tcpdump 速查手册

> 版本适用：tcpdump 4.x+ / libpcap 1.x+
> 需要 root 或 CAP_NET_RAW 权限

目录

1. 基础用法

命令	说明
`tcpdump -D`	列出可用网卡
`tcpdump -i eth0`	指定网卡抓包
`tcpdump -i any`	所有网卡抓包
`tcpdump -c 100`	抓取 100 个包后停止
`tcpdump -n`	不解析主机名（IP 显示）
`tcpdump -nn`	不解析主机名和端口号
`tcpdump -q`	简要输出
`tcpdump -t`	不显示时间戳
`tcpdump -tttt`	显示完整日期+时间

2. 输出控制

选项	说明
`-v`	详细输出（TTL、ID、总长度等）
`-vv`	更详细（NFS、SMB 额外字段）
`-vvv`	最详细
`-X`	十六进制 + ASCII 显示载荷
`-XX`	同上，含链路层头
`-A`	仅 ASCII 显示载荷（适合 HTTP）
`-e`	显示链路层头（MAC 地址）
`-S`	显示绝对序列号（非相对）
`-l`	行缓冲（配合管道实时处理）

组合示例：

```
# 实时 ASCII 查看 HTTP 明文
tcpdump -i eth0 -A -s 0 -l port 80 | grep -i 'host\|GET\|POST'

# 十六进制分析，每行最多抓 1500 字节
tcpdump -i eth0 -XX -s 1500 -c 50
```

---

3. 抓包保存与读取

```
# 保存到 pcap 文件
tcpdump -i eth0 -w capture.pcap

# 读取 pcap 文件
tcpdump -r capture.pcap

# 限制单文件大小（MB），自动轮转
tcpdump -i eth0 -w trace.pcap -C 100

# 按时间轮转（每 3600 秒一个文件）
tcpdump -i eth0 -w 'trace_%Y%m%d_%H%M%S.pcap' -G 3600

# 保留最近 10 个文件
tcpdump -i eth0 -w trace.pcap -C 50 -W 10

# 抓取完整包（snaplen 不截断）
tcpdump -i eth0 -s 0 -w full.pcap
```

---

4. BPF 过滤表达式

4.1 基本语法

```
<类型> <方向> <协议>
```

限定符	可选值	示例
类型 (type)	`host`, `net`, `port`, `portrange`	`host 10.0.0.1`
方向 (dir)	`src`, `dst`, `src or dst`	`src host 10.0.0.1`
协议 (proto)	`tcp`, `udp`, `icmp`, `ip`, `ip6`, `arp`, `ether`	`tcp port 443`

4.2 逻辑运算符

运算符	说明	示例
`and` / `&&`	与	`host 10.0.0.1 and port 80`
`or` / `||`	或	`port 80 or port 443`
`not` / `!`	非	`not port 22`
`()`	分组	`(src host A) and (dst port 80)`

4.3 常用过滤表达式

```
# 按主机
tcpdump host 192.168.1.100
tcpdump src host 10.0.0.1
tcpdump dst host 10.0.0.2

# 按网段
tcpdump net 192.168.1.0/24
tcpdump src net 10.0.0.0/8

# 按端口
tcpdump port 443
tcpdump src port 8080
tcpdump portrange 8000-9000

# 按协议
tcpdump icmp
tcpdump udp
tcpdump 'tcp[tcpflags] & tcp-syn != 0'

# 排除特定流量
tcpdump -i eth0 not port 22 and not arp

# 指定 VLAN
tcpdump vlan 100

# IPv6
tcpdump ip6
tcpdump 'ip6 and tcp port 443'
```

