# TCPDump e WireShark

O objetivo deste compilado de informações sobre ferramentas de Análise de Pacotes é resumir a complexidade do assunto e fornecer os comandos mais comuns para orientação e consulta rápida de Administradores e Analistas de Redes. Vamos explorar o TCPDump e o WireShark

## TCPDump: A Ferramenta Essencial para Análise de Tráfego de Rede

No vasto e complexo universo das redes de computadores, a capacidade de "ver" o que está acontecendo em tempo real é um superpoder. Para administradores de rede, essa habilidade não é apenas desejável, mas fundamental para diagnosticar problemas, otimizar o desempenho e garantir a segurança. É aqui que entra o **TCPDump**, uma ferramenta de linha de comando que se tornou um verdadeiro canivete suíço para qualquer profissional de TI.

Neste artigo didático, vamos mergulhar no TCPDump, explorando desde seus conceitos básicos até comandos avançados e, o mais importante, como interpretar seus resultados para tomar decisões informadas e eficientes na sua rede.

## O Que é o TCPDump e Por Que Ele é Indispensável?

Imagine sua rede como uma rodovia movimentada, onde carros (pacotes de dados) trafegam constantemente. O TCPDump é como um observador privilegiado que pode parar na beira da estrada, inspecionar cada carro que passa e até mesmo filtrar quais carros ele quer ver, tudo isso sem interromper o fluxo do tráfego.

Em termos técnicos, o TCPDump é um **analisador de pacotes de rede** que roda na linha de comando. Ele permite que você intercepte e exiba pacotes TCP/IP e outros protocolos que estão sendo transmitidos ou recebidos por uma interface de rede. Sua principal vantagem é a capacidade de capturar dados diretamente da camada de enlace, oferecendo uma visão crua e detalhada do que está acontecendo.

**Por que ele é indispensável para administradores de rede?**

1. **Diagnóstico de Problemas**: É a primeira linha de defesa quando algo não funciona. Conectividade falha, latência alta, serviços inacessíveis – o TCPDump pode revelar a causa raiz.
2. **Segurança**: Identifique tráfego suspeito, tentativas de invasão, varreduras de portas ou comunicações não autorizadas.
3. **Monitoramento de Desempenho**: Entenda gargalos, retransmissões excessivas ou padrões de tráfego que afetam a performance da rede.
4. **Validação de Configurações**: Verifique se as regras de firewall, rotas ou configurações de serviços estão funcionando como esperado.
5. **Aprendizado e Compreensão**: A melhor forma de entender como os protocolos de rede realmente funcionam é vê-los em ação.

## Primeiros Passos: Instalação e Uso Básico

Antes de começar, você precisará ter o TCPDump instalado em seu sistema Linux ou macOS.

Mais no final deste artigo, vamos explicar como usar a ferramenta **WIRESHARK**, similar ao TCPDump, porém em ambiente Windows.

Na maioria das distribuições Linux, ele pode instalar o TCPDump facilmente:
Veja os comandos abaixo:

- **Debian/Ubuntu**: *sudo apt update && sudo apt install tcpdump*
- **CentOS/RHEL/Fedora**: *sudo yum install tcpdump ou sudo dnf install tcpdump*
- **macOS**: Geralmente vem pré-instalado.

Com o TCPDump pronto, vamos aos comandos básicos.

### *1. Captura Simples*

O comando mais simples é apenas tcpdump. No entanto, ele requer privilégios de superusuário para acessar as interfaces de rede:

***sudo tcpdump***

Ao executar este comando, você verá um fluxo contínuo de pacotes passando pela sua interface de rede padrão. A saída pode ser avassaladora, mas é um bom ponto de partida. Cada linha representa um pacote.

**Exemplo de Saída Padrão:**

*14:30:01.123456 IP 192.168.1.10.54321 > 172.217.160.142.443: Flags [S], seq 12345, win 65535, options [mss 1460,sackOK,TS val 12345678 ecr 0], length 0 14:30:01.123500 IP 172.217.160.142.443 > 192.168.1.10.54321: Flags [S.], seq 67890, ack 12346, win 65535, options [mss 1460,sackOK,TS val 87654321 ecr 12345678], length 0*

### *2. Especificando a Interface*

Se você tem múltiplas interfaces de rede (Ethernet, Wi-Fi, loopback), é crucial especificar qual delas você quer monitorar. Use a opção *-i* (interface).

Para listar as interfaces disponíveis, use ***tcpdump -D*** ou ***ip a***.

Para uma interface Ethernet ***sudo tcpdump -i eth0***

Para uma interface Wi-Fi ***sudo tcpdump -i wlan0***

Para a interface de loopback ***sudo tcpdump -i lo***

### 3. Controlando a Resolução de Nomes

Por padrão, o TCPDump tenta resolver endereços IP para nomes de host e números de porta para nomes de serviço (ex: 80 para http). Isso pode adicionar latência à captura e nem sempre é desejável.

