

NAME

tcpssso - set a socket option on a TCP endpoint

SYNOPSIS

tcpssso -i *id* [*level*] *optname* *optval*

tcpssso -a [*level*] *optname* *optval*

tcpssso -C *cc-algo* [-**S** *stack*] [-**s** *state*] [*level*] *optname* *optval*

tcpssso [-**C** *cc-algo*] [-**S** *stack*] [-**s** *state*] [*level*] *optname* *optval*

tcpssso [-**C** *cc-algo*] [-**S** *stack*] [-**s** *state*] [*level*] *optname* *optval*

DESCRIPTION

The **tcpssso** command applies a *level* level socket option with name *optname* and value *optval* on a TCP endpoint from the command line.

TCP endpoints in the TIME_WAIT state can not be handled by **tcpssso**. TCP endpoints in the SYN_RCVD state can only be handled if their prior state was SYN_SENT.

[*level*] can be specified as a non negative number or a symbolic name like SOL_SOCKET, IPPROTO_IP, IPPROTO_IPV6, or IPPROTO_TCP. If not specified, **tcpssso** deduces the level from *optname*, if provided as a symbolic name. If that is not the case, IPPROTO_TCP is used.

optname can be specified as a non negative number or a symbolic name like SO_DEBUG, IP_TOS, IPV6_TCLASS, TCP_LOG, TCP_CONGESTION, or TCP_FUNCTION_BLK.

optval can be in integer value, which will be converted to a binary value and passed as an int value. If it cannot be parsed as an integer value, it will be processed as a string. If the *optname* is TCP_FUNCTION_BLK then *optval* is converted to a *struct tcp_function_set*.

If **-i** *id* is specified then **tcpssso** will apply the socket option to the TCP endpoint with the *inp_gencnt* provided as *id*. The *inp_gencnt* for existing TCP endpoints can be determined by using `sockstat(1)`.

If **-a** is specified then **tcpssso** will apply the socket option to all TCP endpoints subject to the above state restrictions.

If **-C** *cc-algo* is specified then **tcpssso** will apply the socket option to all TCP endpoints using the TCP congestion control algorithm *cc-algo* and subject to the above state restrictions.

If **-S** *stack* is specified then **tcpssso** will apply the socket option to all TCP endpoints using the TCP stack *stack* and subject to the above state restrictions.

If **-s state** is specified then **tcpssso** will apply the socket option to all TCP endpoints being in the state *state*. *state* is one of CLOSED, LISTEN, SYN_SENT, SYN_RCVD, ESTABLISHED, CLOSE_WAIT, FIN_WAIT_1, CLOSING, LAST_ACK, FIN_WAIT_2. Using SYN_RCVD only applies to TCP endpoints in the state SYN_RCVD if their prior state was SYN_SENT.

If multiple of **-C cc-algo**, **-S stack**, and **-s state** are specified, **tcpssso** will apply the socket option to all TCP endpoints not being in the state TIME_WAIT and using the congestion control algorithm *cc-algo*, being in the state *state*, and using the TCP stack *stack*, if specified.

If none of the **-a**, **-C**, **-S**, or **-s** options are specified then the option **-i** must be specified.

EXIT STATUS

The **tcpssso** utility exits 0 on success, and >0 if an error occurs.

EXAMPLES

To diagnose a problem with a particular TCP connection to sshd(8), first determine its `inp_gencnt` using `sockstat(1)`:

```
# sockstat -4 -c -i -p 22 -P tcp -q
root  sshd      827  4 tcp4 \
      192.168.1.1:22  192.168.1.2:53736  435
```

Then, use the following command to enable Black Box Logging on it:

```
# tcpssso -i 435 TCP_LOG 4
```

To switch all TCP endpoints from using the `freebsd` stack to the `rack` stack use:

```
# tcpssso -S freebsd TCP_FUNCTION_BLK rack
```

The following command will set the congestion control module of all TCP endpoints currently using `cubic` as its congestion control algorithm to the congestion control algorithm `newreno`:

```
# tcpssso -C cubic TCP_CONGESTION newreno
```

SEE ALSO

`sockstat(1)`, `setsockopt(2)`, `tcp(4)`, `tcp_functions(9)`

HISTORY

The **tcpssso** command first appeared in FreeBSD 14.

AUTHORS

Michael Tuexen <*tuexen@FreeBSD.org*>