1. Introdução, terminologia e convenções

► O `tcpdump` é o melhor analisador de tráfego em modo texto que existe. Ele é baseado na `libpcap`, uma poderosa API para a captura de pacotes de rede durante o seu tráfego. Assim, o `tcpdump` mostra as conexões estabelecidas e o tráfego correspondente.

► O `tcpdump` está disponível para os Unix like, como GNU/Linux, BSD, OS X, Solaris etc.

► O `WinDump` é um port do `tcpdump` para o MS Windows. Assim, é idêntico ao `tcpdump`.

► Deste ponto em diante, a palavra `tcpdump` será utilizada como referência tanto para o `tcpdump` quanto para o `WinDump`.

2. Principais chaves do tcpdump

Chave	Função
`-D`	Mostra as interfaces de rede disponíveis.
`-i` *`iface`*	Determina qual interface de rede deverá ser utilizada. Caso nenhuma seja especificada, a primeira mostrada pela chave `-D` será utilizada. É possível utilizar qualquer uma mostrada pela chave `-D`, podendo citá-la pelo nome ou pelo número. Para escutar em todas as interfaces, utilize `any` como `iface`.
`-n`	Não faz resolução de nomes de hosts e nem de portas, acelerando a exibição dos resultados na tela (tempo real). É aconselhável sempre utilizar `-n` nas análises de tráfego.
`-N`	Ao resolver nomes, não mostra o domínio do host.
`-A`	Mostra cabeçalho e payload dos pacotes em ASCII.
`-X`	Idem, mas em hexadecimal e caracteres ASCII.
`-x`	Idem, mas somente em sequências em hexadecimal.
`-v`	Aumenta a quantidade de informações extraídas do cabeçalho do pacote.

2. Principais chaves do tcpdump (continuação)

Chave	Função
`-v`	Aumenta a quantidade de informações extraídas do cabeçalho do pacote.
`-vv`	Idem ao anterior, com mais informações ainda.
`-vvv`	Idem ao anterior, com mais informações.
`-w` *`arq`*	Grava o resultado da captura em um arquivo. É importante ressaltar que se nenhuma outra chave ou expressão de filtragem for utilizada, todo o tráfego passante será gravado. É aconselhável utilizar as chaves `-nv` para acelerar a gravação, por não resolver nomes, e para mostrar detalhes da captura em andamento.
`-r` *`arq`*	Lê um arquivo previamente gravado com `-w`. Diversas chaves poderão ser utilizadas para depurar o resultado.
`-t`	Não mostra a data e a hora na tela.
`-tttt`	Mostra a data e a hora utilizando o padrão `yyyy-mm-dd hh:mm:ss.ssssss`.
`-e`	Mostra também os dados referentes à camada 2 do Modelo OSI (enlace).
`-S`	Exibe os resultados TCP utilizando a sua sequência absoluta, em vez da sequência relativa. Recomendado na análise de sequências TCP.

As chaves mostradas são as principais. Há muitas outras disponíveis, que poderão ser vistas na manpage (`$ man tcpdump`) ou em `http://www.tcpdump.org/tcpdump_man.html`.

3. Expressões de filtragem

O `tcpdump`, por estar baseado na libpcap, utiliza as expressões de filtragem fornecidas por esta. Tais expressões poderão ser vistas no manual on-line da library (`$ man pcap-filter` no Debian) ou em `http://www.manpagez.com/man/7/pcap-filter`.

A seguir, algumas expressões muito utilizadas.

Chave	Função
`host` *`nome-ip`*	Especifica que somente o tráfego envolvendo a máquina em questão, referenciada pelo seu nome ou IP, será mostrado.
`net` *`rede/CIDR`*	Idem ao anterior. No entanto, a filtragem é em relação a uma faixa de rede, em vez de uma máquina única. A expressão de filtragem poderá ser com `CIDR`, como em `192.168.1.0/24`, ou com máscara de rede, como em `192.168.0.16 mask 255.255.255.0`.
`ether host` *`MAC`*	Idem, referindo-se a um endereço `MAC`.
`port` *`porta`*	Idem, referindo-se a uma porta.
`portrange` *`20-90`*	Idem, referindo-se ao range de portas de *`20`* a *`90`*.
`src`	Delimita à origem. Pode ser associado a `host`, `net`, `port`, `portrange` e `ether host`. Exemplos: `src host`, `src net`, `src port`, `ether src host`.
`dst`	Delimita ao destino. Pode ser associado a `host`, `net`, `port`, `portrange` e `ether host`. Ex.: `dst host`.
`not` ou `!`	Operador lógico `NOT`. Utilizado para excluir algo do resultado da pesquisa. Ex.: `! port 80`.
`and` ou `&&`	Operador lógico `AND`. Utilizado para associar duas ou mais expressões, tornando-as obrigatórias no resultado da pesquisa.
`or` ou `||`	Operador lógico `OR`. Utilizado para declarar duas ou mais expressões, fazendo com que, pelo menos uma, apareça no resultado da pesquisa.
`ip`	Mostra somente o tráfego `IPv4`.
`ip6`	Mostra somente o tráfego `IPv6`.
`tcp`	Mostra somente o tráfego `TCP`.

