

NAME

ss – another utility to investigate sockets

SYNOPSIS

ss [*options*] [*FILTER*]

DESCRIPTION

ss is used to dump socket statistics. It allows showing information similar to *netstat*. It can display more TCP and state information than other tools.

OPTIONS

When no option is used ss displays a list of open non-listening sockets (e.g. TCP/UNIX/UDP) that have established connection.

-h, --help

Show summary of options.

-V, --version

Output version information.

-H, --no-header

Suppress header line.

-O, --oneline

Print each socket's data on a single line.

-n, --numeric

Do not try to resolve service names. Show exact bandwidth values, instead of human-readable.

-r, --resolve

Try to resolve numeric address/ports.

-a, --all

Display both listening and non-listening (for TCP this means established connections) sockets.

-l, --listening

Display only listening sockets (these are omitted by default).

-o, --options

Show timer information. For TCP protocol, the output format is:

timer:(<timer_name>,<expire_time>,<retrans>)

<timer_name>

the name of the timer, there are five kind of timer names:

on : means one of these timers: TCP retrans timer, TCP early retrans timer and tail loss probe timer

keepalive: tcp keep alive timer

timewait: timewait stage timer

persist: zero window probe timer

unknown: none of the above timers

<expire_time>

how long time the timer will expire

<retrans>

how many times the retransmission occurred

-e, --extended

Show detailed socket information. The output format is:

uid:<uid_number> ino:<inode_number> sk:<cookie>

<uid_number>

the user id the socket belongs to

<inode_number>

the socket's inode number in VFS

<cookie>

an uuid of the socket

-m, --memory

Show socket memory usage. The output format is:

```
skmem:(r<rmem_alloc>,rb<rcv_buf>,t<wmem_alloc>,tb<snd_buf>,
      f<fwd_alloc>,w<wmem_queued>,
      o<opt_mem>,bl<back_log>)
```

<rmem_alloc>

the memory allocated for receiving packet

<rcv_buf>

the total memory can be allocated for receiving packet

<wmem_alloc>

the memory used for sending packet (which has been sent to layer 3)

<snd_buf>

the total memory can be allocated for sending packet

<fwd_alloc>

the memory allocated by the socket as cache, but not used for receiving/sending packet yet. If need memory to send/receive packet, the memory in this cache will be used before allocate additional memory.

<wmem_queued>

The memory allocated for sending packet (which has not been sent to layer 3)

<ropt_mem>

The memory used for storing socket option, e.g., the key for TCP MD5 signature

<back_log>

The memory used for the sk backlog queue. On a process context, if the process is receiving packet, and a new packet is received, it will be put into the sk backlog queue, so it can be received by the process immediately

-p, --processes

Show process using socket.

-i, --info

Show internal TCP information. Below fields may appear:

ts show string "ts" if the timestamp option is set

sack show string "sack" if the sack option is set

ecn show string "ecn" if the explicit congestion notification option is set

ecnseen show string "ecnseen" if the saw ecn flag is found in received packets

fastopen show string "fastopen" if the fastopen option is set

cong_alg the congestion algorithm name, the default congestion algorithm is "cubic"

wscale:<snd_wscales>:<rcv_wscales>
 if window scale option is used, this field shows the send scale factor and receive scale factor

rto:<icsk_rto>
 tcp re-transmission timeout value, the unit is millisecond

backoff:<icsk_backoff>
 used for exponential backoff re-transmission, the actual re-transmission timeout value is icsk_rto << icsk_backoff

rtt:<rtt>/<rttvar>
 rtt is the average round trip time, rttvar is the mean deviation of rtt, their units are millisecond

ato:<ato>
 ack timeout, unit is millisecond, used for delay ack mode

mss:<mss>
 max segment size

cwnd:<cwnd>
 congestion window size

pmtu:<pmtu>
 path MTU value

ssthresh:<ssthresh>
 tcp congestion window slow start threshold

bytes_acked:<bytes_acked>
 bytes acked

bytes_received:<bytes_received>
 bytes received

segs_out:<segs_out>
 segments sent out

segs_in:<segs_in>
 segments received

send <send_bps>bps
 egress bps

lastsnd:<lastsnd>
 how long time since the last packet sent, the unit is millisecond

lastrcv:<lastrcv>
 how long time since the last packet received, the unit is millisecond

lastack:<lastack>
 how long time since the last ack received, the unit is millisecond

pacing_rate <pacing_rate>bps/<max_pacing_rate>bps
 the pacing rate and max pacing rate

rcv_space:<rcv_space>
 a helper variable for TCP internal auto tuning socket receive buffer

--tos Show ToS and priority information. Below fields may appear:

tos IPv4 Type-of-Service byte
tclass IPv6 Traffic Class byte

class_id

Class id set by net_cls cgroup. If class is zero this shows priority set by SO_PRIORITY.

