```
/proc/sys/net/ipv4/* Variables:

ip_forward - BOOLEAN
        0 - disabled (default)
        not 0 - enabled

        Forward Packets between interfaces.

        This variable is special, its change resets all configuration
        parameters to their default state (RFC1122 for hosts, RFC1812
        for routers)

ip_default_ttl - INTEGER
        Default value of TTL field (Time To Live) for outgoing (but not
        forwarded) IP packets. Should be between 1 and 255 inclusive.
        Default: 64 (as recommended by RFC1700)

ip_no_pmtu_disc - INTEGER
        Disable Path MTU Discovery. If enabled in mode 1 and a
        fragmentation-required ICMP is received, the PMTU to this
        destination will be set to min_pmtu (see below). You will need
        to raise min_pmtu to the smallest interface MTU on your system
        manually if you want to avoid locally generated fragments.

        In mode 2 incoming Path MTU Discovery messages will be
        discarded. Outgoing frames are handled the same as in mode 1,
        implicitly setting IP_PMTUDISC_DONT on every created socket.

        Mode 3 is a hardened pmtu discover mode. The kernel will only
        accept fragmentation-needed errors if the underlying protocol
        can verify them besides a plain socket lookup. Current
        protocols for which pmtu events will be honored are TCP, SCTP
        and DCCP as they verify e.g. the sequence number or the
        association. This mode should not be enabled globally but is
        only intended to secure e.g. name servers in namespaces where
        TCP path mtu must still work but path MTU information of other
        protocols should be discarded. If enabled globally this mode
        could break other protocols.

        Possible values: 0-3
        Default: FALSE

min_pmtu - INTEGER
        default 552 - minimum discovered Path MTU

ip_forward_use_pmtu - BOOLEAN
        By default we don't trust protocol path MTUs while forwarding
        because they could be easily forged and can lead to unwanted
        fragmentation by the router.
        You only need to enable this if you have user-space software
        which tries to discover path mtus by itself and depends on the
        kernel honoring this information. This is normally not the
        case.
        Default: 0 (disabled)
        Possible values:
        0 - disabled
        1 - enabled

fwmark_reflect - BOOLEAN
        Controls the fwmark of kernel-generated IPv4 reply packets that are not
        associated with a socket for example, TCP RSTs or ICMP echo replies).
        If unset, these packets have a fwmark of zero. If set, they have the
```

```
        fwmark of the packet they are replying to.
        Default: 0

fib_multipath_use_neigh - BOOLEAN
        Use status of existing neighbor entry when determining nexthop for
        multipath routes. If disabled, neighbor information is not used and
        packets could be directed to a failed nexthop. Only valid for kernels
        built with CONFIG_IP_ROUTE_MULTIPATH enabled.
        Default: 0 (disabled)
        Possible values:
        0 - disabled
        1 - enabled

fib_multipath_hash_policy - INTEGER
        Controls which hash policy to use for multipath routes. Only valid
        for kernels built with CONFIG_IP_ROUTE_MULTIPATH enabled.
        Default: 0 (Layer 3)
        Possible values:
        0 - Layer 3
        1 - Layer 4

ip_forward_update_priority - INTEGER
        Whether to update SKB priority from "TOS" field in IPv4 header after it
        is forwarded. The new SKB priority is mapped from TOS field value
        according to an rt_tos2priority table (see e.g. man tc-prio).
        Default: 1 (Update priority.)
        Possible values:
        0 - Do not update priority.
        1 - Update priority.

route/max_size - INTEGER
        Maximum number of routes allowed in the kernel.  Increase
        this when using large numbers of interfaces and/or routes.
        From linux kernel 3.6 onwards, this is deprecated for ipv4
        as route cache is no longer used.

neigh/default/gc_thresh1 - INTEGER
        Minimum number of entries to keep.  Garbage collector will not
        purge entries if there are fewer than this number.
        Default: 128

neigh/default/gc_thresh2 - INTEGER
        Threshold when garbage collector becomes more aggressive about
        purging entries. Entries older than 5 seconds will be cleared
        when over this number.
        Default: 512

neigh/default/gc_thresh3 - INTEGER
        Maximum number of non-PERMANENT neighbor entries allowed.  Increase
        this when using large numbers of interfaces and when communicating
        with large numbers of directly-connected peers.
        Default: 1024

neigh/default/unres_qlen_bytes - INTEGER
        The maximum number of bytes which may be used by packets
        queued for each unresolved address by other network layers.
        (added in linux 3.3)
        Setting negative value is meaningless and will return error.
        Default: SK_WMEM_MAX, (same as net.core.wmem_default).
                Exact value depends on architecture and kernel options,
                but should be enough to allow queuing 256 packets
                of medium size.
```

```
neigh/default/unres_qlen - INTEGER
        The maximum number of packets which may be queued for each
        unresolved address by other network layers.
        (deprecated in linux 3.3) : use unres_qlen_bytes instead.
        Prior to linux 3.3, the default value is 3 which may cause
        unexpected packet loss. The current default value is calculated
        according to default value of unres_qlen_bytes and true size of
        packet.
        Default: 101

mtu_expires - INTEGER
        Time, in seconds, that cached PMTU information is kept.

min_adv_mss - INTEGER
        The advertised MSS depends on the first hop route MTU, but will
        never be lower than this setting.

IP Fragmentation:

ipfrag_high_thresh - LONG INTEGER
        Maximum memory used to reassemble IP fragments.

ipfrag_low_thresh - LONG INTEGER
        (Obsolete since linux-4.17)
        Maximum memory used to reassemble IP fragments before the kernel
        begins to remove incomplete fragment queues to free up resources.
        The kernel still accepts new fragments for defragmentation.

ipfrag_time - INTEGER
        Time in seconds to keep an IP fragment in memory.

ipfrag_max_dist - INTEGER
        ipfrag_max_dist is a non-negative integer value which defines the
        maximum "disorder" which is allowed among fragments which share a
        common IP source address. Note that reordering of packets is
        not unusual, but if a large number of fragments arrive from a source
        IP address while a particular fragment queue remains incomplete, it
        probably indicates that one or more fragments belonging to that queue
        have been lost. When ipfrag_max_dist is positive, an additional check
        is done on fragments before they are added to a reassembly queue - if
        ipfrag_max_dist (or more) fragments have arrived from a particular IP
        address between additions to any IP fragment queue using that source
        address, it's presumed that one or more fragments in the queue are
        lost. The existing fragment queue will be dropped, and a new one
        started. An ipfrag_max_dist value of zero disables this check.

        Using a very small value, e.g. 1 or 2, for ipfrag_max_dist can
        result in unnecessarily dropping fragment queues when normal
        reordering of packets occurs, which could lead to poor application
        performance. Using a very large value, e.g. 50000, increases the
        likelihood of incorrectly reassembling IP fragments that originate
        from different IP datagrams, which could result in data corruption.
        Default: 64

INET peer storage:

inet_peer_threshold - INTEGER
        The approximate size of the storage.  Starting from this threshold
        entries will be thrown aggressively.  This threshold also determines
        entries' time-to-live and time intervals between garbage collection
        passes.  More entries, less time-to-live, less GC interval.

inet_peer_minttl - INTEGER
```

```
        Minimum time-to-live of entries.  Should be enough to cover fragment
        time-to-live on the reassembling side.  This minimum time-to-live  is
        guaranteed if the pool size is less than inet_peer_threshold.
        Measured in seconds.

inet_peer_maxttl - INTEGER
        Maximum time-to-live of entries.  Unused entries will expire after
        this period of time if there is no memory pressure on the pool (i.e.
        when the number of entries in the pool is very small).
        Measured in seconds.

TCP variables:

somaxconn - INTEGER
        Limit of socket listen() backlog, known in userspace as SOMAXCONN.
        Defaults to 128.  See also tcp_max_syn_backlog for additional tuning
        for TCP sockets.

tcp_abort_on_overflow - BOOLEAN
        If listening service is too slow to accept new connections,
        reset them. Default state is FALSE. It means that if overflow
        occurred due to a burst, connection will recover. Enable this
        option _only_ if you are really sure that listening daemon
        cannot be tuned to accept connections faster. Enabling this
        option can harm clients of your server.

tcp_adv_win_scale - INTEGER
        Count buffering overhead as bytes/2^tcp_adv_win_scale
        (if tcp_adv_win_scale > 0) or bytes-bytes/2^(-tcp_adv_win_scale),
        if it is <= 0.
        Possible values are [-31, 31], inclusive.
        Default: 1

tcp_allowed_congestion_control - STRING
        Show/set the congestion control choices available to non-privileged
        processes. The list is a subset of those listed in
        tcp_available_congestion_control.
        Default is "reno" and the default setting (tcp_congestion_control).

tcp_app_win - INTEGER
        Reserve max(window/2^tcp_app_win, mss) of window for application
        buffer. Value 0 is special, it means that nothing is reserved.
        Default: 31

tcp_autocorking - BOOLEAN
        Enable TCP auto corking :
        When applications do consecutive small write()/sendmsg() system calls,
        we try to coalesce these small writes as much as possible, to lower
        total amount of sent packets. This is done if at least one prior
        packet for the flow is waiting in Qdisc queues or device transmit
        queue. Applications can still use TCP_CORK for optimal behavior
        when they know how/when to uncork their sockets.
        Default : 1

tcp_available_congestion_control - STRING
        Shows the available congestion control choices that are registered.
        More congestion control algorithms may be available as modules,
        but not loaded.

tcp_base_mss - INTEGER
        The initial value of search_low to be used by the packetization layer
        Path MTU discovery (MTU probing).  If MTU probing is enabled,
        this is the initial MSS used by the connection.
```

```
tcp_congestion_control - STRING
        Set the congestion control algorithm to be used for new
        connections. The algorithm "reno" is always available, but
        additional choices may be available based on kernel configuration.
        Default is set as part of kernel configuration.
        For passive connections, the listener congestion control choice
        is inherited.
        [see setsockopt(listenfd, SOL_TCP, TCP_CONGESTION, "name" ...) ]

tcp_dsack - BOOLEAN
        Allows TCP to send "duplicate" SACKs.

tcp_early_retrans - INTEGER
        Tail loss probe (TLP) converts RTOs occurring due to tail
        losses into fast recovery (draft-ietf-tcpm-rack). Note that
        TLP requires RACK to function properly (see tcp_recovery below)
        Possible values:
                0 disables TLP
                3 or 4 enables TLP
        Default: 3

tcp_ecn - INTEGER
        Control use of Explicit Congestion Notification (ECN) by TCP.
        ECN is used only when both ends of the TCP connection indicate
        support for it.  This feature is useful in avoiding losses due
        to congestion by allowing supporting routers to signal
        congestion before having to drop packets.
        Possible values are:
                0 Disable ECN.  Neither initiate nor accept ECN.
                1 Enable ECN when requested by incoming connections and
                  also request ECN on outgoing connection attempts.
                2 Enable ECN when requested by incoming connections
                  but do not request ECN on outgoing connections.
        Default: 2

tcp_ecn_fallback - BOOLEAN
        If the kernel detects that ECN connection misbehaves, enable fall
        back to non-ECN. Currently, this knob implements the fallback
        from RFC3168, section 6.1.1.1., but we reserve that in future,
        additional detection mechanisms could be implemented under this
        knob. The value is not used, if tcp_ecn or per route (or congestion
        control) ECN settings are disabled.
        Default: 1 (fallback enabled)

tcp_fack - BOOLEAN