3. Expressões de filtragem (continuação)

Chave	Função
`udp`	Mostra somente o tráfego `UDP`.
`icmp / icmp6`	Mostra apenas tráfego `ICMP` ou `ICMP6`.
`arp`	Mostra somente tráfego `ARP`.
`stp`	Apenas tráfego do tipo Spanning Tree Protocol.
`less` *tam*	Mostra apenas pacotes com tamanho `<=` *tam*.
`greater` *tam*	Mostra apenas pacotes com tamanho `>=` *tam*.
`vlan` *id*	Mostra apenas o tráfego relativo à `vlan` que possui a identificação *id*.

4. Exemplos de uso

► Mostrar todo o tráfego de rede, que passa pela primeira interface listada com `# tcpdump -D`, sem resolver nomes. Isso permitirá a visualização do tráfego em tempo real.

```
# tcpdump -n
```

► Tráfego `UDP` no adaptador `eth1`, incluindo o payload (área de dados) em ASCII, sem resolver nomes.

```
# tcpdump -nAi eth1 udp
```

► Tráfego UDP com o host 10.1.1.25, sem resolver nomes.

```
# tcpdump -n host 10.1.1.25 and udp
```

► Tráfego com o host 10.1.1.2, que seja `UDP`, e que tenha como origem ou destino a porta 53, sem resolver nomes.

```
# tcpdump -n host 10.1.1.2 and udp and port 53
```

► Tráfego que envolva o host 10.1.1.25, que seja `UDP`, e que esteja relacionado a qualquer porta, exceto a 53, sem resolver nomes. Também será mostrado o cabeçalho referente à camada de enlace.

```
# tcpdump -ne host 10.1.1.25 and udp and port ! 53
```


► Tráfego `TCP` que seja oriundo ou destinado à porta 80 ou que seja apenas oriundo da 110, sem resolver nomes. Os apóstrofos foram utilizados para evitar a interpretação errônea dos parênteses pelo shell.

```
# tcpdump -n tcp and '(port 80 or src port 110)'
```

► Tráfego `ICMP` referentes a qualquer host que pertença à rede 10.1.0.0/16, sem resolver nomes.

```
# tcpdump -n icmp and net 10.1.0.0/16
```

► Tráfego referente ao host que possua o endereço `MAC` especificado. Não resolve nomes.

```
# tcpdump -n ether host 00:ff:31:22:2d:11
```

5. Filtragem dos campos do protocolo TCP

É possível realizar filtragens, procurando por situações específicas no `TCP`. Para isso, você precisará conhecer a estrutura de cabeçalho do protocolo (RFC 793).

Vamos a um exemplo. Queremos filtrar apenas o tráfego que contenha as flags `ACK` e `RST` ativadas. Segundo a RFC 793, as flags `TCP` `CWR`, `ECE`, `URG`, `ACK`, `PSH`, `RST`, `SYN` e `FIN`, nesta ordem, estão no 14º byte do cabeçalho. Como a contagem inicia em zero, o 14º byte é o campo 13. Assim, precisaremos marcar `1` na flag `RST` e `0` nas restantes. Pela ordem das flags, o resultado final será `00010100` que, em decimal, representa `20`. Resultado:

```
# tcpdump -n tcp[13] = 20
```

Para ver o tráfego que NÃO contenha `ACK` e `RST`, utilize:

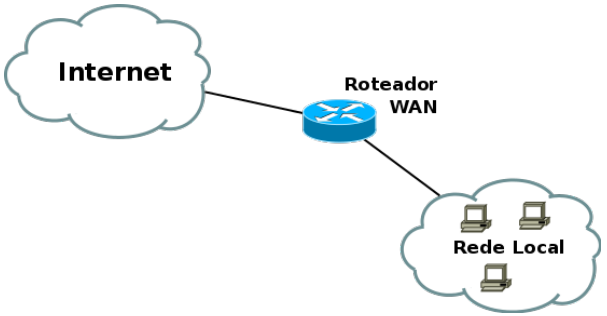
```
# tcpdump -n tcp[13] != 20
```

6. Capturas para estudo

Há diversas capturas para estudo, no wiki do Wireshark, em `http://wiki.wireshark.org/SampleCaptures`. Ajude o wiki deles enviando a sua captura!!!

ANÁLISE DE TRÁFEGO EM REDES TCP/IP COM TCPDUMP E WINDUMP

Versão 1.3 - 16 de junho de 2013



© 2013 by João Eriberto Mota Filho

http://eriberto.pro.br/redes
eriberto@eriberto.pro.br

2048R/2DF0491F: 1D75 E212 B34C F4BF A9E0 D0D8 DE6D E039 C1CF C265
4096R/04EBE9EF: 357D CB0E EC95 A01A EBA1 F0D2 DE63 B9C7 04EB E9EF