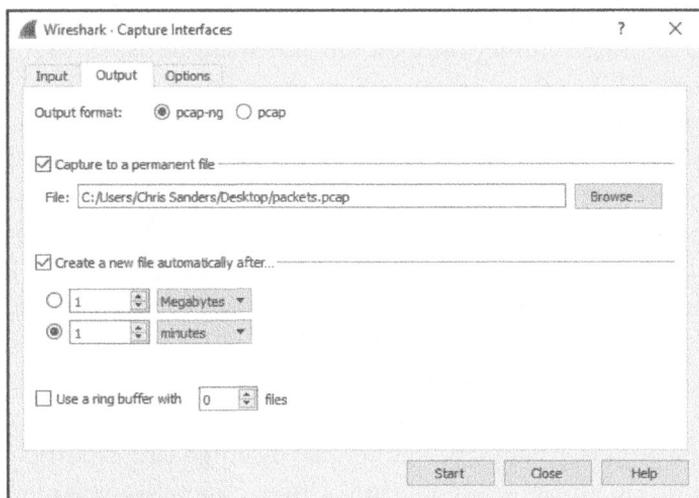
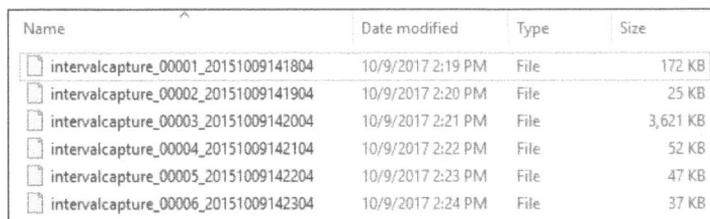


Рис. 4.11. Вкладка Output диалогового окна Capture Interfaces

Если приходится перехватывать большой объем сетевого трафика или выполнять перехват в долгосрочной перспективе, то особенно удобными для этой цели окажутся *наборы файлов* – совокупности нескольких файлов, выделяемые в отдельные группы по особому условию. Чтобы задать набор файлов, установите флажок Create a new file automatically after... (Создать автоматически новый файл после...).

Для сохранения перехваченных пакетов в наборах файлов по заданному размеру или промежутку времени в Wireshark применяются специальные

триггеры. Чтобы активизировать один из таких триггеров, выберите кнопку-переключатель, расположенную ниже упомянутого выше флажка, а затем укажите требуемое пороговое значение и единицу его измерения для срабатывания триггера. Например, можно установить триггер для запуска процесса создания нового файла после каждого перехвата сетевого трафика объемом 1 Мбайт, как показано на рис. 4.12, или через каждую минуту при перехвате сетевого трафика.



Name	Date modified	Type	Size
intervalcapture_00001_20151009141804	10/9/2017 2:19 PM	File	172 KB
intervalcapture_00002_20151009141904	10/9/2017 2:20 PM	File	25 KB
intervalcapture_00003_20151009142004	10/9/2017 2:21 PM	File	3,621 KB
intervalcapture_00004_20151009142104	10/9/2017 2:22 PM	File	52 KB
intervalcapture_00005_20151009142204	10/9/2017 2:23 PM	File	47 KB
intervalcapture_00006_20151009142304	10/9/2017 2:24 PM	File	37 KB

Рис. 4.12. Набор файлов, создаваемый в Wireshark через каждую минуту при перехвате сетевого трафика

Флажок *Use a ring buffer with...* (Использовать кольцевой буфер с...) позволяет указать определенное количество файлов, которые должны быть накоплены в наборе средствами Wireshark, прежде чем перезаписывать их. И хотя термин *кольцевой буфер* имеет разный смысл, в данном случае он, по существу, обозначает набор файлов, где первый файл может быть перезаписан, как только в набор будет введен последний записанный файл, чтобы сохранить вновь поступившие данные. Иными словами, этот флажок устанавливает режим записи файлов по принципу “первым пришел, первым обслужен” (FIFO). Установив этот флажок, можно указать максимальное количество файлов для хранения в кольцевом буфере с циклическим сдвигом. Допустим, что для хранения перехваченного сетевого трафика выбран набор из шести файлов, причем новый файл создается через каждый час. Таким образом, формируется кольцевой буфер размером 6 файлов. Как только будет создан шестой файл, в кольцевом буфере произойдет циклический сдвиг с перезаписью первого файла вместо создания седьмого файла для хранения очередного массива перехваченных данных. Этим гарантируется, что на жестком диске будет храниться не больше шести файлов данных в течение 6 часов с возможностью записи новых данных.

И, наконец, на вкладке *Output* можно также указать конкретный формат файла данных. Так, можно выбрать традиционный формат **.pcap**, если предполагается взаимодействие с сохраненными пакетами в инструментальном средстве, не способном обрабатывать файлы с расширением **.pcapng**.

Вкладка Options

На вкладке Options содержится целый ряд других вариантов выбора режима сохранения перехваченных пакетов, включая параметры настройки отображения, преобразования имен и прерывания перехвата, как показано на рис. 4.13.

