

## Linux网络抓包分析工具

黑客专栏 2022-12-15 18:01 发表于河南

👉👉关注后回复“进群”，拉你进程序员交流群👉👉



架构师大咖

架构师大咖，打造有价值的架构师交流平台。分享架构师干货、教程、课程、资讯。架...

公众号



算法专栏

算法专栏，每日推送。算法是程序员内功，分享算法知识、文章、工具、算法题、教程...

公众号

转自：入门小站



## 一、tcpdump

### 1、作用

**tcpdump** 指令可列出经过指定网络界面的数据包文件头，可以将网络中传送的数据包的“头”完全截获下来提供分析。它支持针对网络层、协议、主机、网络或端口的过滤，并提供 **and**、**or**、**not** 等逻辑语句来帮助你摘取有用信息。

由于它需要将网络接口设置为混杂模式，普通用户不能正常执行，但具备 **root** 权限的用户可以直接执行它来获取网络上的信息

#### 其他抓包工具

- **wireshark**具有图形化和命令行两种版本，可以对 **tcpdump** 抓的包进行分析，其主要功能就是分析数据包。
- **ngrep**它将抓到的包数据以文本形式直接显示出来，适用于包数据包含文本的[抓包]分析(如 HTTP、MySQL)

### 2、命令选项

**tcpdump [选项] [协议] [数据流方向] [范围]**

- **-a** 将网络地址和广播地址转变成名字
- **-A** 以 ASCII 格式打印出所有分组，并将链路层的头最小化

- -b 数据链路层上选择协议，包括 ip/arp/rarp/ipx 都在这一层
- -c 指定收取数据包的次数，即在收到指定数量的数据包后退出 tcpdump
- -d 将匹配信息包的代码以人们能够理解的汇编格式输出
- -dd 将匹配信息包的代码以 c 语言程序段的格式输出
- -ddd 将匹配信息包的代码以十进制的形式输出
- -D 打印系统中所有可以监控的网络接口
- -e 在输出行打印出数据链路层的头部信息
- -f 将外部的 Internet 地址以数字的形式打印出来，即不显示主机名
- -F 从指定的文件中读取表达式，忽略其他的表达式
- -i 指定监听网络接口
- -l 使标准输出变为缓冲形式，可以数据导出到文件
- -L 列出网络接口已知的数据链路
- -n 不把网络地址转换为名字
- -N 不输出主机名中的域名部分，例如 www.baidu.com 只输出 www
- -nn 不进行端口名称的转换
- -P 不将网络接口设置为混杂模式
- -q 快速输出，即只输出较少的协议信息
- -r 从指定的文件中读取数据，一般是 -w 保存的文件
- -w 将捕获到的信息保存到文件中，且不分析和打印在屏幕
- -s 从每个组中读取在开始的 snaplen 个字节，而不是默认的 68 个字节

- -S 将 tcp 的序列号以绝对值形式输出，而不是相对值
- -T 将监听到的包直接解析为指定的类型的报文，常见的类型有 rpc ( 远程过程调用 ) 和 snmp ( 简单网络管理协议 )
- -t 在输出的每一行不打印时间戳
- -tt 在每一行中输出非格式化的时间戳
- -ttt 输出本行和前面以后之间的时间差
- -tttt 在每一行中输出 data 处理的默认格式的时间戳
- -u 输出未解码的 NFS 句柄
- -v 输出稍微详细的信息，例如在 ip 包中可以包括 ttl 和服务类型的信息
- -vv 输出相信的报报文信息

### 3、tcpdump 表达式

关于数据类型的关键字

包括 host、port、net：

host 192.168.100.1 表示一台主机，net 192.168.100.0 表示一个网络网段，port 80 指明端口号为 80，在这里如果没有指明数据类型，那么默认就是 host

数据传输方向的关键字

包括 src、dst、dst or src、dst and src，这些关键字指明了传输的方向，比如 src 192.168.100.1 说明数据包源地址是 192.168.100.1。dst net 192.168.100.0 指明目的网

络地址是 192.168.100.0，默认是监控主机对主机的 src 和 dst，即默认监听本机和目标主机的所有数据

**协议关键字**

包括 ip、arp、rarp、udp

**其他关键字**

- 运算类型：or、and、not、！
- 辅助功能型：gateway、less、broadcast、greater

## 4、tcpdump 捕获方式

tcpdump [协议类型] [源或目标] [主机名称或 IP] [or/and/not/! 条件组合] [源或目标]  
[主机名或 IP] [or/and/not/! 条件组合] [端口] [端口号] ..... [or/and/not/! 条件组合]  
[条件]

```
> tcpdump ip dst 192.168.10.1 and src 192.168.10.10 and port 80 and host !www.baidu.com
```