        This is a legacy option, it has no effect anymore.

tcp_fin_timeout - INTEGER
        The length of time an orphaned (no longer referenced by any
        application) connection will remain in the FIN_WAIT_2 state
        before it is aborted at the local end.  While a perfectly
        valid "receive only" state for an un-orphaned connection, an
        orphaned connection in FIN_WAIT_2 state could otherwise wait
        forever for the remote to close its end of the connection.
        Cf. tcp_max_orphans
        Default: 60 seconds

tcp_frto - INTEGER
        Enables Forward RTO-Recovery (F-RTO) defined in RFC5682.
        F-RTO is an enhanced recovery algorithm for TCP retransmission
        timeouts.  It is particularly beneficial in networks where the
        RTT fluctuates (e.g., wireless). F-RTO is sender-side only
```

```
        modification. It does not require any support from the peer.

        By default it's enabled with a non-zero value. 0 disables F-RTO.

tcp_fwmark_accept - BOOLEAN
        If set, incoming connections to listening sockets that do not have a
        socket mark will set the mark of the accepting socket to the fwmark of
        the incoming SYN packet. This will cause all packets on that connection
        (starting from the first SYNACK) to be sent with that fwmark. The
        listening socket's mark is unchanged. Listening sockets that already
        have a fwmark set via setsockopt(SOL_SOCKET, SO_MARK, ...) are
        unaffected.

        Default: 0

tcp_invalid_ratelimit - INTEGER
        Limit the maximal rate for sending duplicate acknowledgments
        in response to incoming TCP packets that are for an existing
        connection but that are invalid due to any of these reasons:

          (a) out-of-window sequence number,
          (b) out-of-window acknowledgment number, or
          (c) PAWS (Protection Against Wrapped Sequence numbers) check failure

        This can help mitigate simple "ack loop" DoS attacks, wherein
        a buggy or malicious middlebox or man-in-the-middle can
        rewrite TCP header fields in manner that causes each endpoint
        to think that the other is sending invalid TCP segments, thus
        causing each side to send an unterminating stream of duplicate
        acknowledgments for invalid segments.

        Using 0 disables rate-limiting of dupacks in response to
        invalid segments; otherwise this value specifies the minimal
        space between sending such dupacks, in milliseconds.

        Default: 500 (milliseconds).

tcp_keepalive_time - INTEGER
        How often TCP sends out keepalive messages when keepalive is enabled.
        Default: 2hours.

tcp_keepalive_probes - INTEGER
        How many keepalive probes TCP sends out, until it decides that the
        connection is broken. Default value: 9.

tcp_keepalive_intvl - INTEGER
        How frequently the probes are send out. Multiplied by
        tcp_keepalive_probes it is time to kill not responding connection,
        after probes started. Default value: 75sec i.e. connection
        will be aborted after ~11 minutes of retries.

tcp_l3mdev_accept - BOOLEAN
        Enables child sockets to inherit the L3 master device index.
        Enabling this option allows a "global" listen socket to work
        across L3 master domains (e.g., VRFs) with connected sockets
        derived from the listen socket to be bound to the L3 domain in
        which the packets originated. Only valid when the kernel was
        compiled with CONFIG_NET_L3_MASTER_DEV.
        Default: 0 (disabled)

tcp_low_latency - BOOLEAN
        This is a legacy option, it has no effect anymore.
```

```
tcp_max_orphans - INTEGER
        Maximal number of TCP sockets not attached to any user file handle,
        held by system. If this number is exceeded orphaned connections are
        reset immediately and warning is printed. This limit exists
        only to prevent simple DoS attacks, you _must_ not rely on this
        or lower the limit artificially, but rather increase it
        (probably, after increasing installed memory),
        if network conditions require more than default value,
        and tune network services to linger and kill such states
        more aggressively. Let me to remind again: each orphan eats
        up to ~64K of unswappable memory.

tcp_max_syn_backlog - INTEGER
        Maximal number of remembered connection requests, which have not
        received an acknowledgment from connecting client.
        The minimal value is 128 for low memory machines, and it will
        increase in proportion to the memory of machine.
        If server suffers from overload, try increasing this number.

tcp_max_tw_buckets - INTEGER
        Maximal number of timewait sockets held by system simultaneously.
        If this number is exceeded time-wait socket is immediately destroyed
        and warning is printed. This limit exists only to prevent
        simple DoS attacks, you _must_ not lower the limit artificially,
        but rather increase it (probably, after increasing installed memory),
        if network conditions require more than default value.

tcp_mem - vector of 3 INTEGERs: min, pressure, max
        min: below this number of pages TCP is not bothered about its
        memory appetite.

        pressure: when amount of memory allocated by TCP exceeds this number
        of pages, TCP moderates its memory consumption and enters memory
        pressure mode, which is exited when memory consumption falls
        under "min".

        max: number of pages allowed for queueing by all TCP sockets.

        Defaults are calculated at boot time from amount of available
        memory.

tcp_min_rtt_wlen - INTEGER
        The window length of the windowed min filter to track the minimum RTT.
        A shorter window lets a flow more quickly pick up new (higher)
        minimum RTT when it is moved to a longer path (e.g., due to traffic
        engineering). A longer window makes the filter more resistant to RTT
        inflations such as transient congestion. The unit is seconds.
        Default: 300

tcp_moderate_rcvbuf - BOOLEAN
        If set, TCP performs receive buffer auto-tuning, attempting to
        automatically size the buffer (no greater than tcp_rmem[2]) to
        match the size required by the path for full throughput.  Enabled by
        default.

tcp_mtu_probing - INTEGER
        Controls TCP Packetization-Layer Path MTU Discovery.  Takes three
        values:
          0 - Disabled
          1 - Disabled by default, enabled when an ICMP black hole detected
          2 - Always enabled, use initial MSS of tcp_base_mss.

tcp_probe_interval - UNSIGNED INTEGER
```

```
        Controls how often to start TCP Packetization-Layer Path MTU
        Discovery reprobe. The default is reprobing every 10 minutes as
        per RFC4821.

tcp_probe_threshold - INTEGER
        Controls when TCP Packetization-Layer Path MTU Discovery probing
        will stop in respect to the width of search range in bytes. Default
        is 8 bytes.

tcp_no_metrics_save - BOOLEAN
        By default, TCP saves various connection metrics in the route cache
        when the connection closes, so that connections established in the
        near future can use these to set initial conditions.  Usually, this
        increases overall performance, but may sometimes cause performance
        degradation.  If set, TCP will not cache metrics on closing
        connections.

tcp_orphan_retries - INTEGER
        This value influences the timeout of a locally closed TCP connection,
        when RTO retransmissions remain unacknowledged.
        See tcp_retries2 for more details.

        The default value is 8.
        If your machine is a loaded WEB server,
        you should think about lowering this value, such sockets
        may consume significant resources. Cf. tcp_max_orphans.

tcp_recovery - INTEGER
        This value is a bitmap to enable various experimental loss recovery
        features.

        RACK: 0x1 enables the RACK loss detection for fast detection of lost
              retransmissions and tail drops. It also subsumes and disables
              RFC6675 recovery for SACK connections.
        RACK: 0x2 makes RACK's reordering window static (min_rtt/4).
        RACK: 0x4 disables RACK's DUPACK threshold heuristic

        Default: 0x1

tcp_reordering - INTEGER
        Initial reordering level of packets in a TCP stream.
        TCP stack can then dynamically adjust flow reordering level
        between this initial value and tcp_max_reordering
        Default: 3

tcp_max_reordering - INTEGER
        Maximal reordering level of packets in a TCP stream.
        300 is a fairly conservative value, but you might increase it
        if paths are using per packet load balancing (like bonding rr mode)
        Default: 300

tcp_retrans_collapse - BOOLEAN
        Bug-to-bug compatibility with some broken printers.
        On retransmit try to send bigger packets to work around bugs in
        certain TCP stacks.

tcp_retries1 - INTEGER
        This value influences the time, after which TCP decides, that
        something is wrong due to unacknowledged RTO retransmissions,
        and reports this suspicion to the network layer.
        See tcp_retries2 for more details.

        RFC 1122 recommends at least 3 retransmissions, which is the
```

```
        default.

tcp_retries2 - INTEGER
        This value influences the timeout of an alive TCP connection,
        when RTO retransmissions remain unacknowledged.
        Given a value of N, a hypothetical TCP connection following
        exponential backoff with an initial RTO of TCP_RTO_MIN would
        retransmit N times before killing the connection at the (N+1)th RTO.

        The default value of 15 yields a hypothetical timeout of 924.6
        seconds and is a lower bound for the effective timeout.
        TCP will effectively time out at the first RTO which exceeds the
        hypothetical timeout.

        RFC 1122 recommends at least 100 seconds for the timeout,
        which corresponds to a value of at least 8.

tcp_rfc1337 - BOOLEAN
        If set, the TCP stack behaves conforming to RFC1337. If unset,
        we are not conforming to RFC, but prevent TCP TIME_WAIT
        assassination.
        Default: 0

tcp_rmem - vector of 3 INTEGERs: min, default, max
        min: Minimal size of receive buffer used by TCP sockets.
        It is guaranteed to each TCP socket, even under moderate memory
        pressure.
        Default: 4K

        default: initial size of receive buffer used by TCP sockets.
        This value overrides net.core.rmem_default used by other protocols.
        Default: 87380 bytes. This value results in window of 65535 with
        default setting of tcp_adv_win_scale and tcp_app_win:0 and a bit
        less for default tcp_app_win. See below about these variables.

        max: maximal size of receive buffer allowed for automatically
        selected receiver buffers for TCP socket. This value does not override
        net.core.rmem_max.  Calling setsockopt() with SO_RCVBUF disables
        automatic tuning of that socket's receive buffer size, in which
        case this value is ignored.
        Default: between 87380B and 6MB, depending on RAM size.

tcp_sack - BOOLEAN
        Enable select acknowledgments (SACKS).

tcp_comp_sack_delay_ns - LONG INTEGER
        TCP tries to reduce number of SACK sent, using a timer
        based on 5% of SRTT, capped by this sysctl, in nano seconds.
        The default is 1ms, based on TSO autosizing period.

        Default : 1,000,000 ns (1 ms)

tcp_comp_sack_nr - INTEGER
        Max numer of SACK that can be compressed.
        Using 0 disables SACK compression.

        Detault : 44

tcp_slow_start_after_idle - BOOLEAN
        If set, provide RFC2861 behavior and time out the congestion
        window after an idle period.  An idle period is defined at
        the current RTO.  If unset, the congestion window will not
        be timed out after an idle period.
```

```
        Default: 1

tcp_stdurg - BOOLEAN
        Use the Host requirements interpretation of the TCP urgent pointer field.
        Most hosts use the older BSD interpretation, so if you turn this on
        Linux might not communicate correctly with them.
        Default: FALSE

tcp_synack_retries - INTEGER
        Number of times SYNACKs for a passive TCP connection attempt will
        be retransmitted. Should not be higher than 255. Default value
        is 5, which corresponds to 31seconds till the last retransmission
        with the current initial RTO of 1second. With this the final timeout
        for a passive TCP connection will happen after 63seconds.

tcp_syncookies - BOOLEAN
        Only valid when the kernel was compiled with CONFIG_SYN_COOKIES
        Send out syncookies when the syn backlog queue of a socket
        overflows. This is to prevent against the common 'SYN flood attack'
        Default: 1

        Note, that syncookies is fallback facility.
        It MUST NOT be used to help highly loaded servers to stand
        against legal connection rate. If you see SYN flood warnings
        in your logs, but investigation shows that they occur