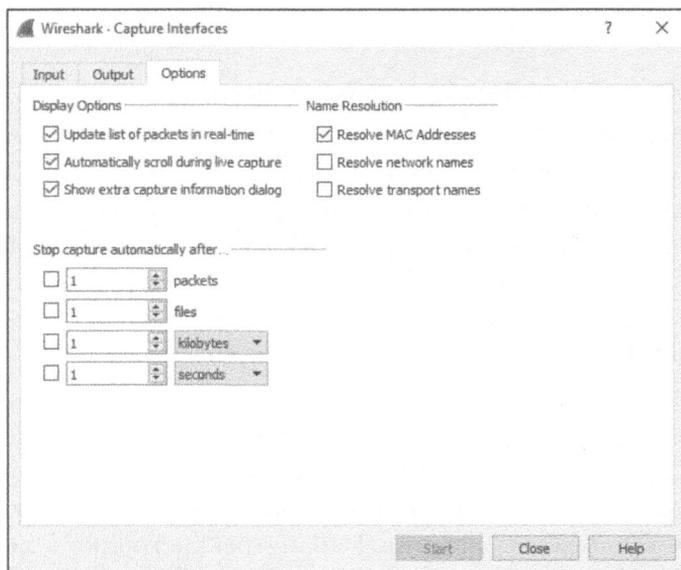


Рис. 4.13. Вкладка Options диалогового окна Capture Interfaces

Параметры настройки отображения

В разделе Display Options (Параметры отображения) находятся элементы, управляющие порядком отображения пакетов в процессе их перехвата. Назначение флажка Update list of packets in real-time (Обновление списка в реальном времени) не требует особых пояснений. Его можно устанавливать вместе с флажком Automatically scroll during live capture (Автоматическая прокрутка в ходе перехвата). Когда оба эти флажка установлены, на экране отображаются все перехваченные пакеты, причем самые последние из них отображаются немедленно.

ПРЕДУПРЕЖДЕНИЕ Если установлены оба флажка, Update list of packets in real-time и Automatically scroll during live capture, то для поддержания соответствующего режима отображения потребуются немало ресурсов ЦП – даже если перехватывается скромный объем данных. Поэтому оба эти флажка лучше сбросить, если только нет особой необходимости просматривать перехваченные пакеты в реальном времени.

Флажок Show extra capture information dialog (Показывать диалоговое окно с дополнительными сведениями) позволяет активизировать или подавить отображение небольшого окна, в котором представлено в процентах количество перехваченных пакетов, отсортированных по их сетевому протоколу. Я лично предпочитаю отображать это информационное окно, поскольку обычно запрещаю активную прокрутку окна отображения пакетов во время их перехвата.

Параметры настройки преобразования имен

В разделе Name Resolution (Преобразование имен) можно установить режим автоматического преобразования MAC-адресов (второго уровня), а также сетевых и транспортных имен (третьего и четвертого уровня соответственно). Более подробно общие вопросы преобразования имен будут рассмотрены в главе 5, “Дополнительные возможности Wireshark”.

Параметры настройки прекращения перехвата

В разделе Stop capture automatically after... (Автоматически прекратить перехват после...) можно задать определенные условия, при которых перехват будет прекращен. Как и при создании набора файлов, прекратить перехват можно по заданному размеру файла или промежутку времени. Но это можно сделать и по указанному количеству пакетов. Параметры настройки в данном разделе удобно сочетать с параметрами настройки, доступными на вкладке Output.

Применение фильтров

С помощью фильтров можно указать пакеты, которые требуются для анализа. Проще говоря, фильтр – это выражение, в котором задаются критерии для включения или исключения пакетов из анализа. Так, если имеются пакеты, которые не требуется просматривать, можно создать фильтр, чтобы избавиться от них. А если требуется просматривать только вполне определенные пакеты, можно создать фильтр, чтобы отображать только их.

В приложении Wireshark предоставляются следующие типы фильтров.

- **Фильтры перехвата.** Применяются в том случае, если требуется перехватывать только те пакеты, которые указаны для включения или исключения в заданном выражении.
- **Фильтры отображения.** Применяются к существующему ряду пакетов с целью скрыть ненужные пакеты или показать нужные, исходя из заданного выражения.

Рассмотрим сначала фильтры перехвата.

Фильтры перехвата

Фильтры данного типа применяются в процессе перехвата пакетов с целью изначально ограничить количество пакетов, предоставляемых для анализа. Одной из главных причин для применения фильтров перехвата служит производительность. Если заранее известно, что анализировать определенную форму сетевого трафика не нужно, ее достаточно отсеять с помощью фильтра перехвата, сэкономив процессорное время, которое обычно затрачивается для перехвата соответствующих пакетов.

Возможность создавать специальные фильтры перехвата окажется удобной в том случае, если приходится иметь дело с большими массивами данных. Их анализ можно ускорить, если отсеять только те пакеты, которые потребуются для решения текущей задачи анализа.

Допустим, решается задача диагностики службы, работающей через порт **262**, но на анализируемом сервере действует целый ряд служб через разные порты. Перехватить и проанализировать сетевой трафик только через один порт — дело совсем не простое. И здесь может пригодиться фильтр перехвата. Для этого достаточно воспользоваться диалоговым окном *Capture Interfaces* следующим образом.

