FS2100 系列 2.5G OC48 过滤卡
用户手册

飞思科技

WWW.FISEE.COM.CN

FS4100 系列 2.5G OC48 过滤卡用户手册

资料编号 DOC- OC48PCI-UM
产品版本 V1.0
资料状态 发行

版权声明

免责声明

本手册依据现有信息制作，其内容如有更改，恕不另行通知。武汉飞思科技有限公司在编写该手册的时候已尽最大努力保证其内容准确可靠，但武汉飞思科技有限公司不对本手册中的遗漏、不准确或错误导致的损失和损害承担责任。

使用者警示

本产品是 A 级通讯产品，当在居住环境中使用时，可能会造成射频干扰，在这种情况下使用者应采取适当的对策以避免影响使用者的正常生活。

手册使用说明

读者对象

本手册的读者对象为安装 FS2100 系列 OC48 过滤卡的工程技术人员和采用 FS2100 系列 OC48 过滤卡的软件开发人员。本手册需要读者具有一定的网络技术知识和经验。

内容介绍

本手册详细介绍了 FS2100 系列 OC48 过滤卡的安装方法，调试命令以及编程接口。

《FS2100 系列 OC48 过滤卡用户手册》共分为六章：

第 1 章 产品综述 详细阐述了 FS2100 系列 OC48 过滤卡的特性和产品规格。
第 2 章 产品硬件结构 详细阐述了 FS2100 系列 OC48 过滤卡的硬件结构和选配部件介绍。
第 3 章 驱动安装 详细阐述了 FS2100 系列 OC48 过滤卡在 windows 和 linux 下驱动和库的安装方法。
第 4 章 调试命令 详细阐述了 FS2100 系列 OC48 过滤卡在使用过程中的需要调试和查看状态的命令。
第 5 章 软件编程接口 详细阐述了 FS2100 系列 OC48 过滤卡提供的软件编程接口。
第 6 章 问与答 详细介绍了 FS2100 系列 OC48 过滤卡在安装过程出现的一些常见问题和解决方案。

获取技术支援

客户在产品使用及网络运行过程中遇到问题时请随时与武汉飞思科技有限公司的服务支持热线联系。此外，客户还可通过武汉飞思科技有限公司网站及时了解最新产品动态，以及下载需要的技术文档。

电话：+86（027）67845143
传真：+86（027）67845062
Email：support@fisee.com.cn
网站：www.fisee.com.cn

手册约定

1．通用格式约定

格 式	意 义
宋体	正文采用宋体表示。
黑体	除一级标题采用宋体**加粗**以外，其余各级标题均采用黑体。
楷体	警告、提示等内容一律用楷体，并且在内容前后增加线条与正文隔离。
“Terminal Display”格式	自定义的“Terminal Display”格式（英文 Courier New； 中文 宋体； 文字大小 8.5）表示屏幕输出信息。此外，屏幕输出信息中夹杂的用户从终端输入的信息采用**加粗**字体表示。

2．命令行格式约定

格 式	意 义
粗体	**命令行关键字（命令中保持不变、必须照输的部分）采用**加粗**字体表示。**
斜体	命令行参数（命令中必须由实际值进行替代的部分）采用*斜体*表示。
[ ]	表示用“[ ]”括起来的部分在命令配置时是可选的。
{ x | y | ... }	表示从两个或多个选项中选取一个。
[x | y | ...]	表示从两个或多个选项中选取一个或者不选。
{ x | y | ... } *	表示从两个或多个选项中选取多个，最少选取一个，最多选取所有选项。
[x | y | ...] *	表示从两个或多个选项中选取多个或者不选。

3．图形界面格式约定

格 式	意 义
< >	带尖括号“< >”表示按钮名，如“单击<确定>按钮”。
[ ]	带方括号“[ ]”表示窗口名、菜单名和数据表，如“弹出[新建用户]窗口”。
/	多级菜单用“/”隔开。如[文件/新建/文件夹]多级菜单表示[文件]菜单下的[新建]子菜单下的[文件夹]菜单项。

4．键盘操作约定

格 式	意 义
加尖括号的字符	表示键名。如<Enter>、<Tab>、<Backspace>、<a>等分别表示回车、制表、退格、小写字母 a。
<键 1 + 键 2>	表示在键盘上同时按下几个键。如<Ctrl+Alt+A>表示同时按下“Ctrl”、“Alt”、“A”这三个键。
<键 1，键 2>	表示先按第一键，释放，再按第二键。如<Alt，F>表示先按<Alt>键，释放后再按<F>键。

5．鼠标操作约定

格 式	意 义
单击	快速按下并释放鼠标的一个按钮。
双击	连续两次快速按下并释放鼠标的一个按钮。
拖动	按住鼠标的一个按钮不放，移动鼠标。

6. 各类标志

小心、注意、警告、危险：提醒操作中应注意的事项。

说明、提示、窍门、思考：对操作内容的描述进行必要的补充和说明。

目 录

第1章 产品综述

1.1 FS2100 系列 OC48 过滤卡简介

FS2100系列OC48过滤卡主要用于对OC48网络中的数据包进行分析，并根据数据包协议头中的某些字段对数据包进行分流。目前，FS2100系列OC48过滤卡支持的分类规则为：目的IP地址＋源IP地址＋协议号＋目的端口号＋源端口号＋TCP标志位。FS2100系列OC48过滤卡具有以下特点：

1. 能够同时支持2路输入，经过规则过滤后，支持2Gbps过滤后数据通过PCI接口采集到服务器。
2. 采用FPGA+CAM的架构，按照五元组（源IP、目的IP、源端口、目的端口和协议）规则对数据进行采集或者丢掉。
3. 采用硬件对数据包进行分类，并将分类好的数据送往不同的应用进程进行处理，能够支持多达16000条分类规则。
4. 数据包采集到服务器时，一次传输 64K 数据和采用零拷贝技术，有效地减少 CPU 占用率。
5. 支持对数据报 VLAN 和 MPLS 头部进行丢掉或者保留处理。
6. 支持对输入的数据包进行截短到指定长度的功能。
7. 支持采样功能，当某一种类别的数据量比较大时，可以对输入的数据进行采样，只允许一部分数据进入。
8. OC48 PCI过滤卡支持对网络上的数据包进行统计；目前可以对65536个源端口、65536个目的端口和256个协议的数据包进行统计。