        because of overload with legal connections, you should tune
        another parameters until this warning disappear.
        See: tcp_max_syn_backlog, tcp_synack_retries, tcp_abort_on_overflow.

        syncookies seriously violate TCP protocol, do not allow
        to use TCP extensions, can result in serious degradation
        of some services (f.e. SMTP relaying), visible not by you,
        but your clients and relays, contacting you. While you see
        SYN flood warnings in logs not being really flooded, your server
        is seriously misconfigured.

        If you want to test which effects syncookies have to your
        network connections you can set this knob to 2 to enable
        unconditionally generation of syncookies.

tcp_fastopen - INTEGER
        Enable TCP Fast Open (RFC7413) to send and accept data in the opening
        SYN packet.

        The client support is enabled by flag 0x1 (on by default). The client
        then must use sendmsg() or sendto() with the MSG_FASTOPEN flag,
        rather than connect() to send data in SYN.

        The server support is enabled by flag 0x2 (off by default). Then
        either enable for all listeners with another flag (0x400) or
        enable individual listeners via TCP_FASTOPEN socket option with
        the option value being the length of the syn-data backlog.

        The values (bitmap) are
          0x1: (client) enables sending data in the opening SYN on the client.
          0x2: (server) enables the server support, i.e., allowing data in
                        a SYN packet to be accepted and passed to the
                        application before 3-way handshake finishes.
          0x4: (client) send data in the opening SYN regardless of cookie
                        availability and without a cookie option.
        0x200: (server) accept data-in-SYN w/o any cookie option present.
        0x400: (server) enable all listeners to support Fast Open by
                        default without explicit TCP_FASTOPEN socket option.
```

```
        Default: 0x1

        Note that that additional client or server features are only
        effective if the basic support (0x1 and 0x2) are enabled respectively.

tcp_fastopen_blackhole_timeout_sec - INTEGER
        Initial time period in second to disable Fastopen on active TCP sockets
        when a TFO firewall blackhole issue happens.
        This time period will grow exponentially when more blackhole issues
        get detected right after Fastopen is re-enabled and will reset to
        initial value when the blackhole issue goes away.
        0 to disable the blackhole detection.
        By default, it is set to 1hr.

tcp_syn_retries - INTEGER
        Number of times initial SYNs for an active TCP connection attempt
        will be retransmitted. Should not be higher than 127. Default value
        is 6, which corresponds to 63seconds till the last retransmission
        with the current initial RTO of 1second. With this the final timeout
        for an active TCP connection attempt will happen after 127seconds.

tcp_timestamps - INTEGER
Enable timestamps as defined in RFC1323.
        0: Disabled.
        1: Enable timestamps as defined in RFC1323 and use random offset for
        each connection rather than only using the current time.
        2: Like 1, but without random offsets.
        Default: 1

tcp_min_tso_segs - INTEGER
        Minimal number of segments per TSO frame.
        Since linux-3.12, TCP does an automatic sizing of TSO frames,
        depending on flow rate, instead of filling 64Kbytes packets.
        For specific usages, it's possible to force TCP to build big
        TSO frames. Note that TCP stack might split too big TSO packets
        if available window is too small.
        Default: 2

tcp_pacing_ss_ratio - INTEGER
        sk->sk_pacing_rate is set by TCP stack using a ratio applied
        to current rate. (current_rate = cwnd * mss / srtt)
        If TCP is in slow start, tcp_pacing_ss_ratio is applied
        to let TCP probe for bigger speeds, assuming cwnd can be
        doubled every other RTT.
        Default: 200

tcp_pacing_ca_ratio - INTEGER
        sk->sk_pacing_rate is set by TCP stack using a ratio applied
        to current rate. (current_rate = cwnd * mss / srtt)
        If TCP is in congestion avoidance phase, tcp_pacing_ca_ratio
        is applied to conservatively probe for bigger throughput.
        Default: 120

tcp_tso_win_divisor - INTEGER
        This allows control over what percentage of the congestion window
        can be consumed by a single TSO frame.
        The setting of this parameter is a choice between burstiness and
        building larger TSO frames.
        Default: 3

tcp_tw_reuse - INTEGER
        Enable reuse of TIME-WAIT sockets for new connections when it is
```

```
        safe from protocol viewpoint.
        0 - disable
        1 - global enable
        2 - enable for loopback traffic only
        It should not be changed without advice/request of technical
        experts.
        Default: 2

tcp_window_scaling - BOOLEAN
        Enable window scaling as defined in RFC1323.

tcp_wmem - vector of 3 INTEGERs: min, default, max
        min: Amount of memory reserved for send buffers for TCP sockets.
        Each TCP socket has rights to use it due to fact of its birth.
        Default: 4K

        default: initial size of send buffer used by TCP sockets.  This
        value overrides net.core.wmem_default used by other protocols.
        It is usually lower than net.core.wmem_default.
        Default: 16K

        max: Maximal amount of memory allowed for automatically tuned
        send buffers for TCP sockets. This value does not override
        net.core.wmem_max.  Calling setsockopt() with SO_SNDBUF disables
        automatic tuning of that socket's send buffer size, in which case
        this value is ignored.
        Default: between 64K and 4MB, depending on RAM size.

tcp_notsent_lowat - UNSIGNED INTEGER
        A TCP socket can control the amount of unsent bytes in its write queue,
        thanks to TCP_NOTSENT_LOWAT socket option. poll()/select()/epoll()
        reports POLLOUT events if the amount of unsent bytes is below a per
        socket value, and if the write queue is not full. sendmsg() will
        also not add new buffers if the limit is hit.

        This global variable controls the amount of unsent data for
        sockets not using TCP_NOTSENT_LOWAT. For these sockets, a change
        to the global variable has immediate effect.

        Default: UINT_MAX (0xFFFFFFFF)

tcp_workaround_signed_windows - BOOLEAN
        If set, assume no receipt of a window scaling option means the
        remote TCP is broken and treats the window as a signed quantity.
        If unset, assume the remote TCP is not broken even if we do
        not receive a window scaling option from them.
        Default: 0

tcp_thin_linear_timeouts - BOOLEAN
        Enable dynamic triggering of linear timeouts for thin streams.
        If set, a check is performed upon retransmission by timeout to
        determine if the stream is thin (less than 4 packets in flight).
        As long as the stream is found to be thin, up to 6 linear
        timeouts may be performed before exponential backoff mode is
        initiated. This improves retransmission latency for
        non-aggressive thin streams, often found to be time-dependent.
        For more information on thin streams, see
        Documentation/networking/tcp-thin.txt
        Default: 0

tcp_limit_output_bytes - INTEGER
        Controls TCP Small Queue limit per tcp socket.
        TCP bulk sender tends to increase packets in flight until it
```

```
        gets losses notifications. With SNDBUF autotuning, this can
        result in a large amount of packets queued on the local machine
        (e.g.: qdiscs, CPU backlog, or device) hurting latency of other
        flows, for typical pfifo_fast qdiscs.  tcp_limit_output_bytes
        limits the number of bytes on qdisc or device to reduce artificial
        RTT/cwnd and reduce bufferbloat.
        Default: 1048576 (16 * 65536)

tcp_challenge_ack_limit - INTEGER
        Limits number of Challenge ACK sent per second, as recommended
        in RFC 5961 (Improving TCP's Robustness to Blind In-Window Attacks)
        Default: 100

UDP variables:

udp_l3mdev_accept - BOOLEAN
        Enabling this option allows a "global" bound socket to work
        across L3 master domains (e.g., VRFs) with packets capable of
        being received regardless of the L3 domain in which they
        originated. Only valid when the kernel was compiled with
        CONFIG_NET_L3_MASTER_DEV.
        Default: 0 (disabled)

udp_mem - vector of 3 INTEGERs: min, pressure, max
        Number of pages allowed for queueing by all UDP sockets.

        min: Below this number of pages UDP is not bothered about its
        memory appetite. When amount of memory allocated by UDP exceeds
        this number, UDP starts to moderate memory usage.

        pressure: This value was introduced to follow format of tcp_mem.

        max: Number of pages allowed for queueing by all UDP sockets.

        Default is calculated at boot time from amount of available memory.

udp_rmem_min - INTEGER
        Minimal size of receive buffer used by UDP sockets in moderation.
        Each UDP socket is able to use the size for receiving data, even if
        total pages of UDP sockets exceed udp_mem pressure. The unit is byte.
        Default: 4K

udp_wmem_min - INTEGER
        Minimal size of send buffer used by UDP sockets in moderation.
        Each UDP socket is able to use the size for sending data, even if
        total pages of UDP sockets exceed udp_mem pressure. The unit is byte.
        Default: 4K

RAW variables:

raw_l3mdev_accept - BOOLEAN
        Enabling this option allows a "global" bound socket to work
        across L3 master domains (e.g., VRFs) with packets capable of
        being received regardless of the L3 domain in which they
        originated. Only valid when the kernel was compiled with
        CONFIG_NET_L3_MASTER_DEV.
        Default: 1 (enabled)

CIPSOv4 Variables:

cipso_cache_enable - BOOLEAN
        If set, enable additions to and lookups from the CIPSO label mapping
        cache.  If unset, additions are ignored and lookups always result in a
```

```
        miss.  However, regardless of the setting the cache is still
        invalidated when required when means you can safely toggle this on and
        off and the cache will always be "safe".
        Default: 1

cipso_cache_bucket_size - INTEGER
        The CIPSO label cache consists of a fixed size hash table with each
        hash bucket containing a number of cache entries.  This variable limits
        the number of entries in each hash bucket; the larger the value the
        more CIPSO label mappings that can be cached.  When the number of
        entries in a given hash bucket reaches this limit adding new entries
        causes the oldest entry in the bucket to be removed to make room.
        Default: 10

cipso_rbm_optfmt - BOOLEAN
        Enable the "Optimized Tag 1 Format" as defined in section 3.4.2.6 of
        the CIPSO draft specification (see Documentation/netlabel for details).
        This means that when set the CIPSO tag will be padded with empty
        categories in order to make the packet data 32-bit aligned.
        Default: 0

cipso_rbm_structvalid - BOOLEAN
        If set, do a very strict check of the CIPSO option when
        ip_options_compile() is called.  If unset, relax the checks done during
        ip_options_compile().  Either way is "safe" as errors are caught else
        where in the CIPSO processing code but setting this to 0 (False) should
        result in less work (i.e. it should be faster) but could cause problems
        with other implementations that require strict checking.
        Default: 0

IP Variables:

ip_local_port_range - 2 INTEGERS
        Defines the local port range that is used by TCP and UDP to
        choose the local port. The first number is the first, the
        second the last local port number.
        If possible, it is better these numbers have different parity.
        (one even and one odd values)
        The default values are 32768 and 60999 respectively.

ip_local_reserved_ports - list of comma separated ranges
        Specify the ports which are reserved for known third-party
        applications. These ports will not be used by automatic port
        assignments (e.g. when calling connect() or bind() with port
        number 0). Explicit port allocation behavior is unchanged.

        The format used for both input and output is a comma separated
        list of ranges (e.g. "1,2-4,10-10" for ports 1, 2, 3, 4 and
        10). Writing to the file will clear all previously reserved
        ports and update the current list with the one given in the
        input.

        Note that ip_local_port_range and ip_local_reserved_ports
        settings are independent and both are considered by the kernel
        when determining which ports are available for automatic port
        assignments.

