Manual do tcpdump em Português

TCPDUMP(1) TCPDUMP(1)

NOME
tcpdump - captura o tráfego em uma rede

SINOPSE
tcpdump [-adeflnNOpqRStvxX] [-c contagem] [-F arquivo] [-i interface] [-m módulo] [-r arquivo] [-s tamanho] [-T tipo] [-w arquivo] [-E algo:secret] [expressão]

DESCRIÇÃO
Tcpdump imprime a saída dos cabeçalhos dos pacotes na interface de rede que combinam com a expressão booleana.

No SunOS com nit ou bpf: Para rodar tcpdump você necessita ter permissão de acesso a /dev/nit ou /dev/bpf*.
No Solaris com dlpi: Você precisa ter acesso de leitura/escrita ao pseudo dispositivo de rede, por exemplo /dev/le.
No HP-UX com dlpi: Você necessita ser root ou ter este instalado com setuid para root.
No IRIS com snoop: Você necessita ser root ou ter este instalado com setuid para root.
No Linux: Você necessita ser root ou ter este instalado com setuid para root.
No Ultrix e Digital UNIX: Somente o super-usuário tem permissão de utilizar o modo de operação promíscuo usando pfconfig(8), qualquer usuário pode rodar o tcpdump.
No BSD: Você necessita ter acesso de leitura em /dev/bpf*.

OPÇÕES:
-a Espera para converter endereços de rede e broadcast para nomes.

-c Sai após receber contagem de pacotes.

-d Captura os pacotes compilados com o código do pacote em um formato humanamente legível para a saída padrão e sai.

-dd Captura os pacotes com o código do pacote como um fragmento de programa em C.

-ddd Captura os pacotes com o código do pacote como números decimais (precedidos de um contador).

-e Imprime a camada de link do cabeçalho na linha capturada.

-E Use algo:secreto para decriptar pacotes IPsec ESP.
Algoritmos costumam ser des-cbc, 3des-cbc, blowfish-cbc, rc3-cbc, cast128-cbc, ou nenhum. O padrão é des-cbc.
A habilidade de descriptar pacotes só estará presente se o tcpdump for compilado com suporte a criptografia habilitado. secreto é o texto ASCII para a chave ESP secreta.
Nós não tentaremos um valor binário arbitrário neste momento.
A opção assume a RFC2406 do ESP, não a RFC1827 também do ESP.
A opção é somente para propósitos de depuração, e o uso desta opção com uma chave secreta TRULY é desencorajada.
Pela apresentação da chave secreta do IPsec na linha de comando você consegue deixar isto visível para outros, via ps(1) e em outras ocasiões.

-f Imprime a saída dos endereços internet numéricamente no lugar de simbólicamente (essa opção tenta descobrir sérios problemas em servidores YP utilizando Sun -- usualmente causa um loop eterno na conversão de não-locais números internet).

-F Utiliza o arquivo para a entrada de um filtro de expressão.
Qualquer expressão adicional passada via linha de comando é ignorada.

-i Escute na interface. Se não especificado o tcpdump irá procurar a lista de interfaces do sistema, interfaces ativas (excluindo a de loopback).

Em sistemas Linux com kernels 2.2 ou superiores, um argumento ``any" para interfaces pode ser usado para capturar pacotes de todas as interfaces.
Note que as capturas em ``any" (todos) os dispositivos não poderá ser feita no modo promíscuo.

-l Deixe a linha de saída "bufferizada". Usualmente se você quizer ver os dados enquanto são capturados.
Ex: ``tcpdump -l | tee dat" ou ``tcpdump -l > dat & tail -f dat".

-n Não converter endereços (Ex: endereços de hosts e números de portas, etc) para nomes.

-N Não imprime a qualificação do domínio dos nomes de host. Ex: se você passar esta flag para o tcpdump, ele imprimirá ``nic" ao invés de ``nic.ddn.mil".

-m Carrega as definições do módulo SMI MIB do arquivo módulo. Esta opção pode ser usada em diversas vezes para carregar severos módulos MIB no tcpdump.

-O Não execute o otimizador de códigos de pacote. Isso usualmente só será utilizado se você suspeitar de uma falha no otimizador.

-p Não coloca a interface em modo promíscuo. Note que esta interface pode precisar entrar em modo promíscuo por alguma outra razão; então, `-p' não poderá ser usado como uma abreviação para `ether host {local-hw- addr} ou ether broadcast'.