1. Выберите команду *Capture⇒Options* из главного меню, чтобы открыть диалоговое окно *Capture Interfaces*.
2. Выберите сетевой интерфейс, где требуется перехватывать пакеты, а затем перейдите к крайнему справа столбцу *Capture Filter* (Фильтр перехвата).
3. Чтобы применить фильтр перехвата, щелкните на этом столбце и введите фильтрующее выражение. В данном случае требуется отфильтровать только сетевой трафик, входящий и исходящий через порт **262**, и поэтому введите **port 262**, как показано на рис. 4.14. (Более подробно фильтрующие выражения рассматриваются в следующем разделе.) Цвет ячейки в данном столбце должен измениться на зеленый, указывая на то, что введено достоверное выражение. Если же оно недостоверно, ячейка будет выделена красным цветом.
4. Задав фильтр, щелкните на кнопке *Start*, чтобы приступить к перехвату.

В итоге вы должны видеть только тот сетевой трафик, который проходит через порт **262**. Следовательно, это даст вам возможность более эффективно анализировать данные этого конкретного трафика.

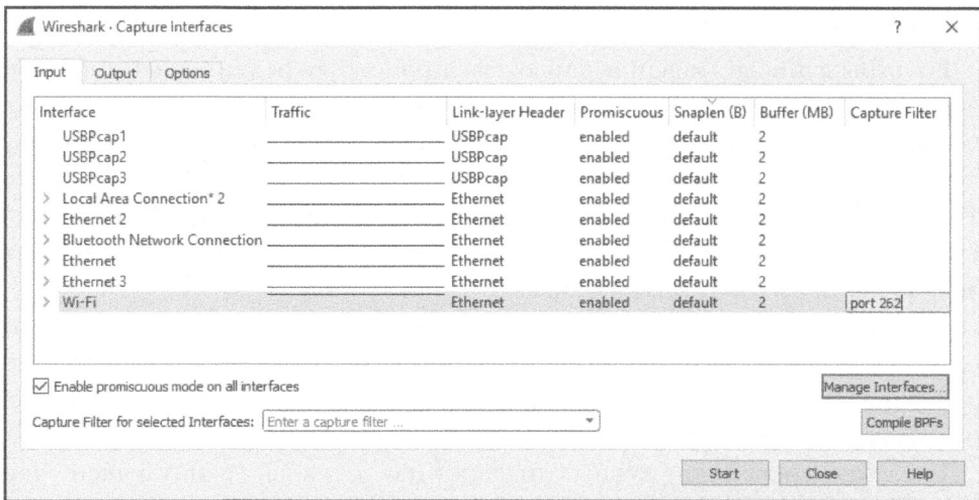


Рис. 4.14. Создание фильтра перехвата в диалоговом окне *Capture Interfaces*

Синтаксис BPF для фильтров перехвата

Фильтры перехвата, применяемые в драйверах `libpcap/WinPcap`, описываются с помощью синтаксиса BPF (Berkeley Packet Filter – фильтр пакетов Беркли). Этот синтаксис характерен для целого ряда приложений анализа пакетов, главным образом потому, что они опираются на библиотеки `libpcap/WinPcap`, в которых для описания фильтров используется синтаксис BPF. Для углубленного анализа сетей на уровне пакетов знание синтаксиса BPF имеет решающее значение.

Фильтр, созданный с помощью синтаксиса BPF, называется *выражением*, причем каждое выражение состоит из одного или нескольких *примитивов*. В свою очередь, примитивы состоят из одного или нескольких *квалификаторов*, перечисленных в табл. 4.2, а также имени или номера идентификатора, как показано на рис. 4.15.

Таблица 4.2. Квалификаторы BPF

Квалификатор	Описание	Примеры
<code>type</code>	Обозначает имя или номер идентификатора, на который делается ссылка	<code>host, net, port</code>
<code>dir</code>	Обозначает направление передачи при обмене данными с сетевым узлом, доступным по имени или номеру идентификатора	<code>src, dst</code>
<code>proto</code>	Ограничивает соответствие конкретному протоколу	<code>ether, ip, tcp, udp, http, ftp</code>

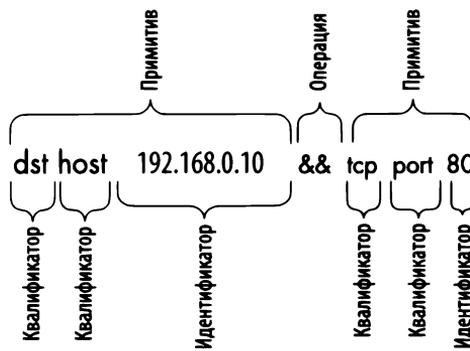


Рис. 4.15. Пример структуры фильтра перехвата

Принимая во внимание составляющие выражения, квалификатор `dst host` и идентификатор `192.168.0.10` совместно образуют примитив. Такой примитив сам является выражением, определяющим перехват сетевого трафика только по IP-адресу получателя `192.168.0.10`.

Для объединения примитивов при создании более сложных выражений можно использовать логические операции. И для этой цели имеются следующие логические операции.

- Логическая операция конкатенации по И (`&&`).
- Логическая операция дизъюнкции по ИЛИ (`||`).
- Логическая операция отрицания по НЕ (`!`).