        You can reserve ports which are not in the current
        ip_local_port_range, e.g.:

        $ cat /proc/sys/net/ipv4/ip_local_port_range
        32000   60999
        $ cat /proc/sys/net/ipv4/ip_local_reserved_ports
```

```
        8080,9148

        although this is redundant. However such a setting is useful
        if later the port range is changed to a value that will
        include the reserved ports.

        Default: Empty

ip_unprivileged_port_start - INTEGER
        This is a per-namespace sysctl.  It defines the first
        unprivileged port in the network namespace.  Privileged ports
        require root or CAP_NET_BIND_SERVICE in order to bind to them.
        To disable all privileged ports, set this to 0.  It may not
        overlap with the ip_local_reserved_ports range.

        Default: 1024

ip_nonlocal_bind - BOOLEAN
        If set, allows processes to bind() to non-local IP addresses,
        which can be quite useful - but may break some applications.
        Default: 0

ip_dynaddr - BOOLEAN
        If set non-zero, enables support for dynamic addresses.
        If set to a non-zero value larger than 1, a kernel log
        message will be printed when dynamic address rewriting
        occurs.
        Default: 0

ip_early_demux - BOOLEAN
        Optimize input packet processing down to one demux for
        certain kinds of local sockets.  Currently we only do this
        for established TCP and connected UDP sockets.

        It may add an additional cost for pure routing workloads that
        reduces overall throughput, in such case you should disable it.
        Default: 1

tcp_early_demux - BOOLEAN
        Enable early demux for established TCP sockets.
        Default: 1

udp_early_demux - BOOLEAN
        Enable early demux for connected UDP sockets. Disable this if
        your system could experience more unconnected load.
        Default: 1

icmp_echo_ignore_all - BOOLEAN
        If set non-zero, then the kernel will ignore all ICMP ECHO
        requests sent to it.
        Default: 0

icmp_echo_ignore_broadcasts - BOOLEAN
        If set non-zero, then the kernel will ignore all ICMP ECHO and
        TIMESTAMP requests sent to it via broadcast/multicast.
        Default: 1

icmp_ratelimit - INTEGER
        Limit the maximal rates for sending ICMP packets whose type matches
        icmp_ratemask (see below) to specific targets.
        0 to disable any limiting,
        otherwise the minimal space between responses in milliseconds.
        Note that another sysctl, icmp_msgs_per_sec limits the number
```

```
        of ICMP packets sent on all targets.
        Default: 1000

icmp_msgs_per_sec - INTEGER
        Limit maximal number of ICMP packets sent per second from this host.
        Only messages whose type matches icmp_ratemask (see below) are
        controlled by this limit.
        Default: 1000

icmp_msgs_burst - INTEGER
        icmp_msgs_per_sec controls number of ICMP packets sent per second,
        while icmp_msgs_burst controls the burst size of these packets.
        Default: 50

icmp_ratemask - INTEGER
        Mask made of ICMP types for which rates are being limited.
        Significant bits: IHGFEDCBA9876543210
        Default mask:     0000001100000011000 (6168)

        Bit definitions (see include/linux/icmp.h):
                0 Echo Reply
                3 Destination Unreachable *
                4 Source Quench *
                5 Redirect
                8 Echo Request
                B Time Exceeded *
                C Parameter Problem *
                D Timestamp Request
                E Timestamp Reply
                F Info Request
                G Info Reply
                H Address Mask Request
                I Address Mask Reply

        * These are rate limited by default (see default mask above)

icmp_ignore_bogus_error_responses - BOOLEAN
        Some routers violate RFC1122 by sending bogus responses to broadcast
        frames.  Such violations are normally logged via a kernel warning.
        If this is set to TRUE, the kernel will not give such warnings, which
        will avoid log file clutter.
        Default: 1

icmp_errors_use_inbound_ifaddr - BOOLEAN

        If zero, icmp error messages are sent with the primary address of
        the exiting interface.

        If non-zero, the message will be sent with the primary address of
        the interface that received the packet that caused the icmp error.
        This is the behaviour network many administrators will expect from
        a router. And it can make debugging complicated network layouts
        much easier.

        Note that if no primary address exists for the interface selected,
        then the primary address of the first non-loopback interface that
        has one will be used regardless of this setting.

        Default: 0

igmp_max_memberships - INTEGER
        Change the maximum number of multicast groups we can subscribe to.
        Default: 20
```

```
        Theoretical maximum value is bounded by having to send a membership
        report in a single datagram (i.e. the report can't span multiple
        datagrams, or risk confusing the switch and leaving groups you don't
        intend to).

        The number of supported groups 'M' is bounded by the number of group
        report entries you can fit into a single datagram of 65535 bytes.

        M = 65536-sizeof (ip header)/(sizeof(Group record))

        Group records are variable length, with a minimum of 12 bytes.
        So net.ipv4.igmp_max_memberships should not be set higher than:

        (65536-24) / 12 = 5459

        The value 5459 assumes no IP header options, so in practice
        this number may be lower.

igmp_max_msf - INTEGER
        Maximum number of addresses allowed in the source filter list for a
        multicast group.
        Default: 10

igmp_qrv - INTEGER
        Controls the IGMP query robustness variable (see RFC2236 8.1).
        Default: 2 (as specified by RFC2236 8.1)
        Minimum: 1 (as specified by RFC6636 4.5)

force_igmp_version - INTEGER
        0 - (default) No enforcement of a IGMP version, IGMPv1/v2 fallback
            allowed. Will back to IGMPv3 mode again if all IGMPv1/v2 Querier
            Present timer expires.
        1 - Enforce to use IGMP version 1. Will also reply IGMPv1 report if
            receive IGMPv2/v3 query.
        2 - Enforce to use IGMP version 2. Will fallback to IGMPv1 if receive
            IGMPv1 query message. Will reply report if receive IGMPv3 query.
        3 - Enforce to use IGMP version 3. The same react with default 0.

        Note: this is not the same with force_mld_version because IGMPv3 RFC3376
        Security Considerations does not have clear description that we could
        ignore other version messages completely as MLDv2 RFC3810. So make
        this value as default 0 is recommended.

conf/interface/*  changes special settings per interface (where
"interface" is the name of your network interface)

conf/all/*        is special, changes the settings for all interfaces

log_martians - BOOLEAN
        Log packets with impossible addresses to kernel log.
        log_martians for the interface will be enabled if at least one of
        conf/{all,interface}/log_martians is set to TRUE,
        it will be disabled otherwise

accept_redirects - BOOLEAN
        Accept ICMP redirect messages.
        accept_redirects for the interface will be enabled if:
        - both conf/{all,interface}/accept_redirects are TRUE in the case
          forwarding for the interface is enabled
        or
        - at least one of conf/{all,interface}/accept_redirects is TRUE in the
          case forwarding for the interface is disabled
```

```
        accept_redirects for the interface will be disabled otherwise
        default TRUE (host)
                FALSE (router)

forwarding - BOOLEAN
        Enable IP forwarding on this interface.  This controls whether packets
        received _on_ this interface can be forwarded.

mc_forwarding - BOOLEAN
        Do multicast routing. The kernel needs to be compiled with CONFIG_MROUTE
        and a multicast routing daemon is required.
        conf/all/mc_forwarding must also be set to TRUE to enable multicast
        routing for the interface

medium_id - INTEGER
        Integer value used to differentiate the devices by the medium they
        are attached to. Two devices can have different id values when
        the broadcast packets are received only on one of them.
        The default value 0 means that the device is the only interface
        to its medium, value of -1 means that medium is not known.

        Currently, it is used to change the proxy_arp behavior:
        the proxy_arp feature is enabled for packets forwarded between
        two devices attached to different media.

proxy_arp - BOOLEAN
        Do proxy arp.
        proxy_arp for the interface will be enabled if at least one of
        conf/{all,interface}/proxy_arp is set to TRUE,
        it will be disabled otherwise

proxy_arp_pvlan - BOOLEAN
        Private VLAN proxy arp.
        Basically allow proxy arp replies back to the same interface
        (from which the ARP request/solicitation was received).

        This is done to support (ethernet) switch features, like RFC
        3069, where the individual ports are NOT allowed to
        communicate with each other, but they are allowed to talk to
        the upstream router.  As described in RFC 3069, it is possible
        to allow these hosts to communicate through the upstream
        router by proxy_arp'ing. Don't need to be used together with
        proxy_arp.

        This technology is known by different names:
          In RFC 3069 it is called VLAN Aggregation.
          Cisco and Allied Telesyn call it Private VLAN.
          Hewlett-Packard call it Source-Port filtering or port-isolation.
          Ericsson call it MAC-Forced Forwarding (RFC Draft).

shared_media - BOOLEAN
        Send(router) or accept(host) RFC1620 shared media redirects.
        Overrides secure_redirects.
        shared_media for the interface will be enabled if at least one of
        conf/{all,interface}/shared_media is set to TRUE,
        it will be disabled otherwise
        default TRUE

secure_redirects - BOOLEAN
        Accept ICMP redirect messages only to gateways listed in the
        interface's current gateway list. Even if disabled, RFC1122 redirect
        rules still apply.
        Overridden by shared_media.
```

```
        secure_redirects for the interface will be enabled if at least one of
        conf/{all,interface}/secure_redirects is set to TRUE,
        it will be disabled otherwise
        default TRUE

send_redirects - BOOLEAN
        Send redirects, if router.
        send_redirects for the interface will be enabled if at least one of
        conf/{all,interface}/send_redirects is set to TRUE,
        it will be disabled otherwise
        Default: TRUE

bootp_relay - BOOLEAN
        Accept packets with source address 0.b.c.d destined
        not to this host as local ones. It is supposed, that
        BOOTP relay daemon will catch and forward such packets.
        conf/all/bootp_relay must also be set to TRUE to enable BOOTP relay
        for the interface
        default FALSE
        Not Implemented Yet.

accept_source_route - BOOLEAN
        Accept packets with SRR option.
        conf/all/accept_source_route must also be set to TRUE to accept packets
        with SRR option on the interface
        default TRUE (router)
                FALSE (host)

accept_local - BOOLEAN
        Accept packets with local source addresses. In combination with
        suitable routing, this can be used to direct packets between two
        local interfaces over the wire and have them accepted properly.
        default FALSE

route_localnet - BOOLEAN
        Do not consider loopback addresses as martian source or destination
        while routing. This enables the use of 127/8 for local routing purposes.
        default FALSE

rp_filter - INTEGER
        0 - No source validation.
        1 - Strict mode as defined in RFC3704 Strict Reverse Path
            Each incoming packet is tested against the FIB and if the interface
            is not the best reverse path the packet check will fail.
            By default failed packets are discarded.
        2 - Loose mode as defined in RFC3704 Loose Reverse Path
            Each incoming packet's source address is also tested against the FIB
            and if the source address is not reachable via any interface
            the packet check will fail.

        Current recommended practice in RFC3704 is to enable strict mode
        to prevent IP spoofing from DDos attacks. If using asymmetric routing
        or other complicated routing, then loose mode is recommended.

        The max value from conf/{all,interface}/rp_filter is used
        when doing source validation on the {interface}.

        Default value is 0. Note that some distributions enable it
        in startup scripts.

arp_filter - BOOLEAN
        1 - Allows you to have multiple network interfaces on the same
        subnet, and have the ARPs for each interface be answered
```