- -**n**: Não resolve nomes de host (apenas IPs).
- -**nn**: Não resolve nomes de host nem nomes de porta (apenas IPs e números de porta).

***sudo tcpdump -i eth0 -n*** Mostra IPs, mas resolve portas (ex: http)
***sudo tcpdump -i eth0 -nn*** Mostra IPs e números de porta (ex: 80)

Para um administrador de rede, ***-nn*** é frequentemente a melhor opção para uma análise rápida e sem atrasos.

### 4. Níveis de Detalhamento (Verbosity)

Para obter mais informações sobre os pacotes, use a opção ***-v*** (verbose).
Você pode aumentar o nível de detalhamento com ***-vv*** e ***-vvv***.

- -***v***: Mostra mais detalhes, como o TTL (Time To Live), ID do pacote, opções TCP, etc.
- **-vv**: Ainda mais detalhes, incluindo campos adicionais de protocolo.
- **-vvv**: O nível máximo de detalhamento, útil para depuração profunda.

Exemplo: ***sudo tcpdump -i eth0 -nn -v***

### 5. Capturando um Número Limitado de Pacotes

Para evitar um fluxo interminável de dados, você pode especificar quantos pacotes deseja capturar com a opção ***-c*** (count).

***sudo tcpdump -i eth0 -nn -c 10*** Captura os primeiros 10 pacotes e para

## Filtrando o Tráfego: Encontrando o Que Realmente Importa

A verdadeira força do TCPDump reside em sua poderosa linguagem de filtragem, que permite isolar exatamente o tráfego que você deseja analisar. Isso é crucial para evitar a sobrecarga de informações.

Os filtros são adicionados como argumentos após as opções do TCPDump.

## 1. Filtros Baseados em Host

Monitore o tráfego de ou para um endereço IP específico.

- **host** <ip_address>: Tráfego de ou para o host.
- **src** host <ip_address>: Tráfego originado do host.
- **dst** host <ip_address>: Tráfego destinado ao host.

**Exemplos:**

***sudo tcpdump -i eth0 -nn host 192.168.1.100*** Todo o tráfego envolvendo 192.168.1.100
***sudo tcpdump -i eth0 -nn src host 10.0.0.5*** Tráfego que sai de 10.0.0.5
***sudo tcpdump -i eth0 -nn dst host 172.16.0.20*** Tráfego que chega em 172.16.0.20

## 2. Filtros Baseados em Porta

Monitore o tráfego de ou para uma porta específica.

- ***port*** <port_number>: Tráfego de ou para a porta.
- ***src port*** <port_number>: Tráfego originado da porta.
- ***dst port*** <port_number>: Tráfego destinado à porta.

**Exemplos:**

***sudo tcpdump -i eth0 -nn port 80*** Tráfego HTTP (porta 80)
***sudo tcpdump -i eth0 -nn dst port 22*** Tráfego SSH (porta 22) chegando
***sudo tcpdump -i eth0 -nn src port 53*** Tráfego DNS (porta 53) saindo

## 3. Filtros Baseados em Protocolo

Monitore tipos específicos de protocolo.

- **tcp**: Apenas pacotes TCP.
- **udp**: Apenas pacotes UDP.
- **icmp**: Apenas pacotes ICMP (usado por ping e traceroute).
- **arp**: Apenas pacotes ARP (Address Resolution Protocol).
- **ip**: Apenas pacotes IP.

**Exemplos:**

***sudo tcpdump -i eth0 -nn icmp*** Apenas pings e respostas
***sudo tcpdump -i eth0 -nn arp*** Apenas requisições e respostas ARP
***sudo tcpdump -i eth0 -nn udp port 53*** Apenas consultas DNS via UDP

## 4. Filtros Baseados em Rede

Monitore o tráfego de ou para uma sub-rede inteira.

- **net** <network_address>/<cidr>: Tráfego de ou para a rede.
- **src net** <network_address>/<cidr>: Tráfego originado da rede.
- **dst net** <network_address>/<cidr>: Tráfego destinado à rede.

**Exemplos:**

***sudo tcpdump -i eth0 -nn net 192.168.1.0/24*** Todo o tráfego da rede 192.168.1.0/24
***sudo tcpdump -i eth0 -nn dst net 10.0.0.0/8*** Tráfego chegando em qualquer host da rede 10.0.0.0/8

## 5. Combinando Filtros (Operadores Lógicos)

Você pode combinar filtros usando operadores lógicos: and, or, not.