**tcpdump**

默认监听在第一块网卡，监听所有经过此网卡的数据包

```
[root@gang ~]#tcpdump
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on virbr0, link-type EN10MB (Ethernet), capture size 262144 bytes
```

CSDN @芒地狼

```
> tcpdump -i ens33
```

监听指定网卡 ens33 的所有传输数据包

```
[root@gang ~]#tcpdump -i ens33
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on ens33, link-type EN10MB (Ethernet), capture size 262144 bytes
17:49:44.834896 IP gang.ssh > 192.168.100.1.57109: Flags [P.], seq 663600428:663600616, ack 2184220387, win 261, length 188
17:49:44.835091 IP 192.168.100.1.57109 > gang.ssh: Flags [.], ack 188, win 4096, length 0
17:49:44.835490 IP gang.44076 > public1.114dns.com.domain: 50592+ PTR? 1.100.168.192.in-addr.arpa. (44)
```

CSDN @芒地狼

```
> tcpdump -i ens33 host 192.168.100.10
```

捕获主机 192.168.100.10 经过网卡 ens33 的所有数据包 ( 也可以是主机名 , 但要求可以解析出 IP 地址 )

```
[root@gang ~]#tcpdump -i ens33 host 192.168.100.10
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on ens33, link-type EN10MB (Ethernet), capture size 262144 bytes
17:52:52.342595 IP gang.ssh > 192.168.100.1.57109: Flags [P.], seq 664815428:664815616, ack 2184222123, win 261, length 188
17:52:52.342748 IP 192.168.100.1.57109 > gang.ssh: Flags [.], ack 188, win 4101, length 0
17:52:52.343001 IP gang.51519 > public1.114dns.com.domain: 17981+ PTR? 1.100.168.192.in-addr.arpa. (44)
17:52:52.357718 IP public1.114dns.com.domain > gang.51519: 17981 NXDomain 0/1
17:52:52.358574 IP gang.42084 > public1.114dns.com.domain: 24666+ PTR? 10.100.168.192.in-addr.arpa. (45)
```

CSDN @芒地狼

```
> tcpdump -i ens33 host 192.168.100.10
18:26:39.485023 IP 192.168.100.1.57109 > gang.ssh: Flags [P.], seq 240580:240580, ack 37, win 261, length 0
18:26:39.485063 IP gang.ssh > 192.168.100.1.57109: Flags [P.], seq 240580:240824, ack 37, win 261, length 36
18:26:39.512584 IP 192.168.100.1.57109 > gang.ssh: Flags [P.], seq 37:73, ack 240824, win 4105, length 36
18:26:39.512644 IP gang.ssh > 192.168.100.1.57109: Flags [P.], seq 240824:240972, ack 73, win 261, length 148
```

CSDN @芒地狼

- 第一列：报文的时间
- 第二列：网络协议 IP
- 第三列：发送方的 ip 地址、端口号、域名，上图显示的是本机的域名，可通过 /  
etc/hosts 查看本机域名
- 第四列：箭头 >，表示数据流向
- 第五列：接收方的 ip 地址、端口号、域名，
- 第六列：冒号
- 第七列：数据包内容，报文头的摘要信息，有 ttl、报文类型、标识值、序列、包的大小等信息

```
> tcpdump host 192.168.130.151 and 192.168.130.152 or 192.168.130.153 192.168.130.152 or 192.168.
```

捕获主机 192.168.56.209 和主机 192.168.56.210 或 192.168.56.211 的所有通信数据包

```
> tcpdump ip host node9 and not www.baidu.com
```

捕获主机 node9 与其他主机之间（不包括 www.baidu.com）通信的 ip 数据包

```
> tcpdump ip host node9 and ! www.baidu.com
```

捕获 node9 与其他所有主机的通信数据包 ( 不包括 www.baidu.com )

```
> tcpdump -i ens33 src node10
```

捕获源主机 node10 发送的所有的经过 ens33 网卡的所有数据包

```
> tcpdump -i ens33 dst host www.baidu.com
```