based on whether or not the kernel would route a packet from
the ARP'd IP out that interface (therefore you must use source
based routing for this to work). In other words it allows control
of which cards (usually 1) will respond to an arp request.

0 - (default) The kernel can respond to arp requests with addresses
from other interfaces. This may seem wrong but it usually makes
sense, because it increases the chance of successful communication.
IP addresses are owned by the complete host on Linux, not by
particular interfaces. Only for more complex setups like load-
balancing, does this behaviour cause problems.

arp_filter for the interface will be enabled if at least one of
conf/{all,interface}/arp_filter is set to TRUE,
it will be disabled otherwise

arp_announce - INTEGER
Define different restriction levels for announcing the local
source IP address from IP packets in ARP requests sent on
interface:
0 - (default) Use any local address, configured on any interface
1 - Try to avoid local addresses that are not in the target's
subnet for this interface. This mode is useful when target
hosts reachable via this interface require the source IP
address in ARP requests to be part of their logical network
configured on the receiving interface. When we generate the
request we will check all our subnets that include the
target IP and will preserve the source address if it is from
such subnet. If there is no such subnet we select source
address according to the rules for level 2.
2 - Always use the best local address for this target.
In this mode we ignore the source address in the IP packet
and try to select local address that we prefer for talks with
the target host. Such local address is selected by looking
for primary IP addresses on all our subnets on the outgoing
interface that include the target IP address. If no suitable
local address is found we select the first local address
we have on the outgoing interface or on all other interfaces,
with the hope we will receive reply for our request and
even sometimes no matter the source IP address we announce.

The max value from conf/{all,interface}/arp_announce is used.

Increasing the restriction level gives more chance for
receiving answer from the resolved target while decreasing
the level announces more valid sender's information.

arp_ignore - INTEGER
Define different modes for sending replies in response to
received ARP requests that resolve local target IP addresses:
0 - (default): reply for any local target IP address, configured
on any interface
1 - reply only if the target IP address is local address
configured on the incoming interface
2 - reply only if the target IP address is local address
configured on the incoming interface and both with the
sender's IP address are part from same subnet on this interface
3 - do not reply for local addresses configured with scope host,
only resolutions for global and link addresses are replied
4-7 - reserved
8 - do not reply for all local addresses