第2章 产品硬件结构

下面介绍 FS2100 系列 OC48 过滤卡的硬件结构和系统配置清单。

2.1 硬件结构

1. FS2102 正面结构如下图所示：

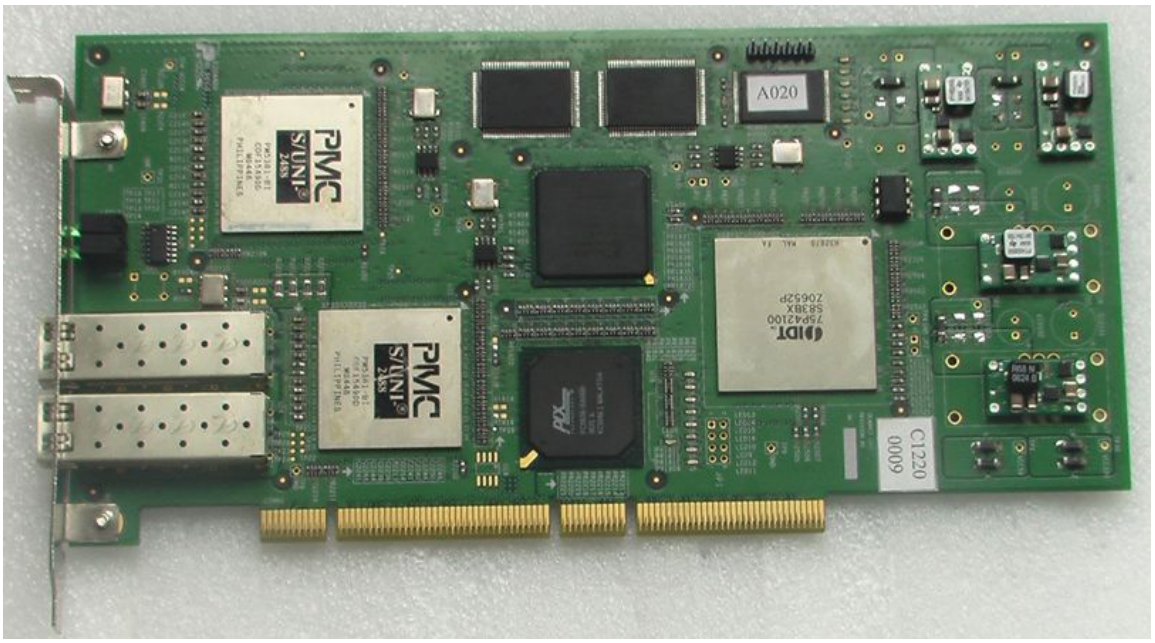

图 2-1 FS2102 正面图

2．FS2102 面板灯的说明

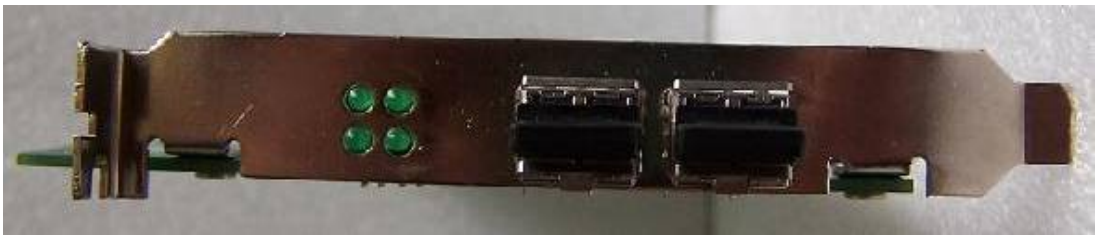

图 2-2 FS2102 测面图

正面向上，光口从左至右为 0 口和 1 口，左边的面板灯对应 0 口，上面为 TX0，下面为 RX0；右边的面板灯对应 1 口，上面的 TX1，下面为 RX1。

2.2 系统配置清单

项 目	部件名称	数 量
1	OC48 过滤卡	1 块
2	2.5G SFP 光收发器	2 个

表 2-1 基本配置清单

第3章 驱动安装

3.1 Windows 下驱动安装

本章介绍了 FS2100 系列 OC48 过滤卡在 windows 下驱动和动态链接库的安装方法。安装所需的文件清单如下：Oc48.sys 和 Oc48.inf 是 OC48 过滤卡的驱动和安装文件；wpcap.lib 和 wpcap.dll 是应用程序需要的动态链接库；include 是需要包含的头文件。