-q Saída rápida (quieta?). Imprime menos informações do protocolo em saídas de linhas mais curtas.

-r Lê os pacotes do arquivo (isso é criado com a opção -w). A entrada padrão será utilizada se o arquivo for ``-".

-s Sniffa tamanho de bytes de arquivo deste pacote.
O default é 68 (com SunOS's NIT, o mínimo é atualmente 96). 68 bytes é adequado para IP, ICMP, TCP e UDP porém pode truncar informações de protocolos de pacotes DNS e NFS (olhe na frente). Os pacotes serão truncados porque um tamanho limitado foi indicado e a saída será ``[|proto]", onde proto é o nome do nível do protocolo onde a truncagem ocorreu.
Note que aumentar muito o tamanho pode causar uma delonga no tempo para processar estes pacotes e, efetivamente, diminuir a quantidade de pacotes capturados.
Isso pode causar perda de pacotes. Você deve limitar o tamanho dos pacotes para o menor possível onde você conseguirá obter a informação que lhe interessar. Coloque o tamanho em 0 para que o tcpdump capture os pacotes completos, independente dos seus tamanhos.

-T Força que os pacotes selecionados pela "expressão" sejam interpretados pelo tipo especificado. Atualmente, os tipos conhecidos são cnfp (Protocolo Cisco NetFlow), rpc (Chamadas de procedimento remotas), rtp (Protocolo de aplicações em tempo real), rtcp (Protocolo de controle de aplicações em tempo real), snmp (Protocolo simples de gerenciamento de redes), vat (Aplicação de Visual Audio), e wb (Placa Branca distribuída).

-R Assume que pacotes ESP/AH são baseados em especificações antigas (RFC1825 a RFC1829). Se especificado, o tcpdump não irá imprimir o campo de prevensão. No entanto, este não é um campo de versão na especificação do protocolo ESP/AH, portanto o tcpdump não poderá deduzir a versão do protocolo.

-S Imprimir o absoluto, em lugar do relativo, número de sequência TCP.

-t Não imprimir o timestamp na linha capturada.

-tt Imprimir um não formatado timestamp na linha capturada.

-v Detalhamento da saída (não muito). Por exemplo, o tempo de vida, identificação, tamanho total e opções do cabeçalho IP serão impressos. Também se habilita opções adicionais de checagem da integridade dos pacotes como verificação do checksum dos

cabeçalhos IP e ICMP.

-vv Saída mais detalhada. Por exemplo, campos adicionais serão impressos em pacotes NFS de resposta.

-vvv Saída mais detalhada ainda. Por exemplo, as opções telnet SB ... SE serão impressas totalmente. Com -X as opções do telnet serão impressas em HEX muito bem.

-w Escreve os pacotes capturados no arquivo no lugar de selecioná-los e imprimí-los. Isso poderá ser impresso utilizando-se a opção -r. A saída padrão será utilizada se o arquivo for ``-".

-x Imprime este pacote (menos seu cabeçalho de camada de link) em hexadecimal. Tamanho em bytes será impresso (opção -s).

-X Se imprime em hexa, imprime em ASCII também. Se a opção -x também for selecionada, o pacote será impresso em hexa/ascii.
Isso é muito útil para analizar novos protocolos. Se a opção -x não estiver selecionada, algumas partes de alguns pacotes serão impressas em hexa/ascii.

expressão
Seleciona quais pacotes serão capturados. Se nenhuma expressão for passada, todos os pacotes da rede serão capturados.
Se informada, apenas os pacotes que tiverem a expressão como sendo verdadeira (combinarem com a expressão) serão capturados.

A expressão consiste em uma ou mais primitivas.
Primitivas usualmente consistem em um identificador (nome ou número) precedidas de um ou mais qualificadores. Existem 3 diferentes qualificadores:

type indica qual identificador (nome ou número) ele se refere. Os tipos possíveis são: host, net e port.
Exemplos:
``host foo"
``net 128.3"
``port 20"
Se nenhum qualificador type for especificado, host será assumido.

dir indica a direção em que a transferência ocorrerá, para e/ou do identificador. As direções possíveis são src, dst, src or dst e src and dst
Exemplos:
``src foo"
``dst net 128.3"
"src or dst port ftp-data"
Se nenhum qualificador dir for especificado, src or dst será assumido. Para camadas de link ``null" (Ex: protocolos de ponto a ponto como o slip) os qualificadores de entrada e saída devem ser utilizados para especificar a direção desejada.

proto Qualificador restrito a estipular um tipo particular de protocolo. As opções existentes de protocolo são:
ether, fddi, tr, ip, ip6, arp, rarp, decnet, tcp e udp.
Ex:
``ether src foo"
``arp net 128.3"
``tcp port 21"
Se não estipulado, todos os protocolos existentes em opção serão assumidos.
Ex:
``src foo" inclui
``(ip or arp or rarp) src foo"
``net bar" inclui
``(ip or arp or rarp) net bar" e
``port 53" inclui

``(tcp or udp) port 53".

