Normalizer Inspector

- Normalizer Inspector Overview, on page 1
- Normalizer Inspector Parameters, on page 2
- Normalizer Inspector Rules, on page 6
- Normalizer Inspector Intrusion Rule Options, on page 7

Normalizer Inspector Overview

Type	Inspector (packet)
Usage	Context
Instance Type	Network
Other Inspectors Required	None
Enabled	`true`

The `normalizer` inspector detects and removes protocol anomalies in packets. The `normalizer` inspector can minimize the chances of attackers creating packets to evade detection in inline deployments.

Note Before your send traffic from your network, you must deploy relevant configurations to managed devices using routed, switched, or transparent interfaces, or inline interface pairs.

You can specify the normalization of any combination of IPv4, IPv6, ICMPv4, ICMPv6, and TCP protocols in packets. The `normalizer` inspector conducts per-packet normalizations and handles most normalizations. The `stream_tcp` inspector handles TCP state-related packet and stream normalizations, including TCP payload normalization.

Inline normalization takes place immediately after decoding and before processing by other inspectors. Normalization proceeds from the inner to outer packet layers.

The `normalizer` inspector does not generate events. The `normalizer` inspector prepares packets for use by other inspectors and in inline deployments. The inspector helps ensure that the packets the system processes are the same as the packets received by the hosts on your network.

Normalizer Inspector Parameters

Locate the `normalizer` scope in your configuration to set the `normalizer` inspector parameters.

ip6

Clears the `Reserved` flag in IPv6 traffic.

Type: boolean

Valid values: `true`, `false`

Default value: `false`

icmp4

Clears the `Reserved` flag in ICMPv4 traffic.

Type: boolean

Valid values: `true`, `false`

Default value: `false`

icmp6

Clears the `Reserved` flag in ICMPv6 traffic.

Type: boolean

Valid values: `true`, `false`

Default value: `false`

ip4.base

Clears the single-bit `Reserved` subfield of the IPv4 Flags header field as well as parameter padding. Fixes urgent pointer/flag issues. We recommend that you enable `ip4.base`.

Type: boolean

Valid values: `true`, `false`

Default value: `false`

ip4.df

Clears the single-bit `Don't Fragment` subfield of the IPv4 Flags header field. Enable `ip4.df` to allow a downstream router to fragment packets instead of dropping them. The `ip4.df` parameter can prevent evasions which create packets to be dropped.

Type: boolean

Valid values: `true`, `false`

Default value: `false`

ip4.rf

Clears the `Reserved` bits on incoming packets.

Type: boolean

Valid values: `true`, `false`

Default value: `false`

ip4.tos

Clears the one byte `Differentiated Services` field, formerly known as `Type of Service`.

Type: boolean

Valid values: `true`, `false`

Default value: `false`

ip4.trim

Truncates packets with excess payload to the datagram length specified in the IP header plus the Layer 2 (for example, Ethernet) header, but does not truncate below the minimum frame length.

Type: boolean

Valid values: `true`, `false`

Default value: `false`

tcp.base

Clears the single-bit `Reserved` subfield of the TCP header as well as option padding bytes. Fixes urgent pointer or flag issues.

Type: boolean

Valid values: `true`, `false`

Default value: `false`

tcp.block

Specfies whether to drop packets during TCP normalization.

When enabled, Snort blocks anomalous TCP packets that, if normalized, would be invalid and likely would be blocked by the receiving host. For example, Snort blocks any SYN packet transmitted subsequent to an established session.

Snort drops any packet that matches any of the following TCP stream inspector rules, regardless of whether the rules are enabled:

- 129:1
- 129:3
- 129:4
- 129:6
- 129:8

- 129:11
- 129:14 through 129:19

Type: boolean

Valid values: `true`, `false`

Default value: `false`

tcp.ecn

Enables per-packet or per-stream normalization of `Explicit Congestion Notification (ECN)` flags.

- Specify `packet` to clear ECN flags on a per-packet basis regardless of negotiation.
- Specify `stream` to clear ECN flags on a per-stream basis if ECN use was not negotiated. If you specify `stream`, you must enable `tcp.require_3whs` in the TCP stream inspector for normalization to take place.
- Specify `off` to disable the `tcp.ecn` parameter.

Type: enum

Valid values: `off`, `packet`, `stream`

Default value: `off`

tcp.ips

Enables normalization of the TCP Data field to ensure consistency in retransmitted data. Any segment that cannot be properly reassembled is dropped.

Type: boolean

Valid values: `true`, `false`

Default value: `true`

tcp.opts

Specifies whether to normalize specific TCP options which you allow in traffic. Snort does not normalize options that you explicitly allow. Snort normalizes options that you do not explicitly allow.

Snort always allows the following TCP options because they are commonly used for optimal TCP performance:

- Maximum Segment Size (MSS)
- Window Scale
- Time Stamp TCP

Snort does not automatically allow other less commonly used options.

When `tcp.opts` is enabled, TCP traffic normalizations include the following:

- Sets all option bytes to No Operation (TCP Option 1), except for MSS, Window Scale, Time Stamp, and any explicitly allowed options.
- Sets the Time Stamp octets to No Operation if Time Stamp is present but invalid, or valid but not negotiated.

- Blocks the packet if Time Stamp is negotiated but not present
- Clears the Time Stamp Echo Reply (TSecr) option field if the Acknowledgment (ACK) control bit is not set.
- Sets the MSS and Window Scale options to No Operation (TCP Option 1) if the SYN control bit is not set.

Type: boolean

Valid values: `true`, `false`

Default value: `false`

tcp.pad

Clears any option padding bytes.

Type: boolean

Valid values: `true`, `false`

Default value: `false`

tcp.req_pay

Clears the TCP header `Urgent Pointer` field and the urgent (URG) control bit if there is no payload.

Type: boolean

Valid values: `true`, `false`

Default value: `false`

tcp.req_urg

Clears the 16-bit TCP header `Urgent Pointer` field if the TCP header urgent (URG) control bit is not set.

Type: boolean

Valid values: `true`, `false`

Default value: `false`

tcp.req_urp

Clears the TCP header `urgent (URG)` control bit if the TCP header `Urgent Pointer` field is not set.

Type: boolean

Valid values: `true`, `false`

Default value: `false`

tcp.resv

Clears the `Reserved` bits in the TCP header.

Type: boolean

Valid values: `true`, `false`

Default value: `false`

tcp.trim_mss

Trims the TCP `Data` field to the Maximum Segment Size (MSS) if the payload is longer than MSS.

Type: boolean

Valid values: `true`, `false`

Default value: `false`

tcp.trim_rst

Clears data from the RST packet.

Type: boolean

Valid values: `true`, `false`

Default value: `false`

tcp.trim_syn

Removes data in TCP synchronization (SYN) packets.

Type: boolean

Valid values: `true`, `false`

Default value: `false`

tcp.trim_win

Trims the TCP `Data` field to the size specified in the `Window` field.

Type: boolean

Valid values: `true`, `false`

Default value: `false`

tcp.urp

Sets the two-byte TCP header `Urgent Pointer` field to the payload length if the pointer is greater than the payload length.

Type: boolean

Valid values: `true`, `false`

Default value: `false`

Normalizer Inspector Rules

The `normalizer` inspector does not have any associated rules.

Normalizer Inspector Intrusion Rule Options

The `normalizer` inspector does not have any intrusion rule options.