捕获所有发送到主机 www.baidu.com 的数据包

监听主机 192.168.56.1 和 192.168.56.210 之间 ip 协议的 80 端口的且排除 www.baidu.com 通信的所有数据包：

```
> tcpdump ip dst 192.168.56.1 and src 192.168.56.210 and port 80 and host ! baidu.com
```

也可以写成 tcpdump ip dst 192.168.56.1 and src 192.168.56.210 and port 80 and host not www.baidu.com · 即 not 和 ! 都是相同的取反的意思



```
> tcpdump arp
```

监控指定主机的通信数据包与 1.9.1 方式相同

```
> tcpdump tcp port 22 and host 192.168.56.210
```

捕获主机 192.168.56.210 接收和发出的 tcp 协议的 ssh 的数据包

```
tcpdump udp port 53
```

监听本机 udp 的 53 端口的数据包，udp 是 dns 协议的端口，这也是一个 dns 域名解析的完整过程

## 5、常用的过滤条件

tcpdump 可以支持逻辑运算符

and: 与运算，所有的条件都需要满足，可用 “and” 和 “&&” 表示

or: 或运行，只要有一个条件满足就可以，可用 “or” 和 “|” 表示

not: 取反，即取反条件，可以用 “not” 和 “!” 表示

```
> tcpdump icmp and src 192.168.100.10 -i ens33 -n
```

过滤 icmp 报文并且源 IP 是 192.168.100.10

多条件格式

在使用多个过滤条件进行组合时，有可能需要用到括号，而括号在 shell 中是特殊符号，又需要使用引号将其包含。用括号的主要作用是逻辑运算符之间存在优先级，!>and > or, 为例条件能够精确所以需要对一些必要的组合括号括起来，而括号的意思相当于加减运算一样，括起来的内容作为一个整体进行逻辑运算。


过滤源地址是 192.168.100.1 并且目的地址是 192.168.20.20 的数据包或者 ARP 协议的包

```
[root@gang ~]# tcpdump '(src 192.168.100.1 and dst 192.168.100.10 ) or arp' -i ens33 -n
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on ens33, link-type EN10MB (Ethernet), capture size 262144 bytes
18:20:03.324656 IP 192.168.100.1.57109 > 192.168.100.10.ssh: Flags [.], ack 664927488, win 4103, length 0
18:20:03.365514 IP 192.168.100.1.57109 > 192.168.100.10.ssh: Flags [.], ack 149, win 4102, length 0
18:20:03.406719 IP 192.168.100.1.57109 > 192.168.100.10.ssh: Flags [.], ack 20, win 4102, length 0
```

```
> tcpdump **src** host 192.168.10.10 -i ens33 -n -c 5
```

过滤源 IP 地址是 192.168.10.10 的包


```
0 packets dropped by kernel
[root@gang ~]# tcpdump src host 192.168.100.10 -i ens33 -n -c 5
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on ens33, link-type EN10MB (Ethernet), capture size 262144 bytes
18:39:19.422691 IP 192.168.100.10.ssh > 192.168.100.1.57109: Flags [P.], seq 665433368:665433556, ack 218425
5431, win 261, length 188
18:39:19.422909 IP 192.168.100.10.ssh > 192.168.100.1.57109: Flags [P.], seq 188:360, ack 1, win 261, length
172
18:39:19.423052 IP 192.168.100.10.ssh > 192.168.100.1.57109: Flags [P.], seq 360:508, ack 1, win 261, length
148
18:39:19.423143 IP 192.168.100.10.ssh > 192.168.100.1.57109: Flags [P.], seq 508:656, ack 1, win 261, length
148
18:39:19.423271 IP 192.168.100.10.ssh > 192.168.100.1.57109: Flags [P.], seq 656:804, ack 1, win 261, length
148
5 packets captured
5 packets received by filter
0 packets dropped by kernel
[root@gang ~]#
```



```
> tcpdump **dst** host 192.168.10.10 -i ens33 -n -c 5
```

过滤目的 IP 地址是 192.168.10.10 的包

```
0 packets dropped by kernel
[root@gang ~]# tcpdump dst host 192.168.100.10 -i ens33 -n -c 5
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on ens33, link-type EN10MB (Ethernet), capture size 262144 bytes
18:41:43.387756 IP 192.168.100.1.57109 > 192.168.100.10.ssh: Flags [.], ack 665436468, win 4104, length 0
18:41:43.428593 IP 192.168.100.1.57109 > 192.168.100.10.ssh: Flags [.], ack 149, win 4103, length 0
18:41:43.468701 IP 192.168.100.1.57109 > 192.168.100.10.ssh: Flags [.], ack 289, win 4103, length 0
18:41:43.509710 IP 192.168.100.1.57109 > 192.168.100.10.ssh: Flags [.], ack 429, win 4102, length 0
18:41:43.535973 IP 192.168.100.1.57109 > 192.168.100.10.ssh: Flags [P.], seq 0:36, ack 569, win 4101, length
36
5 packets captured
5 packets received by filter
0 packets dropped by kernel
[root@gang ~]#
```



