一、前言

在网络分析和监测工具中，RouterOS 抓包被广泛应用于网络安全和故障排除领域。RouterOS 是一种功能强大的路由操作系统，它提供了丰富的功能和工具，支持多种网络协议，用于管理和保护网络。抓包是其中一项重要的功能，它允许管理员捕获、分析和监测网络流量，以便识别潜在的安全威胁和故障。

考虑到各个环境的复杂性及限制性，本文将从图形化、命令行两个维度讲述抓包技巧，同时将深入探讨如何配置和使用 RouterOS 的抓包功能，以及如何解读和分析捕获的网络数据包。

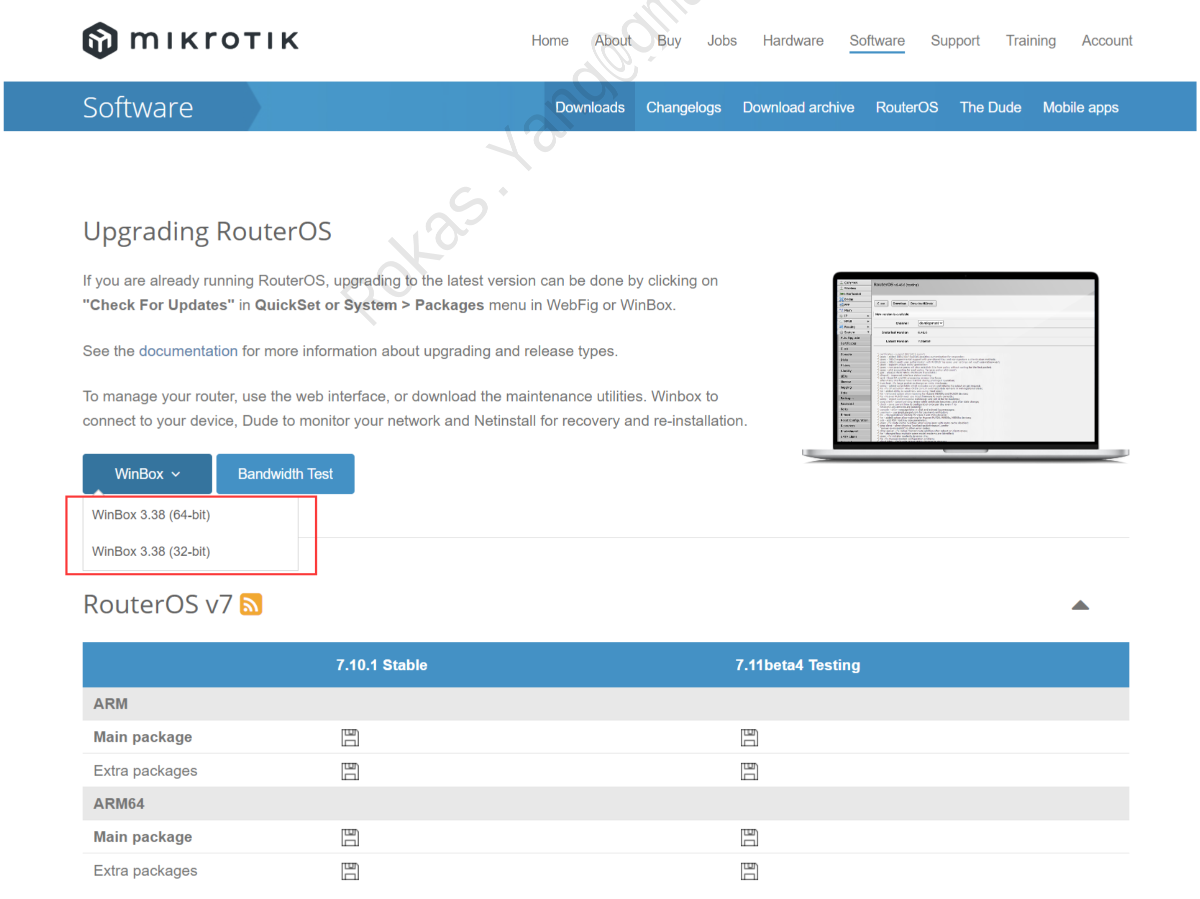
二、图形化抓包

图形化界面抓包是最简单粗暴的方式，如果没有环境限制首推此方式。

1.Packet Sniffer

1）winbox下载

在图形化界面抓包，需要用到Ros的客户端工具winbox，winbox可以在 **Ros的官网** 下载:



MIKROTIK
Home About Buy Jobs Hardware Software Support Training Account
Software
Downloads Changelogs Download archive RouterOS The Dude Mobile apps
WinBox
Bandwidth Test
WinBox 3.38 (64-bit)
WinBox 3.38 (32-bit)

Upgrading RouterOS

If you are already running RouterOS, upgrading to the latest version can be done by clicking on **"Check For Updates"** in **QuickSet or System > Packages** menu in WebFig or WinBox.

See the documentation for more information about upgrading and release types.

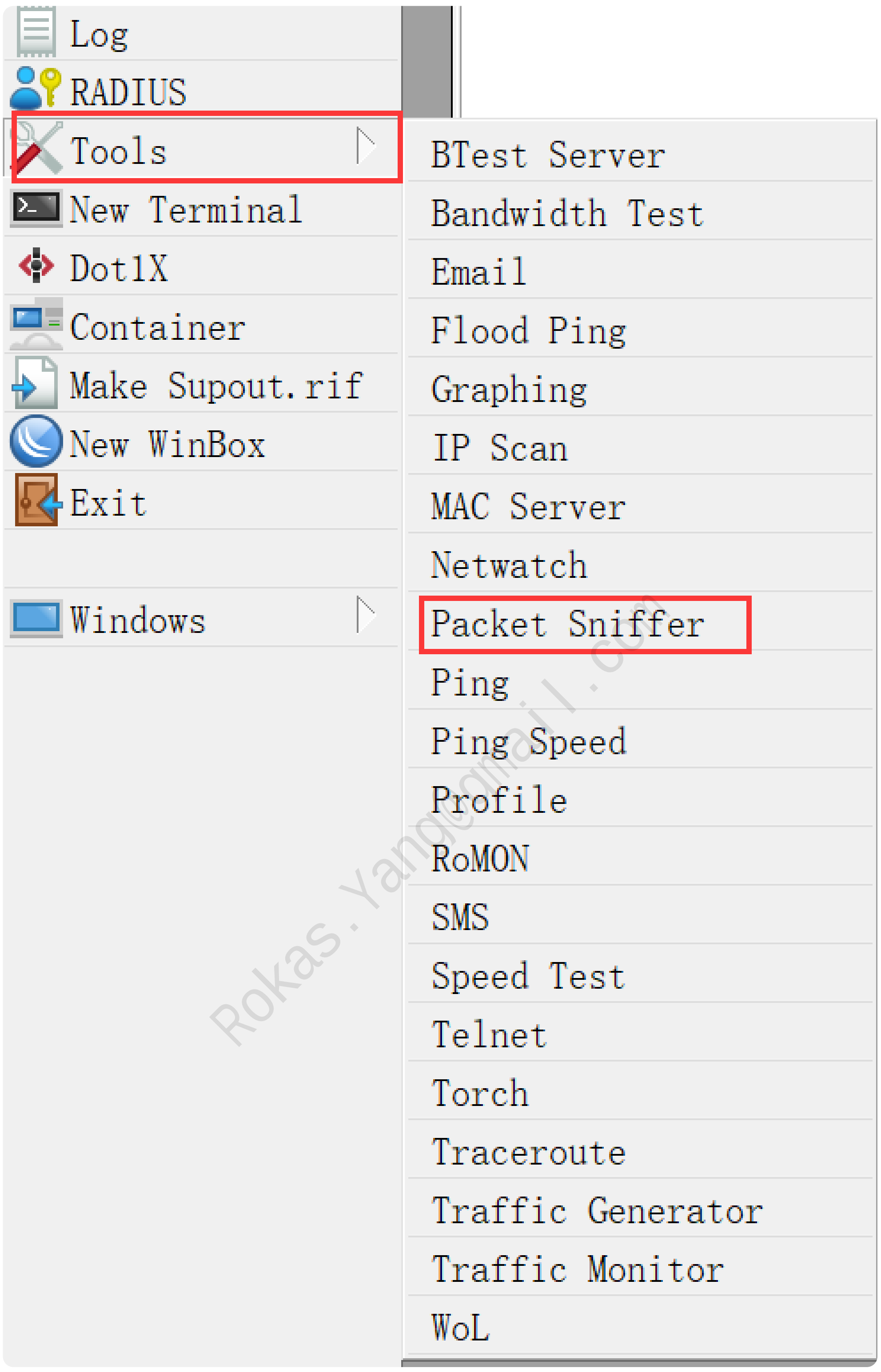
To manage your router, use the web interface, or download the maintenance utilities. Winbox to connect to your device, Dude to monitor your network and Netinstall for recovery and re-installation.

RouterOS v7

	7.10.1 Stable	7.11beta4 Testing
ARM		
Main package	💾	💾
Extra packages	💾	💾
ARM64		
Main package	💾	💾
Extra packages	💾	💾

2）选项概览

使用winbox登录到Ros后，点击 **Tools** --> **Packet Sniffer** 即为抓包工具所在之处：

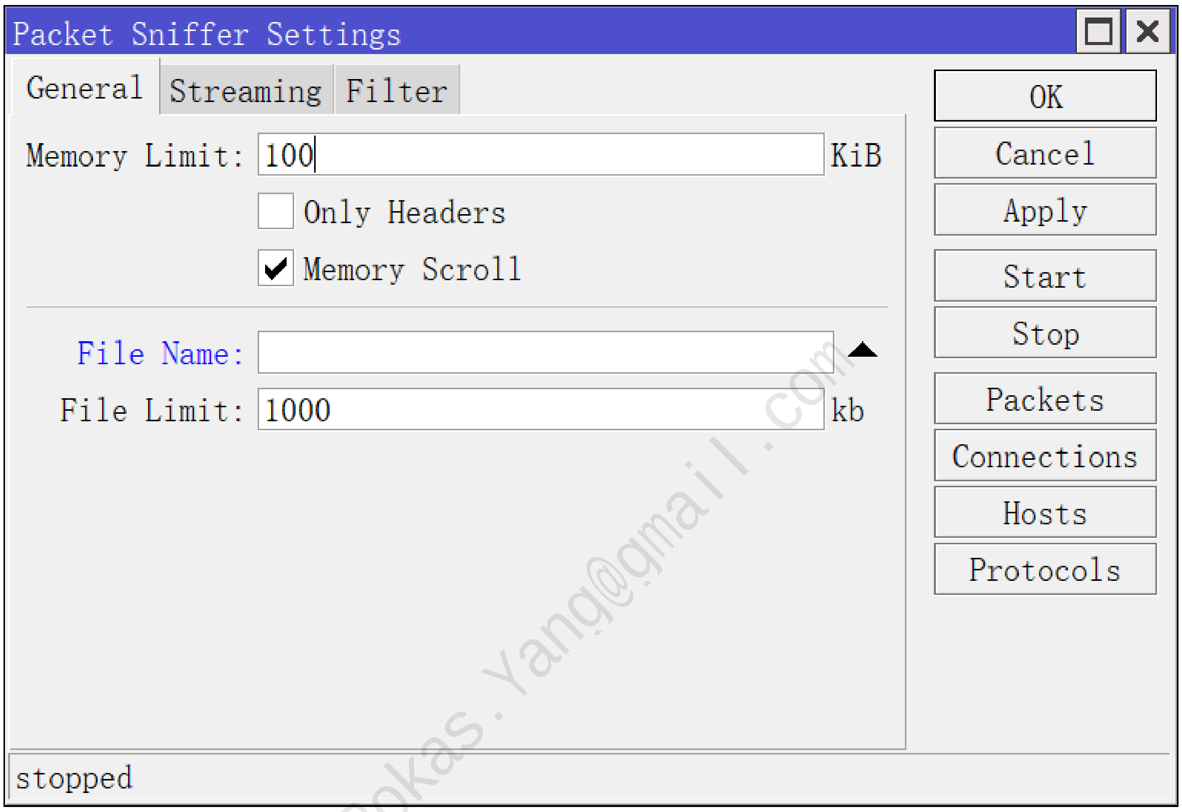


初始界面如下，分为General、Streaming、Filter三部分配置，接下来将说明每个选项的参数含义。

3）General选项说明

通用(General)选项参数含义：

- **Memory Limit**：最大使用内存大小，默认100KB;
- **Only Headers**：只抓取头部信息；
- **Memory Scroll**：捕获的数据包会存储在内存中，直到内存滚动缓冲区被填满。一旦缓冲区达到容量上限，最早捕获的数据包将被丢弃以腾出空间，而新的数据包将继续写入缓冲区；
- **File Name**：抓包后保存到的文件名，如果不想保存可以留空；
- **File Limit**：抓包文件大小限制，默认1000kb。


Packet Sniffer Settings

Memory Limit: 100 KiB

☐ Only Headers

✓ Memory Scroll

File Name:

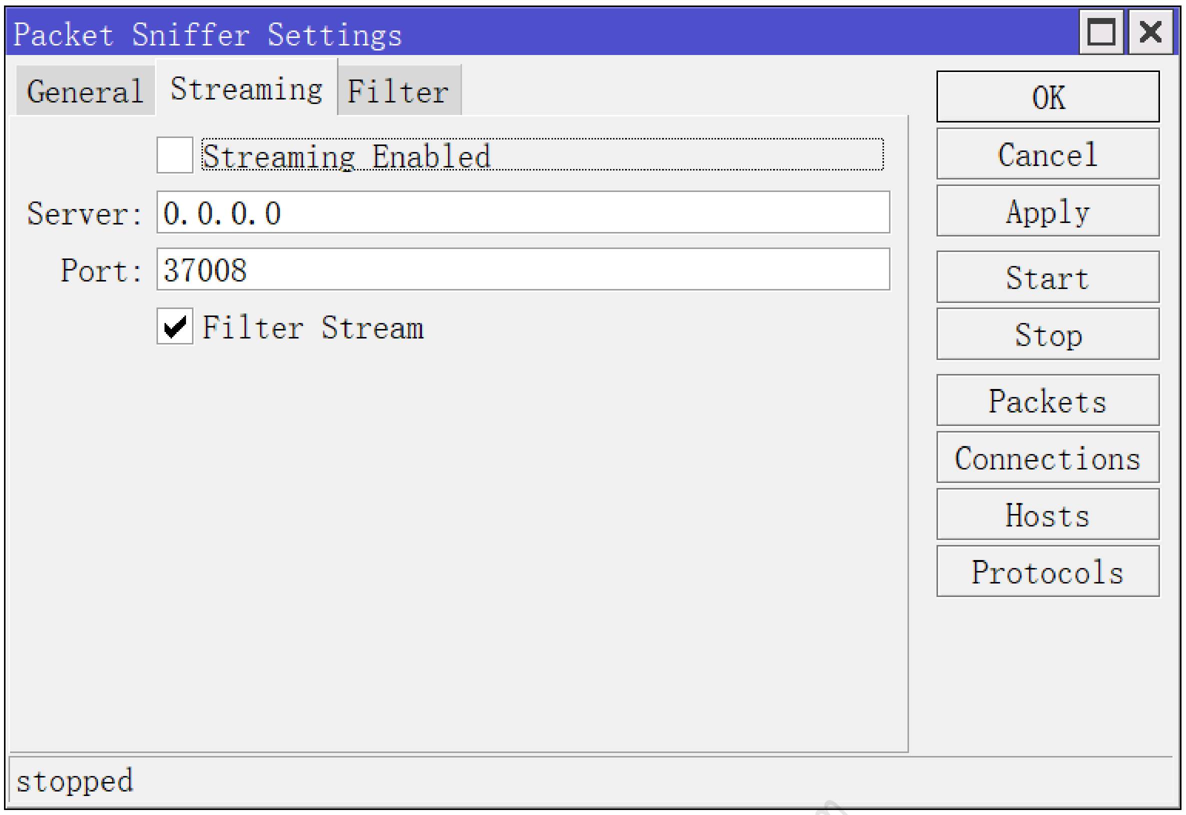
File Limit: 1000 kb

stopped

4）Streaming选项说明

Streaming选项参数含义：

- **Streaming Enabled**：打开Streaming选项，将捕获的数据包转发给指定地址；
- **Server**：转发到对端的IP地址；
- **Port**：转发给对端的端口；
- **Filter Stream**：过滤流，过滤特定流，只有启用此参数后，Filter选项的参数设置才能生效。


