

Wireshark 安装

从网络学堂上下载 Wireshark 安装包，解压后运行其中的可执行文件 Wireshark-win32-1.8.6，然后按照安装程序提示进行安装。安装过程中所有选项采用默认的即可。

Wireshark 使用

1. 启动、停止、重新启动数据包捕获

启动 Wireshark，启动后的界面如下：

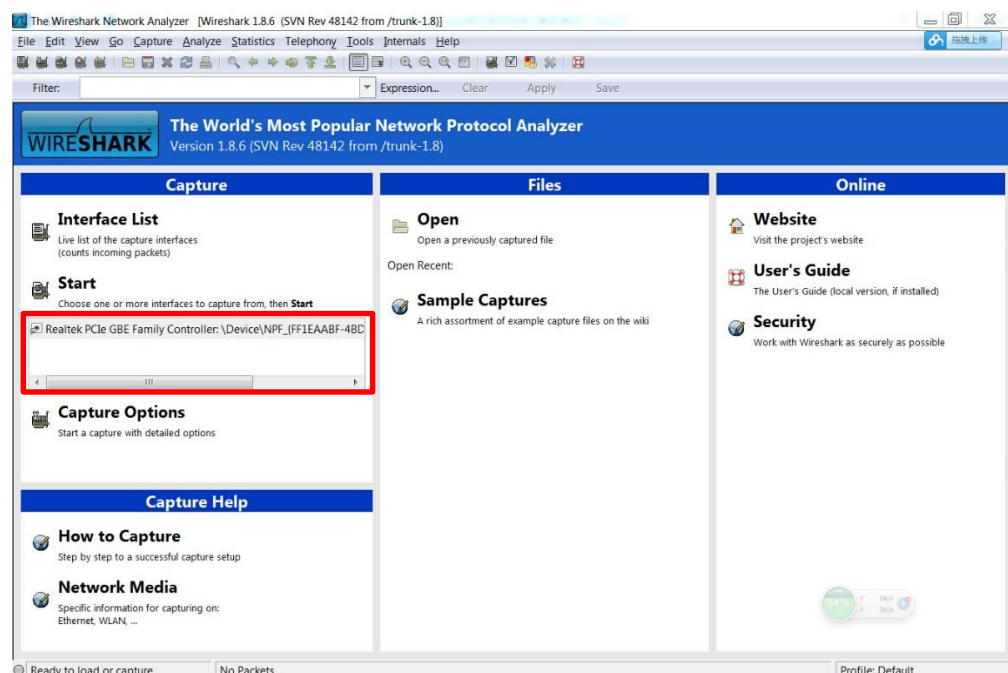


图 1. Wireshark 启动界面

可以看到，在 **Capture** 下面的方框中列出了所有的网卡。根据需要，选择其中的一个或者几个，然后点击 **Start** 开始捕获。在捕获之前，还可以通过“**Capture Options**”里进行一些参数设置，包括存储捕获包的文件选项、何时停止捕获的选项、显示选项以及是否进行名称解析的选项等。一般情况下，使用默认值即可，不用进行特殊设置。

