Snort 3 Reference Manual

REVISION HISTORY			
NUMBER	DATE	DESCRIPTION	NAME
3.1.78.0	2024-01-16 01:22:16 EST		TST

Contents

1 Help

The detail in this reference manual was generated from the various help commands available in Snort. `snort --help` will output:

```
Snort has several options to get more help:

-? list command line options (same as --help)
--help this overview of help
--help-commands [<module prefix>] output matching commands
--help-config [<module prefix>] output matching config options
--help-counts [<module prefix>] output matching peg counts
--help-limits print the int upper bounds denoted by max*
--help-module <module> output description of given module
--help-modules list all available modules with brief help
--help-modules-json dump description of all available modules in JSON format
--help-plugins list all available plugins with brief help
--help-options [<option prefix>] output matching command line options
--help-signals dump available control signals
--list-buffers output available inspection buffers
--list-builtin [<module prefix>] output matching builtin rules
--list-gids [<module prefix>] output matching generators
--list-modules [<module type>] list all known modules
--list-plugins list all known modules
--show-plugins list module and plugin versions

--help* and --list* options preempt other processing so should be last on the
command line since any following options are ignored.  To ensure options like
--markup and --plugin-path take effect, place them ahead of the help or list
options.

Options that filter output based on a matching prefix, such as --help-config
won't output anything if there is no match.  If no prefix is given, everything
matches.

Report bugs to bugs@snort.org.
```

2 Basic Modules

Internal modules which are not plugins are termed "basic". These include configuration for core processing.

2.1 active

Help: configure responses

Type: basic

Usage: global

Configuration:

- int **active.attempts** = 0: number of TCP packets sent per response (with varying sequence numbers) { 0:255 }
- string **active.device**: use *ip* for network layer responses or *eth0* etc for link layer
- string **active.dst_mac**: use format *01:23:45:67:89:ab*
- int **active.max_responses** = 0: maximum number of responses { 0:255 }

- int **active.min_interval** = 255: minimum number of seconds between responses { 1:255 }

Peg counts:

- **active.injects**: total crafted packets encoded and injected (sum)
- **active.failed_injects**: total crafted packet encode + injects that failed (sum)
- **active.direct_injects**: total crafted packets directly injected (sum)
- **active.failed_direct_injects**: total crafted packet direct injects that failed (sum)
- **active.holds_denied**: total number of packet hold requests denied (sum)
- **active.holds_canceled**: total number of packet hold requests canceled (sum)
- **active.holds_allowed**: total number of packet hold requests allowed (sum)

2.2 alerts

Help: configure alerts

Type: basic

Usage: global

Configuration:

- bool **alerts.alert_with_interface_name** = false: include interface in alert info (fast, full, or syslog only)
- int **alerts.detection_filter_memcap** = 1048576: set available MB of memory for detection_filters { 0:max32 }
- int **alerts.event_filter_memcap** = 1048576: set available MB of memory for event_filters { 0:max32 }
- bool **alerts.log_references** = false: include rule references in alert info (full only)
- string **alerts.order**: change the order of rule action application
- int **alerts.rate_filter_memcap** = 1048576: set available MB of memory for rate_filters { 0:max32 }
- string **alerts.reference_net**: set the CIDR for homenet (for use with -l or -B, does NOT change $HOME_NET in IDS mode)
- string **alerts.tunnel_verdicts**: let DAQ handle non-allow verdicts for gtp|teredo|6in4|4in6|4in4|6in6|gre|mpls|vxlan traffic

2.3 attribute_table

Help: configure hosts loading

Type: basic

Usage: global

Configuration:

- string **attribute_table.hosts_file**: filename to load attribute host table from
- int **attribute_table.max_hosts** = 1024: maximum number of hosts in attribute table { 32:max53 }
- int **attribute_table.segments** = 4: number of segments of hosts attribute table. It must be power of 2. { 1:32 }
- int **attribute_table.max_services_per_host** = 8: maximum number of services per host entry in attribute table { 1:65535 }
- int **attribute_table.max_metadata_services** = 9: maximum number of services in rule { 1:255 }

2.4 classifications

Help: define rule categories with priority

Type: basic

Usage: global

Configuration:

- string **`classifications[].name`**: name used with classtype rule option
- int **`classifications[].priority`** = 1: default priority for class { 0:max32 }
- string **`classifications[].text`**: description of class

2.5 daq

Help: configure packet acquisition interface

Type: basic

Usage: global

Configuration:

- string **`daq.module_dirs[].path`**: directory path
- string **`daq.inputs[].input`**: input source
- int **daq.snaplen** = 1518: set snap length (same as -s) { 0:65535 }
- int **daq.batch_size** = 64: set receive batch size (same as --daq-batch-size) { 1: }
- string **`daq.modules[].name`**: DAQ module name (required)
- enum **`daq.modules[].mode`** = *passive*: DAQ module mode { *passive* | *inline* | *read-file* }
- string **`daq.modules[].variables[].variable`**: DAQ module variable (foo[=bar])

Peg counts:

- **daq.pcaps**: total files and interfaces processed (max)
- **daq.received**: total packets received from DAQ (sum)
- **daq.analyzed**: total packets analyzed from DAQ (sum)
- **daq.dropped**: packets dropped (sum)
- **daq.filtered**: packets filtered out (sum)
- **daq.outstanding**: packets unprocessed (now)
- **daq.outstanding_max**: maximum of packets unprocessed (max)
- **daq.injected**: active responses or replacements (sum)
- **daq.allow**: total allow verdicts (sum)
- **daq.block**: total block verdicts (sum)
- **daq.replace**: total replace verdicts (sum)
- **daq.whitelist**: total whitelist verdicts (sum)

- **daq.blacklist**: total blacklist verdicts (sum)
- **daq.ignore**: total ignore verdicts (sum)
- **daq.internal_blacklist**: packets blacklisted internally due to lack of DAQ support (sum)
- **daq.internal_whitelist**: packets whitelisted internally due to lack of DAQ support (sum)
- **daq.skipped**: packets skipped at startup (sum)
- **daq.idle**: attempts to acquire from DAQ without available packets (sum)
- **daq.rx_bytes**: total bytes received (sum)
- **daq.expected_flows**: expected flows created in DAQ (sum)
- **daq.retries_queued**: messages queued for retry (sum)
- **daq.retries_dropped**: messages dropped when overrunning the retry queue (sum)
- **daq.retries_processed**: messages processed from the retry queue (sum)
- **daq.retries_discarded**: messages discarded when purging the retry queue (sum)
- **daq.sof_messages**: start of flow messages received from DAQ (sum)
- **daq.eof_messages**: end of flow messages received from DAQ (sum)
- **daq.other_messages**: messages received from DAQ with unrecognized message type (sum)

2.6 decode

Help: general decoder rules

Type: basic

Usage: context

Rules:

- **116:150** (decode) loopback IP
- **116:151** (decode) same src/dst IP
- **116:293** (decode) two or more IP (v4 and/or v6) encapsulation layers present
- **116:449** (decode) unassigned/reserved IP protocol
- **116:450** (decode) bad IP protocol
- **116:459** (decode) fragment with zero length
- **116:472** (decode) too many protocols present
- **116:473** (decode) ether type out of range

2.7 detection

Help: configure general IPS rule processing parameters

Type: basic

Usage: global

Configuration:

- bool **detection.allow_missing_so_rules** = false: warn (true) or error (false) when an SO rule stub refers to an SO rule that isn't loaded
- bool **detection.global_default_rule_state** = true: enable or disable rules by default (overridden by ips policy settings)
- bool **detection.global_rule_state** = false: apply rule_state against all policies
- bool **detection.hyperscan_literals** = false: use hyperscan for content literal searches instead of boyer-moore
- int **detection.offload_limit** = 99999: minimum sizeof PDU to offload fast pattern search (defaults to disabled) { 0:max32 }
- int **detection.offload_threads** = 0: maximum number of simultaneous offloads (defaults to disabled) { 0:max32 }
- bool **detection.pcre_enable** = true: enable pcre pattern matching
- int **detection.pcre_match_limit** = 1500: limit pcre backtracking, 0 = off { 0:max32 }
- int **detection.pcre_match_limit_recursion** = 1500: limit pcre stack consumption, 0 = off { 0:max32 }
- bool **detection.pcre_override** = true: enable pcre match limit overrides when pattern matching (ie ignore /O)
- bool **detection.pcre_to_regex** = false: enable the use of regex instead of pcre for compatible expressions
- bool **detection.enable_address_anomaly_checks** = false: enable check and alerting of address anomalies
- bool **detection.enable_strict_reduction** = false: enable strict deduplication of rule headers by ports (saves memory, but loses some speed during config reading)
- int **detection.max_continuations_per_flow** = 1024: maximum number of continuations stored simultaneously on the flow { 0:65535 }
- string **`detection.service_extension[].service`**: service to perform extension for
- string **`detection.service_extension[].extend_to[].extend_to_service`**: service to extend to

Peg counts:

- **detection.analyzed**: total packets processed (now)
- **detection.hard_evals**: non-fast pattern rule evaluations (sum)
- **detection.raw_searches**: fast pattern searches in raw packet data (sum)
- **detection.cooked_searches**: fast pattern searches in cooked packet data (sum)
- **detection.pkt_searches**: fast pattern searches in packet data (sum)
- **detection.alt_searches**: alt fast pattern searches in packet data (sum)
- **detection.pdu_searches**: fast pattern searches in service buffers (sum)
- **detection.file_searches**: fast pattern searches in file buffer (sum)
- **detection.offloads**: fast pattern searches that were offloaded (sum)
- **detection.alerts**: alerts not including IP reputation (sum)