Packet Sniffer Settings

General | Streaming | Filter

☐ Streaming Enabled

Server: 0.0.0.0

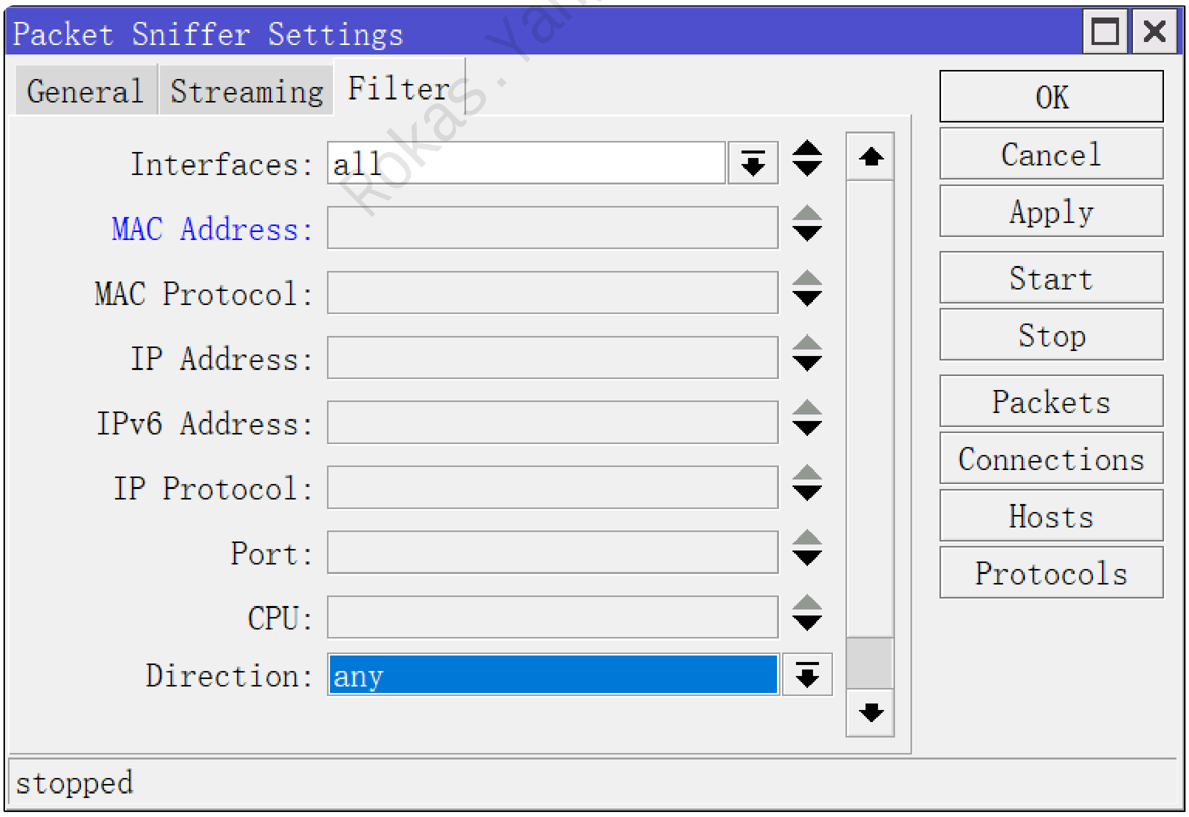
Port: 37008

☑ Filter Stream

stopped

5）Filter选项说明

Filter则为具体的过滤规则，以下参数言简意赅，用来指定特定接口、过滤特定IP、Mac地址、协议类型、端口、出入方向等的报文：


Packet Sniffer Settings

General | Streaming | Filter

Interfaces: all

MAC Address:

MAC Protocol:

IP Address:

IPv6 Address:

IP Protocol:

Port:

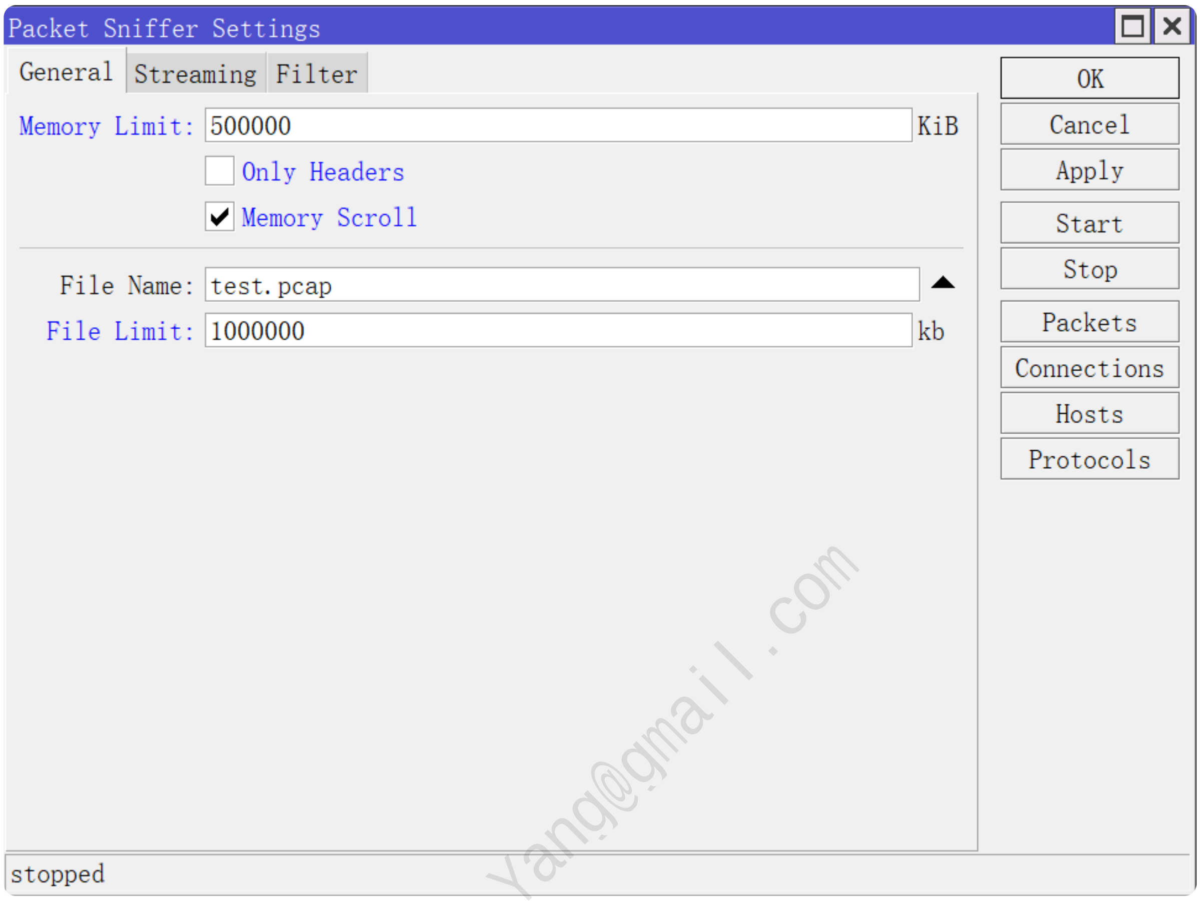
CPU:

Direction: any

stopped

6）抓包设置

比如想把抓包指定到文件test.pcap，则如下设置，抓包文件大小最大为1000Mb，最大使用内存为500MB，抓取Wan口的所有报文，则可以如下设置：


Packet Sniffer Settings

General | Streaming | Filter

Memory Limit: 500000 KiB

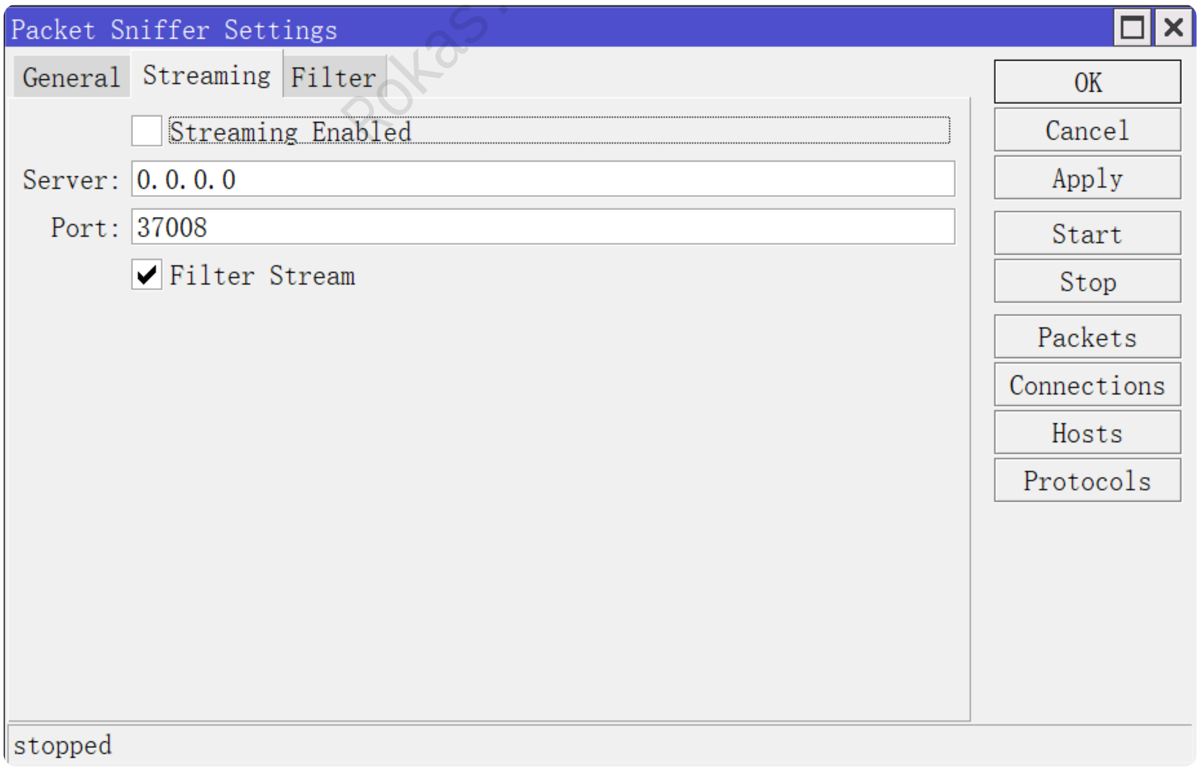
[ ] Only Headers

[✓] Memory Scroll

File Name: test.pcap

File Limit: 1000000 kb

stopped


Packet Sniffer Settings

General | Streaming | Filter

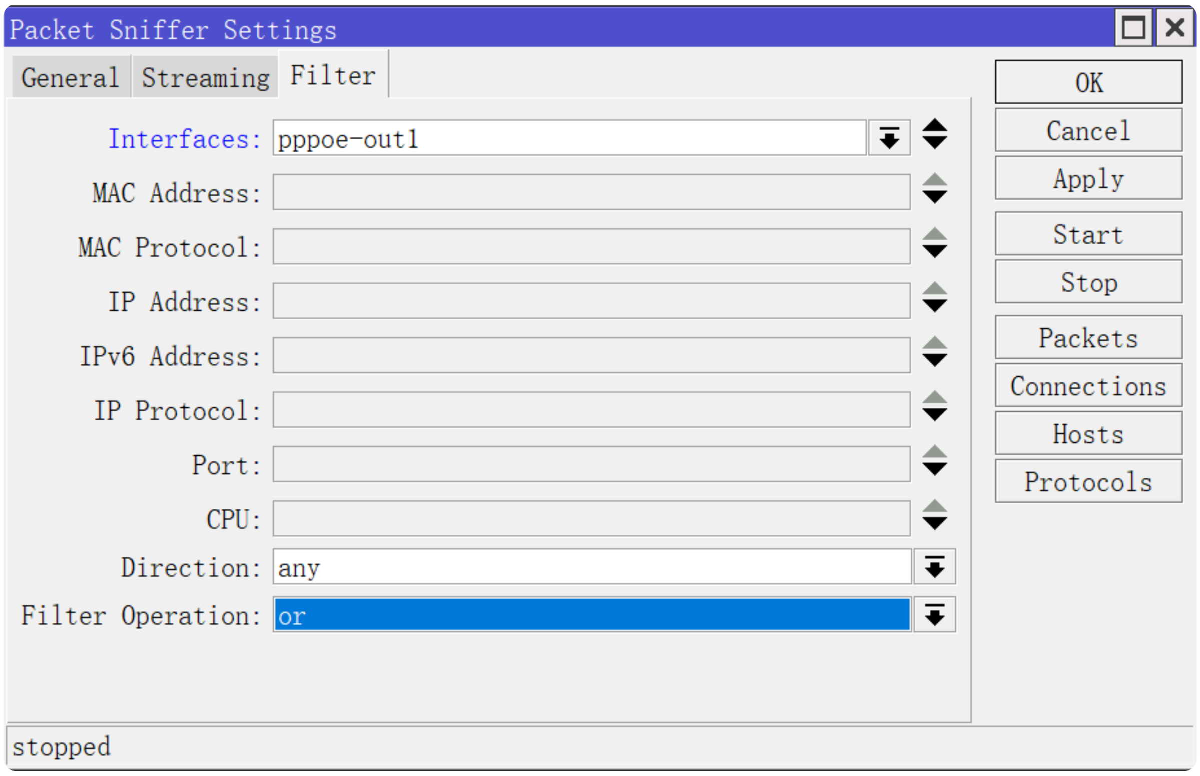
[ ] Streaming Enabled

Server: 0.0.0.0

Port: 37008

[✓] Filter Stream

stopped


Packet Sniffer Settings

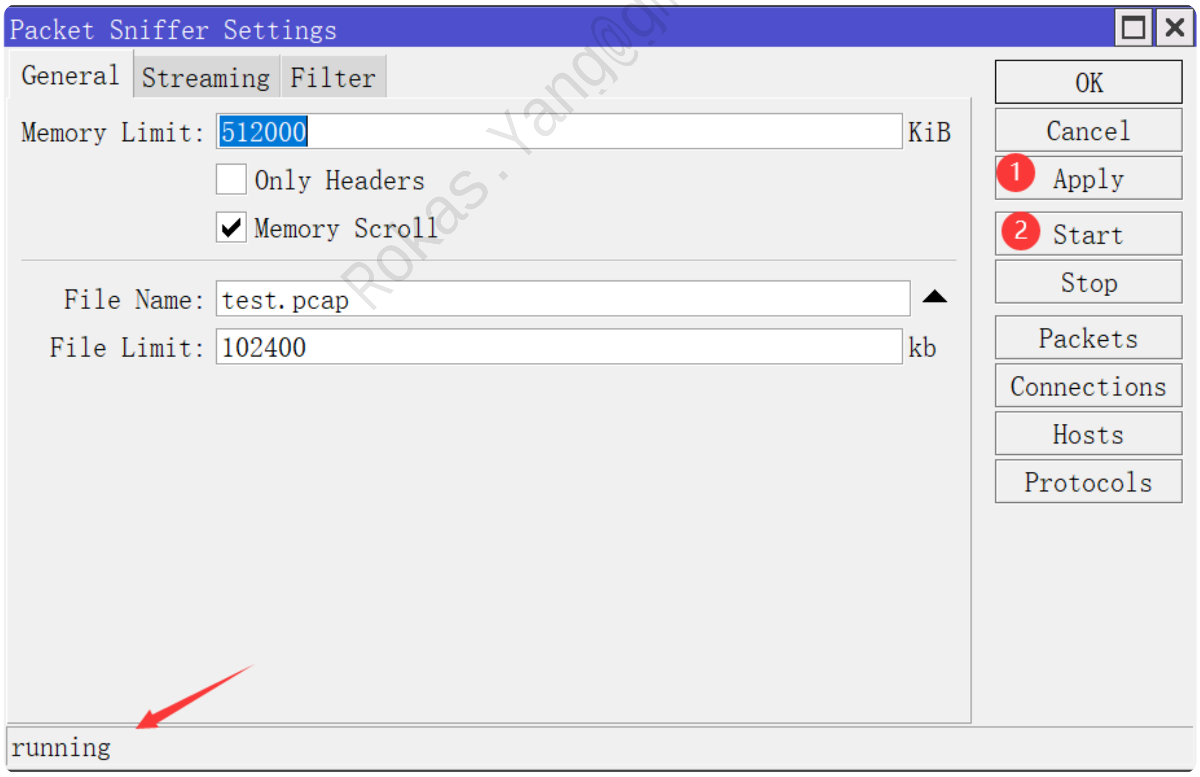
General | Streaming | Filter

Interfaces: pppoe-out1
MAC Address:
MAC Protocol:
IP Address:
IPv6 Address:
IP Protocol:
Port:
CPU:
Direction: any
Filter Operation: or

stopped

7）运行抓包

上述参数设置后，点击右侧的Apply，引用规则，然后点击Start，此时Sniffer已经是running状态：


Packet Sniffer Settings

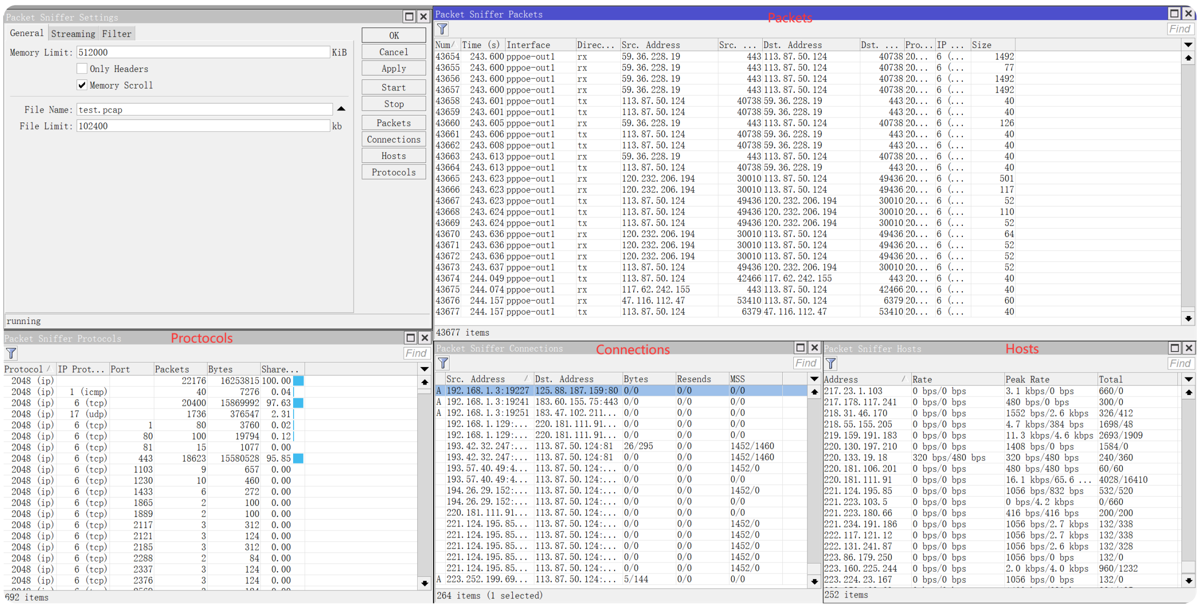
General | Streaming | Filter

Memory Limit: 512000 KiB
[ ] Only Headers
[✓] Memory Scroll

File Name: test.pcap
File Limit: 102400 kb

running

之后可以点击右侧的**Packets**、**Connections**、**Hosts**、**Protocols**，分别实时查看数据包、连接、主机、协议等具体分类：


Packet Sniffer Settings

General | Streaming | Filter

Memory Limit: 512000 KiB

[ ] Only Headers

[✓] Memory Scroll

File Name: test.pcap

File Limit: 102400 kb

OK | Cancel | Apply | Start | Stop | Packets | Connections | Hosts | Protocols

running

Packet Sniffer Packets — Packets

Num	Time (s)	Interface	Direc...	Src. Address	Src. ...	Dst. Address	Dst. ...	Pro...	IP ...	Size
43654	243.600	pppoe-out1	rx	59.36.228.19	443	113.87.50.124	40738	20...	6 (...	1492
43655	243.600	pppoe-out1	rx	59.36.228.19	443	113.87.50.124	40738	20...	6 (...	77
43656	243.600	pppoe-out1	rx	59.36.228.19	443	113.87.50.124	40738	20...	6 (...	1492
43657	243.600	pppoe-out1	rx	59.36.228.19	443	113.87.50.124	40738	20...	6 (...	1492
43658	243.601	pppoe-out1	tx	113.87.50.124	40738	59.36.228.19	443	20...	6 (...	40
43659	243.601	pppoe-out1	tx	113.87.50.124	40738	59.36.228.19	443	20...	6 (...	40
43660	243.605	pppoe-out1	rx	59.36.228.19	443	113.87.50.124	40738	20...	6 (...	126
43661	243.606	pppoe-out1	tx	113.87.50.124	40738	59.36.228.19	443	20...	6 (...	40
43662	243.608	pppoe-out1	tx	113.87.50.124	40738	59.36.228.19	443	20...	6 (...	40
43663	243.613	pppoe-out1	rx	59.36.228.19	443	113.87.50.124	40738	20...	6 (...	40
43664	243.613	pppoe-out1	tx	113.87.50.124	40738	59.36.228.19	443	20...	6 (...	40
43665	243.623	pppoe-out1	rx	120.232.206.194	30010	113.87.50.124	49436	20...	6 (...	501
43666	243.623	pppoe-out1	rx	120.232.206.194	30010	113.87.50.124	49436	20...	6 (...	117
43667	243.623	pppoe-out1	tx	113.87.50.124	49436	120.232.206.194	30010	20...	6 (...	52
43668	243.624	pppoe-out1	tx	113.87.50.124	49436	120.232.206.194	30010	20...	6 (...	110
43669	243.624	pppoe-out1	tx	113.87.50.124	49436	120.232.206.194	30010	20...	6 (...	52
43670	243.636	pppoe-out1	rx	120.232.206.194	30010	113.87.50.124	49436	20...	6 (...	64
43671	243.636	pppoe-out1	rx	120.232.206.194	30010	113.87.50.124	49436	20...	6 (...	52
43672	243.636	pppoe-out1	rx	120.232.206.194	30010	113.87.50.124	49436	20...	6 (...	52
43673	243.637	pppoe-out1	tx	113.87.50.124	49436	120.232.206.194	30010	20...	6 (...	52
43674	244.049	pppoe-out1	tx	113.87.50.124	42466	117.62.242.155	443	20...	6 (...	40
43675	244.074	pppoe-out1	rx	117.62.242.155	443	113.87.50.124	42466	20...	6 (...	40
43676	244.157	pppoe-out1	rx	47.116.112.47	53410	113.87.50.124	6379	20...	6 (...	60
43677	244.157	pppoe-out1	tx	113.87.50.124	6379	47.116.112.47	53410	20...	6 (...	40

43677 items

Packet Sniffer Protocols — Proctocols

Protocol	IP Prot...	Port	Packets	Bytes	Share...
2048 (ip)			22176	16253815	100.00
2048 (ip)	1 (icmp)		40	7276	0.04
2048 (ip)	6 (tcp)		20400	15869992	97.63
2048 (ip)	17 (udp)		1736	376547	2.31
2048 (ip)	6 (tcp)	1	80	3760	0.02
2048 (ip)	6 (tcp)	80	100	19794	0.12
2048 (ip)	6 (tcp)	81	15	1077	0.00
2048 (ip)	6 (tcp)	443	18623	15580528	95.85
2048 (ip)	6 (tcp)	1103	9	657	0.00
2048 (ip)	6 (tcp)	1230	10	460	0.00
2048 (ip)	6 (tcp)	1433	6	272	0.00
2048 (ip)	6 (tcp)	1865	2	100	0.00
2048 (ip)	6 (tcp)	1889	2	100	0.00
2048 (ip)	6 (tcp)	2117	3	312	0.00
2048 (ip)	6 (tcp)	2121	3	124	0.00
2048 (ip)	6 (tcp)	2185	3	312	0.00
2048 (ip)	6 (tcp)	2288	2	84	0.00
2048 (ip)	6 (tcp)	2337	3	124	0.00
2048 (ip)	6 (tcp)	2376	3	124	0.00

692 items

Packet Sniffer Connections — Connections

	Src. Address	Dst. Address	Bytes	Resends	MSS
A	192.168.1.3:19227	125.88.187.159:80	0/0	0/0	0/0
A	192.168.1.3:19241	183.60.155.75:443	0/0	0/0	0/0
A	192.168.1.3:19251	183.47.102.211...	0/0	0/0	0/0
	192.168.1.129:...	220.181.111.91...	0/0	0/0	0/0
	192.168.1.129:...	220.181.111.91...	0/0	0/0	0/0
	193.42.32.247:...	113.87.50.124:81	26/295	0/0	1452/1460
	193.42.32.247:...	113.87.50.124:81	0/0	0/0	1452/1460
	193.57.40.49:4...	113.87.50.124:...	0/0	0/0	1452/0
	193.57.40.49:4...	113.87.50.124:...	0/0	0/0	0/0
	194.26.29.152:...	113.87.50.124:...	0/0	0/0	1452/0
	194.26.29.152:...	113.87.50.124:...	0/0	0/0	0/0
	220.181.111.91...	113.87.50.124:...	0/0	0/0	0/0
	221.124.195.85...	113.87.50.124:...	0/0	0/0	1452/0
	221.124.195.85...	113.87.50.124:...	0/0	0/0	1452/0
	221.124.195.85...	113.87.50.124:...	0/0	0/0	1452/0
	221.124.195.85...	113.87.50.124:...	0/0	0/0	1452/0
	221.124.195.85...	113.87.50.124:...	0/0	0/0	1452/0
A	223.252.199.69...	113.87.50.124:...	5/144	0/0	0/0

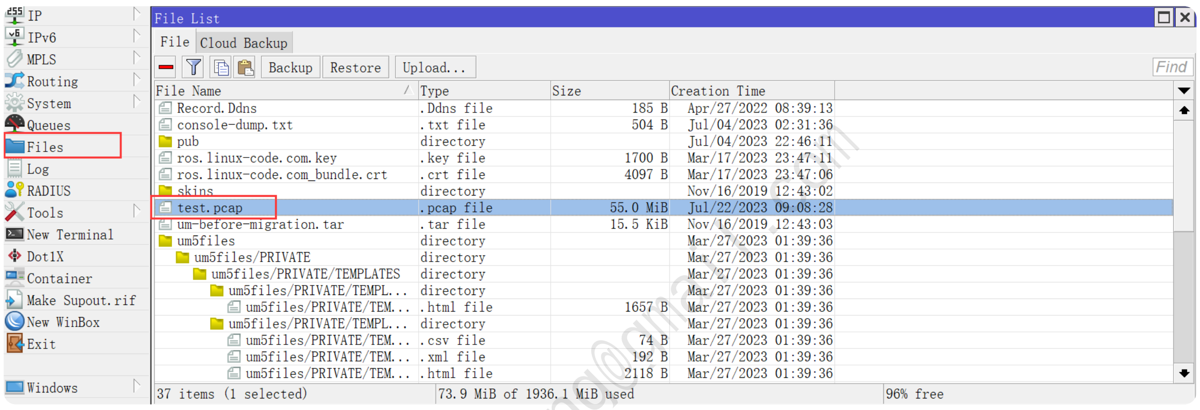
264 items (1 selected)

Packet Sniffer Hosts — Hosts

Address	Rate	Peak Rate	Total
217.23.1.103	0 bps/0 bps	3.1 kbps/0 bps	660/0
217.178.117.241	0 bps/0 bps	480 bps/0 bps	300/0
218.31.46.170	0 bps/0 bps	1552 bps/2.6 kbps	326/412
218.55.155.205	0 bps/0 bps	4.7 kbps/384 bps	1698/48
219.159.191.183	0 bps/0 bps	11.3 kbps/4.6 kbps	2693/1909
220.130.197.210	0 bps/0 bps	1408 bps/0 bps	1584/0
220.133.19.18	320 bps/480 bps	320 bps/480 bps	240/360
220.181.106.201	0 bps/0 bps	480 bps/480 bps	60/60
220.181.111.91	0 bps/0 bps	16.1 kbps/65.6 ...	4028/16410
221.124.195.85	0 bps/0 bps	1056 bps/832 bps	532/520
221.223.103.5	0 bps/0 bps	0 bps/4.2 kbps	0/660
221.223.180.66	0 bps/0 bps	416 bps/416 bps	200/200
221.234.191.186	0 bps/0 bps	1056 bps/2.7 kbps	132/338
222.117.121.12	0 bps/0 bps	1056 bps/2.7 kbps	132/338
222.131.241.87	0 bps/0 bps	1056 bps/2.6 kbps	132/328
223.86.179.250	0 bps/0 bps	1056 bps/0 bps	132/0
223.160.225.244	0 bps/0 bps	2.0 kbps/4.0 kbps	960/1232
223.224.23.167	0 bps/0 bps	1056 bps/0 bps	132/0

252 items

抓完后点击STOP停止抓包，生成了一个55MiB大小的test.pcap文件：



File List

File | Cloud Backup

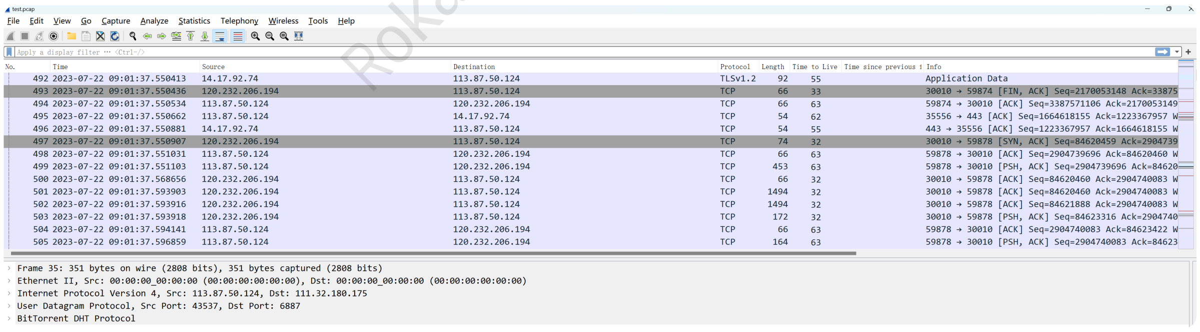
Backup | Restore | Upload...