[`fddi' é atualmente um apelido para `ether'; isso seria idêntico a mensionar ``o nível de link de dados usado nesta específica interface de rede."
Cabeçalhos FDDI possuem campos iguais ao Ethernet, de endereços de origem e destino, e também tipo de pacotes, então você pode filtrar estes campos FDDI analogamente como você faz com estes campos Ethernet.
Cabeçalhos FDDI também contém outros campos, porém você não poderá especificá-los em uma expressão de filtro.

Similarmente, `tr' é um apelido para `ether'; o parágrafo anterior sobre os cabeçalhos FDDI também se aplica a cabeçalhos Token Ring.]

Em adição a isto, existe algumas outras diretivas especiais que não foram mencionadas: gateway, broadcast, less, greater e expressões aritméticas. Todas elas serão descritas mais adiante.

Expressões de filtragem mais complexas podem ser feitas utilizando-se expressões como and, or e not combinadas com as diretivas.

Ex:
``host foo and not port ftp and not port ftp-data".
Para facilitar, listas de qualificadores idênticos podem ser omitidas.
Ex:
``tcp dst port ftp or ftp-data or domain"
é exatamente o mesmo que:
``tcp dst port ftp or tcp dst port ftp-data or tcp dst port domain".

Primitivas (diretivas) permitidas são:

dst host host
Verdadeiro se o campo de destino de pacotes IPv4/v6 for host, este pode ser endereço ou nome.

src host host
Verdadeiro se o campo de origem de pacotes IPv4/v6 for host.

host host
Verdadeiro se o campo de origem ou destino de pacotes IPv4/v6 for host. Qualquer uma destas expressões de host podem ser precedidas por diretivas tais como ip, arp, rarp, ou ip6, assim:
ip host host
isso é o equivalente a:
ether proto ip and host host
Se host for um nome com múltiplos endereços IP, eles serão checados para a correlação.

ether dst ehost
Verdadeiro se o endereço de destino ethernet for ehost. Ehost pode ser um nome de /etc/ethers ou um número (veja ethers(3N) para o formato numérico).

ether src ehost
Verdadeiro se o endereço de origem ethernet for ethos.

ether host ehost
Verdadeiro se o endereço de origem ou de destino ethernet for ehost.

gateway host
Verdadeiro se o host usado no pacote for um gateway. Ex: o endereço de origem ou destino ethernet seja um host diferente do endereço de origem ou destino IP.
Host precisa ser um nome e ser encontrado tanto em /etc/hosts quanto /etc/ethers.
(Uma expressão equivalente seria:
ether host ehost and not host host

que poderia ser usada com nomes ethernet ou números para host / ehost.)
Esta syntax por enquanto não funciona com configurações IPv6.

dst net net
Verdadeiro se o endereço de destino IPv4/v6 do pacote for um número de rede. Net pode ser um nome de /etc/networks ou um número de rede (veja networks(4) para maiores detalhes).

src net net
Verdadeiro se o endereço de origem IPv4/v6 do pacote for um número de rede.

net net
Verdadeiro se o endereço de origem ou destino IPv4/v6 do pacote for um número de rede.

net net mask mask
Verdadeiro se o endereço IP combinar com a rede de mascará especificada. Pode ser qualificada com src ou dst. Esta syntax também não foi implementada para redes IPv6.

net net/len
Verdadeiro se o endereço IPv4/v6 combinar com a rede que possua netmask len bits. Pode ser qualificada com src ou dst.

dst port port
Verdadeiro se o pacote for ip/tcp, ip/udp, ip6/tcp ou ip6/udp e sua porta de destino seja port.
A porta pode ser um número ou um nome usado em /etc/services (veja tcp(4P) e udp(4P)).
Se um nome for usado, tanto o número da porta quanto o protocolo serão checados.
Se um número ou um nome ambíguo for utilizado, somente o número da porta será checado.
Ex:
dst port 513
Irá imprimir tanto tcp/login quanto udp/who
e
port domain
Irá imprimir tanto tcp/domain quanto udp/domain

src port port
Verdadeiro se o pacote tiver a porta de origem port.