The max value from conf/{all,interface}/arp_ignore is used

```
        when ARP request is received on the {interface}

arp_notify - BOOLEAN
        Define mode for notification of address and device changes.
        0 - (default): do nothing
        1 - Generate gratuitous arp requests when device is brought up
            or hardware address changes.

arp_accept - BOOLEAN
        Define behavior for gratuitous ARP frames who's IP is not
        already present in the ARP table:
        0 - don't create new entries in the ARP table
        1 - create new entries in the ARP table

        Both replies and requests type gratuitous arp will trigger the
        ARP table to be updated, if this setting is on.

        If the ARP table already contains the IP address of the
        gratuitous arp frame, the arp table will be updated regardless
        if this setting is on or off.

mcast_solicit - INTEGER
        The maximum number of multicast probes in INCOMPLETE state,
        when the associated hardware address is unknown.  Defaults
        to 3.

ucast_solicit - INTEGER
        The maximum number of unicast probes in PROBE state, when
        the hardware address is being reconfirmed.  Defaults to 3.

app_solicit - INTEGER
        The maximum number of probes to send to the user space ARP daemon
        via netlink before dropping back to multicast probes (see
        mcast_resolicit).  Defaults to 0.

mcast_resolicit - INTEGER
        The maximum number of multicast probes after unicast and
        app probes in PROBE state.  Defaults to 0.

disable_policy - BOOLEAN
        Disable IPSEC policy (SPD) for this interface

disable_xfrm - BOOLEAN
        Disable IPSEC encryption on this interface, whatever the policy

igmpv2_unsolicited_report_interval - INTEGER
        The interval in milliseconds in which the next unsolicited
        IGMPv1 or IGMPv2 report retransmit will take place.
        Default: 10000 (10 seconds)

igmpv3_unsolicited_report_interval - INTEGER
        The interval in milliseconds in which the next unsolicited
        IGMPv3 report retransmit will take place.
        Default: 1000 (1 seconds)

promote_secondaries - BOOLEAN
        When a primary IP address is removed from this interface
        promote a corresponding secondary IP address instead of
        removing all the corresponding secondary IP addresses.

drop_unicast_in_l2_multicast - BOOLEAN
        Drop any unicast IP packets that are received in link-layer
        multicast (or broadcast) frames.
```

```
        This behavior (for multicast) is actually a SHOULD in RFC
        1122, but is disabled by default for compatibility reasons.
        Default: off (0)

drop_gratuitous_arp - BOOLEAN
        Drop all gratuitous ARP frames, for example if there's a known
        good ARP proxy on the network and such frames need not be used
        (or in the case of 802.11, must not be used to prevent attacks.)
        Default: off (0)


tag - INTEGER
        Allows you to write a number, which can be used as required.
        Default value is 0.

xfrm4_gc_thresh - INTEGER
        The threshold at which we will start garbage collecting for IPv4
        destination cache entries.  At twice this value the system will
        refuse new allocations.

igmp_link_local_mcast_reports - BOOLEAN
        Enable IGMP reports for link local multicast groups in the
        224.0.0.X range.
        Default TRUE

Alexey Kuznetsov.
kuznet@ms2.inr.ac.ru

Updated by:
Andi Kleen
ak@muc.de
Nicolas Delon
delon.nicolas@wanadoo.fr




/proc/sys/net/ipv6/* Variables:

IPv6 has no global variables such as tcp_*.  tcp_* settings under ipv4/ also
apply to IPv6 [XXX?].

bindv6only - BOOLEAN
        Default value for IPV6_V6ONLY socket option,
        which restricts use of the IPv6 socket to IPv6 communication
        only.
                TRUE: disable IPv4-mapped address feature
                FALSE: enable IPv4-mapped address feature

        Default: FALSE (as specified in RFC3493)

flowlabel_consistency - BOOLEAN
        Protect the consistency (and unicity) of flow label.
        You have to disable it to use IPV6_FL_F_REFLECT flag on the
        flow label manager.
        TRUE: enabled
        FALSE: disabled
        Default: TRUE

auto_flowlabels - INTEGER
        Automatically generate flow labels based on a flow hash of the
        packet. This allows intermediate devices, such as routers, to
        identify packet flows for mechanisms like Equal Cost Multipath
```

```
        Routing (see RFC 6438).
        0: automatic flow labels are completely disabled
        1: automatic flow labels are enabled by default, they can be
           disabled on a per socket basis using the IPV6_AUTOFLOWLABEL
           socket option
        2: automatic flow labels are allowed, they may be enabled on a
           per socket basis using the IPV6_AUTOFLOWLABEL socket option
        3: automatic flow labels are enabled and enforced, they cannot
           be disabled by the socket option
        Default: 1

flowlabel_state_ranges - BOOLEAN
        Split the flow label number space into two ranges. 0-0x7FFFF is
        reserved for the IPv6 flow manager facility, 0x80000-0xFFFFF
        is reserved for stateless flow labels as described in RFC6437.
        TRUE: enabled
        FALSE: disabled
        Default: true

flowlabel_reflect - BOOLEAN
        Automatically reflect the flow label. Needed for Path MTU
        Discovery to work with Equal Cost Multipath Routing in anycast
        environments. See RFC 7690 and:
        https://tools.ietf.org/html/draft-wang-6man-flow-label-reflection-01
        TRUE: enabled
        FALSE: disabled
        Default: FALSE

fib_multipath_hash_policy - INTEGER
        Controls which hash policy to use for multipath routes.
        Default: 0 (Layer 3)
        Possible values:
        0 - Layer 3 (source and destination addresses plus flow label)
        1 - Layer 4 (standard 5-tuple)

anycast_src_echo_reply - BOOLEAN
        Controls the use of anycast addresses as source addresses for ICMPv6
        echo reply
        TRUE:  enabled
        FALSE: disabled
        Default: FALSE

idgen_delay - INTEGER
        Controls the delay in seconds after which time to retry
        privacy stable address generation if a DAD conflict is
        detected.
        Default: 1 (as specified in RFC7217)

idgen_retries - INTEGER
        Controls the number of retries to generate a stable privacy
        address if a DAD conflict is detected.
        Default: 3 (as specified in RFC7217)

mld_qrv - INTEGER
        Controls the MLD query robustness variable (see RFC3810 9.1).
        Default: 2 (as specified by RFC3810 9.1)
        Minimum: 1 (as specified by RFC6636 4.5)

max_dst_opts_number - INTEGER
        Maximum number of non-padding TLVs allowed in a Destination
        options extension header. If this value is less than zero
        then unknown options are disallowed and the number of known
        TLVs allowed is the absolute value of this number.
```

```
        Default: 8

max_hbh_opts_number - INTEGER
        Maximum number of non-padding TLVs allowed in a Hop-by-Hop
        options extension header. If this value is less than zero
        then unknown options are disallowed and the number of known
        TLVs allowed is the absolute value of this number.
        Default: 8

max_dst_opts_length - INTEGER
        Maximum length allowed for a Destination options extension
        header.
        Default: INT_MAX (unlimited)

max_hbh_length - INTEGER
        Maximum length allowed for a Hop-by-Hop options extension
        header.
        Default: INT_MAX (unlimited)

skip_notify_on_dev_down - BOOLEAN
        Controls whether an RTM_DELROUTE message is generated for routes
        removed when a device is taken down or deleted. IPv4 does not
        generate this message; IPv6 does by default. Setting this sysctl
        to true skips the message, making IPv4 and IPv6 on par in relying
        on userspace caches to track link events and evict routes.
        Default: false (generate message)

IPv6 Fragmentation:

ip6frag_high_thresh - INTEGER
        Maximum memory used to reassemble IPv6 fragments. When
        ip6frag_high_thresh bytes of memory is allocated for this purpose,
        the fragment handler will toss packets until ip6frag_low_thresh
        is reached.

ip6frag_low_thresh - INTEGER
        See ip6frag_high_thresh

ip6frag_time - INTEGER
        Time in seconds to keep an IPv6 fragment in memory.

IPv6 Segment Routing:

seg6_flowlabel - INTEGER
        Controls the behaviour of computing the flowlabel of outer
        IPv6 header in case of SR T.encaps

        -1 set flowlabel to zero.
        0 copy flowlabel from Inner packet in case of Inner IPv6
                (Set flowlabel to 0 in case IPv4/L2)
        1 Compute the flowlabel using seg6_make_flowlabel()

        Default is 0.

conf/default/*:
        Change the interface-specific default settings.


conf/all/*:
        Change all the interface-specific settings.

        [XXX:  Other special features than forwarding?]
```

```
conf/all/forwarding - BOOLEAN
        Enable global IPv6 forwarding between all interfaces.

        IPv4 and IPv6 work differently here; e.g. netfilter must be used
        to control which interfaces may forward packets and which not.

        This also sets all interfaces' Host/Router setting
        'forwarding' to the specified value.  See below for details.

        This referred to as global forwarding.

proxy_ndp - BOOLEAN
        Do proxy ndp.

fwmark_reflect - BOOLEAN
        Controls the fwmark of kernel-generated IPv6 reply packets that are not
        associated with a socket for example, TCP RSTs or ICMPv6 echo replies).
        If unset, these packets have a fwmark of zero. If set, they have the
        fwmark of the packet they are replying to.
        Default: 0

conf/interface/*:
        Change special settings per interface.

        The functional behaviour for certain settings is different
        depending on whether local forwarding is enabled or not.

accept_ra - INTEGER
        Accept Router Advertisements; autoconfigure using them.

        It also determines whether or not to transmit Router
        Solicitations. If and only if the functional setting is to
        accept Router Advertisements, Router Solicitations will be
        transmitted.

        Possible values are:
                0 Do not accept Router Advertisements.
                1 Accept Router Advertisements if forwarding is disabled.
                2 Overrule forwarding behaviour. Accept Router Advertisements
                  even if forwarding is enabled.

        Functional default: enabled if local forwarding is disabled.
                            disabled if local forwarding is enabled.

accept_ra_defrtr - BOOLEAN
        Learn default router in Router Advertisement.

        Functional default: enabled if accept_ra is enabled.
                            disabled if accept_ra is disabled.

accept_ra_from_local - BOOLEAN
        Accept RA with source-address that is found on local machine
        if the RA is otherwise proper and able to be accepted.
        Default is to NOT accept these as it may be an un-intended
        network loop.

        Functional default:
           enabled if accept_ra_from_local is enabled
               on a specific interface.
           disabled if accept_ra_from_local is disabled
               on a specific interface.

accept_ra_min_hop_limit - INTEGER
```

```
        Minimum hop limit Information in Router Advertisement.

        Hop limit Information in Router Advertisement less than this
        variable shall be ignored.

        Default: 1

accept_ra_pinfo - BOOLEAN
        Learn Prefix Information in Router Advertisement.

        Functional default: enabled if accept_ra is enabled.
                            disabled if accept_ra is disabled.

accept_ra_rt_info_min_plen - INTEGER
        Minimum prefix length of Route Information in RA.

        Route Information w/ prefix smaller than this variable shall
        be ignored.

        Functional default: 0 if accept_ra_rtr_pref is enabled.
                            -1 if accept_ra_rtr_pref is disabled.

accept_ra_rt_info_max_plen - INTEGER
        Maximum prefix length of Route Information in RA.

        Route Information w/ prefix larger than this variable shall
        be ignored.

        Functional default: 0 if accept_ra_rtr_pref is enabled.
                            -1 if accept_ra_rtr_pref is disabled.

accept_ra_rtr_pref - BOOLEAN
        Accept Router Preference in RA.

        Functional default: enabled if accept_ra is enabled.
                            disabled if accept_ra is disabled.

accept_ra_mtu - BOOLEAN
        Apply the MTU value specified in RA option 5 (RFC4861). If
        disabled, the MTU specified in the RA will be ignored.

        Functional default: enabled if accept_ra is enabled.
                            disabled if accept_ra is disabled.

accept_redirects - BOOLEAN
        Accept Redirects.

        Functional default: enabled if local forwarding is disabled.
                            disabled if local forwarding is enabled.

accept_source_route - INTEGER
        Accept source routing (routing extension header).

        >= 0: Accept only routing header type 2.
        < 0: Do not accept routing header.

        Default: 0

autoconf - BOOLEAN
        Autoconfigure addresses using Prefix Information in Router
        Advertisements.

        Functional default: enabled if accept_ra_pinfo is enabled.
```

```
                                disabled if accept_ra_pinfo is disabled.

dad_transmits - INTEGER
        The amount of Duplicate Address Detection probes to send.
        Default: 1

forwarding - INTEGER
        Configure interface-specific Host/Router behaviour.

        Note: It is recommended to have the same setting on all
        interfaces; mixed router/host scenarios are rather uncommon.

        Possible values are:
                0 Forwarding disabled
                1 Forwarding enabled

        FALSE (0):

        By default, Host behaviour is assumed.  This means:

        1. IsRouter flag is not set in Neighbour Advertisements.
        2. If accept_ra is TRUE (default), transmit Router
           Solicitations.
        3. If accept_ra is TRUE (default), accept Router
           Advertisements (and do autoconfiguration).
        4. If accept_redirects is TRUE (default), accept Redirects.

        TRUE (1):

        If local forwarding is enabled, Router behaviour is assumed.
        This means exactly the reverse from the above:

        1. IsRouter flag is set in Neighbour Advertisements.
        2. Router Solicitations are not sent unless accept_ra is 2.
        3. Router Advertisements are ignored unless accept_ra is 2.
        4. Redirects are ignored.

        Default: 0 (disabled) if global forwarding is disabled (default),
                 otherwise 1 (enabled).

hop_limit - INTEGER
        Default Hop Limit to set.
        Default: 64

mtu - INTEGER
        Default Maximum Transfer Unit
        Default: 1280 (IPv6 required minimum)

ip_nonlocal_bind - BOOLEAN
        If set, allows processes to bind() to non-local IPv6 addresses,
        which can be quite useful - but may break some applications.
        Default: 0

router_probe_interval - INTEGER
        Minimum interval (in seconds) between Router Probing described