启动捕捉后，Wireshark 界面如图 2 所示：

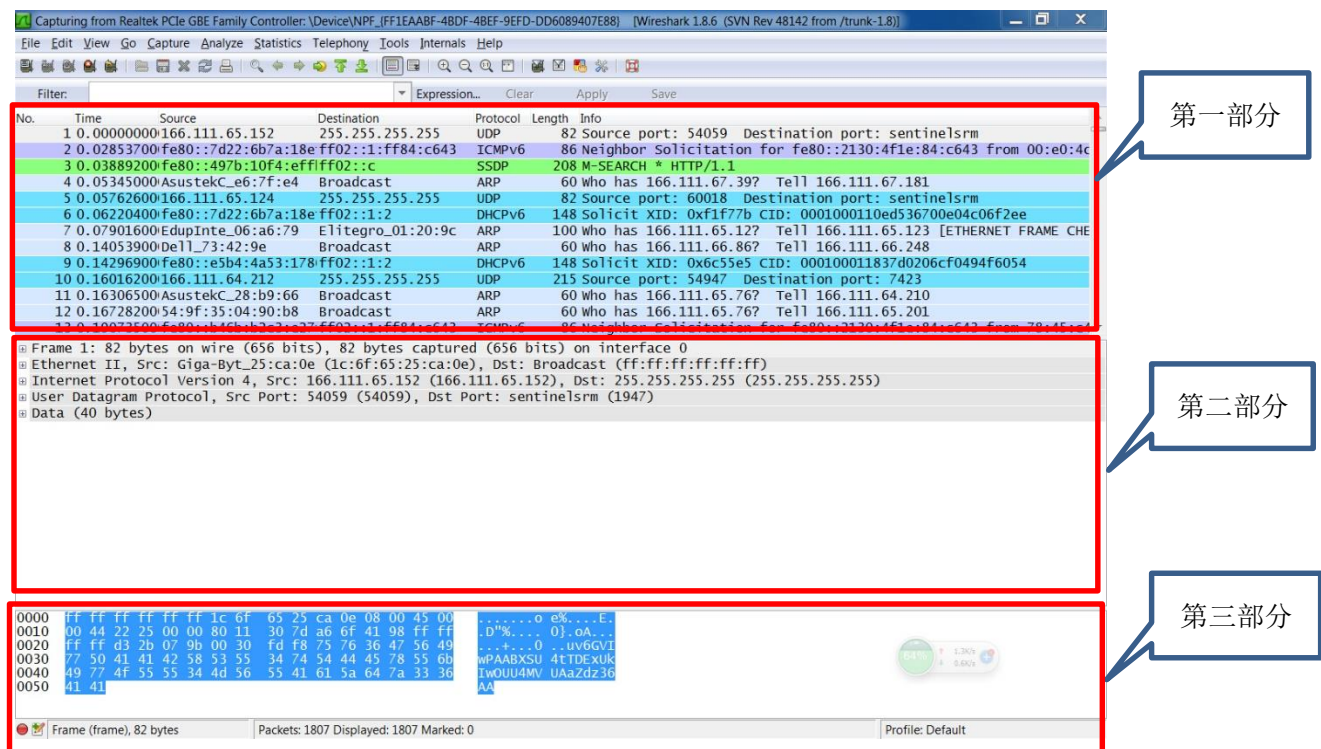


图 2 Wireshark 数据包捕获界面

除了菜单栏，Wireshark 数据包捕获界面主要包括 3 部分。第一部分为“数据包列表”面板。列表中的每行显示捕捉文件的一个包。每一列对应每个数据包的一个协议字段或者协议信息，如源 IP 地址，目的 IP 地址等。通常，因为高层协议会覆盖底层协议，因此在包列表面板看到的都是每个包的最高层协议描述。默认的列包括：



- **No.** 包的编号，编号不会发生改变，即使进行了过滤也同样如此
- **Time** 包的时间戳。包时间戳的格式可以在“View/Time Display Format”中自行设置
- **Source** 显示包的源地址。
- **Destination** 显示包的目标地址。
- **Protocol** 显示包的协议类型的简写
- **Info** 包内容的附加信息

第二部分为“包详情”面板。该面板显示当前包（在包列表面板被选中的包）的协议及协议字段，协议及字段以树状方式组织。可以通过点击左侧的加号展开或折叠它们。

第三部分为“包字节”面板。该面板以 16 进制转储方式显示当前选择包的数据。通常在 16 进制转储形式中，左侧显示包数据偏移量，中间栏以 16 进制表

示，右侧显示为对应的 ASCII 字符。

停止和重新启动数据包捕获

在捕获数据包的过程中，可以通过单击左上侧工具栏中的  图标停止数据包捕获。或通过单击左上侧工具栏中的  图标重启数据包捕获。

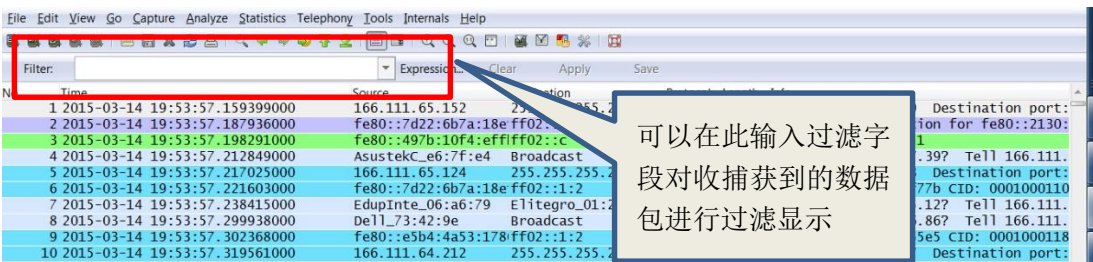
2. 将捕获的数据包保存到文件

可以通过 File->Save as 将捕获的数据包保存到文件中，以供日后分析使用。Wireshark 支持多种保存文件格式，一般默认保存为其原生格式文件(libpcap)，也可以保存为其他格式供其他工具进行读取分析。