port port
Verdadeiro se a porta de origem ou de destino do pacote for port. Todas as expressões de porta anteriores podem ser precedidas com diretivas tcp ou udp, assim:
tcp src port port
Irá capturar apenas pacotes tcp com porta de origem port.

less length
Verdadeiro se o pacote tiver um tamanho menor ou igual a length. Isso seria o equivalente a:
len <= length.

greater length
Verdadeiro se o pacote tiver um tamanho maior ou igual a length. Isso seria o equivalente a:
len >= length.

ip proto protocol
Verdadeiro se o pacote for um pacote IP (veja ip(4P)) com protocolo de tipo protocol.
Protocol pode ser um número ou um dos nomes: icmp, icmp6, igmp, igrp, pim, ah, esp, udp ou tcp.
Note que os identificadores tcp, udp e icmp também são diretivas que podem ser passadas via backslash (l), isso seria no C-Shell.
Perceba que esta primitiva não observa a regra de protocolo do cabeçalho.

ip6 proto protocol
Verdadeiro se o pacote for um pacote IPv6 com protocolo de tipo protocol.
Observe que esta primitiva não observa a regra de protocolo do cabeçalho.

ip6 protochain protocol
Verdadeiro se o pacote for um pacote IPv6, e conter um cabeçalho de protocolo com tipo protocol na regra de protocolo do cabeçalho.
Por exemplo:
ip6 protochain 6
combina com qualquer pacote IPv6 com o protocolo TCP definido na regra do cabeçalho que especifica o protocolo.
O pacote pode conter, por exemplo, cabeçalho de autenticação, roteamento, hop-by-hop, tanto IPv6 quanto TCP. O código BPF emitido por esta primitiva é complexo e não será otimizado pelo otimizador BPF do tcpdump, porque isto ficaria muito lento.

ip protochain protocol
Equivalente ao ip6 protochain protocol, porém este é válido para IPv4.

ether broadcast
Verdadeiro se o pacote for um pacote ethernet de broadcast. A diretiva ether é opcional.

ip broadcast
Verdadeiro se o pacote for um pacote de broadcast IP. Isso irá checar tanto por tudo-zero quanto tudo-um como convenção para broadcast, e olhará para a máscara de subrede local.

ether multicast
Verdadeiro se o pacote for um pacote ethernet de multicast. A diretiva ether é opcional.
Isto é uma abreviação de ``ether[0] & 1 != 0".

ip multicast
Verdadeiro se o pacote for um pacote IP multicast.

ip6 multicast
Verdadeiro se o pacote for um pacote IPv6 multicast.
ether proto protocol
Verdadeiro se o pacote for um pacote ethernet com tipo protocol.
Protocol pode ser um número ou um dos seguintes nomes:
ip, ip6, arp, rarp, atalk, aarp, dec-net, sca, lat, mopdl, moprc, ou iso.

Observe que estes identificadores também são diretivas que podem ser passados via backslash (l).
[No caso do FDDI (ex: ``fddi protocol arp"), a identificação do protocolo vêm do cabeçalho 802.2 Logical Link Control (LLC), e isto poderá usualmente estar no topo da camada do cabeçalho FDDI.
O tcpdump assumirá, enquanto filtra o identificador de protocolo, que todos os pacotes FDDI incluem um cabeçalho LLC e que este cabeçalho estará no formato SNAP. O mesmo se aplica ao Token Ring.]

decnet src host
Verdadeiro se o endereço de origem DECNET for host, isto poderá ser um endereço na forma ``10.123", ou um nome de host DECNET.
[O suporte a nomes de host DECNET só está disponível em sistemas Ultrix configurados para rodar DECNET.]

decnet dst host
Verdadeiro se o endereço de destino DECNET for host.

decnet host host
Verdadeiro se o endereço de destino ou de origem DECNET for host.

ip, ip6, arp, rarp, atalk, aarp, decnet, iso

Abreviações para:
ether proto p
onde p é um dos protocolos mencionados anteriormente.

lat, moprc, mopdl
Abreviações para:
ether proto p
onde p é um dos protocolos mencionados anteriormente.
Perceba que o tcpdump atualmente não possui "know how" para estes protocolos.