Например, следующее выражение служит для перехвата только сетевого трафика, исходящего из источника, расположенного по IP-адресу `192.168.0.10` и проходящего через порт отправителя или получателя `80`:

```
src host 192.168.0.10 && port 80
```

Фильтры имен хостов и адресации

Большинство фильтров создается для анализа конкретного сетевого устройства или группы подобных устройств. В зависимости от обстоятельств фильтрация может основываться на MAC-адресе устройства, адресах IPv4 или IPv6, а также на имени хоста в службе DNS.

Допустим, что требуется проанализировать сетевой трафик конкретного хоста, взаимодействующего с сервером в сети. Чтобы перехватывать весь сетевой трафик, связанный с адресом данного хоста по протоколу IPv4, на сервере можно создать следующий фильтр, используя квалификатор `host`:

```
host 172.16.16.149
```

Если же анализируется сеть, действующая по протоколу IPv6, то фильтр следует создать на основе адреса IPv6, используя квалификатор `host`:

```
host 2001:db8:85a3::8a2e:370:7334
```

Создать фильтр можно и на основе имени хоста сетевого устройства, используя квалификатор `host` следующим образом:

```
host testserver2
```

А если требуется принять во внимание возможное изменение IP-адреса хоста, то фильтр можно создать на основе его MAC-адреса, добавив квалификатор протокола `ether`, как показано ниже.

```
ether host 00-1a-a0-52-e2-a0
```

Квалификаторы направления передачи данных нередко применяются вместе с фильтрами (например, из приведенных выше примеров) для перехвата сетевого трафика в зависимости от того, является ли этот трафик входящим или исходящим из хоста. Например, чтобы перехватить только сетевой трафик, исходящий из конкретного хоста, необходимо дополнить фильтр квалификатором `src` следующим образом:

```
src host 172.16.16.149
```

А для того чтобы перехватить только те данные, которые предназначены для хоста по адресу **172.16.16.149**, следует дополнить приведенный выше фильтр квалификатором `dst`:

```
dst host 172.16.16.149
```

Если в фильтре не употребляется квалификатор типа (`host`, `net` или `port`) с примитивом, то подразумевается квалификатор `host`. Поэтому приведенное ниже выражение, где отсутствует такой квалификатор, равнозначно выражению из предыдущего примера.

```
dst 172.16.16.149
```

Фильтры портов

Помимо фильтрации хостов, создаваемый фильтр можно основывать на портах, применяемых в каждом пакете. Фильтрацию портов можно выполнять для отсеивания служб и приложений, в которых применяются порты известных служб. Так, в следующем примере демонстрируется простой фильтр,

предназначенный для перехвата сетевого трафика, входящего или исходящего только из порта **8080**:

```
port 8080
```

Для перехвата всего сетевого трафика, кроме проходящего через порт **8080**, подойдет следующий фильтр:

```
!port 8080
```

Фильтры портов можно объединять с фильтрами направления передачи данных. Например, чтобы перехватить только сетевой трафик, направляемый веб-серверу для приема через стандартный порт **80** сетевого протокола **HTTP**, в фильтре можно употребить квалификатор `dst` следующим образом:

```
dst port 8080
```

Фильтры протоколов

Фильтры данного типа позволяют отсеивать пакеты по определенным сетевым протоколам. Они служат для сопоставления с сетевыми протоколами, которые не относятся к уровню приложений и не могут быть выявлены только по определенному порту. Так, если требуется просмотреть сетевой трафик только по протоколу **ICMP**, можно воспользоваться следующим фильтром:

```
icmp
```

А для просмотра всего сетевого трафика, кроме протокола **IPv6**, подойдет такой фильтр:

```
!ip6
```

Фильтры полей сетевых протоколов

К числу самых сильных сторон синтаксиса **BPF** относится возможность исследовать каждый байт в заголовке протокола, чтобы создавать специальные фильтры на основании полученных сведений о конкретном протоколе. Специальные фильтры, рассматриваемые в этом подразделе, позволяют извлекать из пакета определенное количество байтов, начиная с конкретного места.

Допустим, требуется отсеять сетевой трафик по полю типа в заголовке **ICMP**. Поле типа находится в самом начале пакета, т.е. имеет нулевое смещение. Чтобы обозначить местоположение исследуемого поля в пакете, достаточно указать в квадратных скобках смещение в байтах после квалификатора

протокола (в данном примере — `icmp[0]`). В итоге будет возвращено однобайтовое значение, которое можно с чем-нибудь сравнить. Например, чтобы получить только пакеты, передаваемые по сетевому протоколу ICMP и представляющие сообщения типа 3 о недостижимости получателя, в фильтрующем выражении следует употребить операцию равенства, как показано ниже.

```
icmp[0] == 3
```

А для того чтобы исследовать только пакеты, передаваемые по сетевому протоколу ICMP и представляющие эхо-запрос (типа 8) или эхо-ответ (типа 0), в фильтрующем выражении можно употребить два примитива и логическую операцию ИЛИ следующим образом:

```
icmp[0] == 8 OR icmp[0] == 0
```

Несмотря на то что специальные фильтры вполне справляются со своей задачей, они основываются только на однобайтовой информации из заголовка пакета. Но в фильтрующем выражении можно также указать длину возвращаемых данных в байтах после числовой величины смещения в квадратных скобках, отделив ее двоеточием.