3.1.1 驱动安装

- 将 OC48 过滤卡插入系统中，出现如图 3-1 所示画面，点击“下一步”。

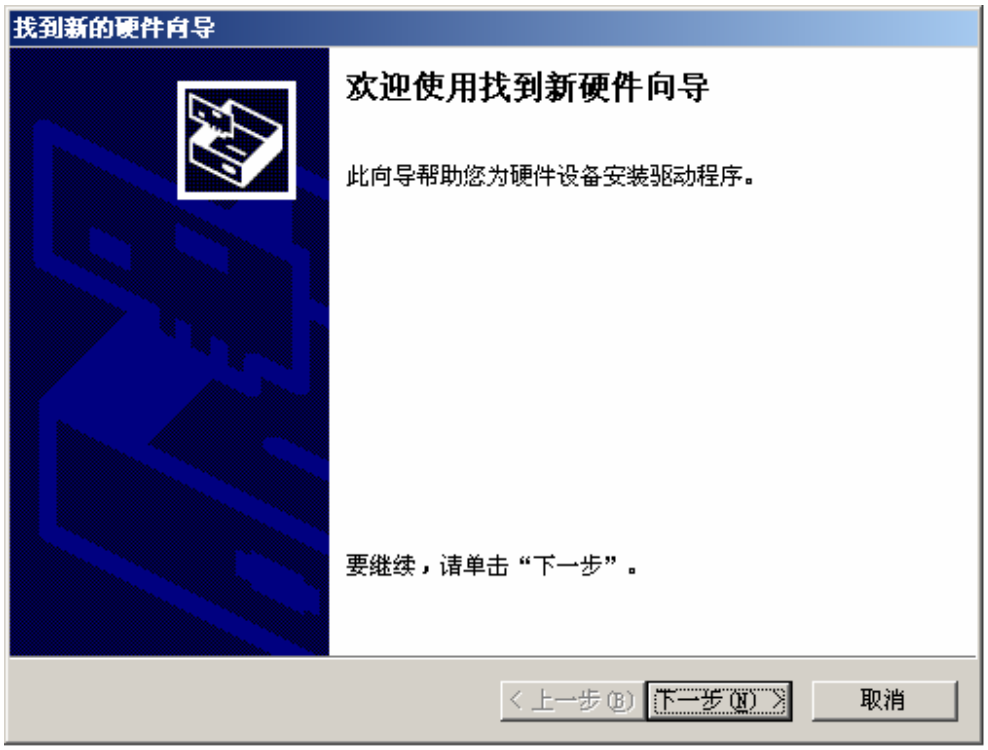

图 3-1 找到新硬件

- 出现如图 3-2 所示画面，选择“搜索适于我的设备的驱动程序”，然后点击“下一步”。

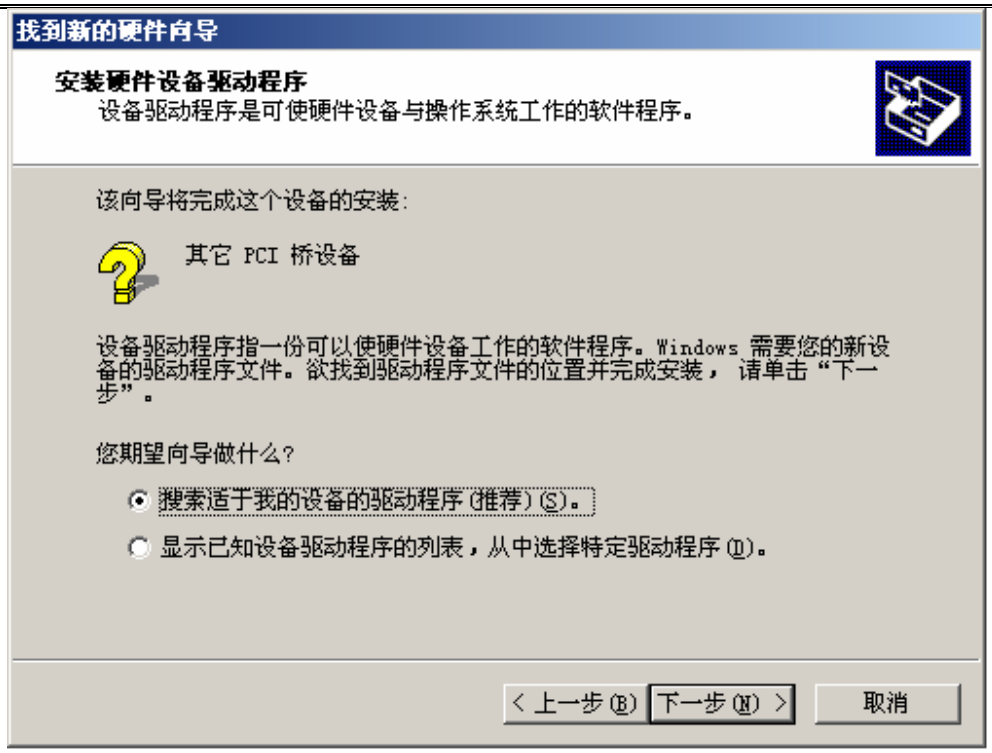

图 3-2 安装硬件设备驱动程序

- 出现如图 3-3 所示画面，选择“指定一个位置”，点击“下一步”。

图 3-3 找到驱动程序文件

- 出现如图 3-4 所示画面，点击“浏览”，选择设备驱动程序所在目录中的“Oc48.inf”文件，点击“确定”。

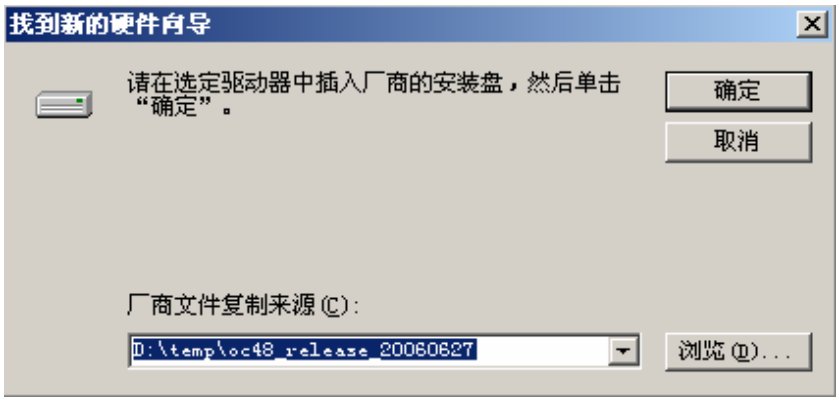

图 3-4 厂商文件复制来源

- 出现如图 3-5 所示画面，点击“下一步”。

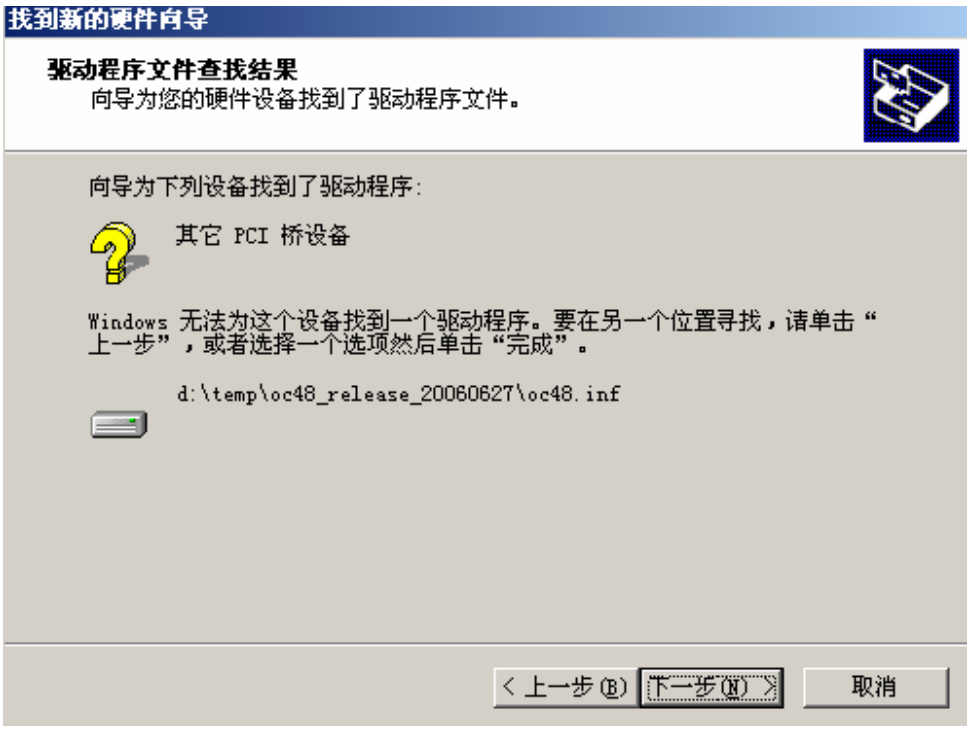

图 3-5 查找驱动程序

- 出现如图 3-6 所示画面，点击“完成”，此时表示已将驱动程序成功安装到系统中。

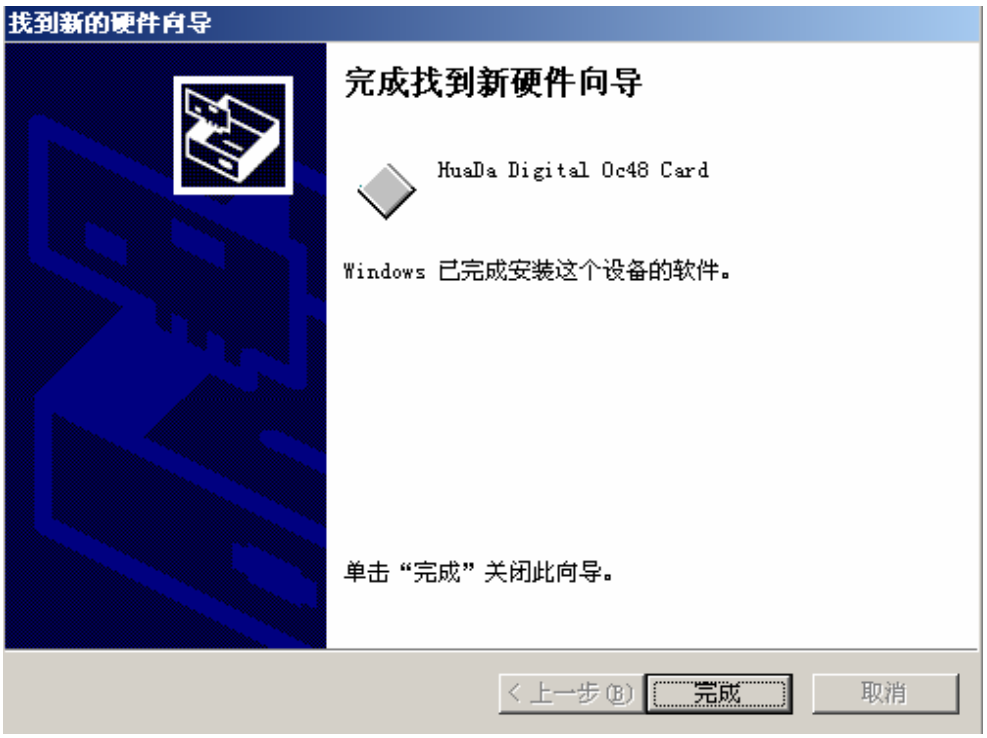

图 3-6 完成驱动程序安装

安装完成后，在“设备管理器”的“其他设备”中出现“Gigabit Ethenet PCI Card”，如图 3-7 所示