        in RFC4191.

        Default: 60

router_solicitation_delay - INTEGER
        Number of seconds to wait after interface is brought up
        before sending Router Solicitations.
        Default: 1
```

```
router_solicitation_interval - INTEGER
        Number of seconds to wait between Router Solicitations.
        Default: 4

router_solicitations - INTEGER
        Number of Router Solicitations to send until assuming no
        routers are present.
        Default: 3

use_oif_addrs_only - BOOLEAN
        When enabled, the candidate source addresses for destinations
        routed via this interface are restricted to the set of addresses
        configured on this interface (vis. RFC 6724, section 4).

        Default: false

use_tempaddr - INTEGER
        Preference for Privacy Extensions (RFC3041).
          <= 0 : disable Privacy Extensions
          == 1 : enable Privacy Extensions, but prefer public
                 addresses over temporary addresses.
          >  1 : enable Privacy Extensions and prefer temporary
                 addresses over public addresses.
        Default:  0 (for most devices)
                 -1 (for point-to-point devices and loopback devices)

temp_valid_lft - INTEGER
        valid lifetime (in seconds) for temporary addresses.
        Default: 604800 (7 days)

temp_prefered_lft - INTEGER
        Preferred lifetime (in seconds) for temporary addresses.
        Default: 86400 (1 day)

keep_addr_on_down - INTEGER
        Keep all IPv6 addresses on an interface down event. If set static
        global addresses with no expiration time are not flushed.
          >0 : enabled
           0 : system default
          <0 : disabled

        Default: 0 (addresses are removed)

max_desync_factor - INTEGER
        Maximum value for DESYNC_FACTOR, which is a random value
        that ensures that clients don't synchronize with each
        other and generate new addresses at exactly the same time.
        value is in seconds.
        Default: 600

regen_max_retry - INTEGER
        Number of attempts before give up attempting to generate
        valid temporary addresses.
        Default: 5

max_addresses - INTEGER
        Maximum number of autoconfigured addresses per interface.  Setting
        to zero disables the limitation.  It is not recommended to set this
        value too large (or to zero) because it would be an easy way to
        crash the kernel by allowing too many addresses to be created.
        Default: 16
```

```
disable_ipv6 - BOOLEAN
        Disable IPv6 operation.  If accept_dad is set to 2, this value
        will be dynamically set to TRUE if DAD fails for the link-local
        address.
        Default: FALSE (enable IPv6 operation)

        When this value is changed from 1 to 0 (IPv6 is being enabled),
        it will dynamically create a link-local address on the given
        interface and start Duplicate Address Detection, if necessary.

        When this value is changed from 0 to 1 (IPv6 is being disabled),
        it will dynamically delete all addresses and routes on the given
        interface. From now on it will not possible to add addresses/routes
        to the selected interface.

accept_dad - INTEGER
        Whether to accept DAD (Duplicate Address Detection).
        0: Disable DAD
        1: Enable DAD (default)
        2: Enable DAD, and disable IPv6 operation if MAC-based duplicate
           link-local address has been found.

        DAD operation and mode on a given interface will be selected according
        to the maximum value of conf/{all,interface}/accept_dad.

force_tllao - BOOLEAN
        Enable sending the target link-layer address option even when
        responding to a unicast neighbor solicitation.
        Default: FALSE

        Quoting from RFC 2461, section 4.4, Target link-layer address:

        "The option MUST be included for multicast solicitations in order to
        avoid infinite Neighbor Solicitation "recursion" when the peer node
        does not have a cache entry to return a Neighbor Advertisements
        message.  When responding to unicast solicitations, the option can be
        omitted since the sender of the solicitation has the correct link-
        layer address; otherwise it would not have be able to send the unicast
        solicitation in the first place. However, including the link-layer
        address in this case adds little overhead and eliminates a potential
        race condition where the sender deletes the cached link-layer address
        prior to receiving a response to a previous solicitation."

ndisc_notify - BOOLEAN
        Define mode for notification of address and device changes.
        0 - (default): do nothing
        1 - Generate unsolicited neighbour advertisements when device is brought
            up or hardware address changes.

ndisc_tclass - INTEGER
        The IPv6 Traffic Class to use by default when sending IPv6 Neighbor
        Discovery (Router Solicitation, Router Advertisement, Neighbor
        Solicitation, Neighbor Advertisement, Redirect) messages.
        These 8 bits can be interpreted as 6 high order bits holding the DSCP
        value and 2 low order bits representing ECN (which you probably want
        to leave cleared).
        0 - (default)

mldv1_unsolicited_report_interval - INTEGER
        The interval in milliseconds in which the next unsolicited
        MLDv1 report retransmit will take place.
        Default: 10000 (10 seconds)
```

```
mldv2_unsolicited_report_interval - INTEGER
        The interval in milliseconds in which the next unsolicited
        MLDv2 report retransmit will take place.
        Default: 1000 (1 second)

force_mld_version - INTEGER
        0 - (default) No enforcement of a MLD version, MLDv1 fallback allowed
        1 - Enforce to use MLD version 1
        2 - Enforce to use MLD version 2

suppress_frag_ndisc - INTEGER
        Control RFC 6980 (Security Implications of IPv6 Fragmentation
        with IPv6 Neighbor Discovery) behavior:
        1 - (default) discard fragmented neighbor discovery packets
        0 - allow fragmented neighbor discovery packets

optimistic_dad - BOOLEAN
        Whether to perform Optimistic Duplicate Address Detection (RFC 4429).
        0: disabled (default)
        1: enabled

        Optimistic Duplicate Address Detection for the interface will be enabled
        if at least one of conf/{all,interface}/optimistic_dad is set to 1,
        it will be disabled otherwise.

use_optimistic - BOOLEAN
        If enabled, do not classify optimistic addresses as deprecated during
        source address selection.  Preferred addresses will still be chosen
        before optimistic addresses, subject to other ranking in the source
        address selection algorithm.
        0: disabled (default)
        1: enabled

        This will be enabled if at least one of
        conf/{all,interface}/use_optimistic is set to 1, disabled otherwise.

stable_secret - IPv6 address
        This IPv6 address will be used as a secret to generate IPv6
        addresses for link-local addresses and autoconfigured
        ones. All addresses generated after setting this secret will
        be stable privacy ones by default. This can be changed via the
        addrgenmode ip-link. conf/default/stable_secret is used as the
        secret for the namespace, the interface specific ones can
        overwrite that. Writes to conf/all/stable_secret are refused.

        It is recommended to generate this secret during installation
        of a system and keep it stable after that.

        By default the stable secret is unset.

addr_gen_mode - INTEGER
        Defines how link-local and autoconf addresses are generated.

        0: generate address based on EUI64 (default)
        1: do no generate a link-local address, use EUI64 for addresses generated
           from autoconf
        2: generate stable privacy addresses, using the secret from
           stable_secret (RFC7217)
        3: generate stable privacy addresses, using a random secret if unset

drop_unicast_in_l2_multicast - BOOLEAN
        Drop any unicast IPv6 packets that are received in link-layer
        multicast (or broadcast) frames.
```

```
        By default this is turned off.

drop_unsolicited_na - BOOLEAN
        Drop all unsolicited neighbor advertisements, for example if there's
        a known good NA proxy on the network and such frames need not be used
        (or in the case of 802.11, must not be used to prevent attacks.)

        By default this is turned off.

enhanced_dad - BOOLEAN
        Include a nonce option in the IPv6 neighbor solicitation messages used for
        duplicate address detection per RFC7527. A received DAD NS will only signal
        a duplicate address if the nonce is different. This avoids any false
        detection of duplicates due to loopback of the NS messages that we send.
        The nonce option will be sent on an interface unless both of
        conf/{all,interface}/enhanced_dad are set to FALSE.
        Default: TRUE

icmp/*:
ratelimit - INTEGER
        Limit the maximal rates for sending ICMPv6 packets.
        0 to disable any limiting,
        otherwise the minimal space between responses in milliseconds.
        Default: 1000

echo_ignore_all - BOOLEAN
        If set non-zero, then the kernel will ignore all ICMP ECHO
        requests sent to it over the IPv6 protocol.
        Default: 0

xfrm6_gc_thresh - INTEGER
        The threshold at which we will start garbage collecting for IPv6
        destination cache entries.  At twice this value the system will
        refuse new allocations.


IPv6 Update by:
Pekka Savola <pekkas@netcore.fi>
YOSHIFUJI Hideaki / USAGI Project <yoshfuji@linux-ipv6.org>


/proc/sys/net/bridge/* Variables:

bridge-nf-call-arptables - BOOLEAN
        1 : pass bridged ARP traffic to arptables' FORWARD chain.
        0 : disable this.
        Default: 1

bridge-nf-call-iptables - BOOLEAN
        1 : pass bridged IPv4 traffic to iptables' chains.
        0 : disable this.
        Default: 1

bridge-nf-call-ip6tables - BOOLEAN
        1 : pass bridged IPv6 traffic to ip6tables' chains.
        0 : disable this.
        Default: 1

bridge-nf-filter-vlan-tagged - BOOLEAN
        1 : pass bridged vlan-tagged ARP/IP/IPv6 traffic to {arp,ip,ip6}tables.
        0 : disable this.
        Default: 0
```

```
bridge-nf-filter-pppoe-tagged - BOOLEAN
        1 : pass bridged pppoe-tagged IP/IPv6 traffic to {ip,ip6}tables.
        0 : disable this.
        Default: 0

bridge-nf-pass-vlan-input-dev - BOOLEAN
        1: if bridge-nf-filter-vlan-tagged is enabled, try to find a vlan
        interface on the bridge and set the netfilter input device to the vlan.
        This allows use of e.g. "iptables -i br0.1" and makes the REDIRECT
        target work with vlan-on-top-of-bridge interfaces.  When no matching
        vlan interface is found, or this switch is off, the input device is
        set to the bridge interface.
        0: disable bridge netfilter vlan interface lookup.
        Default: 0

proc/sys/net/sctp/* Variables:

addip_enable - BOOLEAN
        Enable or disable extension of  Dynamic Address Reconfiguration
        (ADD-IP) functionality specified in RFC5061.  This extension provides
        the ability to dynamically add and remove new addresses for the SCTP
        associations.

        1: Enable extension.

        0: Disable extension.

        Default: 0

pf_enable - INTEGER
        Enable or disable pf (pf is short for potentially failed) state. A value
        of pf_retrans > path_max_retrans also disables pf state. That is, one of
        both pf_enable and pf_retrans > path_max_retrans can disable pf state.
        Since pf_retrans and path_max_retrans can be changed by userspace
        application, sometimes user expects to disable pf state by the value of
        pf_retrans > path_max_retrans, but occasionally the value of pf_retrans
        or path_max_retrans is changed by the user application, this pf state is
        enabled. As such, it is necessary to add this to dynamically enable
        and disable pf state. See:
        https://datatracker.ietf.org/doc/draft-ietf-tsvwg-sctp-failover for
        details.

        1: Enable pf.

        0: Disable pf.

        Default: 1

addip_noauth_enable - BOOLEAN
        Dynamic Address Reconfiguration (ADD-IP) requires the use of
        authentication to protect the operations of adding or removing new
        addresses.  This requirement is mandated so that unauthorized hosts
        would not be able to hijack associations.  However, older
        implementations may not have implemented this requirement while
        allowing the ADD-IP extension.  For reasons of interoperability,
        we provide this variable to control the enforcement of the
        authentication requirement.

        1: Allow ADD-IP extension to be used without authentication.  This
           should only be set in a closed environment for interoperability
           with older implementations.
```

```
        0: Enforce the authentication requirement

        Default: 0

auth_enable - BOOLEAN
        Enable or disable Authenticated Chunks extension.  This extension
        provides the ability to send and receive authenticated chunks and is
        required for secure operation of Dynamic Address Reconfiguration
        (ADD-IP) extension.

        1: Enable this extension.
        0: Disable this extension.

        Default: 0

prsctp_enable - BOOLEAN
        Enable or disable the Partial Reliability extension (RFC3758) which
        is used to notify peers that a given DATA should no longer be expected.

        1: Enable extension
        0: Disable

        Default: 1

max_burst - INTEGER
        The limit of the number of new packets that can be initially sent.  It
        controls how bursty the generated traffic can be.

        Default: 4

association_max_retrans - INTEGER
        Set the maximum number for retransmissions that an association can
        attempt deciding that the remote end is unreachable.  If this value
        is exceeded, the association is terminated.

        Default: 10

max_init_retransmits - INTEGER
        The maximum number of retransmissions of INIT and COOKIE-ECHO chunks
        that an association will attempt before declaring the destination
        unreachable and terminating.

        Default: 8

path_max_retrans - INTEGER
        The maximum number of retransmissions that will be attempted on a given
        path.  Once this threshold is exceeded, the path is considered
        unreachable, and new traffic will use a different path when the
        association is multihomed.

        Default: 5

pf_retrans - INTEGER
        The number of retransmissions that will be attempted on a given path
        before traffic is redirected to an alternate transport (should one
        exist).  Note this is distinct from path_max_retrans, as a path that
        passes the pf_retrans threshold can still be used.  Its only
        deprioritized when a transmission path is selected by the stack.  This
        setting is primarily used to enable fast failover mechanisms without
        having to reduce path_max_retrans to a very low value.  See:
        http://www.ietf.org/id/draft-nishida-tsvwg-sctp-failover-05.txt
        for details.  Note also that a value of pf_retrans > path_max_retrans
        disables this feature. Since both pf_retrans and path_max_retrans can
```