5. 协议专项过滤

5.1 TCP 标志位过滤

```
# SYN 包（新连接请求）
tcpdump 'tcp[tcpflags] & tcp-syn != 0'

# 仅 SYN（不含 SYN-ACK）
tcpdump 'tcp[tcpflags] == tcp-syn'

# SYN-ACK
tcpdump 'tcp[tcpflags] == (tcp-syn|tcp-ack)'

# FIN 包
tcpdump 'tcp[tcpflags] & tcp-fin != 0'

# RST 包（连接异常）
tcpdump 'tcp[tcpflags] & tcp-rst != 0'

# PSH-ACK（数据推送）
tcpdump 'tcp[tcpflags] & (tcp-push|tcp-ack) == (tcp-push|tcp-ack)'

# 使用数值（tcp[13]是标志位字节）
# SYN=0x02, ACK=0x10, FIN=0x01, RST=0x04, PSH=0x08
tcpdump 'tcp[13] & 0x02 != 0'    # 含 SYN
tcpdump 'tcp[13] == 0x12'        # SYN-ACK
tcpdump 'tcp[13] == 0x04'        # 仅 RST
```

TCP 标志位对照表：

位	名称	十六进制	含义
0	FIN	0x01	结束连接
1	SYN	0x02	同步序列号
2	RST	0x04	重置连接
3	PSH	0x08	立即推送数据
4	ACK	0x10	确认
5	URG	0x20	紧急

5.2 HTTP 过滤

```
# GET 请求
tcpdump -A -s 0 'tcp port 80 and (((ip[2:2] - ((ip[0]&0xf)<<2)) - ((tcp[12]&0xf0)>>2)) != 0)' | grep
'GET'

# POST 请求
tcpdump -A -s 0 'tcp dst port 80 and tcp[((tcp[12]&0xf0)>>2):4] = 0x504f5354'

# HTTP 响应
tcpdump -A -s 0 'tcp src port 80 and tcp[((tcp[12]&0xf0)>>2):4] = 0x48545450'

# 简便方式（配合 grep）
tcpdump -i eth0 -A -s 0 -l port 80 | grep -E 'GET|POST|Host:|HTTP/'
```

5.3 DNS 过滤

```
# 所有 DNS 流量
tcpdump port 53

# DNS 查询
tcpdump -vvv dst port 53

# DNS 响应
tcpdump -vvv src port 53

# 特定域名（需要 ASCII 匹配）
tcpdump -i eth0 -l port 53 | grep 'example.com'
```

5.4 ICMP 过滤

```
# 所有 ICMP
tcpdump icmp

# Ping 请求 (type 8)
tcpdump 'icmp[icmptype] == icmp-echo'

# Ping 响应 (type 0)
tcpdump 'icmp[icmptype] == icmp-echoreply'

# 目标不可达 (type 3)
tcpdump 'icmp[icmptype] == icmp-unreach'

# TTL 超时 (type 11)
tcpdump 'icmp[icmptype] == 11'
```

5.5 ARP 过滤

```
# 所有 ARP
tcpdump arp

# ARP 请求
tcpdump 'arp[6:2] == 1'

# ARP 响应
tcpdump 'arp[6:2] == 2'
```

5.6 TLS/SSL 过滤

```
# TLS Client Hello
tcpdump 'tcp port 443 and (tcp[((tcp[12]&0xf0)>>2)]=0x16) and (tcp[((tcp[12]&0xf0)>>2)+5]=0x01)'

# 所有 TLS 握手
tcpdump 'tcp port 443 and (tcp[((tcp[12]&0xf0)>>2)]=0x16)'
```

---

6. 高级技巧

6.1 字节偏移过滤

```
# 语法：proto[offset:size]
# ip[0]      → IP 版本+首部长度
# ip[8]      → TTL
# ip[9]      → 协议号（6=TCP, 17=UDP, 1=ICMP）
# tcp[0:2]   → 源端口
# tcp[2:2]   → 目的端口

# TTL ≤ 5（可能 traceroute）
tcpdump 'ip[8] <= 5'

# IP 分片包
tcpdump '(ip[6:2] & 0x3fff) != 0'

# TCP 窗口为 0（零窗口）
tcpdump 'tcp[14:2] == 0'

# 载荷长度 > 0 的 TCP 包
tcpdump 'tcp and (ip[2:2] - ((ip[0]&0xf)<<2) - ((tcp[12]&0xf0)>>2)) > 0'
```