File Name	Type	Size	Creation Time
Record.Ddns	.Ddns file	185 B	Apr/27/2022 08:39:13
console-dump.txt	.txt file	504 B	Jul/04/2023 02:31:36
pub	directory		Jul/04/2023 22:46:11
ros.linux-code.com.key	.key file	1700 B	Mar/17/2023 23:47:11
ros.linux-code.com_bundle.crt	.crt file	4097 B	Mar/17/2023 23:47:06
skins	directory		Nov/16/2019 12:43:02
test.pcap	.pcap file	55.0 MiB	Jul/22/2023 09:08:28
um-before-migration.tar	.tar file	15.5 KiB	Nov/16/2019 12:43:03
um5files	directory		Mar/27/2023 01:39:36
um5files/PRIVATE	directory		Mar/27/2023 01:39:36
um5files/PRIVATE/TEMPLATES	directory		Mar/27/2023 01:39:36
um5files/PRIVATE/TEMPL...	directory		Mar/27/2023 01:39:36
um5files/PRIVATE/TEM...	.html file	1657 B	Mar/27/2023 01:39:36
um5files/PRIVATE/TEMPL...	directory		Mar/27/2023 01:39:36
um5files/PRIVATE/TEM...	.csv file	74 B	Mar/27/2023 01:39:36
um5files/PRIVATE/TEM...	.xml file	192 B	Mar/27/2023 01:39:36
um5files/PRIVATE/TEM...	.html file	2118 B	Mar/27/2023 01:39:36

37 items (1 selected) | 73.9 MiB of 1936.1 MiB used | 96% free

右击可以把它download下来，之后使用其他工具分析。

抓到的包是最原始的数据，没有任何其他特殊封装：



test.pcap
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter … <Ctrl-/>

No.	Time	Source	Destination	Protocol	Length	Time to Live	Time since previous	Info
492	2023-07-22 09:01:37.550413	14.17.92.74	113.87.50.124	TLSv1.2	92	55		Application Data
493	2023-07-22 09:01:37.550436	120.232.206.194	113.87.50.124	TCP	66	33		30010 → 59874 [FIN, ACK] Seq=2170053148 Ack=33875
494	2023-07-22 09:01:37.550534	113.87.50.124	120.232.206.194	TCP	66	63		59874 → 30010 [ACK] Seq=3387571106 Ack=2170053149
495	2023-07-22 09:01:37.550662	113.87.50.124	14.17.92.74	TCP	54	62		35556 → 443 [ACK] Seq=1664618155 Ack=1223367957 W
496	2023-07-22 09:01:37.550881	14.17.92.74	113.87.50.124	TCP	54	55		443 → 35556 [ACK] Seq=1223367957 Ack=1664618155 W
497	2023-07-22 09:01:37.550907	120.232.206.194	113.87.50.124	TCP	74	32		30010 → 59878 [SYN, ACK] Seq=84620459 Ack=2904739
498	2023-07-22 09:01:37.551031	113.87.50.124	120.232.206.194	TCP	66	63		59878 → 30010 [ACK] Seq=2904739696 Ack=84620460 W
499	2023-07-22 09:01:37.551103	113.87.50.124	120.232.206.194	TCP	453	63		59878 → 30010 [PSH, ACK] Seq=2904739696 Ack=84620
500	2023-07-22 09:01:37.568656	120.232.206.194	113.87.50.124	TCP	66	32		30010 → 59878 [ACK] Seq=84620460 Ack=2904740083 W
501	2023-07-22 09:01:37.593903	120.232.206.194	113.87.50.124	TCP	1494	32		30010 → 59878 [ACK] Seq=84620460 Ack=2904740083 W
502	2023-07-22 09:01:37.593916	120.232.206.194	113.87.50.124	TCP	1494	32		30010 → 59878 [ACK] Seq=84621888 Ack=2904740083 W
503	2023-07-22 09:01:37.593918	120.232.206.194	113.87.50.124	TCP	172	32		30010 → 59878 [PSH, ACK] Seq=84623316 Ack=2904740
504	2023-07-22 09:01:37.594141	113.87.50.124	120.232.206.194	TCP	66	63		59878 → 30010 [ACK] Seq=2904740083 Ack=84623422 W
505	2023-07-22 09:01:37.596859	113.87.50.124	120.232.206.194	TCP	164	63		59878 → 30010 [PSH, ACK] Seq=2904740083 Ack=84623

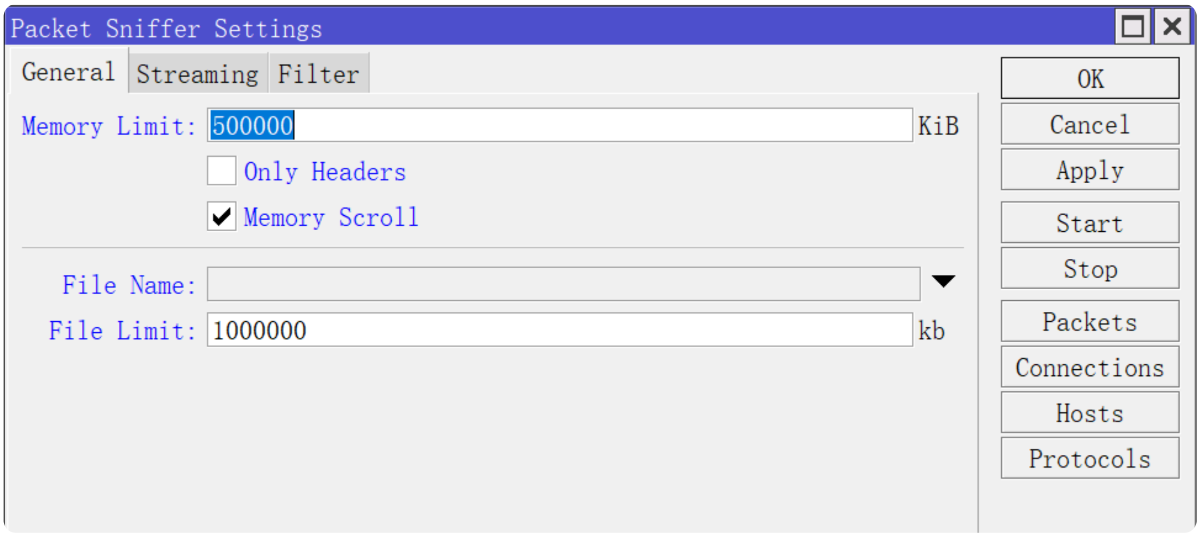
```
> Frame 35: 351 bytes on wire (2808 bits), 351 bytes captured (2808 bits)
> Ethernet II, Src: 00:00:00_00:00:00 (00:00:00:00:00:00), Dst: 00:00:00_00:00:00 (00:00:00:00:00:00)
> Internet Protocol Version 4, Src: 113.87.50.124, Dst: 111.32.180.175
> User Datagram Protocol, Src Port: 43537, Dst Port: 6887
> BitTorrent DHT Protocol
```

2.Wireshark

当我们不想把抓包存放在ros上时，可以通过设置Streaming选项，让ros把收到的包转发给其他服务器。

1）General选项配置

同理，我们限制最大内存为500MiB，File Limit和File Name无需设置，我们不需要保存到文件：


Packet Sniffer Settings

General | Streaming | Filter

Memory Limit: 500000 KiB

☐ Only Headers

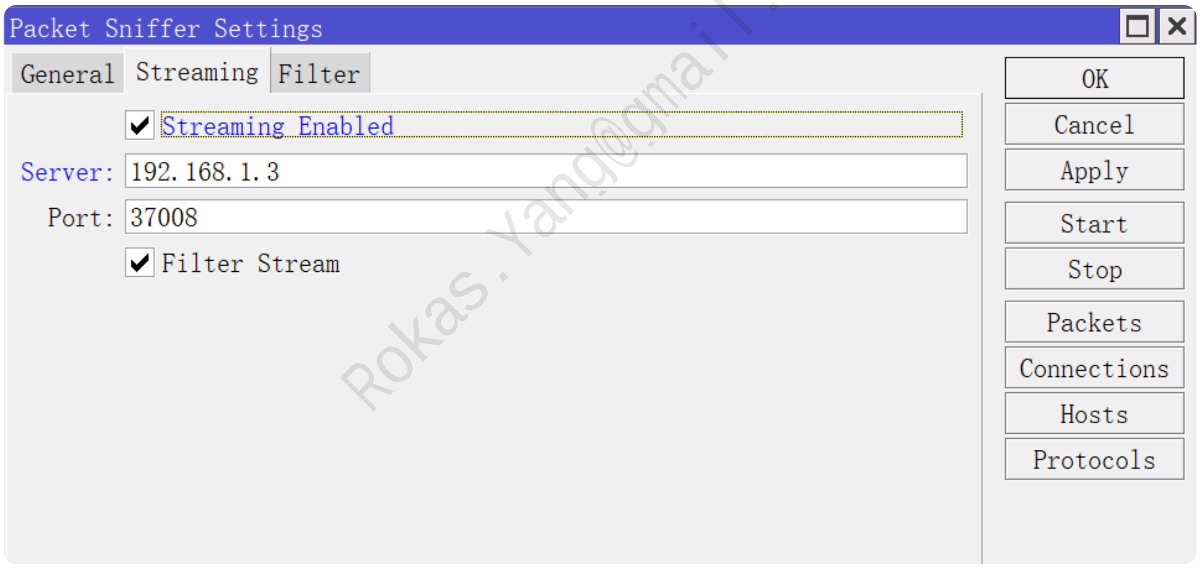
☑ Memory Scroll

File Name:

File Limit: 1000000 kb

2）Streaming选项配置

勾选Streaming Enabled，并且指定Server为接受抓包的目的IP及端口，可以是任何IP，包括内外网：


Packet Sniffer Settings

General | Streaming | Filter

☑ Streaming Enabled

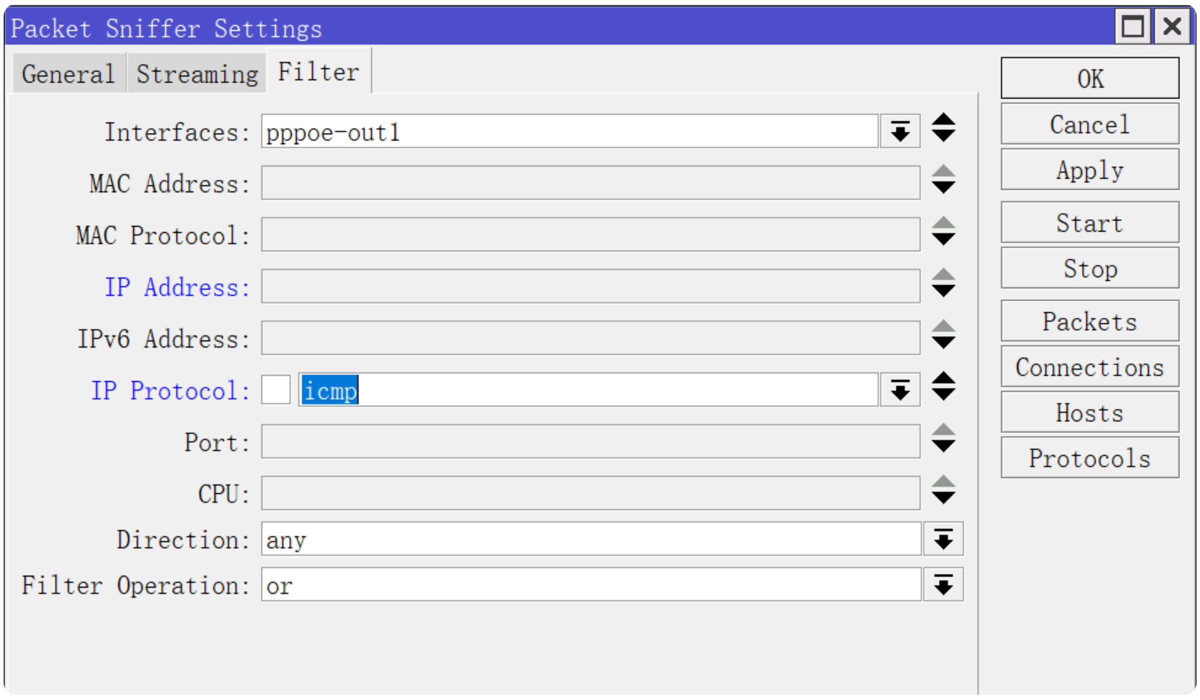
Server: 192.168.1.3

Port: 37008

☑ Filter Stream

3）Filter参数配置

抓取wan口的ICMP协议为示例：



Packet Sniffer Settings

General | Streaming | Filter

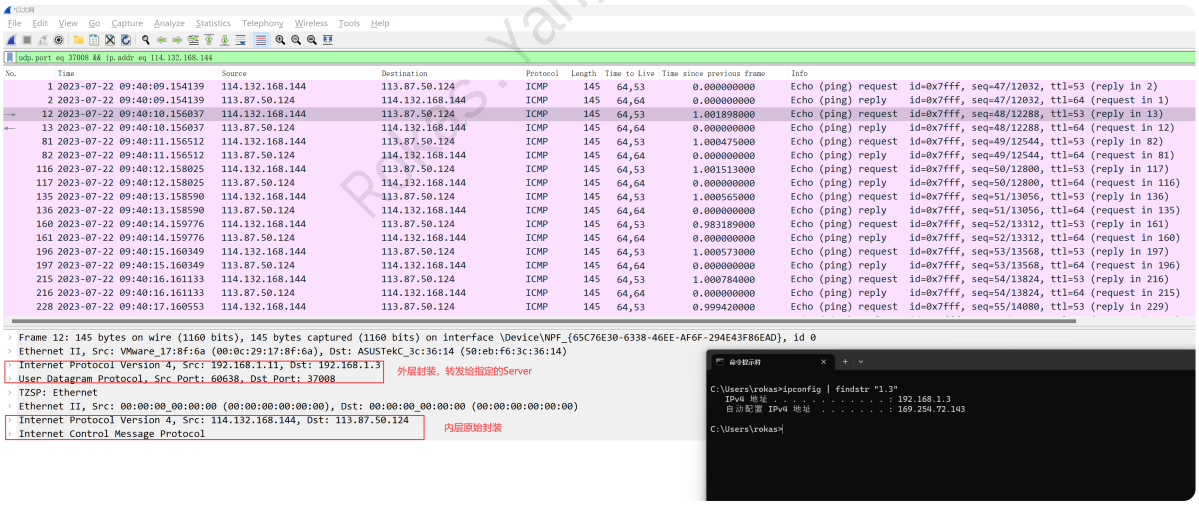
Interfaces:	pppoe-out1
MAC Address:	
MAC Protocol:	
IP Address:	
IPv6 Address:	
IP Protocol:	icmp
Port:	
CPU:	
Direction:	any
Filter Operation:	or