也可以通过 File->Open 打开以前保存的文件，以利用 Wireshark 内置的工具进行进一步分析。

3. 数据包过滤

Wireshark 支持对捕捉到的数据包进行过滤显示。可以通过在数据包捕获界面的 Filter 框（如图 3 所示）输入过滤字段对捕捉到的数据包进行过滤显示。



包详情面板的每个字段都可以作为过滤使用。一般来说，每个基本的过滤字段包括两部分，第一部分为协议，如 ip, tcp, http 等，第二部分为该协议下的特定字段。第二部分也可以省略。例如，在过滤框中输入 tcp 将会只显示所有包含 tcp 协议的数据包；在过滤框中输入 ip.addr==192.168.1.32 将会只显示源或目的 IP 地址为 192.168.1.32 的数据包。

在具体使用时，输入协议名称和属性操作符.之后，wireshark 会自动列出该协议的所有字段，然后可以从中选择需要的字段。

表 1 为 Wireshark 支持的比较操作符。在使用时既可以采用符号，如==；也可以采用英文缩写，如 eq。

English	C-linker	描述及范例
eq	==	Equal ip.addr==10.0.0.5
ne	!=	Not equal ip.addr!=10.0.0.5
gt	>	Greater than frame.pkt_len>10
lt	<	Less than frame.pkt_len<128
ge	>=	Greater than or equal to frame.pkt_len ge 0x100
le	<=	Equal frame.pkt_len <= 0x20

表 1. Wireshark 支持的比较操作符

另外，wireshark 也支持用逻辑操作符将过滤表达式组合在一起使用。

Wireshark 支持的逻辑操作符及其使用示例如表 2 所示。

English	C-linker	描述和范例
and	&&	Logical AND ip.addr==10.0.0.5 and tcp.flags.fin
or		Logical OR ip.addr==10.0.0.5 or ip.addr==192.1.1.1
xor	^^	Logical XOR tr.dst[0:3] == 0.6.29 xor tr.src[0:3] == 0.6.29
not	!	Logical Not not llc
[...]		<p>Substring Operator</p> <p>Wireshark 允许选择一个序列的子序列。在标签后你可以加上一对[]号，在里面包含用逗号分离的列表范围。</p> <p>eht.src[0:3] == 00:00:83</p> <p>上例使用 n:m 格式指定一个范围。在这种情况下，n 是起始位置偏移(0 表示没有偏移，即是第一位，同理 1 表示向右偏移一位，便是第二位)，m 是从指定起始位置的区域长度。</p> <p>eth.src[1-2] == 00:83</p> <p>上例使用 n-m 格式一个范围。在本例中 n 表示起始位置偏移，m</p>

English	C-linker	描述和范例
		<p>表示终止位置偏移</p> <pre>eth.src[:4]=00:00:83:00</pre> <p>上例使用:m 格式, 表示从起始位置到偏移位置 m。等价于 0:m</p> <pre>eth.src[4:]=20:20</pre> <p>上例使用 n: 格式, 表示从最后位置偏移 n 个序列</p> <pre>eth.src[2] == 83</pre> <p>上例使用 n 形式指定一个单独的位置。在此例中序列中的单元已经在偏移量 n 中指定。它等价于 n:1</p> <pre>eth.src[0:3, 102, :4, 4:, 2] == 00:00:83:00:83:00:00:83:00:20:20:83</pre> <p>Wireshark 允许你将多个分号隔开的列表组合在一起表示复合区域, 如上例所示</p>

表 2. Wireshark 支持的逻辑操作符

4. 帮助文档

在使用过程中, 如果遇到任何问题, 可以通过 `help->contents` 查看本地的帮助文档。