- **and** (ou &&): Ambos os critérios devem ser verdadeiros.
- **or** (ou ||): Pelo menos um dos critérios deve ser verdadeiro.
- **not** (ou !) : O critério não deve ser verdadeiro.

**Exemplos:**

Tráfego TCP para a porta 80 do host 192.168.1.100
***sudo tcpdump -i eth0 -nn tcp and host 192.168.1.100 and port 80***

Tráfego de ou para 192.168.1.100, mas que não seja SSH (porta 22)
***sudo tcpdump -i eth0 -nn host 192.168.1.100 and not port 22***

Tráfego DNS (porta 53) ou HTTP (porta 80)
***sudo tcpdump -i eth0 -nn 'port 53 or port 80'*** Use aspas para agrupar

## 6. Filtros de Tamanho de Pacote

Útil para identificar pacotes fragmentados ou muito grandes/pequenos.

- **less** <bytes>: Pacotes menores que o tamanho especificado.
- **greater** <bytes>: Pacotes maiores que o tamanho especificado.

***sudo tcpdump -i eth0 -nn greater 1500*** Pacotes maiores que o MTU padrão

## Entendendo a Saída do TCPDump: O Que Cada Campo Significa

A saída do TCPDump, especialmente com filtros, começa a fazer sentido quando você entende a estrutura. Vamos analisar uma linha típica:

***14:30:01.123456 IP 192.168.1.10.54321 > 172.217.160.142.443: Flags [S], seq 12345, win 65535, options [mss 1460,sackOK,TS val 12345678 ecr 0], length 0***

- 14:30:01.123456: **Timestamp**. A hora exata em que o pacote foi capturado. Fundamental para correlacionar eventos.
- IP: **Protocolo da Camada de Rede**. Neste caso, Internet Protocol. Poderia ser ARP, IP6, etc.
- 192.168.1.10.54321: **Endereço IP de Origem e Porta de Origem**. O host 192.168.1.10 usando a porta efêmera 54321.
- >: Indica a **direção** do tráfego (origem para destino).
- 172.217.160.142.443: **Endereço IP de Destino e Porta de Destino**. O host 172.217.160.142 (provavelmente um servidor Google) na porta 443 (HTTPS).
- Flags [S]: **Flags TCP**. [S] significa SYN (Synchronize), indicando o início de uma conexão TCP.
  Outras flags comuns incluem:
  - [.]: ACK (Acknowledgement)
  - [P]: PSH (Push)
  - [F]: FIN (Finish)
  - [R]: RST (Reset)
  - [U]: URG (Urgent)
  - [S.] ou [SA]: SYN-ACK (resposta ao SYN)
- seq 12345: **Número de Sequência TCP**.
  Indica o número de sequência do primeiro byte de dados neste pacote.
- ack 12346: **Número de Reconhecimento TCP**.
  Indica o próximo número de sequência que o remetente espera receber do receptor.
- win 65535: **Tamanho da Janela TCP**.
  Indica a quantidade de dados que o receptor está disposto a aceitar.
- options [...]: **Opções TCP**.
  Informações adicionais como MSS (Maximum Segment Size), SACK (Selective Acknowledgement), Timestamps.
- length 0: **Comprimento dos Dados**.
  O tamanho da carga útil do pacote (excluindo cabeçalhos).
  Um length 0 em um SYN é normal, pois ele não carrega dados de aplicação.

**Exemplo de Saída UDP (DNS):**

***14:30:02.500000 IP 192.168.1.10.12345 > 8.8.8.8.53: 12345+ A? www.google.com. (32)***

- **IP**: Protocolo IP.
- **192.168.1.10.12345**: Origem (IP e porta efêmera).
- **8.8.8.8.53**: Destino (servidor DNS do Google na porta 53).
- **12345+ A?** www.google.com. (32): Detalhes da consulta DNS. 12345 é o ID da consulta, A? é o tipo de registro (endereço IPv4), www.google.com. é o domínio consultado, e (32) é o tamanho da consulta.

**Exemplo de Saída ICMP (Ping):**

***14:30:03.789012 IP 192.168.1.10 > 8.8.8.8: ICMP echo request, id 1, seq 1, length 64***
***14:30:03.800000 IP 8.8.8.8 > 192.168.1.10: ICMP echo reply, id 1, seq 1, length 64***

- **ICMP echo request**: Requisição de ping.
- **ICMP echo reply**: Resposta ao ping.
- **id 1, seq 1**: Identificador e número de sequência do ping, úteis para correlacionar requisições e respostas.
- **length 64**: Tamanho da carga útil do ICMP.

## Casos de Uso Práticos para Administradores de Rede

Agora que entendemos os comandos e a saída, vamos aplicar o TCPDump em cenários reais de administração de rede.