基于端口进行过滤

```
> tcpdump port 22 -i ens33 -n -c 5
> 过滤端口号为 22 即 ssh 协议的
```

```
0 packets dropped by kernel
[root@gang ~]# tcpdump port 22 -i ens33 -n -c 5
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on ens33, link-type EN10MB (Ethernet), capture size 262144 bytes
18:43:21.154569 IP 192.168.100.10.ssh > 192.168.100.1.57109: Flags [P.], seq 665438436:665438624, ack 218425
8651, win 261, length 188
18:43:21.154732 IP 192.168.100.1.57109 > 192.168.100.10.ssh: Flags [.], ack 188, win 4101, length 0
18:43:21.154751 IP 192.168.100.10.ssh > 192.168.100.1.57109: Flags [P.], seq 188:360, ack 1, win 261, length
172
18:43:21.154862 IP 192.168.100.10.ssh > 192.168.100.1.57109: Flags [P.], seq 360:612, ack 1, win 261, length
252
18:43:21.154923 IP 192.168.100.1.57109 > 192.168.100.10.ssh: Flags [.], ack 612, win 4106, length 0
5 packets captured
6 packets received by filter
0 packets dropped by kernel
[root@gang ~]#
```

入门小站

CSDN @芒地狼

```
> tcpdump portrange 22-433 -i ens33 -n -c 8
```

过滤端口号 22-433 内的数据包

```
[root@gang ~]# tcpdump portrange 22-443 -i ens33 -n -c 8
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on ens33, link-type EN10MB (Ethernet), capture size 262144 bytes
18:45:12.338116 IP 192.168.100.10.ssh > 192.168.100.1.57109: Flags [P.], seq 665441060:665441248, ack 218426
0347, win 261, length 188
18:45:12.338268 IP 192.168.100.1.57109 > 192.168.100.10.ssh: Flags [.], ack 188, win 4103, length 0
18:45:12.338291 IP 192.168.100.10.ssh > 192.168.100.1.57109: Flags [P.], seq 188:360, ack 1, win 261, length
172
18:45:12.338422 IP 192.168.100.10.ssh > 192.168.100.1.57109: Flags [P.], seq 360:612, ack 1, win 261, length
252
18:45:12.338496 IP 192.168.100.1.57109 > 192.168.100.10.ssh: Flags [.], ack 612, win 4101, length 0
18:45:12.338505 IP 192.168.100.10.ssh > 192.168.100.1.57109: Flags [P.], seq 612:760, ack 1, win 261, length
148
18:45:12.338580 IP 192.168.100.10.ssh > 192.168.100.1.57109: Flags [P.], seq 760:1012, ack 1, win 261, length
252
18:45:12.338635 IP 192.168.100.1.57109 > 192.168.100.10.ssh: Flags [.], ack 1012, win 4106, length 0
8 packets captured
9 packets received by filter
0 packets dropped by kernel
```

入门小站

CSDN @芒地狼

## 二、wireshark

### 1、什么是 wireshark

Wireshark 是一个网络封包分析软件。网络封包分析软件的功能是捕获网络数据包，并尽可能显示出最为详细的网络封包资料。Wireshark 使用 WinPCAP 作为接口，直接与网卡

进行数据报文交换

## 2、安装 wireshark

Linux 中有两个版本的 wireshark，一个是 wireshark，这个版本是无图形化界面，基本命令是“tshark”。

一个是 wireshark-gnome（界面版本），这个版本只能安装在支持 GUI 功能的 Linux 的版本中。

```
> yum -y install wireshark // 安装无图形化版本  
> yum -y install wireshark-gnome // 安装图形化版本
```

```
[root@gang ~]#rpm -q wireshark  
未安装软件包 wireshark  
[root@gang ~]#yum -y install wireshark  
已加载插件: fastestmirror, langpacks  
Loading mirror speeds from cached hostfile  
* base: mirrors.huaweicloud.com  
* extras: mirrors.aliyun.com  
* updates: mirrors.aliyun.com
```