点击右侧的Apply应用设置，并且点击Start开始抓包。

4）在目的Server上捕获报文

登录到指定的Server，并且使用抓包工具，抓取37008端口的数据。

通过**udp.port eq 37006**过滤到ros转发过来的报文，首先通过IPIP封装了外层协议，Server才能收到外层转发过来的包，内层则是ros最原始的收发包数据：



*以太网
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
命令提示符

udp.port eq 37008 && ip.addr eq 114.132.168.144

```
No.  Time                        Source           Destination      Protocol Length Time to Live Time since previous frame Info
  1 2023-07-22 09:40:09.154139  114.132.168.144  113.87.50.124    ICMP     145    64,53        0.000000000  Echo (ping) request  id=0x7fff, seq=47/12032, ttl=53 (reply in 2)
  2 2023-07-22 09:40:09.154139  113.87.50.124    114.132.168.144  ICMP     145    64,64        0.000000000  Echo (ping) reply    id=0x7fff, seq=47/12032, ttl=64 (request in 1)
 12 2023-07-22 09:40:10.156037  114.132.168.144  113.87.50.124    ICMP     145    64,53        1.001898000  Echo (ping) request  id=0x7fff, seq=48/12288, ttl=53 (reply in 13)
 13 2023-07-22 09:40:10.156037  113.87.50.124    114.132.168.144  ICMP     145    64,64        0.000000000  Echo (ping) reply    id=0x7fff, seq=48/12288, ttl=64 (request in 12)
 81 2023-07-22 09:40:11.156512  114.132.168.144  113.87.50.124    ICMP     145    64,53        1.000475000  Echo (ping) request  id=0x7fff, seq=49/12544, ttl=53 (reply in 82)
 82 2023-07-22 09:40:11.156512  113.87.50.124    114.132.168.144  ICMP     145    64,64        0.000000000  Echo (ping) reply    id=0x7fff, seq=49/12544, ttl=64 (request in 81)
116 2023-07-22 09:40:12.158025  114.132.168.144  113.87.50.124    ICMP     145    64,53        1.001513000  Echo (ping) request  id=0x7fff, seq=50/12800, ttl=53 (reply in 117)
117 2023-07-22 09:40:12.158025  113.87.50.124    114.132.168.144  ICMP     145    64,64        0.000000000  Echo (ping) reply    id=0x7fff, seq=50/12800, ttl=64 (request in 116)
135 2023-07-22 09:40:13.158590  114.132.168.144  113.87.50.124    ICMP     145    64,53        1.000565000  Echo (ping) request  id=0x7fff, seq=51/13056, ttl=53 (reply in 136)
136 2023-07-22 09:40:13.158590  113.87.50.124    114.132.168.144  ICMP     145    64,64        0.000000000  Echo (ping) reply    id=0x7fff, seq=51/13056, ttl=64 (request in 135)
160 2023-07-22 09:40:14.159776  114.132.168.144  113.87.50.124    ICMP     145    64,53        0.983189000  Echo (ping) request  id=0x7fff, seq=52/13312, ttl=53 (reply in 161)
161 2023-07-22 09:40:14.159776  113.87.50.124    114.132.168.144  ICMP     145    64,64        0.000000000  Echo (ping) reply    id=0x7fff, seq=52/13312, ttl=64 (request in 160)
196 2023-07-22 09:40:15.160349  114.132.168.144  113.87.50.124    ICMP     145    64,53        1.000573000  Echo (ping) request  id=0x7fff, seq=53/13568, ttl=53 (reply in 197)
197 2023-07-22 09:40:15.160349  113.87.50.124    114.132.168.144  ICMP     145    64,64        0.000000000  Echo (ping) reply    id=0x7fff, seq=53/13568, ttl=64 (request in 196)
215 2023-07-22 09:40:16.161133  114.132.168.144  113.87.50.124    ICMP     145    64,53        1.000784000  Echo (ping) request  id=0x7fff, seq=54/13824, ttl=53 (reply in 216)
216 2023-07-22 09:40:16.161133  113.87.50.124    114.132.168.144  ICMP     145    64,64        0.000000000  Echo (ping) reply    id=0x7fff, seq=54/13824, ttl=64 (request in 215)
228 2023-07-22 09:40:17.160553  114.132.168.144  113.87.50.124    ICMP     145    64,53        0.999420000  Echo (ping) request  id=0x7fff, seq=55/14080, ttl=53 (reply in 229)
```

```
> Frame 12: 145 bytes on wire (1160 bits), 145 bytes captured (1160 bits) on interface \Device\NPF_{65C76E30-6338-46EE-AF6F-294E43F86EAD}, id 0
> Ethernet II, Src: VMware_17:8f:6a (00:0c:29:17:8f:6a), Dst: ASUSTekC_3c:36:14 (50:eb:f6:3c:36:14)
> Internet Protocol Version 4, Src: 192.168.1.11, Dst: 192.168.1.3
> User Datagram Protocol, Src Port: 60638, Dst Port: 37008
> TZSP: Ethernet
> Ethernet II, Src: 00:00:00_00:00:00 (00:00:00:00:00:00), Dst: 00:00:00_00:00:00 (00:00:00:00:00:00)
> Internet Protocol Version 4, Src: 114.132.168.144, Dst: 113.87.50.124
> Internet Control Message Protocol
```

外层封装，转发给指定的Server

内层原始封装

```
C:\Users\rokas>ipconfig | findstr "1.3"
   IPv4 地址 . . . . . . . . . . . . : 192.168.1.3
   自动配置 IPv4 地址  . . . . . . . : 169.254.72.143

C:\Users\rokas>
```

三、命令行抓包

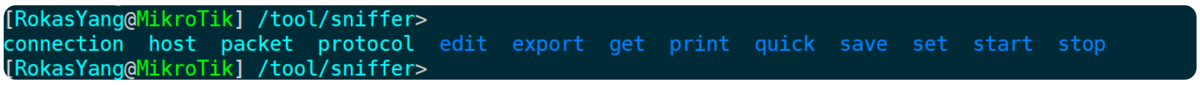
命令行抓包就是把上述图形化界面的参数设置一遍，没有其它差异。

1.Sniffer

首先ssh到routeros，执行以下命令进入到sniffer：

```
[RokasYang@MikroTik] > /tool/sniffer/
[RokasYang@MikroTik] /tool/sniffer>
```

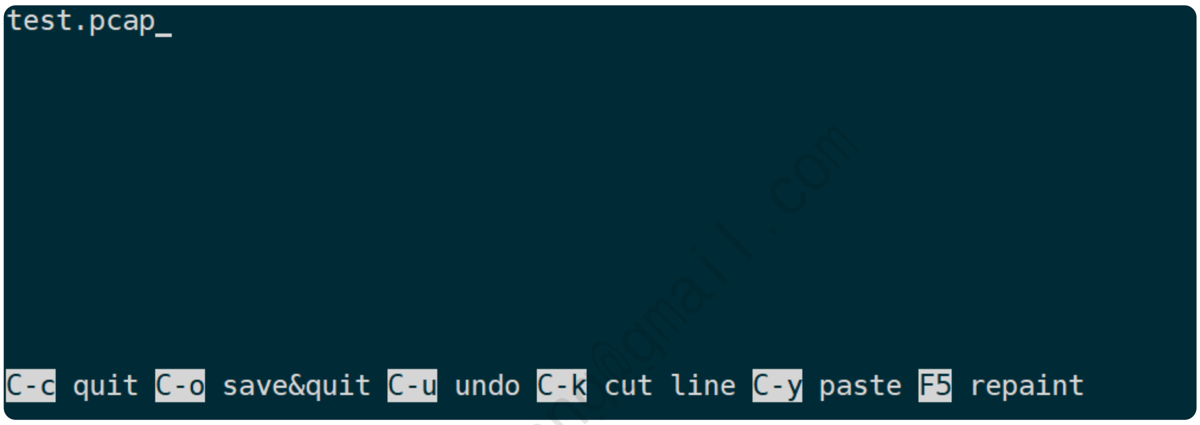
按tab可以看到sniffper下面有如下命令：


```
[RokasYang@MikroTik] /tool/sniffer>
connection  host  packet  protocol  edit  export  get  print  quick  save  set  start  stop
[RokasYang@MikroTik] /tool/sniffer>
```

1）edit file-name/edit file-limit

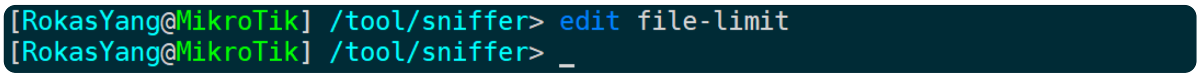
如果想把包保存到ros，执行edit edit file-name命令，之后进入编辑器模式，写入文件命令后，ctrl + o 保存退出。

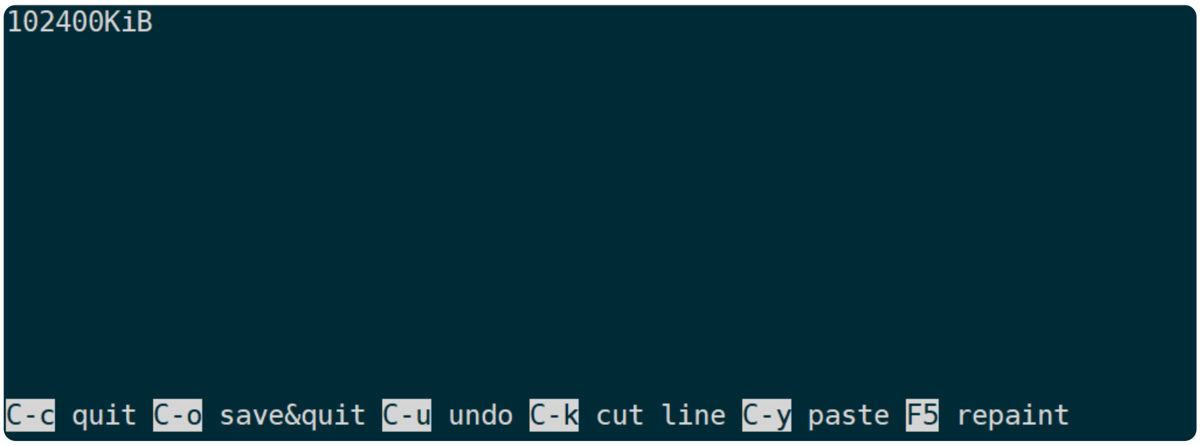
```
[RokasYang@MikroTik] /tool/sniffer> edit file-name
```


```
test.pcap_









C-c quit C-o save&quit C-u undo C-k cut line C-y paste F5 repaint
```

之后编辑抓包文件大小限制：

```
[RokasYang@MikroTik] /tool/sniffer> edit file-limit
[RokasYang@MikroTik] /tool/sniffer>
```

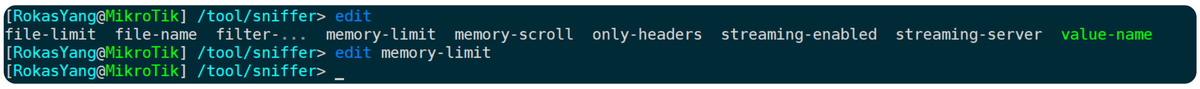

```
[RokasYang@MikroTik] /tool/sniffer> edit file-limit
[RokasYang@MikroTik] /tool/sniffer> _
```


```
102400KiB









C-c quit C-o save&quit C-u undo C-k cut line C-y paste F5 repaint
```

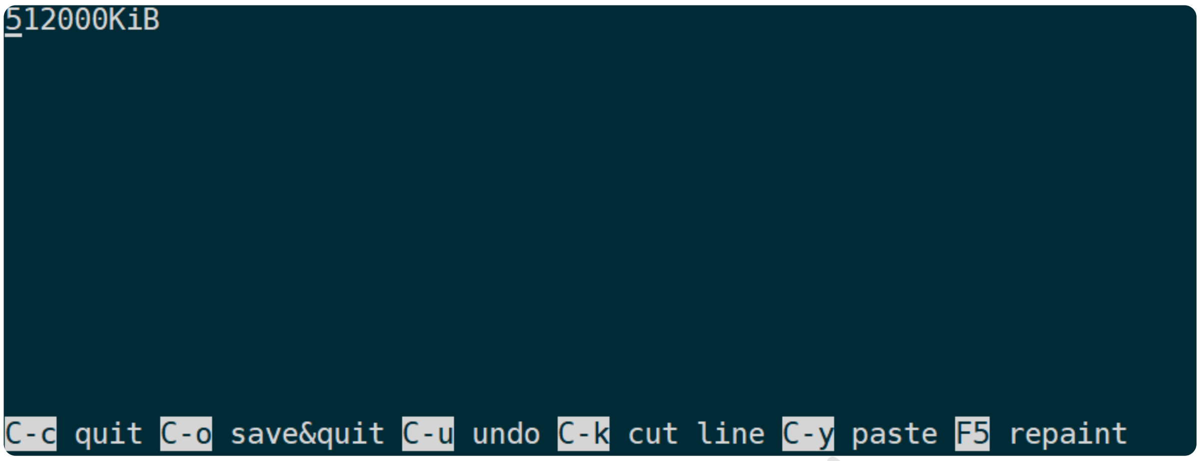
2）edit memory-limit

同理，编辑内存限制，512MiB。

```
[RokasYang@MikroTik] /tool/sniffer> edit memory-limit
```



```
[RokasYang@MikroTik] /tool/sniffer> edit
file-limit  file-name  filter-...  memory-limit  memory-scroll  only-headers  streaming-enabled  streaming-server  value-name
[RokasYang@MikroTik] /tool/sniffer> edit memory-limit
[RokasYang@MikroTik] /tool/sniffer> _
```