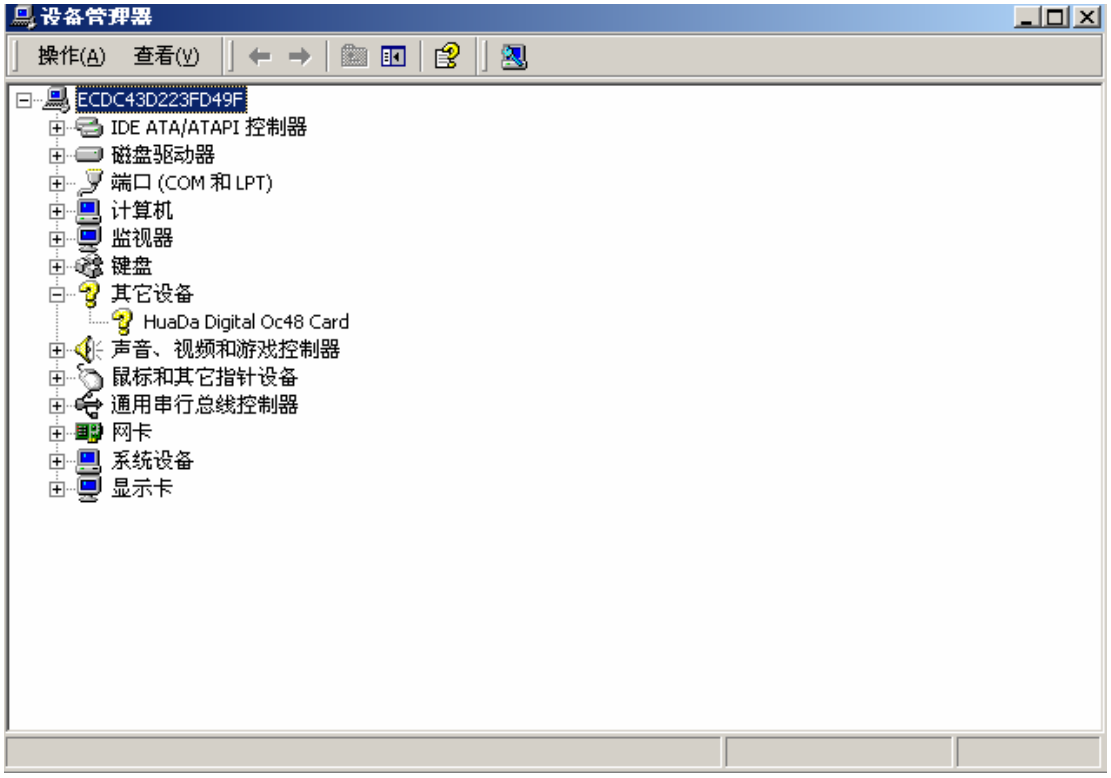

图 3-7 设备管理器

3.1.2 动态链接库安装

运行 WinPcap_3_1.exe，安装 winpcap 库，然后用附件中 wpcap.dll 替换 windows 安装目录下 system32 中的 wpcap.dll 文件。并将 wpcap.lib 添加到你们程序的工程中，并包含附件中的头文件。

3.1.3 OC48 过滤卡测试

example 中有一个 test 程序，用于对安装好的 FS2100 系列 OC48 过滤卡进行测试。用户可以使用这个测试程序通过 OC48 过滤卡来采包。点击 test 程序，出现“please input the rule”提示，然后直接回车，此时会采集所有数据包，并在每收到 1000 个数据包时打印一条信息，如图 3-8 所示：

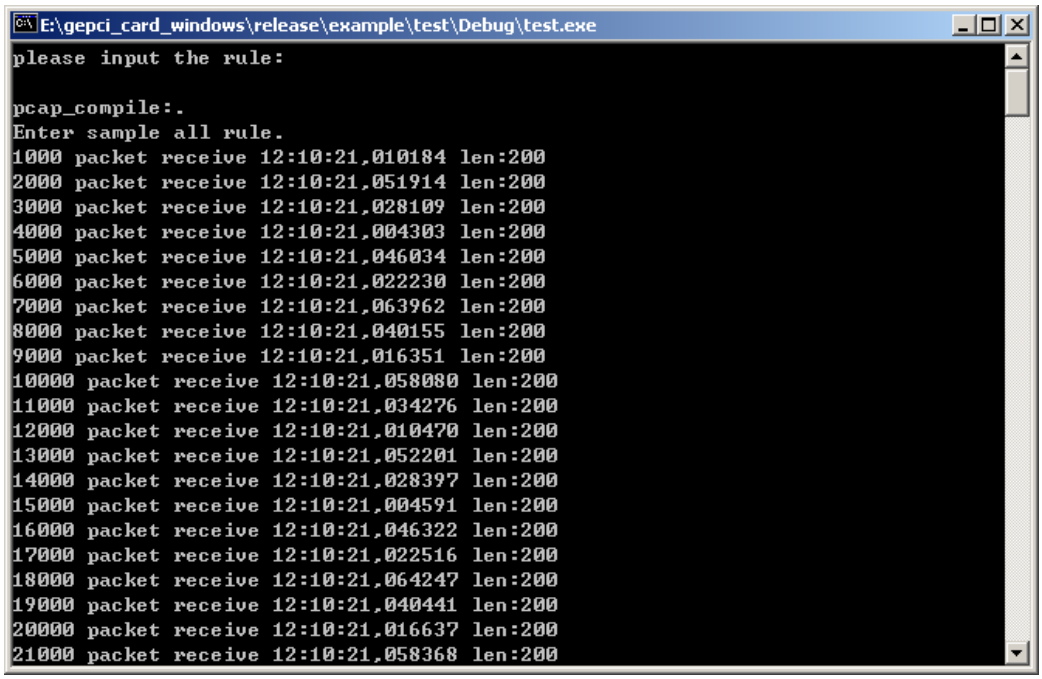

图 3-8 采集所有数据报测试程序

如果要采集端口号是 80 的数据包时，请先点击 test 程序，出现“please input the rule”提示，然后输入过滤规则，如：“port 80”，过滤卡将采集所有端口号为 80 的数据包，并在每收到 1000 个数据包时打印一条信息，如图 3-9 所示：