入门小站  
CSDN @ 芒地狼

```
[root@gang ~]#yum -y install wireshark-gnome  
已加载插件: fastestmirror, langpacks  
Loading mirror speeds from cached hostfile  
* base: mirrors.huaweicloud.com  
* extras: mirrors.aliyun.com  
* updates: mirrors.aliyun.com  
正在解决依赖关系
```

入门小站  
CSDN @ 芒地狼



注:这里的通过 yum 进行安装，需要提前做好 epel 源（即红帽操作系统额外拓展包），装上了 EPEL 之后，就相当于添加了一个第三方源。官方的 rpm repository 提供的 rpm 包也不够丰富，很多时候需要自己编译那太辛苦了，而 EPEL 可以解决官方 yum 源数据包不够丰富的情况。

### 安装epel源

```
> yum -y install epel-release
```

```

[root@gang ~]#yum -y install epel-release
已加载插件：fastestmirror, langpacks
Loading mirror speeds from cached hostfile
* base: mirrors.huaweicloud.com
* extras: mirrors.aliyun.com

```

### 3、tshark 命令

tshark 是 wireshark 的命令行工具

tshark 选项 参数

- i：指定捕获的网卡接口，不设置默认第一个非环回口接口
- D：显示所有可用的网络接口列表
- f：指定条件表达式，与 tcpdump 相同
- s：设置每个抓包的大小，默认 65535，多于这个大小的数据将不会被截取。
- c：捕获指定数量的数据包后退出
- w：后接文件名，将抓包的结果输出到 .pcap 文件中，可以借助其他网络分析工具进行分
- p：设置网络接口以非混合模式工作，即只关心和本机有关的流量
- r：后接文件路径，用于分析保持好的网络包文件，比如 tcpdump 的输出文件
- n：禁止所有地址名字解析，即禁止域名解析，默认是允许所有
- N：指定对某一层的地址名字解析，如果 -n 和 -N 同时存在，则 -n 将被忽略，如果两者都不写，则会默认
- m：代表数据链路层
- n：代表网络层
- t：代表传输层
- V：设置将解码结果的细节输出，否则解码结果仅显示一个 packet 一行的 summary
- t：设置结果的时间格式
  - ad：表示带日期的绝对时间
  - a：表示不带日期的绝对时间
  - r：表示从第一个包到现在的相对时间
  - d：表示两个相邻包之间的增量时间

```
tshark -f "icmp" -i ens33 -V -c 1
```