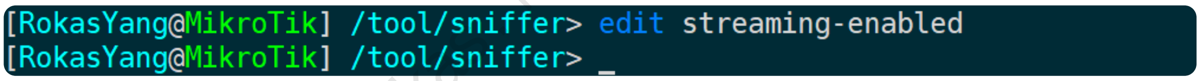


```
512000KiB






C-c quit C-o save&quit C-u undo C-k cut line C-y paste F5 repaint
```

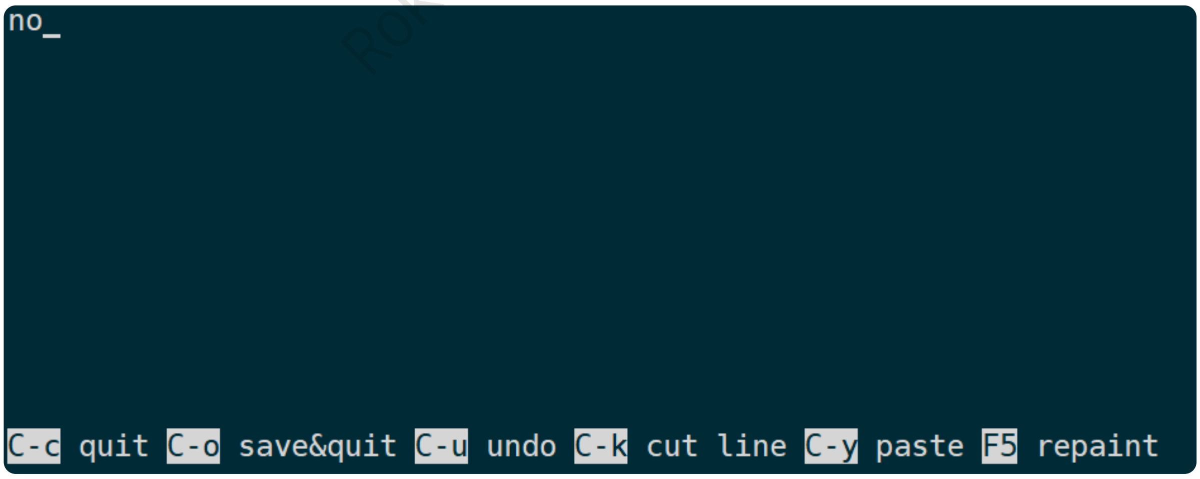
3）edit streaming-enabled

抓包保存到本地文件，并且并不需要将报文转发给其它server，确保此选项是no状态：

```
[RokasYang@MikroTik] /tool/sniffer> edit streaming-enabled
```



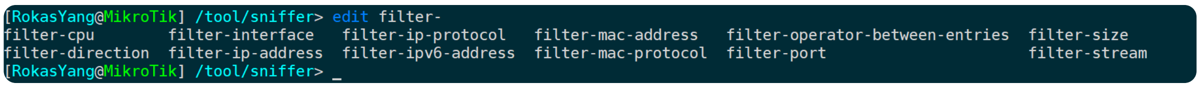
```
[RokasYang@MikroTik] /tool/sniffer> edit streaming-enabled
[RokasYang@MikroTik] /tool/sniffer> _
```



```
no_






C-c quit C-o save&quit C-u undo C-k cut line C-y paste F5 repaint
```

4）edit filter

编辑过滤规则，和图形界面的参数都能对上，需要设置哪个就edit哪个。



```
[RokasYang@MikroTik] /tool/sniffer> edit filter-
filter-cpu        filter-interface  filter-ip-protocol    filter-mac-address    filter-operator-between-entries  filter-size
filter-direction  filter-ip-address  filter-ipv6-address  filter-mac-protocol   filter-port                      filter-stream
[RokasYang@MikroTik] /tool/sniffer> _
```

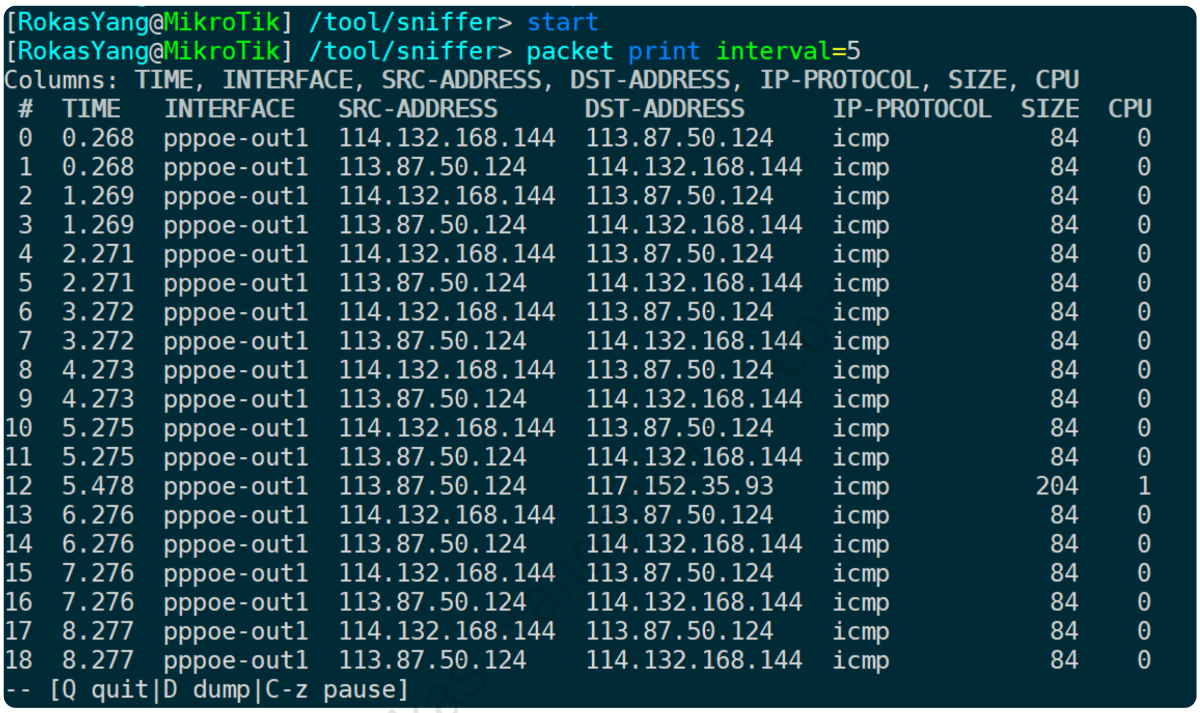
比如抓pppoe-out1口的ICMP报文，则可以设置：

```
[RokasYang@MikroTik] /tool/sniffer> edit filter-interface   # 写入pppoe-out1
[RokasYang@MikroTik] /tool/sniffer> edit filter-ip-protocol # 写入icmp
```

5） start/packet print

执行start命令便开始抓包，使用packet print命令可以输出抓到的请求：

```
[RokasYang@MikroTik] /tool/sniffer> start
[RokasYang@MikroTik] /tool/sniffer> packet print interval=5
```



```
[RokasYang@MikroTik] /tool/sniffer> start
[RokasYang@MikroTik] /tool/sniffer> packet print interval=5
Columns: TIME, INTERFACE, SRC-ADDRESS, DST-ADDRESS, IP-PROTOCOL, SIZE, CPU
 #  TIME   INTERFACE   SRC-ADDRESS      DST-ADDRESS      IP-PROTOCOL  SIZE  CPU
 0  0.268  pppoe-out1  114.132.168.144  113.87.50.124    icmp           84    0
 1  0.268  pppoe-out1  113.87.50.124    114.132.168.144  icmp           84    0
 2  1.269  pppoe-out1  114.132.168.144  113.87.50.124    icmp           84    0
 3  1.269  pppoe-out1  113.87.50.124    114.132.168.144  icmp           84    0
 4  2.271  pppoe-out1  114.132.168.144  113.87.50.124    icmp           84    0
 5  2.271  pppoe-out1  113.87.50.124    114.132.168.144  icmp           84    0
 6  3.272  pppoe-out1  114.132.168.144  113.87.50.124    icmp           84    0
 7  3.272  pppoe-out1  113.87.50.124    114.132.168.144  icmp           84    0
 8  4.273  pppoe-out1  114.132.168.144  113.87.50.124    icmp           84    0
 9  4.273  pppoe-out1  113.87.50.124    114.132.168.144  icmp           84    0
10  5.275  pppoe-out1  114.132.168.144  113.87.50.124    icmp           84    0
11  5.275  pppoe-out1  113.87.50.124    114.132.168.144  icmp           84    0
12  5.478  pppoe-out1  113.87.50.124    117.152.35.93    icmp          204    1
13  6.276  pppoe-out1  114.132.168.144  113.87.50.124    icmp           84    0
14  6.276  pppoe-out1  113.87.50.124    114.132.168.144  icmp           84    0
15  7.276  pppoe-out1  114.132.168.144  113.87.50.124    icmp           84    0
16  7.276  pppoe-out1  113.87.50.124    114.132.168.144  icmp           84    0
17  8.277  pppoe-out1  114.132.168.144  113.87.50.124    icmp           84    0
18  8.277  pppoe-out1  113.87.50.124    114.132.168.144  icmp           84    0
-- [Q quit|D dump|C-z pause]
```

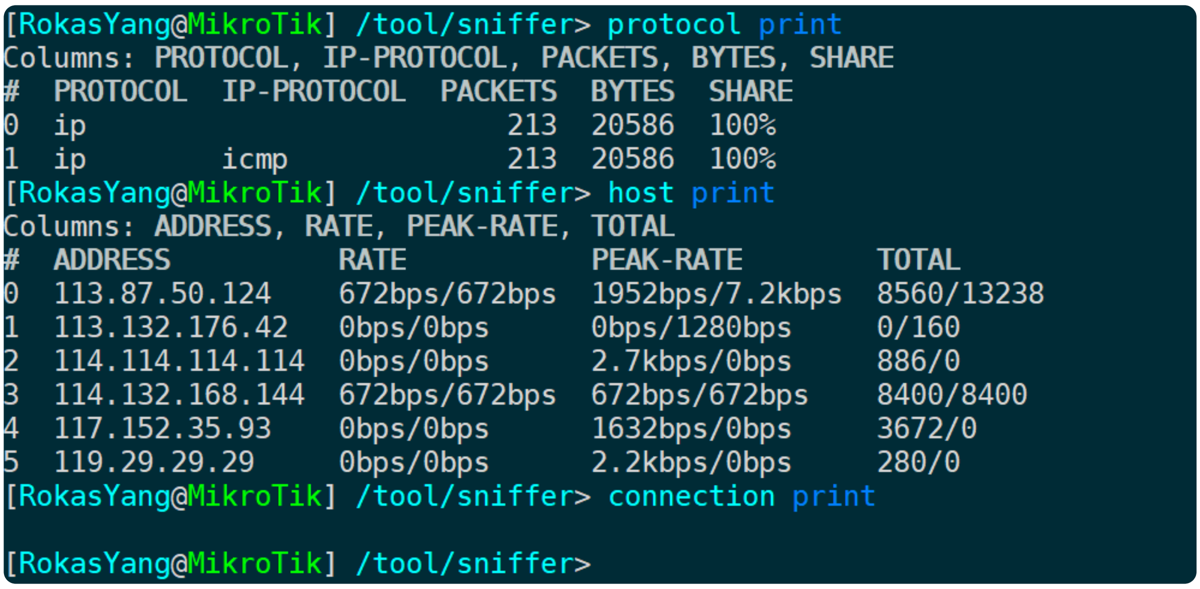
其中interval表示刷新间隔，这里设置为每5s刷新一次。

6） connection/protocol/host print

抓包过程中，连接信息，协议信息，主机信息都可以试试展示出来，并设置刷新间隔，因为指定抓取的ICMP协议，没有连接的概念，所以connection print是过滤不到内容的。

```
[RokasYang@MikroTik] /tool/sniffer> protocol print
Columns: PROTOCOL, IP-PROTOCOL, PACKETS, BYTES, SHARE
#  PROTOCOL  IP-PROTOCOL  PACKETS  BYTES  SHARE
0  ip                         213  20586  100%
1  ip        icmp             213  20586  100%
[RokasYang@MikroTik] /tool/sniffer> host print
Columns: ADDRESS, RATE, PEAK-RATE, TOTAL
#  ADDRESS          RATE           PEAK-RATE        TOTAL
0  113.87.50.124    672bps/672bps  1952bps/7.2kbps  8560/13238
1  113.132.176.42   0bps/0bps      0bps/1280bps     0/160
2  114.114.114.114  0bps/0bps      2.7kbps/0bps     886/0
3  114.132.168.144  672bps/672bps  672bps/672bps    8400/8400
4  117.152.35.93    0bps/0bps      1632bps/0bps     3672/0
```

```
5  119.29.29.29     0bps/0bps      2.2kbps/0bps     280/0
[RokasYang@MikroTik] /tool/sniffer> connection print
```

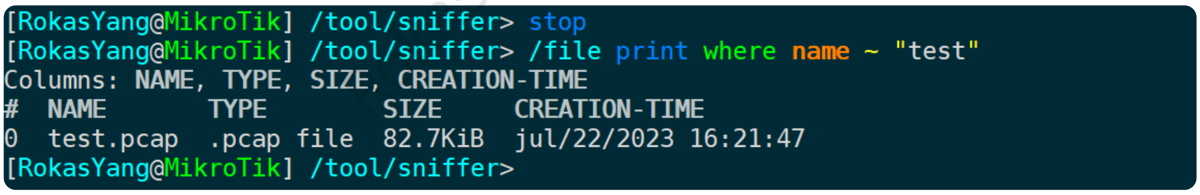


```
[RokasYang@MikroTik] /tool/sniffer> protocol print
Columns: PROTOCOL, IP-PROTOCOL, PACKETS, BYTES, SHARE
#  PROTOCOL  IP-PROTOCOL  PACKETS  BYTES  SHARE
0  ip                         213  20586  100%
1  ip        icmp             213  20586  100%
[RokasYang@MikroTik] /tool/sniffer> host print
Columns: ADDRESS, RATE, PEAK-RATE, TOTAL
#  ADDRESS          RATE          PEAK-RATE         TOTAL
0  113.87.50.124    672bps/672bps  1952bps/7.2kbps  8560/13238
1  113.132.176.42   0bps/0bps      0bps/1280bps     0/160
2  114.114.114.114  0bps/0bps      2.7kbps/0bps     886/0
3  114.132.168.144  672bps/672bps  672bps/672bps    8400/8400
4  117.152.35.93    0bps/0bps      1632bps/0bps     3672/0
5  119.29.29.29     0bps/0bps      2.2kbps/0bps     280/0
[RokasYang@MikroTik] /tool/sniffer> connection print

[RokasYang@MikroTik] /tool/sniffer>
```