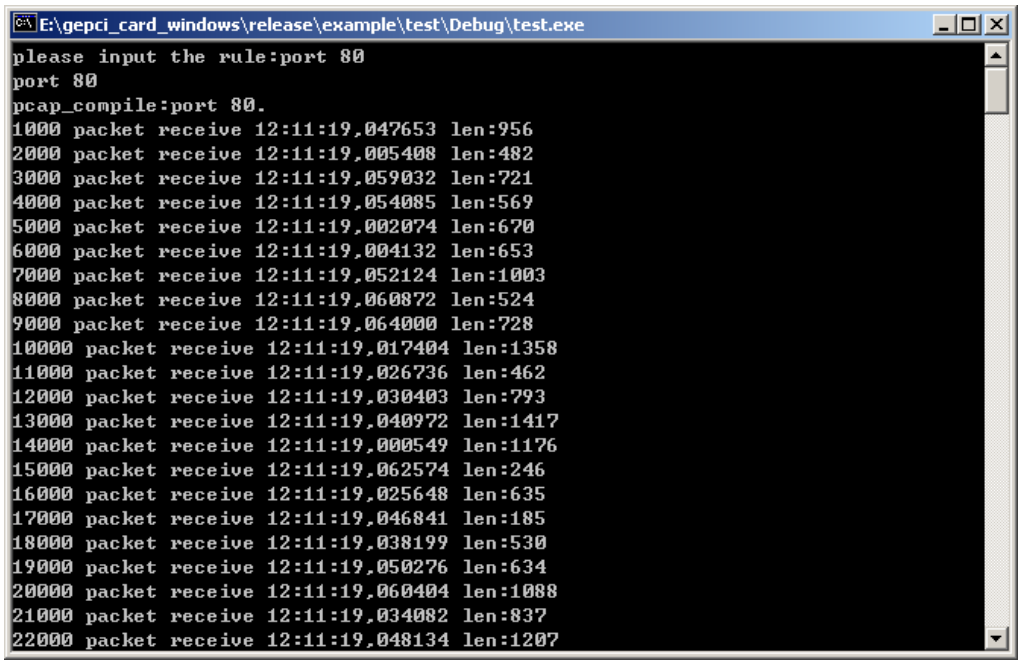

图 3-9 规则过滤测试程序

如果能正常完成图 3-8 的测试程序，表明驱动程序安装正确，否则请重新安装驱动程序。

3.2 Linux 下驱动安装

3.2.1 驱动安装

- 运行命令 tar xvzf oc48pci_card_linux_release.tgz 解开软件包。
- 运行命令 cd oc48_card_linux, 进入解开的软件包中。
- 运行命令 cd oc48_card_driver 进入驱动安装目录。
- 运行命令./build,安装驱动。

3.2.2 动态链接库安装

- 运行命令 cd oc48_card_libpcap 进入动态链接库安装目录。
- 运行命令./build，安装 libpcap 库。

3.2.3 OC48 过滤卡测试

- 运行命令 cd tcpdump-3.8.3 进入测试目录，运行./tcpdump -nne,采集所有数据包，如图 3-10 所示：

```
[fb@localhost tcpdump-3.8.3]$ ./tcpdump -nne
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on gfilter, link-type EN10MB (Ethernet), capture size 96 bytes
11:22:30.019866 00:00:00:00:00:1f > 00:00:00:00:00:00, ethertype IPv4 (0x0800), leng
th 891: IP 198.19.1.2.1024 > 198.19.1.1.1025: . 123456:124293(837) ack 234567 win 40
96
11:22:30.019941 00:00:00:00:00:1f > 00:00:00:00:00:00, ethertype IPv4 (0x0800), leng
th 282: IP 198.19.1.2.1024 > 198.19.1.1.1025: . 0:228(228) ack 1 win 4096
11:22:30.021811 00:00:00:00:00:1f > 00:00:00:00:00:00, ethertype IPv4 (0x0800), leng
th 1300: IP 198.19.1.2.1024 > 198.19.1.1.1025: . 0:1246(1246) ack 1 win 4096
11:22:30.021935 00:00:00:00:00:1f > 00:00:00:00:00:00, ethertype IPv4 (0x0800), leng
th 480: IP 198.19.1.2.1024 > 198.19.1.1.1025: . 0:426(426) ack 1 win 4096
11:22:30.023109 00:00:00:00:00:1f > 00:00:00:00:00:00, ethertype IPv4 (0x0800), leng
th 888: IP 198.19.1.2.1024 > 198.19.1.1.1025: . 0:834(834) ack 1 win 4096
11:22:30.023182 00:00:00:00:00:1f > 00:00:00:00:00:00, ethertype IPv4 (0x0800), leng
th 276: IP 198.19.1.2.1024 > 198.19.1.1.1025: . 0:222(222) ack 1 win 4096
11:22:30.025031 00:00:00:00:00:1f > 00:00:00:00:00:00, ethertype IPv4 (0x0800), leng
th 1288: IP 198.19.1.2.1024 > 198.19.1.1.1025: . 0:1234(1234) ack 1 win 4096
11:22:30.025150 00:00:00:00:00:1f > 00:00:00:00:00:00, ethertype IPv4 (0x0800), leng
th 457: IP 198.19.1.2.1024 > 198.19.1.1.1025: . 0:403(403) ack 1 win 4096
11:22:30.026235 00:00:00:00:00:1f > 00:00:00:00:00:00, ethertype IPv4 (0x0800), leng
th 842: IP 198.19.1.2.1024 > 198.19.1.1.1025: . 0:788(788) ack 1 win 4096
11:22:30.026286 00:00:00:00:00:1f > 00:00:00:00:00:00, ethertype IPv4 (0x0800), leng
th 185: IP 198.19.1.2.1024 > 198.19.1.1.1025: . 0:131(131) ack 1 win 4096
11:22:30.027796 00:00:00:00:00:1f > 00:00:00:00:00:00, ethertype IPv4 (0x0800), leng
th 1106: IP 198.19.1.2.1024 > 198.19.1.1.1025: . 0:1052(1052) ack 1 win 4096
```

图 3-10 采集所有数据报测试程序

第4章 调试命令

本章详细介绍了用于调试和查看 FS2100 系列 OC48 过滤卡状态和统计的命令。

4.1 Logout

该命令主要用于退出调试程序。

命令格式

logout

参数说明

无。

具体描述

logout

说明：退出调试程序。

4.2 set filter-rule

该命令用于配置过滤卡的规则。

命令格式

set filter-rule [index *priority* **] [protocol** *pt* **] [srcip** *sip* **] [dstip** *dip* **] [srcport** *sp* **] [dstport** *dp* **] action { drop | forward | sample /base /ratio }**

参数说明

index	设置过滤规则索引号，十六进制数，范围：[0x30~0x3ffe]。
protocol	设置过滤规则的协议类型，范围：tcp / udp 。
srcip	设置过滤规则的源 IP，例如 192.168.88.1 。
dstip	设置过滤规则的目的 IP，例如 192.168.88.1 。
srcport	设置过滤规则的源端口，例如 8080。
dstport	设置过滤规则的目的端口，例如 8080。
drop	将符合规则的数据包丢掉。
forward	将符合过滤规则的数据包转发。
sample	数据包采样率设置。
base	设置采样率的进制。
ration	设置采样率。

具体描述

set filter-rule index 66 protocol dstip 192.168.88.2 srcport 1024 action forward

说明：设置一条索引号为 0x66 的规则，将目的 ip 为 192.168.88.2，且源端口为 1024 的 tcp 包送往服务器。

4.3 show filter-rule

该命令用于显示已配置的过滤规则。

命令格式

show filter-rule [*num***]**

参数说明

num 要查看的规则的索引号，省略代表显示所有规则。

具体描述

show filter-rule

说明：查看所有的过滤规则。结果如图所示：

```
OC48PCI>set filter-rule srcport 80 action forward

OC48PCI>show filter-rule
index  sip             dip             sp     dp     pt   forward   ratio
48     0.0.0.0         0.0.0.0         80     0      tcp  yes       N/A
```