过滤 icmp 报文，并展开详细信息

```
tshark -f "arp" -i ens33
```

过滤 arp 报文

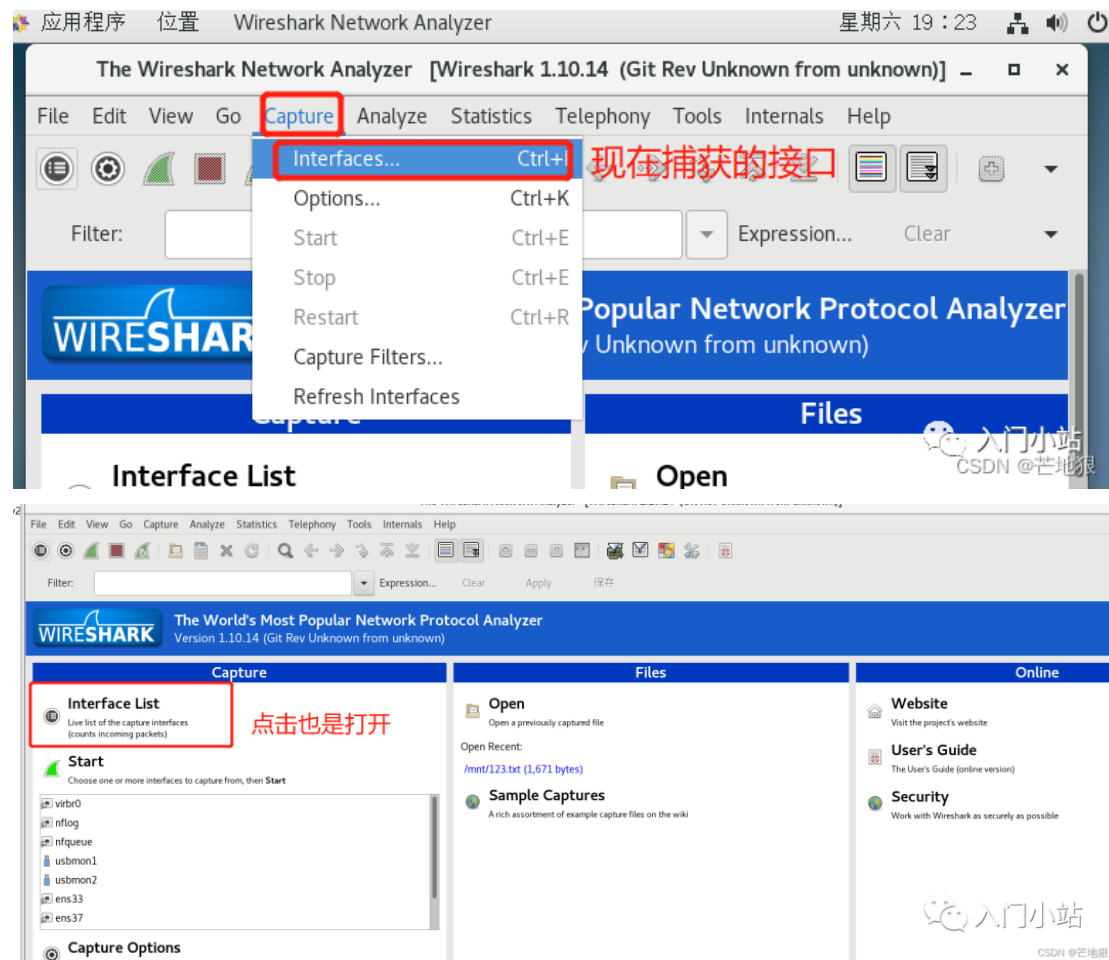


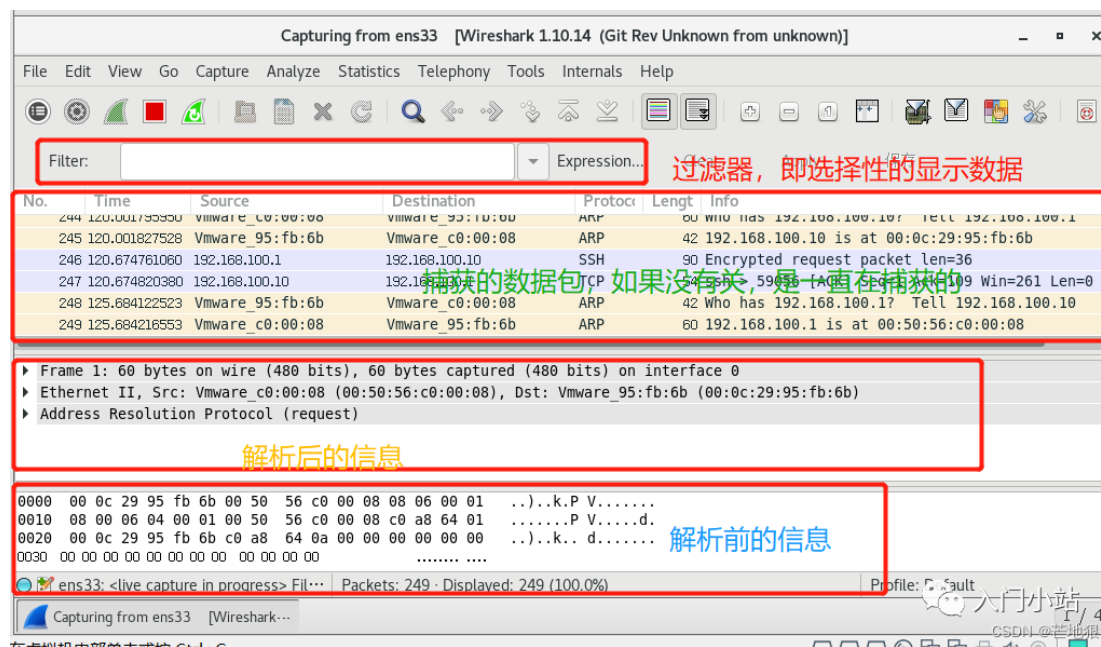
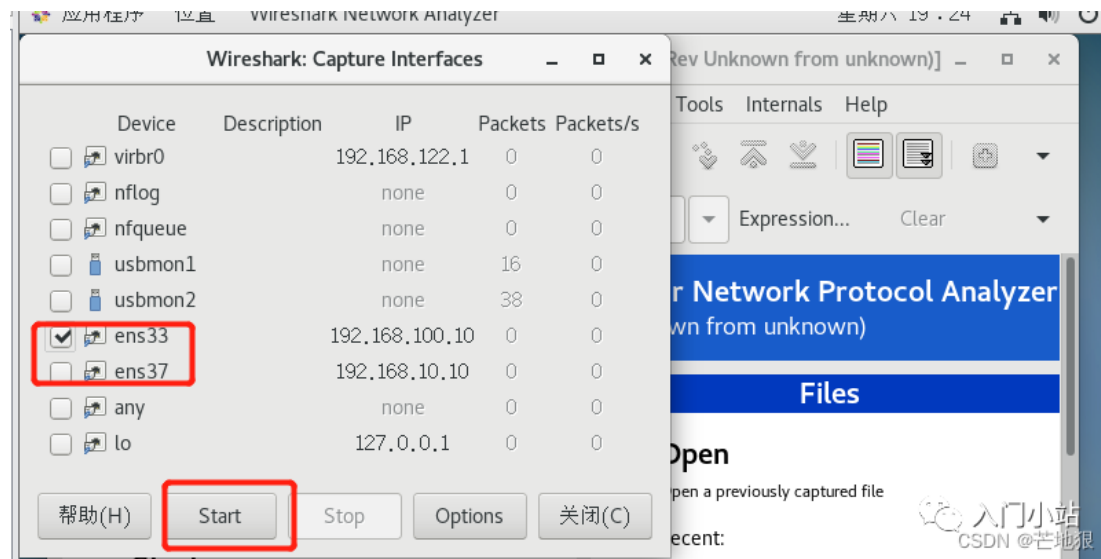
```
Address Resolution Protocol (request)
  Hardware type: Ethernet (1)
  Protocol type: IP (0x0800)
  Hardware size: 6
  Protocol size: 4
  Opcode: request (1)
  Sender MAC address: Vmware_ea:34:0e (00:0c:29:ea:34:0e)
  Sender IP address: 192.168.100.20 (192.168.100.20)
  Target MAC address: 00:00:00 00:00:00 (00:00:00:00:00:00)
  Target IP address: 192.168.100.2 (192.168.100.2)
```