### *1. Diagnóstico de Conectividade*

**Problema**: Um usuário relata que não consegue acessar um servidor web interno (192.168.1.50).

**Ação**: Verifique se os pacotes HTTP estão chegando e sendo respondidos.

***sudo tcpdump -i eth0 -nn host 192.168.1.50 and port 80***

**O que procurar**:

- **Pacotes SYN de origem para destino**: O cliente está tentando iniciar a conexão?
- **Pacotes SYN-ACK de destino para origem**: O servidor está respondendo ao pedido de conexão?
- **Pacotes ACK de origem para destino**: O cliente está confirmando o SYN-ACK?
- **Pacotes RST**: Se você vê muitos [R], a conexão está sendo resetada, o que pode indicar um firewall bloqueando ou um serviço não rodando.

- **Nenhuma saída**: Se não há tráfego, o problema pode ser na camada 2 (ARP, switch), roteamento ou firewall antes do servidor.

**Problema**: O DNS parece estar lento ou não funcionando.

**Ação**: Monitore as consultas e respostas DNS.

***sudo tcpdump -i eth0 -nn port 53***

**O que procurar**:

- **Consultas UDP de origem para o servidor DNS (porta 53)**: O cliente está enviando as consultas?
- **Respostas UDP do servidor DNS para origem**: O servidor DNS está respondendo?
- **Tempo entre consulta e resposta**: A latência é aceitável?
- **Respostas com** ou SERVFAIL: O servidor DNS está indicando que o domínio não existe ou que houve um erro.

## *2. Identificação de Tráfego Suspeito/Segurança*

**Problema**: Suspeita de varredura de portas na rede.

**Ação**: Procure por muitos pacotes SYN sem as respostas correspondentes, vindo de um único IP para várias portas.

***sudo tcpdump -i eth0 -nn 'tcp[tcpflags] & (tcp-syn) != 0 and tcp[tcpflags] & (tcp-ack) == 0'***

**O que procurar**:

- Um único IP de origem enviando muitos pacotes [S] para diferentes IPs e portas de destino. Isso pode indicar uma varredura de portas.
- Pacotes com flags TCP incomuns ou combinações estranhas.

**Problema**: Identificar tráfego não autorizado ou de um protocolo inesperado.

**Ação**: Filtre por protocolos ou portas que não deveriam estar ativos.

# Exemplo: Procurar por tráfego Telnet (porta 23) em uma rede que só deveria usar SSH
***sudo tcpdump -i eth0 -nn port 23***

# Exemplo: Procurar por tráfego de um host específico que não deveria estar se comunicando
***sudo tcpdump -i eth0 -nn host 192.168.1.200 and not host <seu_gateway>***

**O que procurar**:

- Conexões para portas não padrão.
- Comunicações entre hosts que não deveriam interagir.

- Tráfego de broadcast ou multicast excessivo que pode indicar problemas de configuração ou ataques.

## 3. Monitoramento de Desempenho

**Problema**: Aplicação lenta, suspeita de problemas de rede.

**Ação**: Capture o tráfego entre o cliente e o servidor da aplicação.

***sudo tcpdump -i eth0 -nn host <ip_cliente> and host <ip_servidor> and port <porta_aplicacao>***

**O que procurar**:

- **Retransmissões TCP**: Pacotes com o mesmo número de sequência sendo enviados várias vezes. Isso indica perda de pacotes na rede.
- **Janelas TCP pequenas**: Se o win (window size) estiver consistentemente baixo, o receptor pode estar sobrecarregado ou o buffer TCP está pequeno.
- **Latência**: Observe os timestamps entre o envio de um pacote e o ACK correspondente. Um atraso significativo pode indicar latência da rede ou do servidor.
- **Comprimento dos pacotes**: Pacotes muito pequenos podem indicar overhead excessivo para pouca carga útil.

## 4. Análise de Protocolos Específicos

**DHCP**: Para verificar se um cliente está recebendo um endereço IP corretamente.

***sudo tcpdump -i eth0 -nn port 67 or port 68***

**O que procurar**:

- **DHCP Discover** (cliente -> broadcast)
- **DHCP Offer** (servidor -> cliente)
- **DHCP Request** (cliente -> servidor)
- **DHCP ACK** (servidor -> cliente)
- A ausência de qualquer uma dessas etapas indica um problema no processo de atribuição de IP.

**ARP**: Para diagnosticar problemas de resolução de endereços na camada 2.

***sudo tcpdump -i eth0 -nn arp***

**O que procurar**:

- **ARP Request**: Um host perguntando "Quem tem o IP X.X.X.X? Diga ao Y.Y.Y.Y".