4.4 delete filter-rule

该命令用于删除已配置的过滤规则。

命令格式

delete filter-rule *num*

参数说明

num 要删除的规则的索引号。

具体描述

delete filter-rule 66

说明：删除索引号为 0x66 的过滤规则。

4.5 delete all-filter

该命令用于删除所有已配置的过滤规则。

命令格式

delete all-filter

参数说明

无。

具体描述

delete all-filter

说明：删除所有已配置的过滤规则。

4.6 set pmrc long-pkt

该命令用于设置 pmrc 入口的最长包长。

命令格式

set pmrc long-pkt {*size*}

参数说明

size 设置 pmrc 入口的最长包，十进制数，范围：[536~65535]。

具体描述

set pmrc long-pkt 1500

说明：设置 pmrc 入口的最长包长为 1500。

4.7 set pmrc short-pkt

该命令用于设置 pmrc 入口的最短包长。

命令格式

set pmrc short-pkt {*size*}

参数说明

size 设置 pmrc 入口的最短包长，十进制数，范围：[1~1500]。

具体描述

set pmrc long-pkt 30

说明：设置 pmrc 入口的最长包长为 30。

4.8 set pmrc rx-mode

该命令用于设置 pmrc 入口开关。

命令格式

set pmrc rx-mode {*type*}

参数说明

type pmrc 入口开关值，范围：enable / disable。

具体描述

set pmrc rx-mode disable

说明：关闭 pmrc 入口。

4.9 set pmrc vlan

该命令用于设置 pmrc 的 vlan 处理模式。

命令格式

set pmrc vlan {*type*}

参数说明

type pmrc 的 vlan 处理模式，范围：remove / bypass。

具体描述

set pmrc vlan remove

说明：设置 pmrc 的 vlan 处理模式为 remove。

4.10 set pmrc mpls

该命令用于设置 pmrc 的 mpls 处理模式。

命令格式

set pmrc mpls {*type*}

参数说明

type pmrc 的 mpls 处理模式，范围：remove / bypass。

具体描述

set pmrc mpls remove

说明：设置 pmrc 的 mpls 处理模式为 remove。

4.11 show pmrc config

该命令用于查看 pmrc 的配置信息。

命令格式

show pmrc config

参数说明

无。

具体描述

show pmrc config

说明：显示 pmrc 的配置信息。结果如图所示：

```
OC48PCI>show pmrc config
In function f_get_pmrc_config.
rx_parity EVEN
long_pkt=1540
short_pkt=40
rx enable
remove VLAN
remove MPLS
```

4.12 show pmrc statistics

该命令用于查看 pmrc 的统计信息。

命令格式

show pmrc statistics

参数说明

无。

具体描述

show pmrc statistics

说明：显示 pmrc 的统计信息。结果如图所示：

```
OC48PCI>show pmrc statistics
channel A
 per_rx_pkt_cnt=0 per_tx_pkt_cnt=0
 rx_long_pkt=0 rx_short_pkt=0
 rx_pkt_cnt=0-0 rx_byte_cnt=0-0
 rx_byte_rate = 0 rx_pkt_rate = 0

channel B
 per_rx_pkt_cnt=0 per_tx_pkt_cnt=0
 rx_long_pkt=0 rx_short_pkt=0
 rx_pkt_cnt=0-0 rx_byte_cnt=0-0
 rx_byte_rate = 0 rx_pkt_rate = 0

MI Transmitted to PLXIF: Packets 0  Frames 0 Useful Frames 0
MI Transmitted to PLXIF: Packets 0  Frames 0 Useful Frames 0
CAM Interface to MUX search keys : received  0  sented 0
```

4.13 set framer fcs

该命令用于设置 OC48 接口的 fcs。

命令格式

set framer fcs *type*

参数说明

type fcs 的类型，范围：crc-16/crc-32

具体描述

set framer fcs crc-32

说明：设置 OC48 接口的 fcs 类型为 crc-32。

4.14 set framer scramble

该命令用于设置 OC48 接口的 scramble。

命令格式

set framer scramble *type*

参数说明

type Scramble 模式，范围：enable/disable

具体描述

set framer scramble enable

说明：设置 OC48 接口的绕码为 enable。

4.15 show framer config

该命令用于查看 OC48 接口的配置信息。

命令格式

show framer config

参数说明

无。

具体描述

show framer config

说明：显示 OC48 接口的配置信息。结果如图所示：

```
OC48PCI>show pmc5381 config
--------pm5381 config---------
min pkt = 8
max pkt = 32768
threshold = 7
burst = 3
Scramble : Enable
FCS length : Crc-32
```

4.16 show framer statistics

该命令用于查看 OC48 接口的统计信息。

命令格式

show framer statistics *port*

参数说明

Port 端口号，范围：[1~2]。

具体描述

show framer statistics 1

说明：显示 OC48 接口 1 的统计信息。结果如图所示：

```
OC48PCI>show pmc5381 statistics 1
(047H)section BIP: 24543  (048H)line BIP: 01952  (049H)line REI: 64480
path BIP error counter: 49088 00000 00000 00000 00000 00000 00000 00000 00000
00000 00000 00000 00000 00000 00000 00000 00000 00000 00000 00000 00000 00000
00000 00000 00000 00000 00000 00000 00000 00000 00000 00000 00000 00000 00000
00000 00000 00000 00000 00000 00000 00000 00000 00000 00000 00000 00000 00000
path REI error counter: 00000 00000 00000 00000 00000 00000 00000 00000 00000
00000 00000 00000 00000 00000 00000 00000 00000 00000 00000 00000 00000 00000
00000 00000 00000 00000 00000 00000 00000 00000 00000 00000 00000 00000 00000
00000 00000 00000 00000 00000 00000 00000 00000 00000 00000 00000 00000 00000
positive justification: 00000 00000 00000 00000 00000 00000 00000 00000 00000
00000 00000 00000 00000 00000 00000 00000 00000 00000 00000 00000 00000 00000
00000 00000 00000 00000 00000 00000 00000 00000 00000 00000 00000 00000 00000
00000 00000 00000 00000 00000 00000 00000 00000 00000 00000 00000 00000 00000
negative justification: 00000 00000 00000 00000 00000 00000 00000 00000 00000
00000 00000 00000 00000 00000 00000 00000 00000 00000 00000 00000 00000 00000
00000 00000 00000 00000 00000 00000 00000 00000 00000 00000 00000 00000 00000
00000 00000 00000 00000 00000 00000 00000 00000 00000 00000 00000 00000 00000
(749H)RX byte (hi): 0000000000      (747-748H)RX byte (lo): 0911842184
(74a-74bH)RX packet: 0000000000     (74cH)RX erred FCS HCS: 00000
(74dH)RX aborted packet: 00000      (74eH)RX min length packet error: 00000
(74fH)RX max len packet err: 00000  (76bH)RX FIFO indirect cells packets: 00000
(76cH)RX FIFO accepted: 0000000000  (76eH)RX FIFO dropped: 00000
```

4.17 show framer status

该命令用于查看 OC48 接口的状态。

命令格式

show framer status *port*

参数说明

Port 端口号，范围：[1~2]。

具体描述

show framer status 1

说明：显示 OC48 接口 1 的状态。

4.18 show statistics

该命令用于查看端口号的统计信息。

命令格式

show statistics *type data*

参数说明

type 范围：pt（协议类型）/ sp（源端口）/ dp（目的端口）。

data 十进制数，范围：[0~65535]。

具体描述

show statistics sp 4000

说明：显示源端口号 4000 的统计信息。

4.19 set card

该命令用于选择设备。

命令格式

set card {*num***}**

参数说明

num 卡号，范围：1 到使用的卡数。

具体描述

set card 1

说明：选择 1 号卡。

第5章 应用编程接口

5.1 接口概述

下表列出各调用接口的名称以及功能的概述,在其后部分将对每一接口及其功能进行详细的描述：

函数名	功能概述
pcap_open_live	获取采集句柄。
pcap_compile	解析规则和配置规则。
pcap_set_default_rule	配置设备的默认规则。
pcap_del_filter	删除设备配置的规则
pcap_loop	开始进行采集.该函数内部处于循环状态。
pcap_breakloop	中断采集。
pcap_close	销毁采集句柄，释放资源。
pcap_stats	获取应用进程收包和丢包的统计信息。
pcap_card_statistics	获取过滤采集卡每个入口包数、字节数、速率以及出口包数、字节数、速率等统计信息。
pcap_ptport_statistics	获取网络上某种协议、某个源端口或某个目的端口的包数和字节数。
pcap_version	库版本号。