入门小站  
CSDN @芒地狼

#### 4、图形化界面





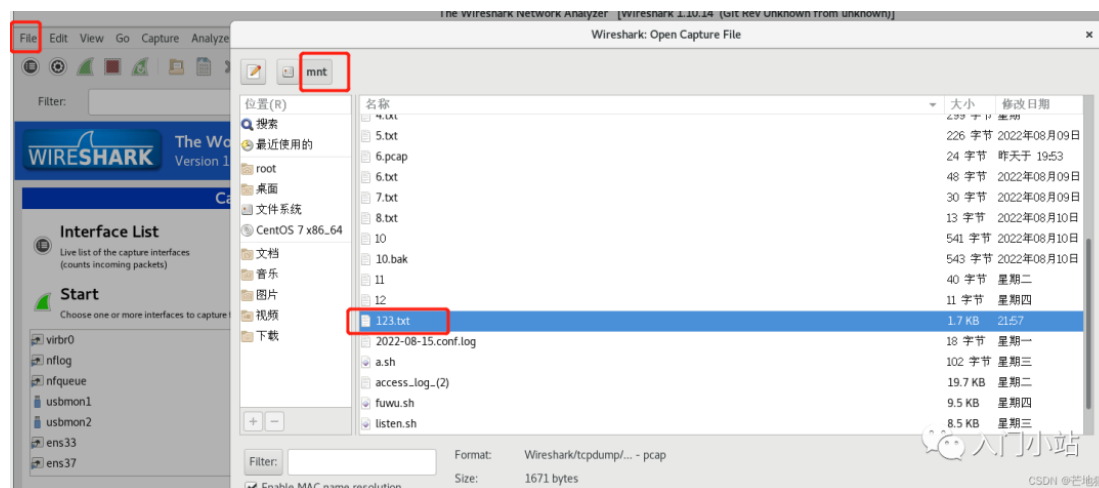


### 三、Tcpdump 和 wireshark 合用

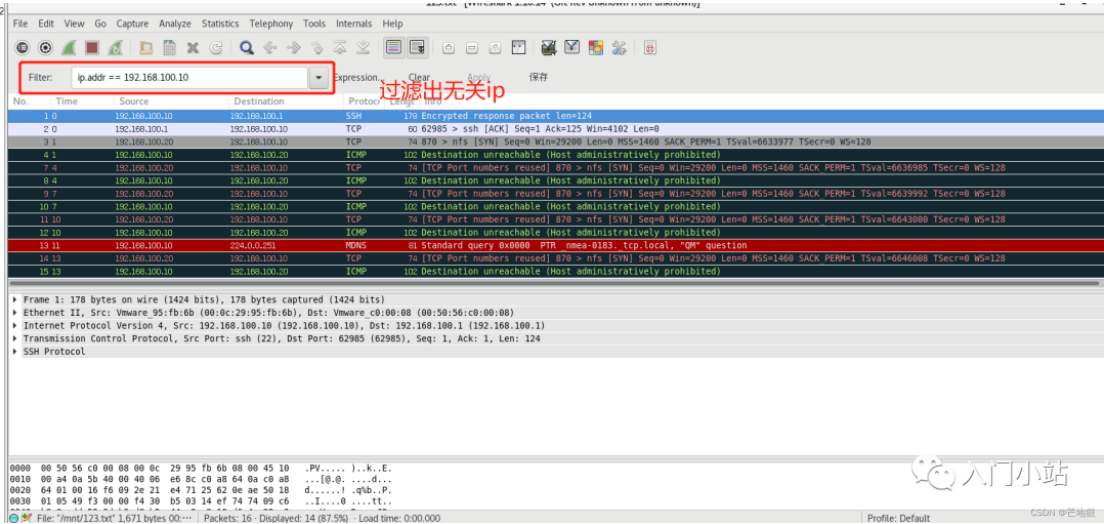
Tcpdump 解析报文信息没有 wireshark 详细，所以可以通过 Tcpdump 捕获数据并输出，再通过 wireshark 进行解析，输出文件格式为 .pcap 或者其他

```
[root@gang mnt]# tcpdump -i ens33 -nn host 192.168.100.10 -w /mnt/123.txt
tcpdump: listening on ens33, link-type EN10MB (Ethernet), capture size 262144 bytes
^C16 packets captured
17 packets received by filter
0 packets dropped by kernel
[root@gang mnt]# ls
1      12      1.txt      2.txt      4        6.pcap     access_log_(2)  nfs.sh
10     123.txt    2          3          4.sh      6.txt      a.sh            test.txt
10.bak  1.sh      2022-08-15.conf.log  3.sh      4.txt     7.txt      fuwu.sh
11     1.sh.0    2.sh      3.txt     5.txt     8.txt     listen.sh
[root@gang mnt]#
```

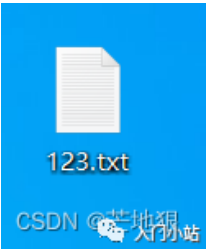
在虚拟机上通过 wireshark 读取



使用 `ip.addr == [ip 地址号]` 可以过滤掉无关 ip



图形读取



用 wireshark 直接打开查看

## 总结

tcpdump 和 wireshark 两种单以抓包的功能来看，是相似的，两者的命令行的选项也是有相同，但是 tcpdump 对数据包分析的能力不是很好，同时目前很多 Linux 内置安装了

tcpdump 这个工具，所以可以通过 tcpdump 把数据包抓出并存放到我们自定义的文件(.pcap)中，再通过把文件取出用 wireshark 进行分析排障

-End-

最近有一些小伙伴，让我帮忙找一些 面试题 资料，于是我翻遍了收藏的 5T 资料后，汇总整理出来，可以说是程序员面试必备！所有资料都整理到网盘了，欢迎下载！



110道Python面试题汇总.pdf



200页Java面试题.pdf



java面试题.pdf



Java面试题手册.pdf



linux及Python语法面试题.pdf



python面试宝典.pdf



面试题整理.pdf



爬虫及网络编程面试题.pdf



数据分析面试可能会问fk.pdf



数据结构与算法面试题.pdf



Python入门到精通

Python入门到精通：人生苦短，我用Python！Python每日推送、Python教程、Pyth...

公众号

点击👉卡片，关注后回复【面试题】即可获得

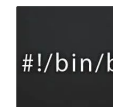
[在看点这里](#) 好文分享给更多人↓↓

阅读原文

喜欢此内容的人还喜欢

## 9 个非常实用的 Shell 拿来就用脚本实例！

计算科学与信息化



## 10个实用 Linux Shell 脚本案例

咸鱼爱搞机



## Linux grep排除过滤输出

myfreax