Допустим, требуется создать фильтр для перехвата всех пакетов, передаваемых по сетевому протоколу ICMP для недостижимого получателя, хоста и определяемых по типу 3 и коду 1. Однобайтовые поля с этими данными расположены один за другим в заголовке пакета, начиная с нулевого смещения. С этой целью можно создать фильтр, где проверяются два начальных байта с нулевым смещением в заголовке пакета, которые сравниваются с шестнадцатеричным значением `0301` (т.е. тип 3 и код 1) следующим образом:

```
icmp[0:2] == 0x0301
```

Зачастую требуется перехватывать только пакеты, передаваемые по сетевому протоколу TCP с установленным флагом RST. Более подробно о протоколе TCP речь пойдет в главе 8, “Протоколы транспортного уровня”, а пока что достаточно сказать, что флаги располагаются в поле пакета TCP со смещением 13. Это поле интересно тем, что его длина равна одному байту, где каждый бит обозначает отдельный флаг. Как поясняется в приложении Б, “Интерпретация пакетов”, каждый бит с установленным в нем флагом представлен в такой байте числом по основанию 2. Так, первый бит представлен числом 1, второй бит — числом 2, третий — числом 4 и т.д. В пакете TCP можно одновременно установить несколько флагов. Следовательно, для эффективной их фильтрации одного значения `tcp[13]` недостаточно, поскольку установленному биту RST могут соответствовать несколько значений.

Вместо этого необходимо указать местоположение проверяемого флага в байте, дополнив фильтрующее выражение знаком амперсанда (&) и числом, представляющим бит, в котором хранится этот флаг. В частности, флаг RST хранится в бите, представленном в данном байте числом 4, которое обозначает, что флаг RST установлен. Таким образом, фильтр для рассматриваемых здесь целей будет выглядеть следующим образом:

```
tcp[13] & 4 == 4
```

А для того чтобы просмотреть все пакеты с установленным флагом PSH, который хранится в бите, представленном числом 8 в поле пакета TCP со смещением 13, достаточно воспользоваться следующим фильтром:

```
tcp[13] & 8 == 8
```

Примеры выражений для фильтров перехвата

Зачастую удачный или неудачный исход анализа пакетов зависит от умения создавать фильтры, пригодные для конкретной ситуации. В табл. 4.3 перечислены некоторые примеры выражений для фильтров перехвата, которые часто применяются при анализе пакетов.

Таблица 4.3. Наиболее употребительные фильтры перехвата

Фильтр	Описание
<code>tcp[13] & 32 == 32</code>	Пакеты TCP с установленным флагом URG
<code>tcp[13] & 16 == 16</code>	Пакеты TCP с установленным флагом ACK
<code>tcp[13] & 8 == 8</code>	Пакеты TCP с установленным флагом PSH
<code>tcp[13] & 4 == 4</code>	Пакеты TCP с установленным флагом RST
<code>tcp[13] & 2 == 2</code>	Пакеты TCP с установленным флагом SYN
<code>tcp[13] & 1 == 1</code>	Пакеты TCP с установленным флагом FIN
<code>tcp[13] == 18</code>	Пакеты TCP с установленными флагами SYN и ACK
<code>ether host 00:00:00:00:00:00</code>	Входящий и исходящий сетевой трафик по указанному MAC-адресу
<code>!ether host 00:00:00:00:00:00</code>	Входящий и исходящий сетевой трафик, кроме указанного MAC-адреса
<code>broadcast</code>	Только широковещательный трафик
<code>icmp</code>	Трафик только по сетевому протоколу ICMP
<code>icmp[0:2] == 0x0301</code>	Трафик по сетевому протоколу ICMP для недостижимого получателя и хоста
<code>ip</code>	Трафик только по сетевому протоколу IPv4
<code>ip6</code>	Трафик только по сетевому протоколу IPv6
<code>udp</code>	Трафик только по сетевому протоколу UDP

Фильтры отображения

К этому типу относятся фильтры, применение которых к файлу перехвата означает, что в Wireshark должны быть отображены только те пакеты, которые соответствуют данному фильтру. Фильтр отображения можно создать в текстовом поле Filter, расположенном над панелью Packet List.

Фильтры отображения применяются чаще, чем фильтры перехвата, поскольку они позволяют отсеивать ненужные данные не удаляя при этом их физически из файла перехвата. Так, если потребуется отменить первоначальное условие перехвата, для этого достаточно просто очистить фильтрующее выражение. Кроме того, фильтры отображения оказываются намного более эффективными благодаря поддержке со стороны обширной библиотеки дешифратора пакетов, доступной в Wireshark.

Например, в некоторых случаях фильтр отображения может применяться для отсеивания неуместного широковещательного трафика из файла перехвата. При этом на панели Packet List отсеиваются те широковещательные пакеты ARP, которые не имеют никакого отношения к текущей анализируемой проблеме в сети. Но поскольку эти широковещательные пакеты ARP могут оказаться полезными в дальнейшем, то лучше отсеять их временно, а не удалять полностью.