- **ARP Reply**: A resposta "Eu sou o X.X.X.X e meu MAC é ZZZZ".
- **ARP Gratuitous**: Anúncios de um host sobre seu próprio IP/MAC.
- ARP excessivo pode indicar problemas de segurança (ARP spoofing) ou configuração de rede.

## Salvando e Analisando Capturas

Para análises mais aprofundadas ou para compartilhar com colegas, é essencial salvar a captura de pacotes em um arquivo.

### *1. Salvando para um Arquivo*

Use a opção -**w** (write) para salvar os pacotes em um arquivo no formato **.pcap** (Packet Capture).

***sudo tcpdump -i eth0 -nn -s 0 -w minha_captura.pcap -c 1000***

- **-s 0**: Define o "snapshot length" para 0, o que significa capturar o pacote inteiro. Isso é crucial para análises detalhadas, pois o padrão do TCPDump pode truncar pacotes.
- **-w** minha_captura.pcap: O nome do arquivo onde a captura será salva.
- **-c 1000**: Captura 1000 pacotes e então para. Sempre use -c ou -G (tempo) para evitar arquivos de captura gigantescos.

### *2. Lendo de um Arquivo*

Para analisar um arquivo **.pcap** salvo anteriormente, use a opção **-r** (read).

***sudo tcpdump -r minha_captura.pcap -nn***

Você pode aplicar os mesmos filtros ao ler de um arquivo, o que é extremamente útil para focar em partes específicas de uma captura grande.

***sudo tcpdump -r minha_captura.pcap -nn host 192.168.1.100 and port 80***

### *3. Análise com Ferramentas Gráficas*

Arquivos .pcap são o padrão da indústria e podem ser abertos por ferramentas gráficas como o **Wireshark**.

O Wireshark oferece uma interface muito mais amigável para navegar, filtrar e analisar pacotes, com decodificação de protocolos e gráficos estatísticos. A combinação do TCPDump para captura rápida no servidor e o Wireshark para análise detalhada no desktop é uma prática comum e poderosa.

## Dicas Avançadas e Boas Práticas para Administradores

- **Permissões**: Lembre-se sempre de que o TCPDump precisa de privilégios de root (**sudo**) para funcionar.
- **Tamanho do Snapshot**: Sempre use **-s** 0 ao salvar capturas para análise posterior, a menos que você tenha um motivo muito específico para truncar os pacotes. Pacotes truncados podem omitir informações cruciais.
- **Modo Promíscuo**: Por padrão, o TCPDump geralmente opera em modo promíscuo (captura todos os pacotes que vê na interface, mesmo os não destinados a ele). Se você quiser garantir que ele *não* esteja em modo promíscuo (apenas pacotes para o seu host), use **-p**.
  No entanto, para a maioria das análises de rede, o modo promíscuo é desejado.
- **Captura em Segundo Plano**: Para capturas de longa duração, use ferramentas como nohup ou screen/tmux para que o TCPDump continue rodando mesmo se sua sessão SSH for desconectada.
- **Cuidado com o Volume de Dados**: Capturar todo o tráfego em uma rede movimentada pode gerar arquivos gigantescos rapidamente e consumir recursos do sistema. Use filtros agressivamente e limite o tempo ou o número de pacotes.
- **Ética e Privacidade**: A capacidade de inspecionar o tráfego de rede é poderosa e vem com grande responsabilidade. Sempre obtenha permissão antes de capturar tráfego em redes que você não possui ou gerencia.
  Esteja ciente das leis de privacidade e segurança de dados.
- **Expressões Regulares**: O TCPDump suporta expressões regulares mais complexas para filtros, permitindo uma granularidade ainda maior. Por exemplo, você pode filtrar por conteúdo dentro do pacote, embora isso seja mais avançado e consuma mais recursos.