vlan [vlan_id]
Verdadeiro se o pacote for um pacote VLAN IEEE 802.1Q Se [vlan_id] for especificado, somente será verdadeiro o pacote que obedeça a especificação [vlan_id].
Observe que esta é a primeira diretiva vlan encontrada nas mudanças em expressão com offsets de decodificação para definir que este pacote é um pacote VLAN.

tcp, udp, icmp
Abreviações para:
ip proto p or ip6 proto p
onde p é um dos protocolos mencionados anteriormente.

iso proto protocol
Verdadeiro se o pacote for um pacote OSI com tipo de protocolo protocol. Protocol pode ser um número ou um dos seguintes nomes: clnp, esis ou isis.

clnp, esis, isis
Abreviações para:
iso proto p
onde p é um dos protocolos mencionados anteriormente. Observe que o tcpdump faz um trabalho incompleto para selecionar estes protocolos.

expr relop expr
Verdadeiro se a relação holds, onde relop é um dos seguintes: >, <, >=, <=, =, !=, e expr é uma expressão aritmética composta por uma constante inteira (expressa na syntax padrão da linguagem C), os operadores binários normais, a saber: +, -, *, /, &, |, um operador de tamanho e um pacote especial de dados acessórios.
Para acessar o interior dos dados de um pacote, use a seguinte syntax:
proto [expr : size]
Proto é um dos seguintes:
ether, fddi, tr, ip, arp, rarp, tcp, udp, icmp ou ip6, e indica a camada de protocolo para a operação de índice. Observe que tcp, udp e outros protocolos de camada mais alta, somente se aplicam ao IPv4, não ao IPv6 (isso será corrigido no futuro).
O final do byte, relativo a camada indicada do protocolo, é passado em expr.
Size (tamanho) é opcional e indica o número de bytes no campo de interesse; este pode ser: one, two ou four, sendo o padrão one.
O operador de tamanho, indicado pela diretiva len, informa o tamanho do pacote.

Por exemplo:
``ether[0] & 1 != 0"
Captura todo o tráfego multicast.
A expressão:
``ip[0] & 0xf != 5"
Captura todos os pacotes IP com opções.
A expressão:
``ip[6:2] & 0x1fff = 0"
Captura somente datagramas não fragmentados e fragmentos zero dos datagramas fragmentados.

Esta checagem pode ser expecificamente aplicada para operações de índice em tcp e udp. Então, tcp[0] sempre menciona o primeiro byte do cabeçalho TCP, e nunca o primeiro byte de um fragmento.

Diretivas podem ser combinadas usando-se:

Diretivas e operadores entre parênteses (parênteses são especiais para o Shell e serão interpretados).

Negação: (`!' ou `not').

Concatenação (e): (`&&' ou `and').

Alternação (ou): (`||' ou `or').

Negação deve ser precedida. Alternação e concatenação necessitam de precedência e associam esquerda com direita.

Se um identificador for passado sem uma diretiva, a mais recente diretiva será assumida.

Por exemplo:
not host vs and ace
será igual à:
not host vs and host ace
mas não pode ser confundido com:
not (host vs or ace)

Argumentos de expressão podem ser passados ao tcpdump como um simples argumento, ou como múltiplos argumentos.
Geralmente, se a expressão contiver metacaracteres de SHELL, será mais fácil passar como um simples argumento entre aspas.
Múltiplos argumentos são concatenados com espaços antes de serem analizados.

EXEMPLOS
Para imprimir todos os pacotes vindos de ou para o departamento sundown tcpdump host sundown

Para imprimir o tráfego entre helios e hot ou ace:
tcpdump host helios and (hot or ace)

Para imprimir todos os pacotes IP entre ace e qualquer host exceto helios:
tcpdump ip host ace and not helios

Para imprimir todo tráfego entre hosts locais e hosts em Berkeley:
tcpdump net ucb-ether

Para imprimir todo tráfego ftp para a internet através do gateway snup:
(Observe que a expressão está entre parênteses para prevenir que o shell interprete o que está entre parênteses)
tcpdump 'gateway snup and (port ftp or ftp-data)'

Para imprimir o tráfego para redes fora da rede local (se você for gateway para outra rede, esta regra pode não prever isto em sua rede local).
tcpdump ip and not net localnet

Para imprimir pacotes de início e final de conexões (SYN e FIN) TCP envolvendo hosts não-locais.
tcpdump 'tcp[13] & 3 != 0 and not src and dst net localnet'

Para imprimir pacotes IP maiores do que 576 bytes enviados através do gateway snup:
tcpdump 'gateway snup and ip[2:2] > 576'

Para imprimir pacotes de broadcast ou multicast IP que não estão sendo enviados via broadcast ou multicat ethernet
tcpdump 'ether[0] & 1 = 0 and ip[16] >= 224'

Para imprimir todos os pacotes ICMP que não sejam echo request/reply
(não sejam ping's)
tcpdump 'icmp[0] != 8 and icmp[0] != 0'

SEE ALSO
traffic(1C), nit(4P), bpf(4), pcap(3)

AUTHORS
The original authors are:

Van Jacobson, Craig Leres and Steven McCanne, all of the Lawrence Berkeley National Laboratory, University of Cali- fornia, Berkeley, CA.