6.2 配合其他工具

```
# 实时抓包 → Wireshark 分析
tcpdump -i eth0 -s 0 -U -w - | wireshark -k -i -

# 实时统计连接数
tcpdump -i eth0 -nn -l | awk '{print $3}' | cut -d. -f1-4 | sort | uniq -c | sort -rn | head

# 提取 URL
tcpdump -i eth0 -A -s 0 -l port 80 | egrep -o 'GET [^ ]+'

# 配合 tshark 做统计
tcpdump -i eth0 -w - | tshark -r - -q -z io,stat,1

# 实时过滤发送到 syslog
tcpdump -i eth0 -l port 80 | logger -t tcpdump
```

6.3 性能优化

```
# 内核缓冲区（单位 KB，减少丢包）
tcpdump -i eth0 -B 4096

# 设置快照长度（只抓头部，提升性能）
tcpdump -i eth0 -s 96

# 使用 BPF（内核层过滤，比用户态快）
# tcpdump 默认就是 BPF，确保过滤条件尽量精确

# 禁用名称解析（减少 DNS 查询开销）
tcpdump -i eth0 -nn

# 直接写文件，不打印到终端
tcpdump -i eth0 -w /tmp/cap.pcap -q
```

---

7. 故障排查清单

7.1 网络不通 / 连接超时

```
# 1. 检查是否有 ARP 响应（二层可达）
tcpdump -i eth0 -nn arp and host <目标IP>

# 2. 检查 ICMP 是否可达
tcpdump -i eth0 icmp and host <目标IP>

# 3. 检查 TCP 三次握手是否完成
tcpdump -i eth0 -nn 'host <目标IP> and tcp[tcpflags] & (tcp-syn|tcp-ack) != 0'

# 4. 是否收到 RST（被拒绝）
tcpdump -i eth0 -nn 'host <目标IP> and tcp[tcpflags] & tcp-rst != 0'

# 5. 是否收到 ICMP 不可达
tcpdump -i eth0 'icmp[icmptype] == icmp-unreach and host <目标IP>'
```

排查流程：

```
ARP 无响应 → 检查二层（VLAN/交换机/MAC）
SYN 发出无 SYN-ACK → 检查路由/防火墙/目标服务
收到 RST → 目标端口未监听 或 防火墙拒绝
收到 ICMP Unreachable → 路由不可达 / 管理性禁止
```

7.2 连接建立慢 / DNS 问题

```
# 检查 DNS 查询和响应时间
tcpdump -i eth0 -nn -tttt port 53 and host <DNS服务器>

# 检查 DNS 响应码（NXDOMAIN/SERVFAIL）
tcpdump -i eth0 -vvv port 53 | grep -E 'NXDomain|ServFail|Refused'

# 检查 TCP 重传（可能丢包）
tcpdump -i eth0 -nn 'tcp[tcpflags] == tcp-syn' and host <目标>
# 如果出现多个 SYN 包 → 对端无响应（SYN 重传）
```

7.3 数据传输慢 / 丢包

```
# 检查 TCP 重传
tcpdump -i eth0 -nn host <目标IP> | grep -i 'retransmit\|dup ack'

# 零窗口（接收方缓冲区满）
tcpdump -i eth0 -nn 'host <目标IP> and tcp[14:2] == 0'

# TCP 窗口缩放 / 窗口变小
tcpdump -i eth0 -vv host <目标IP> | grep -i 'win'

# 检查 MTU 问题（ICMP Fragmentation Needed）
tcpdump -i eth0 'icmp[icmptype] == 3 and icmp[icmpcode] == 4'

# 大包是否被丢弃
tcpdump -i eth0 -nn 'host <目标IP> and greater 1400'
```

7.4 TLS/SSL 握手问题

```
# 查看完整的 TLS 握手过程
tcpdump -i eth0 -nn -vv 'host <目标> and tcp port 443 and (tcp[((tcp[12]&0xf0)>>2)]=0x16)'

# Client Hello 发出但无 Server Hello → 证书/协议不兼容
# 握手后立即 RST → 证书验证失败
# Alert 消息 → 解析 alert 级别和描述
```