Чтобы отсеять все пакеты ARP в окне перехвата, установите курсор в текстовом поле Filter, расположенном над панелью Packet List, и введите **!arp**. В итоге все пакеты ARP будут удалены из списка (рис. 4.16). Чтобы удалить введенный фильтр отображения, щелкните на кнопке X, а для того чтобы сохранить этот фильтр для последующего применения, щелкните на кнопке со знаком “плюс” (+).

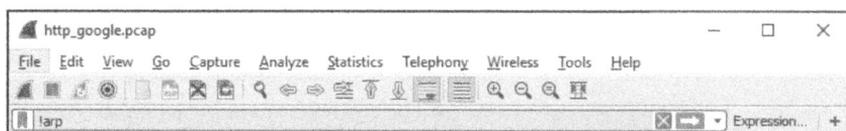


Рис. 4.16. Создание фильтра отображения в текстовом поле Filter, расположенном над подокном Packet List

Фильтры отображения можно применять двумя способами. Один из них состоит в непосредственном употреблении подходящего синтаксиса, как демонстрировалось в предыдущем примере, а другой – в применении диалогового окна Display Filter Expression (Выражение для фильтра отображения) для создания фильтра в диалоговом режиме. Это более простой способ для тех, кто только начинает пользоваться фильтрами. Рассмотрим оба способа применения фильтров отображения, начав с более простого.

Диалоговое окно Display Filter Expression

Возможности диалогового окна Display Filter Expression, приведенного на рис. 4.17, существенно упрощают начинающим пользователям Wireshark задачу по созданию фильтров перехвата и отображения. Для доступа к этому окну выберите команду Filter⇒Expression из главного меню.

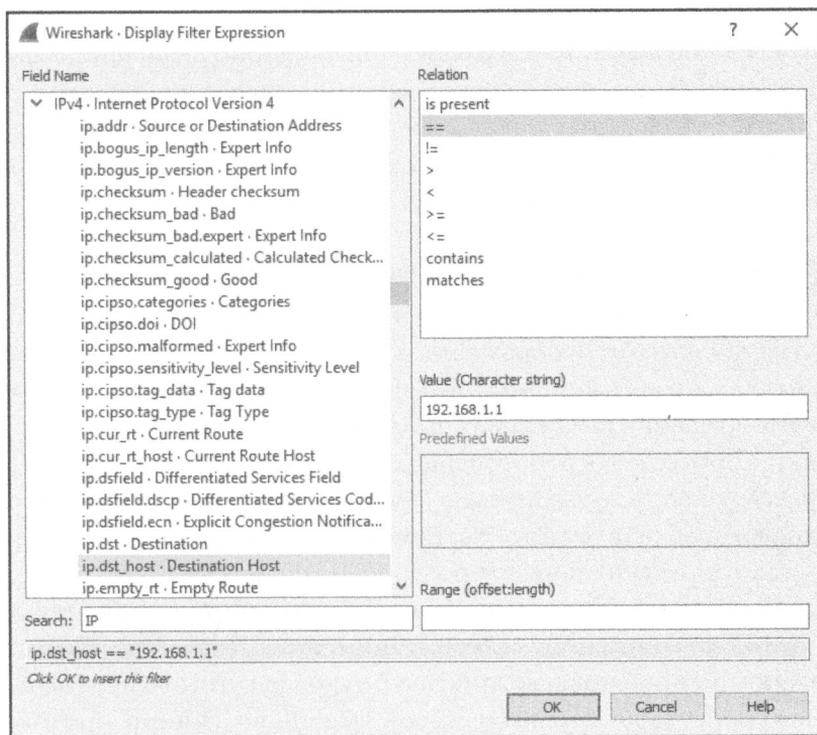


Рис. 4.17. В диалоговом окне Display Filter Expression заметно упрощается задача создания фильтров в Wireshark

Слева в данном окне перечислены все доступные поля сетевого протокола, где указаны все возможные критерии фильтрации. Чтобы создать фильтр, выполните следующие действия.

1. Щелкните на стрелке рядом с названием сетевого протокола, чтобы просмотреть связанные с ним поля критериев. Найдя нужный критерий в качестве основания для создания фильтра, щелкните на нем кнопкой мыши, чтобы выбрать его.
2. Выберите порядок сравнения выбранного поля со значением из критерия. Такое сравнение обозначается с помощью операции больше, меньше, равно и т.д.

3. Составьте фильтрующее выражение, указав значение из критерия, с которым должно сравниваться выбранное поле. Это значение можно указать вручную или выбрать из списка значений, предопределенных в Wireshark.
4. Полученный в итоге фильтр появится в нижней части экрана. По завершении щелкните на кнопке ОК, чтобы ввести его на панели фильтров.

Диалоговое окно `Display Filter Expression` очень удобно для начинающих пользователей Wireshark, но, приобретая некоторый опыт составления фильтров, вы обнаружите, что вводить фильтрующие выражения вручную намного эффективнее. Структура синтаксиса для создания фильтров отображения очень проста, хотя и довольно эффективна.