- **detection.total_alerts**: alerts including IP reputation (sum)
- **detection.logged**: logged packets (sum)
- **detection.passed**: passed packets (sum)
- **detection.match_limit**: fast pattern matches not processed (sum)
- **detection.queue_limit**: events not queued because queue full (sum)
- **detection.log_limit**: events queued but not logged (sum)
- **detection.event_limit**: events filtered (sum)
- **detection.alert_limit**: events previously triggered on same PDU (sum)
- **detection.context_stalls**: times processing stalled to wait for an available context (sum)
- **detection.offload_busy**: times offload was not available (sum)
- **detection.onload_waits**: times processing waited for onload to complete (sum)
- **detection.offload_fallback**: fast pattern offload search fallback attempts (sum)
- **detection.offload_failures**: fast pattern offload search failures (sum)
- **detection.offload_suspends**: fast pattern search suspends due to offload context chains (sum)
- **detection.pcre_match_limit**: total number of times pcre hit the match limit (sum)
- **detection.pcre_recursion_limit**: total number of times pcre hit the recursion limit (sum)
- **detection.pcre_error**: total number of times pcre returns error (sum)
- **detection.cont_creations**: total number of continuations created (sum)
- **detection.cont_recalls**: total number of continuations recalled (sum)
- **detection.cont_flows**: total number of flows using continuation (sum)
- **detection.cont_evals**: total number of condition-met continuations (sum)
- **detection.cont_matches**: total number of continuations matched (sum)
- **detection.cont_mismatches**: total number of continuations mismatched (sum)
- **detection.cont_max_num**: peak number of simultaneous continuations per flow (max)
- **detection.cont_match_distance**: total number of bytes jumped over by matched continuations (sum)
- **detection.cont_mismatch_distance**: total number of bytes jumped over by mismatched continuations (sum)
- **detection.buf_dumps**: total number of IPS buffers collected from matched rules (sum)

2.8 event_filter

Help: configure thresholding of events

Type: basic

Usage: context

Configuration:

- int **`event_filter[].gid`** = 1: rule generator ID { 0:8129 }
- int **`event_filter[].sid`** = 1: rule signature ID { 0:max32 }

- enum **`event_filter[].type`**: 1st count events | every count events | once after count events { *limit* | *threshold* | *both* }
- enum **`event_filter[].track`**: filter only matching source or destination addresses { *by_src* | *by_dst* }
- int **`event_filter[].count`** = 0: number of events in interval before tripping; -1 to disable { -1:max31 }
- int **`event_filter[].seconds`** = 0: count interval { 0:max32 }
- string **`event_filter[].ip`**: restrict filter to these addresses according to track

Peg counts:

- **event_filter.no_memory_local**: number of times event filter ran out of local memory (sum)
- **event_filter.no_memory_global**: number of times event filter ran out of global memory (sum)

2.9 event_queue

Help: configure event queue parameters

Type: basic

Usage: context

Configuration:

- int **event_queue.max_queue** = 8: maximum events to queue { 1:max32 }
- int **event_queue.log** = 3: maximum events to log { 1:max32 }
- enum **event_queue.order_events** = *content_length*: criteria for ordering incoming events { *priority*|*content_length* }
- bool **event_queue.process_all_events** = false: process just first action group or all action groups

2.10 file_policy

Help: configure file policy

Type: basic

Usage: context

Configuration:

- bool **file_policy.enable_type** = true: enable type ID
- bool **file_policy.enable_signature** = false: enable signature calculation
- bool **file_policy.enable_capture** = false: enable file capture
- int **file_policy.verdict_delay** = 0: number of queries to return final verdict { 0:max53 }
- int **`file_policy.rules[].when.file_type_id`** = 0: unique ID for file type in file magic rule { 0:max32 }
- string **`file_policy.rules[].when.sha256`**: SHA 256
- enum **`file_policy.rules[].use.verdict`** = *unknown*: what to do with matching traffic { *unknown* | *log* | *stop* | *block* | *reset* }
- bool **`file_policy.rules[].use.enable_file_type`** = false: true/false → enable/disable file type identification
- bool **`file_policy.rules[].use.enable_file_signature`** = false: true/false → enable/disable file signature
- bool **`file_policy.rules[].use.enable_file_capture`** = false: true/false → enable/disable file capture

2.11 high_availability

Help: implement flow tracking high availability

Type: basic

Usage: global

Configuration:

- bool **high_availability.enable** = false: enable high availability
- bool **high_availability.daq_channel** = false: enable use of daq data plane channel
- bit_list **high_availability.ports**: side channel message port list { 65535 }
- int **high_availability.min_age** = 0: minimum session life in milliseconds before HA updates { 0:max32 }
- int **high_availability.min_sync** = 0: minimum interval in milliseconds between HA updates { 0:max32 }

Peg counts:

- **high_availability.msgs_recv**: total messages received (sum)
- **high_availability.update_msgs_recv**: update messages received (sum)
- **high_availability.update_msgs_recv_no_flow**: update messages received without a local flow (sum)
- **high_availability.update_msgs_consumed**: update messages fully consumed (sum)
- **high_availability.delete_msgs_consumed**: deletion messages consumed (sum)
- **high_availability.daq_stores**: states stored via daq (sum)
- **high_availability.daq_imports**: states imported via daq (sum)
- **high_availability.key_mismatch**: messages received with a flow key mismatch (sum)
- **high_availability.msg_version_mismatch**: messages received with a version mismatch (sum)
- **high_availability.msg_length_mismatch**: messages received with an inconsistent total length (sum)
- **high_availability.truncated_msgs**: truncated messages received (sum)
- **high_availability.unknown_key_type**: messages received with an unknown flow key type (sum)
- **high_availability.unknown_client_idx**: messages received with an unknown client index (sum)
- **high_availability.client_consume_errors**: client data consume failure count (sum)

2.12 host_cache

Help: global LRU cache of host_tracker data about hosts

Type: basic

Usage: global

Configuration:

- string **host_cache.dump_file**: file name to dump host cache on shutdown; won't dump by default
- int **host_cache.memcap** = 8388608: maximum host cache size in bytes { 512:maxSZ }
- int **host_cache.segments** = 4: number of host cache segments. It must be power of 2. { 1:32 }

Commands:

- **host_cache.dump**(file_name): dump host cache
- **host_cache.delete_host**(host_ip): delete host from host cache
- **host_cache.delete_network_proto**(host_ip, proto): delete network protocol from host
- **host_cache.delete_transport_proto**(host_ip, proto): delete transport protocol from host
- **host_cache.delete_service**(host_ip, port, proto): delete service from host
- **host_cache.delete_client**(host_ip, id, service, version): delete client from host
- **host_cache.get_stats**(): get current host cache usage and pegs
- **host_cache.get_segment_stats**(segment): get usage and pegs for cache segment(s)

Peg counts:

- **host_cache.adds**: lru cache added new entry (sum)
- **host_cache.alloc_prunes**: lru cache pruned entry to make space for new entry (sum)
- **host_cache.bytes_in_use**: current number of bytes in use (now)
- **host_cache.items_in_use**: current number of items in the cache (now)
- **host_cache.find_hits**: lru cache found entry in cache (sum)
- **host_cache.find_misses**: lru cache did not find entry in cache (sum)
- **host_cache.reload_prunes**: lru cache pruned entry for lower memcap during reload (sum)
- **host_cache.removes**: lru cache found entry and removed it (sum)
- **host_cache.replaced**: lru cache found entry and replaced it (sum)

2.13 host_tracker

Help: configure hosts

Type: basic

Usage: global

Configuration:

- addr **`host_tracker[].ip`**: hosts address / cidr
- port **`host_tracker[].services[].port`**: port number
- enum **`host_tracker[].services[].proto`**: IP protocol { *ip* | *tcp* | *udp* }

Peg counts:

- **host_tracker.service_adds**: host service adds (sum)
- **host_tracker.service_finds**: host service finds (sum)

2.14 hosts

Help: configure hosts

Type: basic

Usage: global

Configuration:

- addr **hosts[].ip** = *0.0.0.0/32*: hosts address / CIDR
- enum **hosts[].frag_policy**: defragmentation policy { *first* | *linux* | *bsd* | *bsd_right* | *last* | *windows* | *solaris* }
- enum **hosts[].tcp_policy**: TCP reassembly policy { *first* | *last* | *linux* | *old_linux* | *bsd* | *macos* | *solaris* | *irix* | *hpux11* | *hpux10* | *windows* | *win_2003* | *vista* | *proxy* }
- string **hosts[].services[].name**: service identifier
- enum **hosts[].services[].proto** = *tcp*: IP protocol { *tcp* | *udp* }
- port **hosts[].services[].port**: port number

Peg counts:

- **hosts.total_hosts**: maximum number of entries in the host attribute table (max)
- **hosts.hosts_pruned**: number of LRU hosts pruned due to configured resource limits (sum)
- **hosts.dynamic_host_adds**: number of host additions after initial host file load (sum)
- **hosts.dynamic_service_adds**: number of service additions after initial host file load (sum)
- **hosts.dynamic_service_updates**: number of service updates after initial host file load (sum)
- **hosts.service_list_overflows**: number of service additions that failed due to configured resource limits (sum)

2.15 inspection

Help: configure basic inspection policy parameters

Type: basic

Usage: inspect

Configuration:

- int **inspection.id** = 0: correlate policy and events with other items in configuration { 0:max64 }
- string **inspection.uuid**: correlate events by uuid
- enum **inspection.mode** = *inline-test*: set policy mode { *inline* | *inline-test* }
- int **inspection.max_aux_ip** = 16: maximum number of auxiliary IPs per flow to detect and save (-1 = disable, 0 = detect but don't save, 1+ = save in FIFO manner) { -1:127 }

2.16 ips

Help: configure IPS rule processing

Type: basic

Usage: detect

Configuration:

- string **`ips.action_map[].replace`**: action you want to change
- string **`ips.action_map[].with`**: action you want to use instead
- string **ips.action_override**: use this action for all rules (applied before action_map)
- enum **ips.default_rule_state** = *inherit*: enable or disable ips rules { *no* | *yes* | *inherit* }
- bool **ips.enable_builtin_rules** = false: enable events from builtin rules w/o stubs
- int **ips.id** = 0: correlate unified2 events with configuration { 0:max64 }
- string **ips.include**: snort rules and includes
- enum **ips.mode**: set policy mode { *tap* | *inline* | *inline-test* }
- bool **ips.obfuscate_pii** = true: mask all but the last 4 characters of credit card, SSN, phone number, and email
- string **ips.rules**: snort rules and includes (may contain states too)
- string **ips.states**: snort rule states and includes (may contain rules too)
- string **ips.uuid** = *00000000-0000-0000-0000-000000000000*: IPS policy uuid
- string **ips.variables.nets.$var**: IPS policy variable
- string **ips.variables.paths.$var**: IPS policy variable
- string **ips.variables.ports.$var**: IPS policy variable

2.17 js_norm

Help: JavaScript normalizer

Type: basic

Usage: inspect

Configuration:

- int **js_norm.bytes_depth** = -1: number of input JavaScript bytes to normalize (-1 unlimited) { -1:max53 }
- int **js_norm.identifier_depth** = 65536: max number of unique JavaScript identifiers to normalize { 0:65536 }
- int **js_norm.max_tmpl_nest** = 32: maximum depth of template literal nesting that enhanced JavaScript normalizer will process { 0:255 }
- int **js_norm.max_bracket_depth** = 256: maximum depth of bracket nesting that enhanced JavaScript normalizer will process { 1:65535 }
- int **js_norm.max_scope_depth** = 256: maximum depth of scope nesting that enhanced JavaScript normalizer will process { 1:65535 }
- string **`js_norm.ident_ignore[].ident_name`**: name of the identifier to ignore
- string **`js_norm.prop_ignore[].prop_name`**: name of the object property to ignore

Rules:

- **154:1** (js_norm) nested unescape functions
- **154:2** (js_norm) mixed unescape sequence
- **154:3** (js_norm) bad token
- **154:4** (js_norm) unexpected HTML script opening tag
- **154:5** (js_norm) unexpected HTML script closing tag
- **154:6** (js_norm) max number of unique identifiers reached
- **154:7** (js_norm) excessive bracket nesting
- **154:8** (js_norm) data gaps during normalization
- **154:9** (js_norm) excessive scope nesting

Peg counts:

- **js_norm.bytes**: total number of bytes processed (sum)
- **js_norm.identifiers**: total number of unique identifiers processed (sum)
- **js_norm.identifier_overflows**: total number of unique identifier limit overflows (sum)

2.18 latency

Help: packet and rule latency monitoring and control

Type: basic

Usage: context

Configuration:

- int **latency.packet.max_time** = 500: set timeout for packet latency thresholding (usec) { 0:max53 }
- bool **latency.packet.fastpath** = false: fastpath expensive packets (max_time exceeded)
- int **latency.rule.max_time** = 500: set timeout for rule evaluation (usec) { 0:max53 }
- bool **latency.rule.suspend** = false: temporarily suspend expensive rules
- int **latency.rule.suspend_threshold** = 5: set threshold for number of timeouts before suspending a rule { 1:max32 }
- int **latency.rule.max_suspend_time** = 30000: set max time for suspending a rule (ms, 0 means permanently disable rule) { 0:max32 }

Rules:

- **134:1** (latency) rule tree suspended due to latency
- **134:2** (latency) rule tree re-enabled after suspend timeout
- **134:3** (latency) packet fastpathed due to latency

Peg counts:

- **latency.total_packets**: total packets monitored (sum)

- **latency.total_usecs**: total usecs elapsed (sum)
- **latency.max_usecs**: maximum usecs elapsed (sum)
- **latency.packet_timeouts**: packets that timed out (sum)
- **latency.total_rule_evals**: total rule evals monitored (sum)
- **latency.rule_eval_timeouts**: rule evals that timed out (sum)
- **latency.rule_tree_enables**: rule tree re-enables (sum)

2.19 memory

Help: memory management configuration

Type: basic

Usage: global

Configuration:

- int **memory.cap** = 0: set the process cap on memory in bytes (0 to disable) { 0:maxSZ }
- int **memory.interval** = 50: approximate ms between memory epochs (0 to disable) { 0:max32 }
- int **memory.prune_target** = 1048576: bytes to prune per packet thread prune cycle { 1:max32 }
- int **memory.threshold** = 100: scale cap to account for heap overhead { 1:100 }

Peg counts:

- **memory.start_up_use**: memory used before packet processing (now)
- **memory.cur_in_use**: current memory used (now)
- **memory.max_in_use**: maximum memory used (max)
- **memory.epochs**: number of memory updates (sum)
- **memory.allocated**: total amount of memory allocated by packet threads (now)
- **memory.deallocated**: total amount of memory deallocated by packet threads (now)
- **memory.reap_cycles**: number of actionable over-limit conditions (sum)
- **memory.reap_attempts**: attempts to reclaim memory (sum)
- **memory.reap_failures**: failures to reclaim memory (sum)
- **memory.reap_aborts**: abort pruning before target due to process under limit (sum)
- **memory.reap_decrease**: total amount of the decrease in thread memory while process over limit (sum)
- **memory.reap_increase**: total amount of the increase in thread memory while process over limit (sum)
- **memory.app_all**: total bytes allocated by application (now)
- **memory.active**: total bytes allocated in active pages (now)
- **memory.resident**: maximum bytes physically resident (now)
- **memory.retained**: total bytes not returned to OS (now)

2.20 network

Help: configure basic network parameters

Type: basic

Usage: context

Configuration:

- multi **network.checksum_drop** = *none*: drop if checksum is bad { all | ip | noip | tcp | notcp | udp | noudp | icmp | noicmp | none }
- multi **network.checksum_eval** = *all*: checksums to verify { all | ip | noip | tcp | notcp | udp | noudp | icmp | noicmp | none }
- int **network.id** = 0: correlate unified2 events with configuration { 0:18446744073709551614 }
- int **network.min_ttl** = 1: alert / normalize packets with lower TTL / hop limit (you must enable rules and / or normalization also) { 1:255 }
- int **network.new_ttl** = 1: use this value for responses and when normalizing { 1:255 }
- int **network.layers** = 40: the maximum number of protocols that Snort can correctly decode { 3:255 }
- int **network.max_ip6_extensions** = 0: the maximum number of IP6 options Snort will process for a given IPv6 layer before raising 116:456 (0 = unlimited) { 0:255 }
- int **network.max_ip_layers** = 0: the maximum number of IP layers Snort will process for a given packet before raising 116:293 (0 = unlimited) { 0:255 }