be changed by userspace application, a variable pf_enable is used to
disable pf state.

Default: 0

rto_initial - INTEGER
The initial round trip timeout value in milliseconds that will be used
in calculating round trip times. This is the initial time interval
for retransmissions.

Default: 3000

rto_max - INTEGER
The maximum value (in milliseconds) of the round trip timeout. This
is the largest time interval that can elapse between retransmissions.

Default: 60000

rto_min - INTEGER
The minimum value (in milliseconds) of the round trip timeout. This
is the smallest time interval the can elapse between retransmissions.

Default: 1000

hb_interval - INTEGER
The interval (in milliseconds) between HEARTBEAT chunks. These chunks
are sent at the specified interval on idle paths to probe the state of
a given path between 2 associations.

Default: 30000

sack_timeout - INTEGER
The amount of time (in milliseconds) that the implementation will wait
to send a SACK.

Default: 200

valid_cookie_life - INTEGER
The default lifetime of the SCTP cookie (in milliseconds). The cookie
is used during association establishment.

Default: 60000

cookie_preserve_enable - BOOLEAN
Enable or disable the ability to extend the lifetime of the SCTP cookie
that is used during the establishment phase of SCTP association

1: Enable cookie lifetime extension.
0: Disable

Default: 1

cookie_hmac_alg - STRING
Select the hmac algorithm used when generating the cookie value sent by
a listening sctp socket to a connecting client in the INIT-ACK chunk.
Valid values are:
* md5
* sha1
* none
Ability to assign md5 or sha1 as the selected alg is predicated on the
configuration of those algorithms at build time (CONFIG_CRYPTO_MD5 and
CONFIG_CRYPTO_SHA1).

```
        Default: Dependent on configuration.  MD5 if available, else SHA1 if
        available, else none.

rcvbuf_policy - INTEGER
        Determines if the receive buffer is attributed to the socket or to
        association.   SCTP supports the capability to create multiple
        associations on a single socket.  When using this capability, it is
        possible that a single stalled association that's buffering a lot
        of data may block other associations from delivering their data by
        consuming all of the receive buffer space.  To work around this,
        the rcvbuf_policy could be set to attribute the receiver buffer space
        to each association instead of the socket.  This prevents the described
        blocking.

        1: rcvbuf space is per association
        0: rcvbuf space is per socket

        Default: 0

sndbuf_policy - INTEGER
        Similar to rcvbuf_policy above, this applies to send buffer space.

        1: Send buffer is tracked per association
        0: Send buffer is tracked per socket.

        Default: 0

sctp_mem - vector of 3 INTEGERs: min, pressure, max
        Number of pages allowed for queueing by all SCTP sockets.

        min: Below this number of pages SCTP is not bothered about its
        memory appetite. When amount of memory allocated by SCTP exceeds
        this number, SCTP starts to moderate memory usage.

        pressure: This value was introduced to follow format of tcp_mem.

        max: Number of pages allowed for queueing by all SCTP sockets.

        Default is calculated at boot time from amount of available memory.

sctp_rmem - vector of 3 INTEGERs: min, default, max
        Only the first value ("min") is used, "default" and "max" are
        ignored.

        min: Minimal size of receive buffer used by SCTP socket.
        It is guaranteed to each SCTP socket (but not association) even
        under moderate memory pressure.

        Default: 4K

sctp_wmem  - vector of 3 INTEGERs: min, default, max
        Currently this tunable has no effect.

addr_scope_policy - INTEGER
        Control IPv4 address scoping - draft-stewart-tsvwg-sctp-ipv4-00

        0   - Disable IPv4 address scoping
        1   - Enable IPv4 address scoping
        2   - Follow draft but allow IPv4 private addresses
        3   - Follow draft but allow IPv4 link local addresses

        Default: 1
```

```
/proc/sys/net/core/*
        Please see: Documentation/sysctl/net.txt for descriptions of these entries.


/proc/sys/net/unix/*
max_dgram_qlen - INTEGER
        The maximum length of dgram socket receive queue

        Default: 10
```