NAME

tcpprof − report profile of network traffic

SYNOPSIS

tcpprof [**-?hdnpR**] [**-f** *filter expr*] [**-i** *interface*] [**-P** *port*] [**-r** *filename*] [**-s** *seconds*] [**-S** *letters*] [**-t** *lines*]

DESCRIPTION

tcpprof reports a profile of network traffic by ranking it by link type, ip protocol, TCP/UDP port, ip address, or network address.

Network information is collected either by reading data from *filename*, or by directly monitoring the network interface *interface*. The default action for **tcpprof** is to automatically search for an appropriate interface, and to generate a profile before it exits.

When reading data from *filename*, **tcpprof** will display the profile and exit immediately after the entire file has been processed. When collecting data from *interface*, **tcpprof** will keep running unless the **-s** option had been specified.

OPTIONS

The options are as follows:

-f *filter expr*

Filter the packets according the rules given by *filter expr*. For the syntax of these rules, see tcpdump(1). The argument must be quoted if it contains spaces in order to separate it from other options.

-h, **-**? Display version and a brief help message.

-d **tcpprof** will track the source and destination information separately, where applicable, and identify source data with a ">" and destination data with "<". For example, a "http <" statistic signifies all traffic with destination port 80 (http). This option only applies to port, host and network statistics.

-i *interface*

Do a live capture (rather than read from a file) on the interface *interface* given on the command line. If *interface* is "auto" then **tcpprof** tries to find an appropriate one by itself.

-P *port* This tells **tcpprof** to ignore TCP and UDP ports greater than or equal to *port* when displaying port statistics. This is not the same as filtering these port numbers out of the data set. This way, packets with i.e. the source port above *port* and the destination port below *port* will be able to still count the lower port number as a statistic. In addition, this doesn't affect the other statistic types (link, protocol, etc.)

-p Set the interface into non-promiscuous mode (promiscuous is the default) when doing live captures.

-r *filename*

Read all data from *filename*, which may be a regular file, a named pipe or "-" to read it's data from standard input. Acceptable file formats include pcap (tcpdump(1) files) and "snoop" format files. *filename* is usually a file created by the tcpdump(1) command using the "-w" option.

-S *letters*

Tells **tcpprof** which statistics to display. *letters* must be a string of one or more of the following letters:

l show stats about the link layer

i show stats about all ip protocols

p show stats about TCP/UDP ports

h show stats about hosts/ip addresses

n show stats about network addresses

a a synonym for "liphn"

-s *seconds*
When monitoring an interface, **tcpprof** runs for only *seconds* seconds, and then quits. Has no effect when reading data from a file.

-t *lines* When printing a profile of the data, **tcpprof** will display a maximum of *lines* lines for each statistic.

SIGNALS

Upon receiving a SIGINT, **tcpprof** will print any remaining statistics, and then exit.

FILES

`/dev/bpf`**n** the packet filter device

EXAMPLES

```
tcpprof -i fxp0 -S a
```

Displays a complete profile of network data passing through the fxp0 network interface, after the user enters ^C (control C).

```
tcpprof -r file.dump -S a
```

Displays a complete profile of network data from the `tcpdump`(1) generated file "file.dump".

SEE ALSO

`tcpdump`(1), `pcap`(3), `bpf`(4)

HISTORY

tcpprof was first written along side tcpstat in Winter 1998 using FreeBSD 3.0, and then finaly retrofited for Linux in Spring 2000. It became installed along with tcpstat since version 1.5.

AUTHORS

Paul Herman ⟨pherman@frenchfries.net⟩
Cologne, Germany.

Please send all bug reports to this address.

BUGS

Not tested with link types other than Ethernet, PPP, and "None" types.

There may be problems reading non-IPv4 packets across platforms when reading null type link layers. This is due to a lack of a standardized packet type descriptor in libpcap for this link type.

Snoop file formats cannot be read from stdin or named pipes.