# TCPdump - Instalação e Uso Básico

### *1. Instalação (Exemplo para Debian/Ubuntu)*

***sudo apt update && sudo apt install tcpdump***

### *2. Captura Simples*

***sudo tcpdump***

### *3. Especificando a Interface (Listar interfaces)*

***tcpdump -D # ou ip a***

### *4. Especificando a Interface (Capturar em eth0)*

***sudo tcpdump -i eth0***

### 5. Controlando a Resolução de Nomes (Apenas IPs, resolve portas)

**sudo tcpdump -i eth0 -n**

### 6. Controlando a Resolução de Nomes (Apenas IPs e números de porta)

**sudo tcpdump -i eth0 -nn**

### 7. Níveis de Detalhamento (Verbose)

**sudo tcpdump -i eth0 -nn -v**

### 8. Capturando um Número Limitado de Pacotes

**sudo tcpdump -i eth0 -nn -c 10**

## Filtrando o Tráfego

### 1. Filtros Baseados em Host (Tráfego de/para um host)

**sudo tcpdump -i eth0 -nn host 192.168.1.100**

### 2. Filtros Baseados em Host (Tráfego originado de um host)

**sudo tcpdump -i eth0 -nn src host 10.0.0.5**

### 3. Filtros Baseados em Host (Tráfego destinado a um host)

**sudo tcpdump -i eth0 -nn dst host 172.16.0.20**

### 4. Filtros Baseados em Porta (Tráfego de/para a porta 80)

**sudo tcpdump -i eth0 -nn port 80**

### 5. Filtros Baseados em Porta (Tráfego SSH chegando)

**sudo tcpdump -i eth0 -nn dst port 22**

### 6. Filtros Baseados em Protocolo (Apenas ICMP)

**sudo tcpdump -i eth0 -nn icmp**

### 7. Filtros Baseados em Protocolo (Apenas ARP)

**sudo tcpdump -i eth0 -nn arp**

### 8. Filtros Baseados em Protocolo (Apenas consultas DNS via UDP)

**sudo tcpdump -i eth0 -nn udp port 53**

### 9. Filtros Baseados em Rede (Tráfego de/para a rede 192.168.1.0/24)

**sudo tcpdump -i eth0 -nn net 192.168.1.0/24**

### 10. Filtros Baseados em Rede (Tráfego chegando em qualquer host da rede 10.0.0.0/8)

**sudo tcpdump -i eth0 -nn dst net 10.0.0.0/8**

### 11. Combinando Filtros (TCP para porta 80 do host 192.168.1.100)

**sudo tcpdump -i eth0 -nn tcp and host 192.168.1.100 and port 80**

### 12. Combinando Filtros (Tráfego de/para 192.168.1.100, exceto SSH)

**sudo tcpdump -i eth0 -nn host 192.168.1.100 and not port 22**

### 13. Combinando Filtros (Tráfego DNS ou HTTP)

**sudo tcpdump -i eth0 -nn 'port 53 or port 80'**

### 14. Filtros de Tamanho de Pacote (Pacotes maiores que 1500 bytes)

**sudo tcpdump -i eth0 -nn greater 1500**

## Casos de Uso Práticos

### 1. Diagnóstico de Conectividade (Tráfego HTTP para um servidor)

**sudo tcpdump -i eth0 -nn host 192.168.1.50 and port 80**

### 2. Diagnóstico de Conectividade (Monitorar consultas e respostas DNS)

**sudo tcpdump -i eth0 -nn port 53**

### 3. Identificação de Tráfego Suspeito (Varredura de portas - SYN sem ACK)

**sudo tcpdump -i eth0 -nn 'tcp[tcpflags] & (tcp-syn) != 0 and tcp[tcpflags] & (tcp-ack) == 0'**

### 4. Identificação de Tráfego Suspeito (Procurar Telnet em rede que só deveria usar SSH)

**sudo tcpdump -i eth0 -nn port 23**

### 5. Monitoramento de Desempenho (Tráfego entre cliente e servidor de aplicação)

***sudo tcpdump -i eth0 -nn host <ip_cliente> and host <ip_servidor> and port <porta_aplicacao>***

### *6. Análise de Protocolos Específicos (DHCP)*

***sudo tcpdump -i eth0 -nn port 67 or port 68***

### *7. Análise de Protocolos Específicos (ARP)*

***sudo tcpdump -i eth0 -nn arp***

## Salvando e Analisando Capturas

### *1. Salvando para um Arquivo (Captura de 1000 pacotes)*

***sudo tcpdump -i eth0 -nn -s 0 -w minha_captura.pcap -c 1000***

### *2. Lendo de um Arquivo*

***sudo tcpdump -r minha_captura.pcap -nn***

### *3. Lendo de um Arquivo com Filtro*

***sudo tcpdump -r minha_captura.pcap -nn host 192.168.1.100 and port 80***

## Dicas Avançadas (Captura em Segundo Plano com `nohup`)

***nohup sudo tcpdump -i eth0 -nn -s 0 -w long_capture.pcap host 192.168.1.100 &***

Espero que esta apresentação dos comandos em "telas" facilite a sua compreensão e aplicação prática do TCPDump em ambiente Linux.

# Analisando no Windows

## 1. Alternativa Mais Comum e Poderosa: Wireshark (com Npcap)

A ferramenta mais popular e robusta para análise de tráfego de rede no Windows (e em outras plataformas) é o **Wireshark**.
Ele é um analisador de protocolo de rede gráfico que oferece uma interface muito mais rica e amigável para capturar e analisar pacotes do que o tcpdump na linha de comando.