5.2 基本数据结构

5.2.1 pcap_t

pcap_t 为设备句柄,用户使用该句柄访问设备和采集数据包。

5.2.2 pcap_handler

pcap_handler 为回调函数的指针.对采集到的数据包进行处理.该类型的声明如下:

```
typedef void (*pcap_handler)(u_char *usr_data, const struct pcap_pkthdr *hdr,
                const u_char *p);
```

其中,usr_data 为用户自定义数据；hdr 为数据包信息，p 为数据包指针，后两项是系统传入的参数，用户直接使用即可。

5.2.3 pcap_pkthdr

pcap_pkthdr 包含数据包信息，目前只支持数据封包的长度。

```
struct pcap_pkthdr {
        struct timeval ts;     /* time stamp */
        bpf_u_int32 caplen;/* length of portion present */
        bpf_u_int32 len;     /* length this packet (off wire) */
        };
```

5.2.4 pcap_stat

pcap_stat 包含应用进程收包和丢包的统计信息，具体结构如下：

```
struct pcap_stat {
    u_int ps_recv;       /* number of packets received */
    u_int ps_drop;       /* number of packets dropped */
    u_int ps_ifdrop;     /* drops by interface XXX not yet supported */
};
```

5.2.5 card_statistics

card_statistics 包含 OC48 过滤卡每个入口的包数、字节数和每个出口的包数、字节数的 64 位统计数，同时提供了每个入口和出口的包速率（pps）和字节速率（Bps），具体结构如下：

```
struct card_statistics
{
   unsigned int    rx_pkt_cnt[4];      /* Received Packets Counter low 32 bit */
   unsigned int    rx_byte_cnt[4];    /* Received Bytes Counter low 32 bit */
   unsigned int    rx_pkt_cnt_hi[4]; /* Received Packets Counter high 32 bit */
   unsigned int    rx_byte_cnt_hi[4]; /* Received Bytes Counter high 32 bit */
   unsigned int    rx_pkt_rate[4];      /* Received Packets Rate */
   unsigned int    rx_byte_rate[4];    /* Received Bytes Rate */

   unsigned int    ixf_tx_pkt_cnt[4]; /* Sent Packets Counter low 32 bit */
   unsigned int    ixf_tx_byte_cnt[4]; /* Sent Bytes Counter low 32 bit */
   unsigned int    ixf_tx_pkt_cnt_hi[4]; /* Sent Packets Counter high 32 bit*/
   unsigned int    ixf_tx_byte_cnt_hi[4]; /* Sent Bytes Counter high 32 bit */
   unsigned int       ixf_tx_pkt_rate[4]; /* Sent Packets Rate */
   unsigned int       ixf_tx_byte_rate[4]; /* Sent Bytes Rate */
```

```
};
```

5.2.6 card_ptport_statistics

card_ptport_statistics 包含某种协议、某个源端口或者某个目的端口数据包的包数和字节数，具体结构如下：

```
struct card_ptport_statistics
{
    unsigned short type;/*0:no data , 1:pt , 2:sp , 3:dp */
    unsigned short data;/*sp or dp or pt*/
    unsigned int byte_cnt; /*byte count*/
    unsigned int pkt_cnt; /*packet count*/
};
```

5.3 功能详述

5.3.1 pcap_open_live

通过该函数获取采集句柄. 应用程序通过采集句柄访问设备。

函数原型

pcap_t * pcap_open_live(const char *device, int snaplen, int promisc, int to_ms, char *errbuf)

入口参数

device	用于描述要打开设备的名称,如果该参数为 NULL,则表明使用默认的设备. 一般情况下采用默认值。当该参数为 ethN, N 取大等于零的整数时，该 libpcap 包为标准 libpcap 包。
snaplen	采集数据封包的最大长度.即如果封包实际含有大于 snaplen 的长度,但也只采集该封包的前 snaplen 个字节，在本版本中不关心此参数。
promisc	描述本应用所需要的缓存空间大小，以 2M 字节为基本单位。一般情况下采用参数 1（表示需要分配 2M 字节空间）。
to_ms	超时参数，在本版本中不关心此参数。

出口参数

errbuf	如果该参数不为 NULL,则 errbuf 指向的内存空间一定能够容纳 GCAP_ERROR_BUFFER_SIZE 个字符,使用该口进行过滤规则过程中发生错误时,该接口会将发生错误的描述信息放入该指针指

向空间,如果该参数为 NULL,则表明用户并不在意错误描述信息。

返回值

创建成功,返回设备句柄,否则返回 NULL,如果 errbuf 不为 NULL,则通过 errbuf 传递错误信息。

5.3.2 pcap_compile

该函数用于解析规则和配置规则到设备。

函数原型

int pcap_compile (pcap_t *handle, struct bpf_program *program, char *buf, int optimize, bpf_u_int32 mask)

入口参数

handle	通过 pcap_open_live 获得的采集句柄.用该句柄访问设备。该参数无默认值。
program	在本版本中不关心此参数,留扩展用。
buf	保存需要设置的过滤规则。规则的关键字用空格分开，比如 buf = "tcp port 80" 此规则解析为过滤 tcp 类型的数据包并且数据包的目的或者源端口为 80。
optimize	在本版本中不关心此参数,留扩展用。
mask	在本版本中不关心此参数,留扩展用。

返回值

成功返回 0,否则返回-1,错误原因可能是用户向该函数传递了非法的参数。

5.3.3 pcap_set_default_rule

该函数用于配置设备的默认规则。设备有一条默认规则用来处理不匹配任何规则的数据流。丢弃和转发是默认的两种处理。

函数原型

int pcap_set_default_rule(pcap_t * handle, char *act_str);

入口参数

handle	通过 pcap_open_live 获得的采集句柄.用句柄访问设备。该参数无默认值。
act_str	保存需要设置的默认过滤规则。丢弃和转发是默认的两种处理，

比如 act_str = "drop" 丢弃不匹配任何规则的数据流；act_str = "forward" 从 0 口转发不匹配任何规则的数据流。

返回值

成功返回 0,否则返回-1,错误原因可能是用户向该函数传递了非法的参数。

5.3.4 pcap_del_rule

该函数用于删除设备配置的规则。

函数原型

int pcap_del_filter(pcap_t *pcap,int rule_id);

入口参数

pcap	通过 pcap_open_live 获得的采集句柄.用句柄访问设备。该参数无默认值。
rule_id	需要被删除规则的索引号。

返回值

成功返回 0,否则返回-1。

5.3.5 pcap_loop

开始进行采集.该函数内部处于循环状态,直到用户规定的 cnt 个数据封包为止.该函数每接收到一个数据封包,都会调用 callback 注册函数对数据包处理. 用户可调用下面描述的 gcap_breakloop 结束循环状态。

函数原型

int pcap_loop (pcap_t *handle, int cnt, pcap_handler callback, u_char *user) ;

入口参数

handle	通过 pcap_open_live 获得的设备句柄.用句柄访问设备。该参数无默认值。
cnt	限定接收数据包的个数，当接收满 cnt 个数据后 pcap_loop 返回。当 cnt 小于等于零时表示无限循环接收收据包。当没有数据包时此函数返回。
callback	注册的回调函数，该函数每接收到一个数据封包,都会调用 callback 注册函数对数据包处理。
user	用户自定义数据，传递给 callback 使用。

返回值

成功返回 0 ,否则返回-1,错误原因可能是用户传递了错误的参数。

5.3.6 pcap_breakloop

停止在指定数据采集句柄上的所有数据采集处理.如果在一线程中对某采集句柄进行了 pcap_loop 操作,可以在另外的线程中调用 pcap_breakloop 来中止该循环。

函数原型

void pcap_breakloop (pcap_t *handle) ;

入口参数

handle 通过 pcap_open_live 获得的设备句柄.用句柄访问设备。该参数无默认值。

5.3.7 pcap_close

销毁采集句柄.采集工作完成后,使用该函数释放采集句柄本身所使用的资源。

函数原型

int pcap_close(pcap_t * handle) ;

入口参数

handle 待销毁的采集句柄。

返回值

销毁成功返回 0, 否则返回-1, 失败的原因可能是该句柄仍然处于使用中, 或者是用户向该函数传递了非法的参数。

5.3.8 pcap_stats

获取对应采集描述句柄的采集信息.