Commands:

- **network.set_policy**(id): set the network policy for commands given the user policy id

2.21 output

Help: configure general output parameters

Type: basic

Usage: global

Configuration:

- bool **output.dump_chars_only** = false: turns on character dumps (same as -C)
- bool **output.dump_payload** = false: dumps application layer (same as -d)
- bool **output.dump_payload_verbose** = false: dumps raw packet starting at link layer (same as -X)
- int **output.event_trace.max_data** = 0: maximum amount of packet data to capture { 0:65535 }
- bool **output.quiet** = false: suppress normal logging on stdout (same as -q)
- string **output.logdir** = .: where to put log files (same as -l)
- bool **output.show_year** = false: include year in timestamp in the alert and log files (same as -y)
- int **output.tagged_packet_limit** = 256: maximum number of packets tagged for non-packet metrics { 0:max32 }
- bool **output.verbose** = false: be verbose (same as -v)
- bool **output.obfuscate** = false: obfuscate the logged IP addresses (same as -O)
- bool **output.wide_hex_dump** = false: output 20 bytes per lines instead of 16 when dumping buffers

Rules:

- **2:1** (output) tagged packet

2.22 packet_tracer

Help: generate debug trace messages for packets

Type: basic

Usage: global

Configuration:

- bool **packet_tracer.enable** = false: enable summary output of state that determined packet verdict
- enum **packet_tracer.output** = *console*: select where to send packet trace { *console* | *file* }

Commands:

- **packet_tracer.enable**(proto, src_ip, src_port, dst_ip, dst_port): enable packet tracer debugging
- **packet_tracer.disable**(): disable packet tracer

2.23 packets

Help: configure basic packet handling

Type: basic

Usage: global

Configuration:

- bool **packets.address_space_agnostic** = false: determines whether DAQ address space info is used to track fragments and connections
- string **packets.bpf_file**: file with BPF to select traffic for Snort
- int **packets.limit** = 0: maximum number of packets to process before stopping (0 is unlimited) { 0:max53 }
- int **packets.skip** = 0: number of packets to skip before before processing { 0:max53 }
- bool **packets.mpls_agnostic** = true: determines whether MPLS labels are used to track fragments and connections
- bool **packets.vlan_agnostic** = false: determines whether VLAN tags are used to track fragments and connections

2.24 payload_injector

Help: payload injection utility

Type: basic

Usage: global

Peg counts:

- **payload_injector.http_injects**: total number of http injections (sum)
- **payload_injector.http2_injects**: total number of http2 injections (sum)
- **payload_injector.http2_translate_err**: total number of http2 page translation errors (sum)
- **payload_injector.http2_mid_frame**: total number of attempts to inject mid-frame (sum)

2.25 process

Help: configure basic process setup

Type: basic

Usage: global

Configuration:

- string **process.chroot**: set chroot directory (same as -t)
- string **process.threads[].cpuset**: pin the associated thread to this cpuset
- int **process.threads[].thread**: set cpu affinity for the <cur_thread_num> thread that runs { 0:65535 }
- enum **process.threads[].type**: define which threads will have specified affinity, by their type { *other|packet|main* }
- string **process.threads[].name**: define which threads will have specified affinity, by thread name
- bool **process.daemon** = false: fork as a daemon (same as -D)
- bool **process.dirty_pig** = false: shutdown without internal cleanup
- string **process.set_gid**: set group ID (same as -g)
- string **process.set_uid**: set user ID (same as -u)
- int **process.umask**: set process umask (same as -m) { 0x000:0x1FF }
- bool **process.utc** = false: use UTC instead of local time for timestamps
- int **process.watchdog_timer** = 0: watchdog timer for packet threads (seconds, 0 to disable) { 0:60 }
- int **process.watchdog_min_thread_count** = 1: minimum unresponsive threads for watchdog to trigger { 1:65535 }

2.26 profiler

Help: configure profiling of rules and/or modules

Type: basic

Usage: global

Configuration:

- bool **profiler.modules.show** = true: show module time profile stats
- int **profiler.modules.count** = 0: limit results to count items per level (0 = no limit) { 0:max32 }
- enum **profiler.modules.sort** = *total_time*: sort by given field { *none* | *checks* | *avg_check* | *total_time* }
- int **profiler.modules.max_depth** = -1: limit depth to max_depth (-1 = no limit) { -1:255 }
- bool **profiler.memory.show** = true: show module memory profile stats
- int **profiler.memory.count** = 0: limit results to count items per level (0 = no limit) { 0:max32 }
- enum **profiler.memory.sort** = *total_used*: sort by given field { *none* | *allocations* | *total_used* | *avg_allocation* }
- int **profiler.memory.max_depth** = -1: limit depth to max_depth (-1 = no limit) { -1:255 }
- int **profiler.memory.dump_file_size** = 1073741824: files will be rolled over if they exceed this size { 4096:max53 }
- bool **profiler.rules.show** = true: show rule time profile stats
- int **profiler.rules.count** = 0: print results to given level (0 = all) { 0:max32 }