It is currently being maintained by tcpdump.org.

The current version is available via http:

http://www.tcpdump.org/

The original distribution is available via anonymous ftp:

ftp://ftp.ee.lbl.gov/tcpdump.tar.Z

IPv6/IPsec support is added by WIDE/KAME project. This program uses Eric Young's SSLeay library, under specific configuration.

http://www.tcpdump.org/

The original distribution is available via anonymous ftp:

ftp://ftp.ee.lbl.gov/tcpdump.tar.Z

IPv6/IPsec support is added by WIDE/KAME project. This program uses Eric Young's SSLeay library, under specific configuration.

BUGS
Please send problems, bugs, questions, desirable enhance- ments, etc. to:

tcpdump-workers@tcpdump.org

Please send source code contributions, etc. to:

patches@tcpdump.org

NIT doesn't let you watch your own outbound traffic, BPF will. We recommend that you use the latter.

On Linux systems with 2.0[.x] kernels:

packets on the loopback device will be seen twice;

packet filtering cannot be done in the kernel, so that all packets must be copied from the kernel in order to be filtered in user mode;

all of a packet, not just the part that's within the snapshot length, will be copied from the kernel (the 2.0[.x] packet capture mechanism, if asked to copy only part of a packet to userland, will not report the true length of the packet; this would cause most IP packets to get an error from tcp- dump).

We recommend that you upgrade to a 2.2 or later kernel.

Some attempt should be made to reassemble IP fragments or, at least to compute the right length for the higher level protocol.

Name server inverse queries are not dumped correctly: the (empty) question section is printed rather than real query in the answer section. Some believe that inverse queries are themselves a bug and prefer to fix the program gener- ating them rather than tcpdump.

A packet trace that crosses a daylight savings time change will give skewed time stamps (the time change is ignored).

Filter expressions that manipulate FDDI or Token Ring headers assume that all FDDI and Token Ring packets are SNAP-encapsulated Ethernet packets. This is true for IP, ARP, and DECNET Phase IV, but is not true for protocols such as ISO CLNS. Therefore, the filter may inadvertently accept certain packets that do not properly match the fil- ter expression.

Filter expressions on fields other than those that manipu- late Token Ring headers will not correctly handle source- routed Token Ring packets.

ip6 proto should chase header chain, but at this moment it does not. ip6 protochain is supplied for this behavior.

Arithmetic expression against transport layer headers,

A packet trace that crosses a daylight savings time change will give skewed time stamps (the time change is ignored).

Filter expressions that manipulate FDDI or Token Ring headers assume that all FDDI and Token Ring packets are SNAP-encapsulated Ethernet packets. This is true for IP, ARP, and DECNET Phase IV, but is not true for protocols such as ISO CLNS. Therefore, the filter may inadvertently accept certain packets that do not properly match the fil- ter expression.

Filter expressions on fields other than those that manipu- late Token Ring headers will not correctly handle source- routed Token Ring packets.

ip6 proto should chase header chain, but at this moment it does not. ip6 protochain is supplied for this behavior.

Arithmetic expression against transport layer headers, like tcp[0], does not work against IPv6 packets. It only looks at IPv4 packets.

TRADUTOR:

Nome : Rodrigo Rubira Branco
Email : rodrigo@br.tcpdump.org
Site : http://www.br.tcpdump.org
Empresa: Firewalls Security (http://www.firewalls.com.br)
Nota do Tradutor: Agradeço a empresa Firewalls Security por ter me concedido tempo para trabalhar e auxiliar no projeto deste excelente packet sniffer, podendo contribuir de diversas formas para o mesmo, e cedendo espaço em seu servidor para armazenar um mirror completo do TCPDUMP. Percebam que a tradução do manual completo ainda não foi feita, e espero contar com a colaboração de todos para que o seja o mais rápido possível. Obrigado!