7.5 防火墙 / NAT 排查

```
# 在两端同时抓包对比
# 源端:
tcpdump -i eth0 -nn host <目标IP> -w src_side.pcap
# 目的端:
tcpdump -i eth0 -nn host <源IP> -w dst_side.pcap

# 检查 NAT 后的真实源 IP
tcpdump -i eth0 -nn -e host <目标IP>

# SNAT 验证（出口）
tcpdump -i <出口网卡> -nn src host <内部IP>

# 逐跳排查（每个节点抓包）
```

7.6 应用层排查

```
# HTTP 状态码
tcpdump -i eth0 -A -s 0 -l port 80 | grep 'HTTP/1\.[01] [0-9]'

# MySQL 连接（默认 3306）
tcpdump -i eth0 -nn port 3306 -X | head -50

# Redis 命令
tcpdump -i eth0 -A port 6379

# SMTP 会话
tcpdump -i eth0 -A port 25
```

---

8. 实战场景速查

场景 1：找出谁在扫描我

```
tcpdump -i eth0 -nn 'tcp[tcpflags] == tcp-syn' | \
  awk '{print $3}' | cut -d. -f1-4 | sort | uniq -c | sort -rn | head -20
```

场景 2：检测 ARP 欺骗

```
# 监控 ARP 应答，检查 MAC 变化
tcpdump -i eth0 -nn arp | grep 'is-at'
```

场景 3：排查服务间调用

```
# 抓取两个服务之间的通信
tcpdump -i eth0 -nn -A \
  '(host 10.0.1.10 and host 10.0.1.20) and (port 8080 or port 8443)'
```

场景 4：抓取特定大小的包

```
# 小于 64 字节（可能异常）
tcpdump -i eth0 less 64

# 大于 1000 字节
tcpdump -i eth0 greater 1000

# 指定范围
tcpdump -i eth0 'greater 500 and less 1000'
```

场景 5：监控新建连接速率

```
# 每秒新建连接数
tcpdump -i eth0 -nn 'tcp[tcpflags] == tcp-syn' -l | \
  awk '{print substr($1,1,8)}' | uniq -c
```

场景 6：抓取非标准端口流量

```
# 排除常见端口，查看异常通信
tcpdump -i eth0 -nn 'tcp and not port 22 and not port 80 and not port 443 and not port 53'
```

场景 7：远程抓包（通过 SSH）

```
# 远程主机抓包，本地 Wireshark 实时分析
ssh user@remote 'sudo tcpdump -i eth0 -s 0 -U -w -' | wireshark -k -i -

# 保存到本地文件
ssh user@remote 'sudo tcpdump -i eth0 -s 0 -w - port 80' > remote_capture.pcap
```

附录：常用选项速查表

选项	说明
`-i <接口>`	指定网卡
`-n` / `-nn`	不解析主机名 / 不解析主机+端口
`-c <数量>`	抓包数量限制
`-s <字节>`	快照长度（0=完整包）
`-w <文件>`	保存为 pcap
`-r <文件>`	读取 pcap
`-C <MB>`	文件大小轮转
`-G <秒>`	时间轮转
`-W <数量>`	最大文件数
`-B <KB>`	内核缓冲区大小
`-A`	ASCII 显示载荷
`-X` / `-XX`	十六进制+ASCII
`-v` / `-vv` / `-vvv`	详细程度递增
`-e`	显示 MAC 地址
`-l`	行缓冲输出
`-t` / `-tttt`	省略时间 / 完整时间
`-S`	绝对序列号
`-q`	简要输出
`-D`	列出可用接口
`-z <用户>`	抓包后降权到指定用户
`--immediate-mode`	立即递送（低延迟）

附录：退出码

退出码	含义
0	正常结束
1	命令行错误
2	编译过滤表达式错误
4	权限不足（需要 root）

Tip: 生产环境抓包务必加 `-c` 或 `-G`/`-W` 限制，避免磁盘写满。