7） stop

停止抓包，执行stop即可，前面定义的test.pcap会保存在file路径下：

```
[RokasYang@MikroTik] /tool/sniffer> /file print where name ~ "test"
Columns: NAME, TYPE, SIZE, CREATION-TIME
#  NAME       TYPE       SIZE     CREATION-TIME
0  test.pcap  .pcap file  82.7KiB  jul/22/2023 16:21:47
[RokasYang@MikroTik] /tool/sniffer>
```



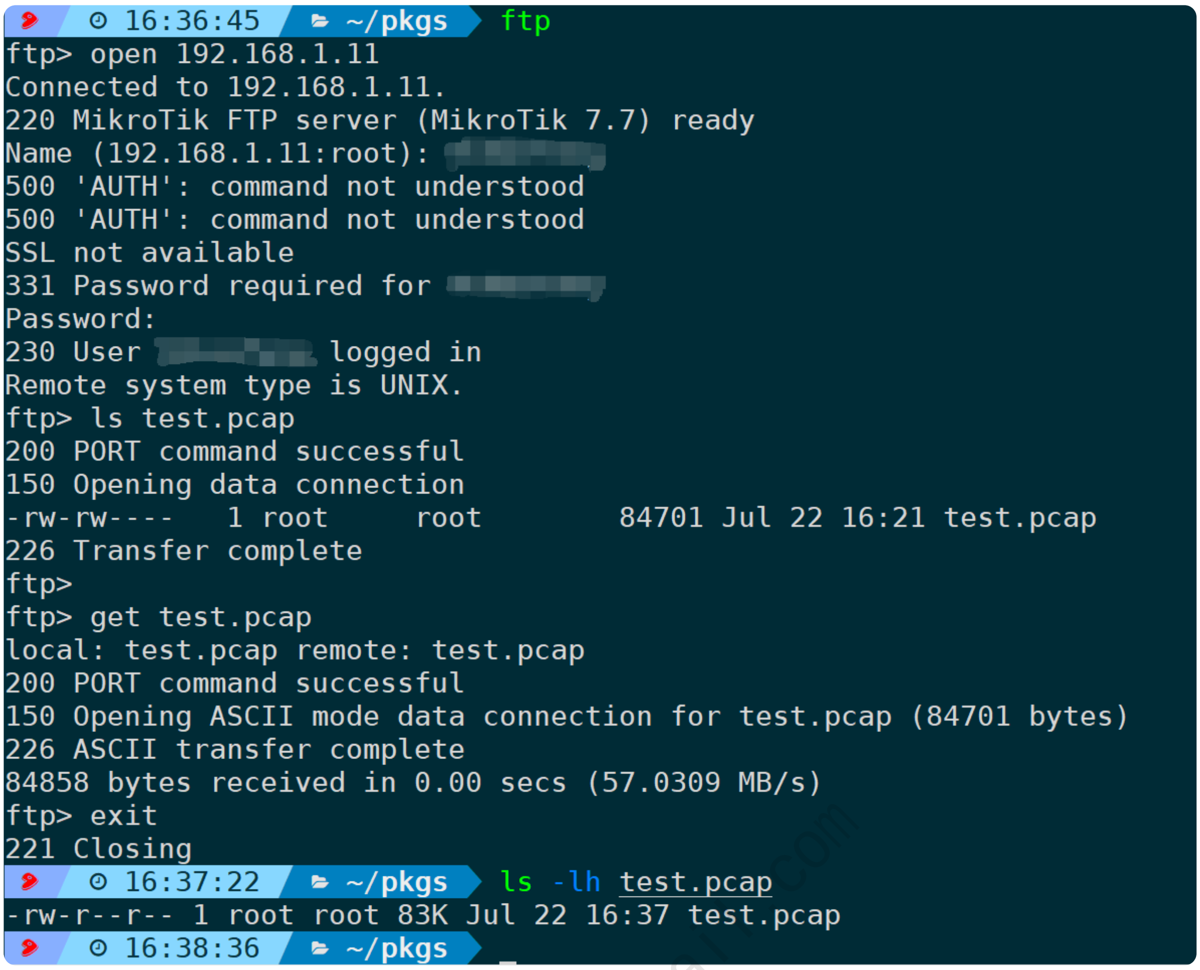
```
[RokasYang@MikroTik] /tool/sniffer> stop
[RokasYang@MikroTik] /tool/sniffer> /file print where name ~ "test"
Columns: NAME, TYPE, SIZE, CREATION-TIME
#  NAME       TYPE        SIZE     CREATION-TIME
0  test.pcap  .pcap file  82.7KiB  jul/22/2023 16:21:47
[RokasYang@MikroTik] /tool/sniffer>
```

8）开启ftp服务传送pcap文件

当没有图形化界面，不方便从winbox取出文件时，临时使用ftp服务来传递文件：

```
[RokasYang@MikroTik] /tool/sniffer> /ip service enable ftp
```

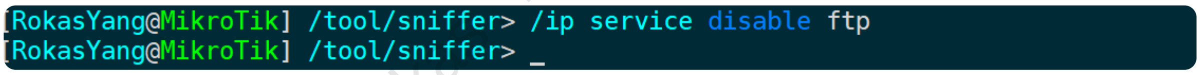
ftp服务开启后，使用ftp客户端将文件get到本地：



```
 16:36:45  ~/pkgs  ftp
ftp> open 192.168.1.11
Connected to 192.168.1.11.
220 MikroTik FTP server (MikroTik 7.7) ready
Name (192.168.1.11:root): [redacted]
500 'AUTH': command not understood
500 'AUTH': command not understood
SSL not available
331 Password required for [redacted]
Password:
230 User [redacted] logged in
Remote system type is UNIX.
ftp> ls test.pcap
200 PORT command successful
150 Opening data connection
-rw-rw----   1 root     root        84701 Jul 22 16:21 test.pcap
226 Transfer complete
ftp>
ftp> get test.pcap
local: test.pcap remote: test.pcap
200 PORT command successful
150 Opening ASCII mode data connection for test.pcap (84701 bytes)
226 ASCII transfer complete
84858 bytes received in 0.00 secs (57.0309 MB/s)
ftp> exit
221 Closing
 16:37:22  ~/pkgs  ls -lh test.pcap
-rw-r--r-- 1 root root 83K Jul 22 16:37 test.pcap
 16:38:36  ~/pkgs  _
```

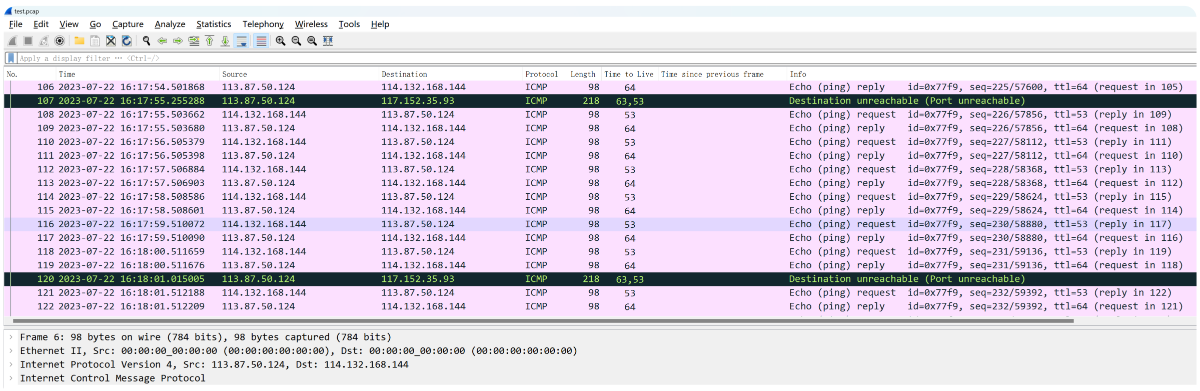
文件下载后，按需将ftp服务关闭即可：

```
[RokasYang@MikroTik] /tool/sniffer> /ip service disable ftp
```



```
[RokasYang@MikroTik] /tool/sniffer> /ip service disable ftp
[RokasYang@MikroTik] /tool/sniffer> _
```

之后便能使用wireshark分析抓包文件了：



test.pcap
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ··· <Ctrl-/>

```
No.   Time                        Source           Destination      Protocol Length Time to Live Time since previous frame Info
  106 2023-07-22 16:17:54.501868  113.87.50.124    114.132.168.144  ICMP       98    64                                     Echo (ping) reply    id=0x77f9, seq=225/57600, ttl=64 (request in 105)
  107 2023-07-22 16:17:55.255288  113.87.50.124    117.152.35.93    ICMP      218    63,53                                  Destination unreachable (Port unreachable)
  108 2023-07-22 16:17:55.503662  114.132.168.144  113.87.50.124    ICMP       98    53                                     Echo (ping) request  id=0x77f9, seq=226/57856, ttl=53 (reply in 109)
  109 2023-07-22 16:17:55.503680  113.87.50.124    114.132.168.144  ICMP       98    64                                     Echo (ping) reply    id=0x77f9, seq=226/57856, ttl=64 (request in 108)
  110 2023-07-22 16:17:56.505379  114.132.168.144  113.87.50.124    ICMP       98    53                                     Echo (ping) request  id=0x77f9, seq=227/58112, ttl=53 (reply in 111)
  111 2023-07-22 16:17:56.505398  113.87.50.124    114.132.168.144  ICMP       98    64                                     Echo (ping) reply    id=0x77f9, seq=227/58112, ttl=64 (request in 110)
  112 2023-07-22 16:17:57.506884  114.132.168.144  113.87.50.124    ICMP       98    53                                     Echo (ping) request  id=0x77f9, seq=228/58368, ttl=53 (reply in 113)
  113 2023-07-22 16:17:57.506903  113.87.50.124    114.132.168.144  ICMP       98    64                                     Echo (ping) reply    id=0x77f9, seq=228/58368, ttl=64 (request in 112)
  114 2023-07-22 16:17:58.508586  114.132.168.144  113.87.50.124    ICMP       98    53                                     Echo (ping) request  id=0x77f9, seq=229/58624, ttl=53 (reply in 115)
  115 2023-07-22 16:17:58.508601  113.87.50.124    114.132.168.144  ICMP       98    64                                     Echo (ping) reply    id=0x77f9, seq=229/58624, ttl=64 (request in 114)
  116 2023-07-22 16:17:59.510072  114.132.168.144  113.87.50.124    ICMP       98    53                                     Echo (ping) request  id=0x77f9, seq=230/58880, ttl=53 (reply in 117)
  117 2023-07-22 16:17:59.510090  113.87.50.124    114.132.168.144  ICMP       98    64                                     Echo (ping) reply    id=0x77f9, seq=230/58880, ttl=64 (request in 116)
  118 2023-07-22 16:18:00.511659  114.132.168.144  113.87.50.124    ICMP       98    53                                     Echo (ping) request  id=0x77f9, seq=231/59136, ttl=53 (reply in 119)
  119 2023-07-22 16:18:00.511676  113.87.50.124    114.132.168.144  ICMP       98    64                                     Echo (ping) reply    id=0x77f9, seq=231/59136, ttl=64 (request in 118)
  120 2023-07-22 16:18:01.015005  113.87.50.124    117.152.35.93    ICMP      218    63,53                                  Destination unreachable (Port unreachable)
  121 2023-07-22 16:18:01.512188  114.132.168.144  113.87.50.124    ICMP       98    53                                     Echo (ping) request  id=0x77f9, seq=232/59392, ttl=53 (reply in 122)
  122 2023-07-22 16:18:01.512209  113.87.50.124    114.132.168.144  ICMP       98    64                                     Echo (ping) reply    id=0x77f9, seq=232/59392, ttl=64 (request in 121)

> Frame 6: 98 bytes on wire (784 bits), 98 bytes captured (784 bits)
> Ethernet II, Src: 00:00:00_00:00:00 (00:00:00:00:00:00), Dst: 00:00:00_00:00:00 (00:00:00:00:00:00)
> Internet Protocol Version 4, Src: 113.87.50.124, Dst: 114.132.168.144
> Internet Control Message Protocol
```

2.Tcpdump

以上大部分配置都可以沿用，然后设置如下几个参数，让ros把数据包传送给指定的Server，Server端再使用tcpdump来抓包保存。

1）edit file-name

```
[RokasYang@MikroTik] /tool/sniffer> edit file-name
[RokasYang@MikroTik] /tool/sniffer>
```

清空文件名，因为不需要保存到ros下，ctrl + o 保存退出。

2）edit streaming-enabled

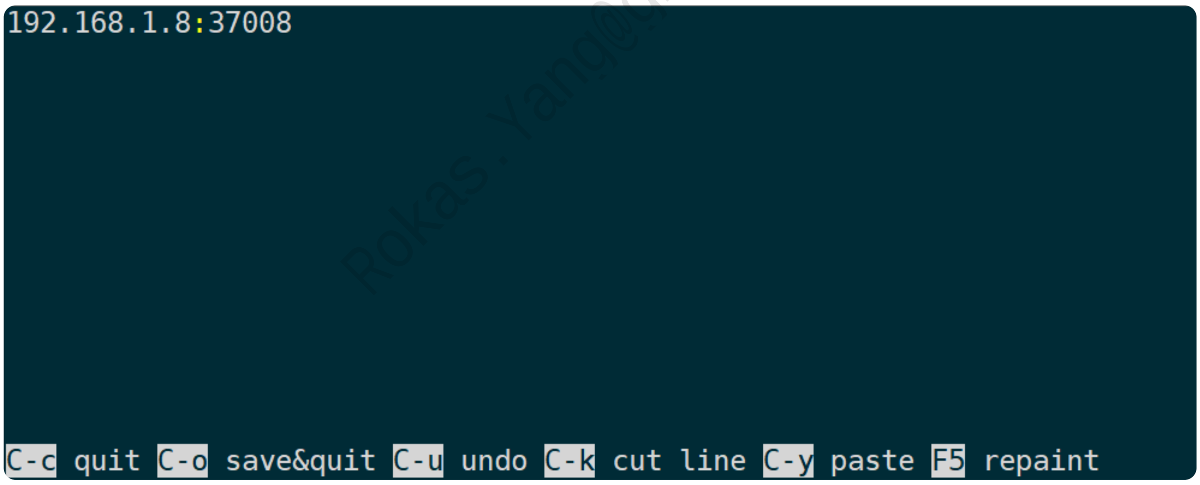
将streaming-enabled改成yes，ctrl + o保存退出。

```
[RokasYang@MikroTik] /tool/sniffer> edit streaming-enabled
[RokasYang@MikroTik] /tool/sniffer>
```

3）edit streaming-server

写入server接收端的IP:PORT地址，ctrl + o保存退出。

```
[RokasYang@MikroTik] /tool/sniffer> edit streaming-server
[RokasYang@MikroTik] /tool/sniffer>
```



```
192.168.1.8:37008

C-c quit C-o save&quit C-u undo C-k cut line C-y paste F5 repaint
```