**Como funciona:**

1. **Instalação do Wireshark**: Você baixa e instala o Wireshark do site oficial: https://www.wireshark.org/#download

2. **Npcap**: Durante a instalação do Wireshark, ele geralmente inclui e instala o **Npcap** (que é o sucessor do WinPcap).
   Npcap é a biblioteca de captura de pacotes para Windows que permite ao Wireshark (e outras ferramentas) interceptar o tráfego de rede diretamente da interface.

3. **Captura e Análise**: Uma vez instalado, você abre o Wireshark, seleciona a interface de rede que deseja monitorar e clica em "Start capturing packets".

**Vantagens do Wireshark:**

- **Interface Gráfica**: Muito mais fácil de visualizar, filtrar e navegar pelos pacotes.
- **Decodificação de Protocolos**: Decodifica centenas de protocolos, mostrando os campos de cada cabeçalho de forma legível.
- **Filtros Avançados**: Possui uma linguagem de filtro poderosa (display filters e capture filters) que é, em muitos aspectos, mais intuitiva que a do tcpdump para iniciantes.
- **Estatísticas e Gráficos**: Oferece diversas ferramentas para análise estatística e visualização gráfica do tráfego.
- **Salvar/Abrir Arquivos PCAP**: Salva as capturas no formato .pcap (o mesmo do tcpdump), permitindo que você as abra e analise posteriormente.

**Como o Wireshark se relaciona**:

Embora o Wireshark seja gráfico, ele usa a mesma lógica de captura de pacotes que o tcpdump. Na verdade, os filtros de captura do Wireshark são baseados na mesma sintaxe de filtro do tcpdump (chamada de BPF - Berkeley Packet Filter).
Então, muitos dos filtros que você aprendeu para tcpdump podem ser adaptados para os filtros de captura do Wireshark.

## 2. Usando o `tcpdump` via WSL (Windows Subsystem for Linux)

Se você realmente precisa usar o comando tcpdump especificamente, a melhor maneira no Windows é através do **Windows Subsystem for Linux (WSL)**.
O WSL permite que você execute um ambiente Linux completo (como Ubuntu, Debian, etc.) diretamente no Windows, sem a necessidade de uma máquina virtual separada.

**Como funciona:**

1. **Instalar WSL**: Siga as instruções da Microsoft para instalar o WSL e uma distribuição Linux de sua escolha (ex: Ubuntu): https://learn.microsoft.com/pt-br/windows/wsl/
2. **Instalar** no WSL: Abra o terminal WSL e instale o tcpdump como faria em qualquer sistema Linux:
3. **Recomendação para WSL**: Use o WSL se você precisa *praticar* comandos tcpdump em um ambiente Linux, mas para *análise real* do tráfego da sua máquina Windows, o Wireshark é a escolha mais prática e eficiente.

## 3. Ferramentas de Linha de Comando Similares (Menos Comuns)

Existem algumas ferramentas de linha de comando para Windows que tentam replicar a funcionalidade do tcpdump, mas são menos conhecidas e mantidas do que o Wireshark. Um exemplo é o **WinDump**, que é uma porta do tcpdump para Windows, mas sua manutenção é menos ativa e ele ainda depende do WinPcap (a versão mais antiga do Npcap).

**Recomendação**: Evite WinDump e similares, pois o Wireshark (com Npcap) ou WSL oferecem soluções mais modernas e bem suportadas.

## Comandos Básicos do Wireshark

O Wireshark é a ferramenta de análise de protocolo de rede mais popular do mundo. Para o usuário iniciante, os "comandos" mais importantes não são comandos de terminal, mas sim as expressões de filtro e os atalhos de teclado que permitem isolar e analisar o tráfego de rede relevante.

Existem dois tipos principais de filtros no Wireshark, que usam sintaxes diferentes:

1.**Filtros de Captura (Capture Filters)**: Aplicados antes da captura, limitando quais pacotes são salvos no arquivo. Usam a sintaxe libpcap (a mesma do tcpdump).

2.**Filtros de Exibição (Display Filters):** Aplicados após a captura, limitando quais pacotes são exibidos na tela. São mais poderosos e flexíveis.

## 1. Filtros de Captura (Capture Filters)

Os Filtros de Captura são essenciais para reduzir o tamanho do arquivo de captura (pcap) e o consumo de recursos, pois descartam o tráfego indesejado antes que ele seja processado pelo Wireshark.

## 2. Filtros de Exibição (Display Filters)

Os Filtros de Exibição são usados na barra de filtro principal do Wireshark para refinar a visualização dos pacotes já capturados. Eles são muito mais expressivos e usam nomes de campos de protocolo.

Com esses filtros e atalhos, você terá o controle necessário para realizar as análises básicas e avançadas no Wireshark.