函数原型

int pcap_stats (pcap_t *handle, struct pcap_stat *stats);

入口参数

handle 通过 pcap_open_live 获得的采集句柄.用句柄访问设备。该参数无默认值。

出口参数

stats pcap 对应的采集句柄的统计信息.包括接收到和丢弃的数据包统计信息。

返回值

销毁成功返回 0, 否则返回-1。

5.3.9 pcap_card_statistics

获取过滤采集卡每个入口包数、字节数、速率以及出口包数、字节数、速率等统计信息。

函数原型

int pcap_card_statistics(pcap_t *pcap,struct card_statistics *entry);

入口参数

pcap 通过 pcap_open_live 获得的采集句柄，用句柄访问设备。该参数无默认值。

出口参数

entry 过滤采集卡的入口和出口的统计信息。

返回值

获取成功返回 0, 否则返回-1。

5.3.10 pcap_ptport_statistics

获取网络上某种协议、某个源端口或某个目的端口的包数和字节数。

函数原型

int pcap_ptport_statistics(pcap_t *pcap,struct card_ptport_statistics *entry);

入口参数

pcap 通过 pcap_open_live 获得的采集句柄，用句柄访问设备。该参数无默认值。

出口参数

entry 某种协议或某个源目的端口的数据结构。

返回值

获取成功返回 0, 否则返回-1。

5.3.11 pcap_version

获取 libpcap 库的版本号。

函数原型

uint32_t pcap_version() ;

返回值

libpcap 库的版本号, 高 16 位返回主版本号, 低 16 位返回次版本号。

5.4 规则配置

5.4.1 基本规则

规则是过滤网络数据包的根据，本系统从网络上过滤数据包，检查网络上数据包是否匹配设置的规则，假如匹配则接受或者按比率采集数据；否则就丢弃数据包。

基本规则是不可再分的规则，本系统中的基本规则如下表所示，每一种编号代表一种基本规则：

①	**src port** value
②	**dst port** value
③	**src host** value (如 1.1.1.1)
④	**dst host** value (如 2.2.2.2)
⑤	**src net** value (如 1.1.0.0)
⑥	**dst net** value (如 2.2.0.0)
⑦	**tcp**
⑧	**udp**
⑨	**icmp**
⑩	**igmp**
⑪	**pt** value

5.4.2 规则的组合运算

该版本 libgcap 支持两种规则运算：“与”和“或”运算。

“与”运算用”and”表示，比如用“src port 80 and dst port 8080”表示基本规则①与上基本规则②形成一条新的规则。

“或”运算用”or”表示，比如用“src port 80 or dst port 8080”表示基本规则①或上基本规则②形成一条新的规则。有些关键字可以用来简化“或”运算，如下表所示：

port value	**dst port** value **or src port** value
net value	**dst net** value **or src net** value
host value	**dst host** value **or src host** value

按照从左至右的顺序解析规则，“与” 和“或”具有同等的有先级别。比如规则 “src port 80 and dst port 8080 or net 192.168.88.1”可以解析为以下规则的组合：

“src port 80 and dst port 8080”

“src net 192.168.88.1”

“dst net 192.168.88.1”

5.4.3 过滤规则

过滤规则是规则加上动作构成的，是配置到设备中的规则。

Filter rule = rule + action,不加 action 则默认为 forward。 本系统目前支持的 action 有以下几种：

Index	Action
①	**Drop**
②	**Forward**
③	**Sample10** 和 **Sample16**

Drop：代表符合该规则的数据包将会被丢弃。

Forward：代表符合该规则的所有数据包将会被送往服务器或者从端口转发出去。

Sample：代表符合该规则的数据包只有一部分可以送往服务器或从端口转发出

去，另外一部分会被丢弃，但是会保证 session 的完整性。进行 sample 时，会将数据包的源 ip，目的 ip，源端口和目的端口通过 hash 运算算出一个 0－15 的值，然后以这个值去索引一个 16bit 的数中的某一位，当这位为 1 时，将数据包送往服务器，否则将这个数据包丢弃。对于 sample，提供了两种方式，Sample10_后面直接接一个 0－15 的数值，表示 16 分之几的 session 会被送往服务器。Sample16_后面直接接一个 16bit 的数。Sample10_5 等价于 sample16_001F。

5.5 接口函数使用举例

5.5.1 举例一……接收 tcp port 80 数据包

```
#include <pcap.h>
#include <stdio.h>

void pcap_print(u_char *user, const struct pcap_pkthdr *h, const
u_char *p)
{
    int i;

    for ( i = 0; i <h->len; i++)
    {
            printf( "%02x-", p[i] );
    }
    printf("\n");
}


int main()
{
   pcap_t *handle; /* the handle */
   char *dev=NULL; /* the device */
   char errbuf[PCAP_ERRBUF_SIZE]; /* err buf */
   struct bpf_program filter; /* filter, no used */
   char filter_app[] = "tcp port 80"; /* the rule */
   bpf_u_int32 mask; /* no used */
```

```
    bpf_u_int32 net; /* no used */
    struct pcap_pkthdr header; /* header pointer */
    const u_char *packet;

    /* Define the device */
    handle = pcap_open_live(dev, 1600, 1, 0, errbuf);
    pcap_compile(handle, &filter, filter_app, 0, net);

    pcap_setfilter(handle, &filter);

    pcap_loop(handle, -1, pcap_print, NULL); //this function
                                        //to test the pcap_loop
    pcap_close(handle);

    return(0);
}
```

第6章 问与答

- **问：**安装好 FS2100 系列 OC48 过滤卡后，在 windows 中，系统检测不到过滤卡；或者在 linux 中插入驱动会出错。
  答：将 FS2100 系列 OC48 过滤卡从服务器中取出，用橡皮或餐巾纸轻轻擦拭 OC48 过滤卡的金手指部分。
- **问：**安装好 FS2100 系列 OC48 过滤卡后，驱动安装正常，并且对应口的状态灯是亮的，但是 OC48 过滤卡采集不到数据包。
  答：运行 parse 程序，修改一下 OC48 过滤卡的 scramble 和 fcs 参数。
- **问：**如何查看每个端口输入的数据包个数，字节数，以及输入的数据包速率（pps）和字节速率（Bps）。
  答：运行 parse 程序，输入 show pmrc statistics 命令，即可查看每个端口输入的数据包情况。
- **问：**OC48 过滤卡提供的 libpcap 库是否支持标准的网卡的 libpcap 库。
  答：是支持的，在条用 pcap_open_live 打开设备时，输入的设备名为 eth 开头时，将调用标准的网卡 libpcap 库，要使用 OC48 过滤卡，输入的设备名应为 crad 或者为空。