4）Server端使用tcpdump

在server端开启一个tcpdump进程，抓取37008端口，写入到文件server.pcap：

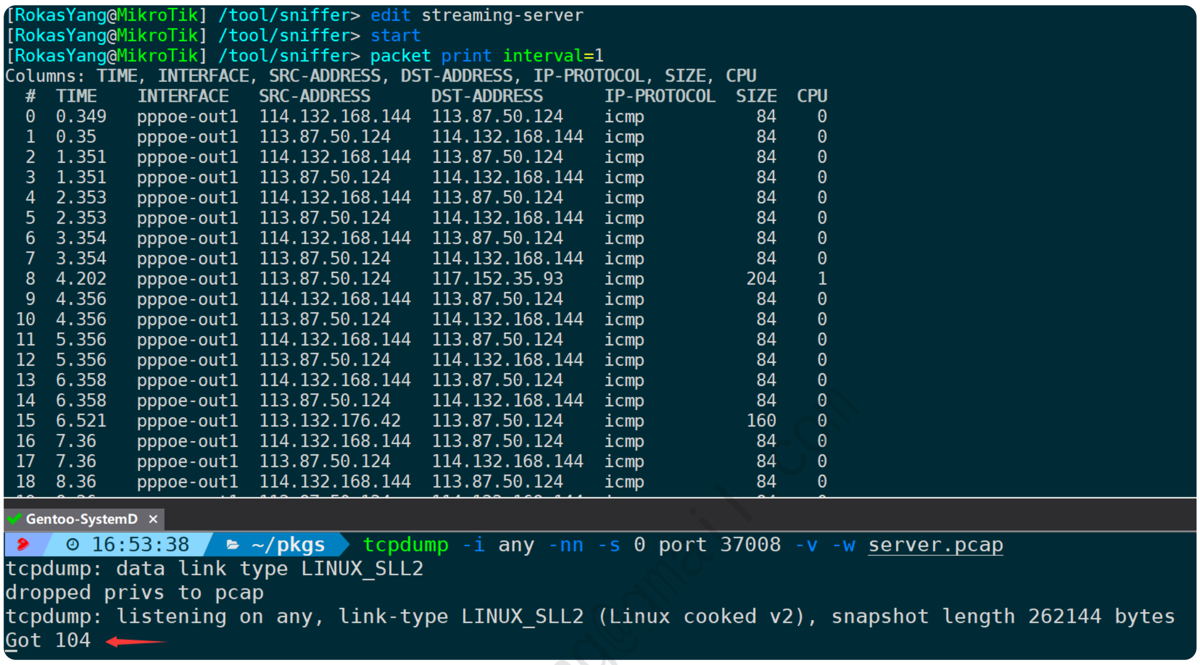
```
tcpdump -i any -nn -s 0 port 37008 -v -w server.pcap
```

5） start

在ros上执行start开始抓包：

```
[RokasYang@MikroTik] /tool/sniffer> start
```

server端可以实时看到抓包数量，ros也可以通过 `packet print interval=1` 来查看数据并且1s刷新一次：



```
[RokasYang@MikroTik] /tool/sniffer> edit streaming-server
[RokasYang@MikroTik] /tool/sniffer> start
[RokasYang@MikroTik] /tool/sniffer> packet print interval=1
Columns: TIME, INTERFACE, SRC-ADDRESS, DST-ADDRESS, IP-PROTOCOL, SIZE, CPU
  #  TIME   INTERFACE    SRC-ADDRESS      DST-ADDRESS      IP-PROTOCOL  SIZE  CPU
  0  0.349  pppoe-out1  114.132.168.144  113.87.50.124    icmp           84    0
  1  0.35   pppoe-out1  113.87.50.124    114.132.168.144  icmp           84    0
  2  1.351  pppoe-out1  114.132.168.144  113.87.50.124    icmp           84    0
  3  1.351  pppoe-out1  113.87.50.124    114.132.168.144  icmp           84    0
  4  2.353  pppoe-out1  114.132.168.144  113.87.50.124    icmp           84    0
  5  2.353  pppoe-out1  113.87.50.124    114.132.168.144  icmp           84    0
  6  3.354  pppoe-out1  114.132.168.144  113.87.50.124    icmp           84    0
  7  3.354  pppoe-out1  113.87.50.124    114.132.168.144  icmp           84    0
  8  4.202  pppoe-out1  113.87.50.124    117.152.35.93    icmp          204    1
  9  4.356  pppoe-out1  114.132.168.144  113.87.50.124    icmp           84    0
 10  4.356  pppoe-out1  113.87.50.124    114.132.168.144  icmp           84    0
 11  5.356  pppoe-out1  114.132.168.144  113.87.50.124    icmp           84    0
 12  5.356  pppoe-out1  113.87.50.124    114.132.168.144  icmp           84    0
 13  6.358  pppoe-out1  114.132.168.144  113.87.50.124    icmp           84    0
 14  6.358  pppoe-out1  113.87.50.124    114.132.168.144  icmp           84    0
 15  6.521  pppoe-out1  113.132.176.42   113.87.50.124    icmp          160    0
 16  7.36   pppoe-out1  114.132.168.144  113.87.50.124    icmp           84    0
 17  7.36   pppoe-out1  113.87.50.124    114.132.168.144  icmp           84    0
 18  8.36   pppoe-out1  114.132.168.144  113.87.50.124    icmp           84    0
```

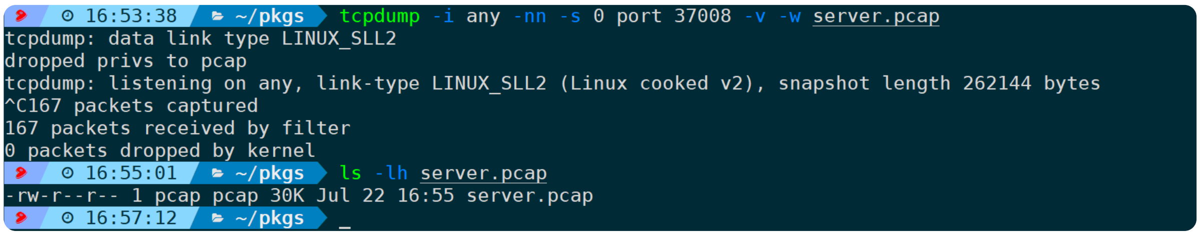
```
 16:53:38  ~/pkgs  tcpdump -i any -nn -s 0 port 37008 -v -w server.pcap
tcpdump: data link type LINUX_SLL2
dropped privs to pcap
tcpdump: listening on any, link-type LINUX_SLL2 (Linux cooked v2), snapshot length 262144 bytes
Got 104
```

6） stop

停止抓包：

```
[RokasYang@MikroTik] /tool/sniffer> stop
```

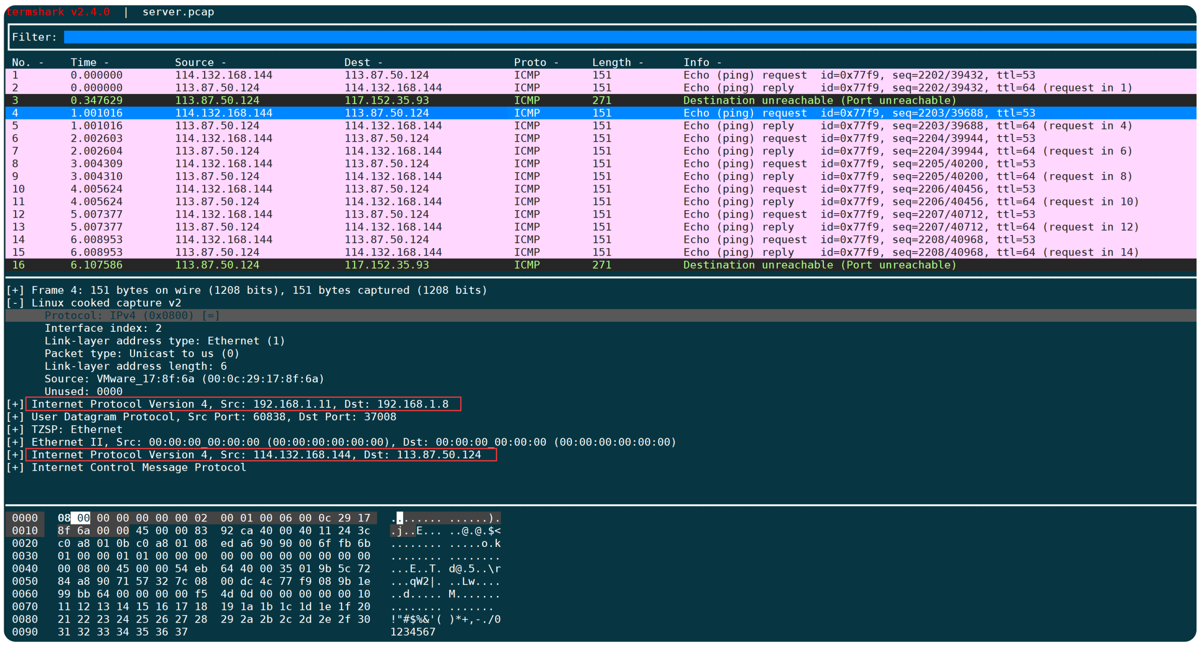
Server按ctrl + c终止就行。



```
 16:53:38  ~/pkgs  tcpdump -i any -nn -s 0 port 37008 -v -w server.pcap
tcpdump: data link type LINUX_SLL2
dropped privs to pcap
tcpdump: listening on any, link-type LINUX_SLL2 (Linux cooked v2), snapshot length 262144 bytes
^C167 packets captured
167 packets received by filter
0 packets dropped by kernel
 16:55:01  ~/pkgs  ls -lh server.pcap
-rw-r--r-- 1 pcap pcap 30K Jul 22 16:55 server.pcap
 16:57:12  ~/pkgs  _
```

之后便可以看到server.pcap文件。

如果你不想把文件传送到windows使用wireshark来分析，那么命令行模式的termshark也是不错的选择：



```
termshark v2.4.0  |  server.pcap
Filter:
No. -    Time -           Source -                  Dest -                    Proto -     Length -      Info -
1        0.000000         114.132.168.144           113.87.50.124             ICMP        151           Echo (ping) request  id=0x77f9, seq=2202/39432, ttl=53
2        0.000000         113.87.50.124             114.132.168.144           ICMP        151           Echo (ping) reply    id=0x77f9, seq=2202/39432, ttl=64 (request in 1)
3        0.347629         113.87.50.124             117.152.35.93             ICMP        271           Destination unreachable (Port unreachable)
4        1.001016         114.132.168.144           113.87.50.124             ICMP        151           Echo (ping) request  id=0x77f9, seq=2203/39688, ttl=53
5        1.001016         113.87.50.124             114.132.168.144           ICMP        151           Echo (ping) reply    id=0x77f9, seq=2203/39688, ttl=64 (request in 4)
6        2.002603         114.132.168.144           113.87.50.124             ICMP        151           Echo (ping) request  id=0x77f9, seq=2204/39944, ttl=53
7        2.002604         113.87.50.124             114.132.168.144           ICMP        151           Echo (ping) reply    id=0x77f9, seq=2204/39944, ttl=64 (request in 6)
8        3.004309         114.132.168.144           113.87.50.124             ICMP        151           Echo (ping) request  id=0x77f9, seq=2205/40200, ttl=53
9        3.004310         113.87.50.124             114.132.168.144           ICMP        151           Echo (ping) reply    id=0x77f9, seq=2205/40200, ttl=64 (request in 8)
10       4.005624         114.132.168.144           113.87.50.124             ICMP        151           Echo (ping) request  id=0x77f9, seq=2206/40456, ttl=53
11       4.005624         113.87.50.124             114.132.168.144           ICMP        151           Echo (ping) reply    id=0x77f9, seq=2206/40456, ttl=64 (request in 10)
12       5.007377         114.132.168.144           113.87.50.124             ICMP        151           Echo (ping) request  id=0x77f9, seq=2207/40712, ttl=53
13       5.007377         113.87.50.124             114.132.168.144           ICMP        151           Echo (ping) reply    id=0x77f9, seq=2207/40712, ttl=64 (request in 12)
14       6.008953         114.132.168.144           113.87.50.124             ICMP        151           Echo (ping) request  id=0x77f9, seq=2208/40968, ttl=53
15       6.008953         113.87.50.124             114.132.168.144           ICMP        151           Echo (ping) reply    id=0x77f9, seq=2208/40968, ttl=64 (request in 14)
16       6.107586         113.87.50.124             117.152.35.93             ICMP        271           Destination unreachable (Port unreachable)

[+] Frame 4: 151 bytes on wire (1208 bits), 151 bytes captured (1208 bits)
[-] Linux cooked capture v2
      Protocol: IPv4 (0x0800) [=]
      Interface index: 2
      Link-layer address type: Ethernet (1)
      Packet type: Unicast to us (0)
      Link-layer address length: 6
      Source: VMware_17:8f:6a (00:0c:29:17:8f:6a)
      Unused: 0000
[+] Internet Protocol Version 4, Src: 192.168.1.11, Dst: 192.168.1.8
[+] User Datagram Protocol, Src Port: 60838, Dst Port: 37008
[+] TZSP: Ethernet
[+] Ethernet II, Src: 00:00:00_00:00:00 (00:00:00:00:00:00), Dst: 00:00:00_00:00:00 (00:00:00:00:00:00)
[+] Internet Protocol Version 4, Src: 114.132.168.144, Dst: 113.87.50.124
[+] Internet Control Message Protocol

0000   08 00 00 00 00 00 00 02  00 01 00 06 00 0c 29 17   ........ ......).
0010   8f 6a 00 00 45 00 00 83  92 ca 40 00 40 11 24 3c   .j..E... ..@.@.$<
0020   c0 a8 01 0b c0 a8 01 08  ed a6 90 90 00 6f fb 6b   ........ .....o.k
0030   01 00 00 01 01 00 00 00  00 00 00 00 00 00 00 00   ........ ........
0040   00 08 00 45 00 00 54 eb  64 40 00 35 01 9b 5c 72   ...E..T. d@.5..\r
0050   84 a8 90 71 57 32 7c 08  00 dc 4c 77 f9 08 9b 1e   ...qW2|. ..Lw....
0060   99 bb 64 00 00 00 00 f5  4d 0d 00 00 00 00 00 10   ..d..... M.......
0070   11 12 13 14 15 16 17 18  19 1a 1b 1c 1d 1e 1f 20   ........ .......
0080   21 22 23 24 25 26 27 28  29 2a 2b 2c 2d 2e 2f 30   !"#$%&'( )*+,-./0
0090   31 32 33 34 35 36 37                               1234567
```

可以清晰看到ros使用IPIP隧道来封装转发数据包给server端。

四、总结

RouterOS作为业内一款强大的软路由操作系统，对于各种专业网络协议的覆盖率，使它原生支持捕获报文，非常利于网络排障、路由排障、协议排障等，可谓专业级别的路由操作系统。

本文以捕获ICMP为示例，避免篇幅过长，其它过滤参数包括IP地址过滤、Mac地址过滤、端口过滤等没有一一展示，根据实际需求场景抓最有用的包，让网络排障事半功倍。