Структура синтаксиса для создания фильтров отображения

Чем чаще вы станете пользоваться Wireshark, тем больше вам придется пользоваться фильтрами отображения непосредственно в главном окне данного приложения ради экономии времени. Правда, синтаксис для создания фильтров отображения несложен и придерживается стандартной структуры. Зачастую эта структура сосредоточена на сетевых протоколах и следует формату *протокол.средство.подчиненное_средство*. В этом нетрудно убедиться, глядя на диалоговое окно `Display Filter Expression`. А теперь рассмотрим ряд примеров, демонстрирующих порядок построения фильтров отображения.

Чаще всего фильтры перехвата или отображения применяются для просмотра пакетов по отдельному сетевому протоколу. Допустим, требуется провести диагностику проблемы, возникшей на уровне сетевого протокола TCP, и с этой целью перехватить только сетевой трафик по данному протоколу. Для этого вполне подойдет простой фильтр `tcp`.

А теперь рассмотрим эту задачу под другим углом зрения. Допустим, что в ходе диагностики проблемы на уровне сетевого протокола TCP довольно интенсивно использовалась утилита `Ping`, что привело к формированию большого объема сетевого трафика по протоколу `ICMP`. Чтобы отсеять этот трафик из файла перехвата, достаточно применить фильтр `expression !icmp`.

Операции сравнения позволяют сравнивать отдельные значения. Например, в ходе диагностики сетей, действующих по протоколу TCP/IP, нередко требуется просматривать все пакеты со ссылками на конкретный IP-адрес. С помощью операции сравнения на равенство (`==`) можно создать следующий фильтр, отображающий все пакеты с IP-адресом `192.168.0.1`:

```
ip.addr==192.168.0.1
```

А теперь допустим, что требуется просмотреть только те пакеты, длина которых не превышает 128 байт. С этой целью можно использовать операцию сравнения “меньше или равно” (\leq) в следующем фильтре:

```
frame.len<=128
```

В табл. 4.4 перечислены все операции сравнения, используемые в фильтрующих выражениях, составляемых в Wireshark.

Таблица 4.4. Операции сравнения, используемые в фильтрах Wireshark

Операция	Описание
=	Равно
!=	Не равно
>	Больше
<	Меньше
>=	Больше или равно
<=	Меньше или равно

Логические операции позволяют объединять несколько фильтрующих выражений в одном операторе. И благодаря этому заметно повышается эффективность применяемых фильтров.

Допустим, требуется отобразить пакеты только по двум IP-адресам. Чтобы составить единое выражение для фильтрации пакетов, содержащих оба IP-адреса, можно воспользоваться логической операцией `or` следующим образом:

```
ip.addr==192.168.0.1 or ip.addr==192.168.0.2
```

В табл. 4.5 перечислены все логические операции, употребляемые в фильтрующих выражениях, составляемых в Wireshark.

Таблица 4.5. Логические операции, употребляемые в фильтрах Wireshark

Операция	Описание
<code>and</code>	Оба задаваемых условия должны быть истинны
<code>or</code>	Истинным должно быть хотя бы одно задаваемое условие
<code>xor</code>	Истинным должно быть одно и только одно задаваемое условие
<code>not</code>	Ни одно из задаваемых условий не истинно

Примеры выражений для фильтров отображения

Несмотря на то что принципы, по которым составляются фильтрующие выражения, довольно просты, в процессе создания новых фильтров для решения различных задач приходится употреблять ряд особых ключевых слов

и операций. В табл. 4.6 приведены некоторые примеры выражений для наиболее употребительных фильтров отображения, а полный их перечень можно найти в справочном руководстве по фильтрам отображения, доступном по адресу <http://www.wireshark.org/docs/dfref/>.

Таблица 4.6. Наиболее употребительные фильтры отображения

Фильтр	Описание
<code>!tcp.port==3389</code>	Отсеять сетевой трафик по протоколу RDP
<code>tcp.flags.syn==1</code>	Отобразить пакеты TCP с установленным флагом SYN
<code>tcp.flags.reset==1</code>	Отобразить пакеты TCP с установленным флагом RST
<code>!arp</code>	Удалить сетевой трафик по протоколу ARP
<code>http</code>	Отобразить весь сетевой трафик по протоколу HTTP
<code>tcp.port==23 tcp.port==21</code>	Отобразить сетевой трафик по протоколу Telnet или FTP
<code>smtp pop imap</code>	Отобразить сетевой трафик электронной почты (по протоколу SMTP, POP или IMAP)

Сохранение фильтров

Начав однажды создавать немалое количество фильтров перехвата и отображения, можно в какой-то момент обнаружить, что некоторые из них применяются довольно часто. Правда, их не придется вводить всякий раз, когда они потребуются, поскольку в Wireshark предоставляется возможность сохранять фильтры для дальнейшего применения. Так, чтобы сохранить специальный фильтр перехвата, выполните следующие действия.

1. Выберите команду `Capture` ⇔ `Capture Filters` (Перехват ⇔ Фильтры перехвата), чтобы открыть диалоговое окно `Capture Filter`.
2. Создайте новый фильтр, щелкнув на кнопке со знаком “плюс” (+) в левой нижней части диалогового окна.
3. Введите имя нового фильтра в поле `Filter Name` (Имя фильтра).
4. Введите фильтрующее выражение в поле `Filter String` (Строка фильтра).
5. Щелкните на кнопке `OK`, чтобы сохранить фильтрующее выражение.