### Resumo para Administradores de Redes no Windows:

- **Para a maioria dos casos de uso e para uma análise profunda e visual**: Use o **Wireshark**.
  Ele é a ferramenta padrão da indústria e oferece tudo o que você precisa e mais.

- **Se você precisa da sintaxe e da experiência de linha de comandos** para fins de aprendizado ou scripts em um ambiente Linux no Windows: Instale o **WSL** e o tcpdump dentro dele. Tenha em mente as limitações de captura de interfaces físicas diretamente do WSL.

Em ambientes profissionais Windows, o Wireshark é a ferramenta de escolha para análise de pacotes. Ele é a evolução natural do conceito de tcpdump para uma plataforma gráfica, mantendo a mesma capacidade de captura de baixo nível.

| | Categoria | Comando/Sintaxe/Atalho | Descrição/Exemplo |
|---|---|---|---|
| **Filtro de Captura** | Protocolo | tcp | Captura apenas pacotes TCP. |
| | Protocolo | udp | Captura apenas pacotes UDP. |
| | Protocolo | icmp | Captura apenas pacotes ICMP (ping). |
| | Host (IP) | host 192.168.1.100 | Captura tráfego de ou para o IP 192.168.1.100. |
| | Host (IP) | src host 10.0.0.5 | Captura tráfego *originado* do IP 10.0.0.5. |
| | Host (IP) | dst host 172.16.0.1 | Captura tráfego *destinado* ao IP 172.16.0.1. |
| | Porta | port 80 | Captura tráfego de ou para a porta 80 (HTTP). |
| | Porta | tcp port 22 | Captura apenas tráfego TCP na porta 22 (SSH). |
| | Porta | udp port 53 | Captura apenas tráfego UDP na porta 53 (DNS). |
| | Combinação | host 192.168.1.100 and port 443 | Captura tráfego de ou para o host 192.168.1.100 na porta 443. |
| | Combinação | not arp and not icmp | Captura todo o tráfego, exceto ARP e ICMP. |
| | Rede | net 192.168.1.0/24 | Captura tráfego de ou para a rede 192.168.1.0/24. |
| **Filtro de Exibição** | Protocolo | http | Exibe apenas pacotes HTTP. |
| | Protocolo | dns | Exibe apenas pacotes DNS. |
| | Protocolo | tcp | Exibe apenas pacotes TCP. |
| | Endereço IP | ip.addr == 192.168.1.1 | Exibe pacotes onde o IP de origem *ou* destino é 192.168.1.1. |
| | Endereço IP | ip.src == 10.0.0.5 | Exibe pacotes onde o IP de origem é 10.0.0.5. |
| | Endereço IP | ip.dst == 172.16.0.1 | Exibe pacotes onde o IP de destino é 172.16.0.1. |
| | Porta | tcp.port == 80 | Exibe pacotes TCP de ou para a porta 80. |
| | Porta | udp.srcport == 53 | Exibe pacotes UDP onde a porta de origem é 53. |
| | Operadores Lógicos | && (AND) | Exemplo: http.request and ip.addr == 192.168.1.10 |
| | Operadores Lógicos | \|\| (OR) | Exemplo: tcp.port == 80 or tcp.port == 443 |
| | Operadores Lógicos | ! (NOT) | Exemplo: !arp |
| | Conteúdo Específico | http.request.method == "GET" | Exibe apenas requisições HTTP GET. |
| | Conteúdo Específico | tcp.flags.syn == 1 and tcp.flags.ack == 0 | Exibe apenas pacotes SYN (início de conexão TCP). |
| | Conteúdo Específico | contains "senha" | Exibe pacotes que contêm a string "senha" em qualquer lugar. |
| **Atalho de Teclado** | Captura | Ctrl + K | Abre a caixa de diálogo de Opções de Captura. |
| | Captura | Ctrl + E | Inicia/Para a captura de pacotes. |
| | Captura | Ctrl + D | Para a captura e exibe os resultados. |
| | Navegação | Ctrl + N | Vai para o próximo pacote. |
| | Navegação | Ctrl + P | Vai para o pacote anterior. |
| | Navegação | Ctrl + G | Vai para um pacote específico (por número). |
| | Navegação | Ctrl + F | Abre a caixa de diálogo de busca de pacotes. |
| | Visualização | Ctrl + , | Move para o próximo pacote marcado. |
| | Visualização | Ctrl + . | Move para o pacote anterior marcado. |
| | Visualização | Ctrl + M | Marca/Desmarca o pacote selecionado. |
| | Visualização | Ctrl + / | Define o pacote selecionado como o primeiro de uma conversa. |
| | Arquivo | Ctrl + O | Abre um arquivo de captura (pcap) existente. |
| | Arquivo | Ctrl + S | Salva o arquivo de captura atual. |