-K, --kill

Attempts to forcibly close sockets. This option displays sockets that are successfully closed and silently skips sockets that the kernel does not support closing. It supports IPv4 and IPv6 sockets only.

-s, --summary

Print summary statistics. This option does not parse socket lists obtaining summary from various sources. It is useful when amount of sockets is so huge that parsing /proc/net/tcp is painful.

-E, --events

Continually display sockets as they are destroyed

-Z, --context

As the **-p** option but also shows process security context.

For **netlink(7)** sockets the initiating process context is displayed as follows:

1. If valid pid show the process context.
2. If destination is kernel (pid = 0) show kernel initial context.
3. If a unique identifier has been allocated by the kernel or netlink user, show context as "unavailable". This will generally indicate that a process has more than one netlink socket active.

-z, --contexts

As the **-Z** option but also shows the socket context. The socket context is taken from the associated inode and is not the actual socket context held by the kernel. Sockets are typically labeled with the context of the creating process, however the context shown will reflect any policy role, type and/or range transition rules applied, and is therefore a useful reference.

-N NSNAME, --net=NSNAME

Switch to the specified network namespace name.

-b, --bpf

Show socket BPF filters (only administrators are allowed to get these information).

-4, --ipv4

Display only IP version 4 sockets (alias for -f inet).

-6, --ipv6

Display only IP version 6 sockets (alias for -f inet6).

-0, --packet

Display PACKET sockets (alias for -f link).

-t, --tcp

Display TCP sockets.

-u, --udp

Display UDP sockets.

-d, --dccp

Display DCCP sockets.

-w, --raw

Display RAW sockets.

-x, --unix

Display Unix domain sockets (alias for -f unix).

- S, --sctp**
Display SCTP sockets.
- vsock**
Display vsock sockets (alias for -f vsock).
- xdp** Display XDP sockets (alias for -f xdp).
- f FAMILY, --family=FAMILY**
Display sockets of type FAMILY. Currently the following families are supported: unix, inet, inet6, link, netlink, vsock, xdp.
- A QUERY, --query=QUERY, --socket=QUERY**
List of socket tables to dump, separated by commas. The following identifiers are understood: all, inet, tcp, udp, raw, unix, packet, netlink, unix_dgram, unix_stream, unix_seqpacket, packet_raw, packet_dgram, dccp, sctp, vsock_stream, vsock_dgram, xdp. Any item in the list may optionally be prefixed by an exclamation mark (!) to exclude that socket table from being dumped.
- D FILE, --diag=FILE**
Do not display anything, just dump raw information about TCP sockets to FILE after applying filters. If FILE is - stdout is used.
- F FILE, --filter=FILE**
Read filter information from FILE. Each line of FILE is interpreted like single command line option. If FILE is - stdin is used.
- FILTER := [state STATE-FILTER] [EXPRESSION]**
Please take a look at the official documentation for details regarding filters.

STATE-FILTER

STATE-FILTER allows to construct arbitrary set of states to match. Its syntax is sequence of keywords state and exclude followed by identifier of state.

Available identifiers are:

All standard TCP states: **established**, **syn-sent**, **syn-recv**, **fin-wait-1**, **fin-wait-2**, **time-wait**, **closed**, **close-wait**, **last-ack**, **listening** and **closing**.

all - for all the states

connected - all the states except for **listening** and **closed**

synchronized - all the **connected** states except for **syn-sent**

bucket - states, which are maintained as minisockets, i.e. **time-wait** and **syn-recv**

big - opposite to **bucket**

USAGE EXAMPLES

ss -t -a Display all TCP sockets.

ss -t -a -Z
Display all TCP sockets with process SELinux security contexts.

ss -u -a Display all UDP sockets.

ss -o state established '(dport = :ssh or sport = :ssh)'
Display all established ssh connections.

ss -x src /tmp/.X11-unix/*

Find all local processes connected to X server.

ss -o state fin-wait-1 '(sport = :http or sport = :https)' dst 193.233.7/24

List all the tcp sockets in state FIN-WAIT-1 for our apache to network 193.233.7/24 and look at their timers.

ss -a -A 'all,!tcp'

List sockets in all states from all socket tables but TCP.

SEE ALSO

ip(8),

RFC 793 - <https://tools.ietf.org/rfc/rfc793.txt> (TCP states)

AUTHOR

ss was written by Alexey Kuznetsov, <kuznet@ms2.inr.ac.ru>.

This manual page was written by Michael Prokop <mika@grml.org> for the Debian project (but may be used by others).