А для того чтобы сохранить специальный фильтр отображения, выполните приведенные ниже действия.

1. Введите фильтр на панели `Filter`, расположенной над панелью `Packet List` в главном окне Wireshark, а затем щелкните на кнопке с зеленой лентой на левом краю данной панели.
2. Выберите команду `Save this Filter` (Сохранить этот фильтр) из всплывающего меню. В отдельном диалоговом окне будет представлен список

сохраненных фильтров отображения (рис. 4.18). Здесь можно присвоить своему фильтру имя, прежде чем сохранять его, щелкнув на кнопке ОК.

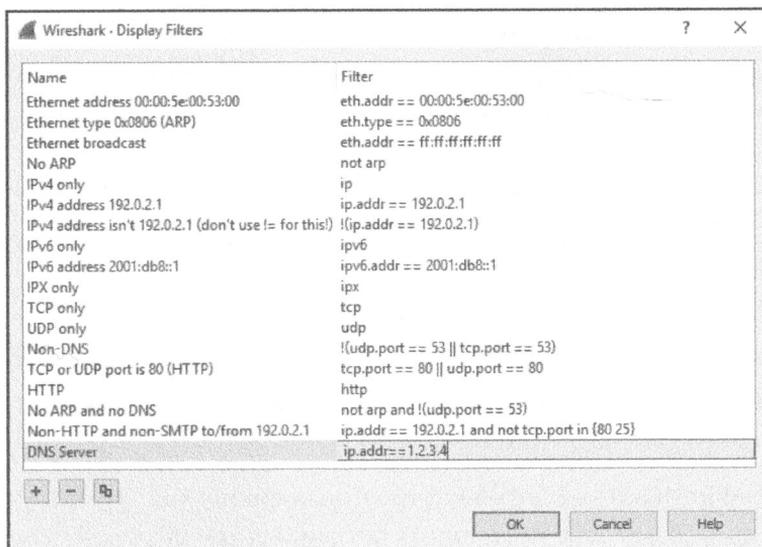


Рис. 4.18. Фильтры отображения можно сохранять непосредственно из главного окна Wireshark

Помещение фильтров отображения на панель инструментов

Если у вас имеются фильтры, которыми вы часто пользуетесь, для удобства взаимодействия с ними проще всего поместить их метки на упомянутой выше панели Filter. Для этого выполните следующие действия.

1. Введите свой фильтр на панели Filter, расположенной над панелью Packet List в главном окне программы Wireshark, а затем щелкните на кнопке со знаком “плюс” (+) справа на данной панели.
2. Ниже панели Filter появится новая панель, где вы можете указать имя своего фильтра в поле Label (Метка; рис. 4.19). Указанная вами метка послужит для представления фильтра на панели инструментов. Введя метку в данном поле, щелкните на кнопке ОК, чтобы создать метку для быстрого доступа к вашему фильтру на панели Filter.

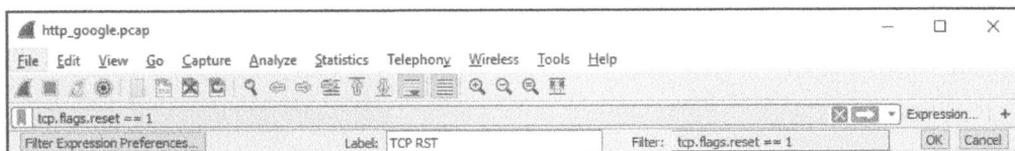


Рис. 4.19. Помещение метки быстрого доступа к фильтру на панель Filter

Как показано на рис. 4.20, в данном случае удалось создать метку быстрого доступа к фильтру для оперативного отображения любых пакетов TCP с установленным флагом RST. Фильтры, помещаемые на панель инструментов, сохраняются в файле конфигурации, упоминавшемся в главе 3, “Введение в Wireshark”. Тем самым существенно расширяются ваши возможности выявлять в самых разных ситуациях затруднения, возникающие в сети на уровне пакетов.

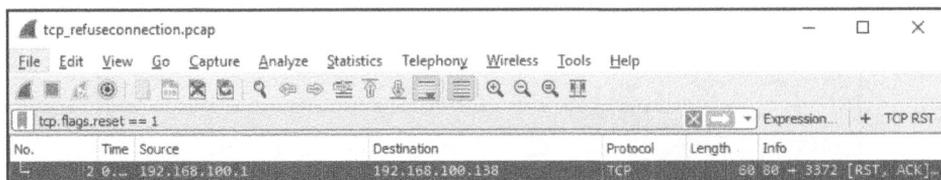


Рис. 4.20. Фильтрация пакетов с помощью меток быстрого доступа на панели инструментов

В состав Wireshark входит ряд встроенных фильтров, которые служат отличными примерами для создания собственных фильтров. Воспользуйтесь ими вместе со справочными страницами Wireshark при создании собственных фильтров. Мы будем часто пользоваться фильтрами в примерах, демонстрируемых далее в этой книге.