- enum **profiler.rules.sort** = *total_time*: sort by given field { *none* | *checks* | *avg_check* | *total_time* | *matches* | *no_matches* | *avg_match* | *avg_no_match* }

Commands:

- **profiler.rule_start**(): enable rule profiler
- **profiler.rule_stop**(): disable rule profiler
- **profiler.rule_status**(): print rule profiler status
- **profiler.rule_dump**(output): print rule statistics in table or json format (json format prints dates as Unix epoch)
- **profiler.module_start**(): enable module time profiling
- **profiler.module_stop**(): disable module time profiling
- **profiler.module_dump**(): print module time profiling statistics
- **profiler.module_status**(): show module time profiler status

2.27 rate_filter

Help: configure rate filters (which change rule actions)

Type: basic

Usage: inspect

Configuration:

- int **`rate_filter[].gid`** = 1: rule generator ID { 0:8129 }
- int **`rate_filter[].sid`** = 1: rule signature ID { 0:max32 }
- enum **`rate_filter[].track`** = *by_src*: filter only matching source or destination addresses { *by_src* | *by_dst* | *by_rule* }
- int **`rate_filter[].count`** = 1: number of events in interval before tripping { 0:max32 }
- int **`rate_filter[].seconds`** = 1: count interval { 0:max32 }
- dynamic **`rate_filter[].new_action`** = alert: take this action on future hits until timeout { alert | block | drop | file_id | log | pass | react | reject | rewrite }
- int **`rate_filter[].timeout`** = 1: count interval { 0:max32 }
- string **`rate_filter[].apply_to`**: restrict filter to these addresses according to track

Peg counts:

- **rate_filter.no_memory**: number of times rate filter ran out of memory (sum)

2.28 references

Help: define reference systems used in rules

Type: basic

Usage: global

Configuration:

- string **`references[].name`**: name used with reference rule option
- string **`references[].url`**: where this reference is defined

2.29 search_engine

Help: configure fast pattern matcher

Type: basic

Usage: global

Configuration:

- int **search_engine.bleedover_port_limit** = 1024: maximum ports in rule before demotion to any-any port group { 1:max32 }
- bool **search_engine.bleedover_warnings_enabled** = false: print warning if a rule is demoted to any-any port group
- bool **search_engine.enable_single_rule_group** = false: put all rules into one group
- bool **search_engine.debug** = false: print verbose fast pattern info
- bool **search_engine.debug_print_nocontent_rule_tests** = false: print rule group info during packet evaluation
- bool **search_engine.debug_print_rule_group_build_details** = false: print rule group info during compilation
- bool **search_engine.debug_print_rule_groups_uncompiled** = false: prints uncompiled rule group information
- bool **search_engine.debug_print_rule_groups_compiled** = false: prints compiled rule group information
- int **search_engine.max_pattern_len** = 0: truncate patterns when compiling into state machine (0 means no maximum) { 0:max32 }
- int **search_engine.max_queue_events** = 5: maximum number of matching fast pattern states to queue per packet { 2:100 }
- bool **search_engine.detect_raw_tcp** = false: detect on TCP payload before reassembly
- dynamic **search_engine.search_method** = ac_bnfa: set fast pattern algorithm - choose available search engine { ac_bnfa | ac_full | hyperscan | lowmem }
- dynamic **search_engine.offload_search_method**: set fast pattern offload algorithm - choose available search engine { ac_bnfa | ac_full | hyperscan | lowmem }
- string **search_engine.rule_db_dir**: deserialize rule databases from given directory
- bool **search_engine.show_fast_patterns** = false: print fast pattern info for each rule
- bool **search_engine.split_any_any** = true: evaluate any-any rules separately to save memory
- int **search_engine.queue_limit** = 0: maximum number of fast pattern matches to queue per packet (0 is unlimited) { 0:max32 }

Peg counts:

- **search_engine.max_queued**: maximum fast pattern matches queued for further evaluation (max)
- **search_engine.total_flushed**: total fast pattern matches processed (sum)
- **search_engine.total_inserts**: total fast pattern hits (sum)
- **search_engine.total_overruns**: fast pattern matches discarded due to overflow (sum)
- **search_engine.total_unique**: total unique fast pattern hits (sum)
- **search_engine.non_qualified_events**: total non-qualified events (sum)
- **search_engine.qualified_events**: total qualified events (sum)
- **search_engine.searched_bytes**: total bytes searched (sum)

2.30 side_channel

Help: implement the side-channel asynchronous messaging subsystem

Type: basic

Usage: global

Configuration:

- bit_list **`side_channel[].ports`**: side channel message port list { 65535 }
- string **`side_channel[].connectors[].connector`**: connector handle
- string **`side_channel[].connector`**: connector handle

Peg counts:

- **side_channel.packets**: total packets (sum)

2.31 snort

Help: command line configuration and shell commands

Type: basic

Usage: global

Configuration:

- string **snort.-?**: <option prefix> output matching command line option quick help (same as --help-options) { (optional) }
- string **snort.-A**: <mode> set alert mode: none, cmg, or alert_*
- addr **snort.-B** = *255.255.255.255/32*: <mask> obfuscated IP addresses in alerts and packet dumps using CIDR mask
- implied **snort.-C**: print out payloads with character data only (no hex)
- string **snort.-c**: <conf> use this configuration
- implied **snort.-D**: run Snort in background (daemon) mode
- implied **snort.-d**: dump the Application Layer
- implied **snort.-e**: display the second layer header info
- implied **snort.-f**: turn off fflush() calls after binary log writes
- int **snort.-G**: <0xid> (same as --logid) { 0:65535 }
- string **snort.-g**: <gname> run snort gid as <gname> group (or gid) after initialization
- implied **snort.-H**: make hash tables deterministic
- implied **snort.-h**: show help overview (same as --help)
- string **snort.-i**: <iface>... list of interfaces
- port **snort.-j**: <port> to listen for Telnet connections
- enum **snort.-k** = *all*: <mode> checksum mode; default is all { *all|noip|notcp|noudp|noicmp|none* }
- string **snort.-L**: <mode> logging mode (none, dump, pcap, or log_*)
- string **snort.-l**: <logdir> log to this directory instead of current directory

- implied **snort.-M**: log messages to syslog (not alerts)
- int **snort.-m**: <umask> set the process file mode creation mask { 0x000:0x1FF }
- int **snort.-n**: <count> stop after count packets { 0:max53 }
- implied **snort.-O**: obfuscate the logged IP addresses
- implied **snort.-Q**: enable inline mode operation
- implied **snort.-q**: quiet mode - suppress normal logging on stdout
- string **snort.-R**: <rules> include this rules file in the default policy
- string **snort.-r**: <pcap>... (same as --pcap-list)
- int **snort.-s**: <snap> (same as --snaplen); default is 1518 { 0:65535 }
- implied **snort.-T**: test and report on the current Snort configuration
- string **snort.-t**: <dir> chroots process to <dir> after initialization
- implied **snort.-U**: use UTC for timestamps
- string **snort.-u**: <uname> run snort as <uname> or <uid> after initialization
- implied **snort.-V**: (same as --version)
- implied **snort.-v**: be verbose
- implied **snort.-X**: dump the raw packet data starting at the link layer
- implied **snort.-x**: same as --pedantic
- implied **snort.-y**: include year in timestamp in the alert and log files
- int **snort.-z**: <count> maximum number of packet threads (same as --max-packet-threads); 0 gets the number of CPU cores reported by the system; default is 1 { 0:max32 }
- implied **snort.--alert-before-pass**: evaluate alert rules before pass rules; default is pass rules first
- string **snort.--bpf**: <filter options> are standard BPF options, as seen in TCPDump
- string **snort.--c2x**: output hex for given char (see also --x2c)
- string **snort.--control-socket**: <file> to create unix socket
- implied **snort.--create-pidfile**: create PID file, even when not in Daemon mode
- string **snort.--daq**: <type> select packet acquisition module (default is pcap)
- int **snort.--daq-batch-size**: <size> set the DAQ receive batch size; default is 64 { 1: }
- string **snort.--daq-dir**: <dir> tell snort where to find desired DAQ
- implied **snort.--daq-list**: list packet acquisition modules available in optional dir, default is static modules only
- enum **snort.--daq-mode**: <mode> select DAQ module operating mode (overrides automatic selection) { *passive* | *inline* | *read-file* }
- string **snort.--daq-var**: <name=value> specify extra DAQ configuration variable
- implied **snort.--dirty-pig**: don't flush packets on shutdown
- string **snort.--dump-builtin-options**: additional options to include with --dump-builtin-rules stubs
- string **snort.--dump-builtin-rules**: [<module prefix>] output stub rules for selected modules { (optional) }
- select **snort.--dump-config**: dump config in json format { all | top }

- implied **snort.--dump-config-text**: dump config in text format
- implied **snort.--dump-dynamic-rules**: output stub rules for all loaded rules libraries
- string **snort.--dump-defaults**: [<module prefix>] output module defaults in Lua format { (optional) }
- string **snort.--dump-rule-databases**: dump rule databases to given directory (hyperscan only)
- implied **snort.--dump-rule-deps**: dump rule dependencies in json format for use by other tools
- implied **snort.--dump-rule-meta**: dump configured rule info in json format for use by other tools
- implied **snort.--dump-rule-state**: dump configured rule state in json format for use by other tools
- implied **snort.--dump-version**: output the version, the whole version, and only the version
- implied **snort.--enable-inline-test**: enable Inline-Test Mode Operation
- implied **snort.--enable-test-features**: enable features used in testing
- implied **snort.--gen-msg-map**: dump configured rules in gen-msg.map format for use by other tools
- implied **snort.--help**: show help overview
- string **snort.--help-commands**: [<module prefix>] output matching commands { (optional) }
- string **snort.--help-config**: [<module prefix>] output matching config options { (optional) }
- string **snort.--help-counts**: [<module prefix>] output matching peg counts { (optional) }
- implied **snort.--help-limits**: print the int upper bounds denoted by max*
- string **snort.--help-module**: <module> output description of given module
- implied **snort.--help-modules**: list all available modules with brief help
- implied **snort.--help-modules-json**: dump description of all available modules in JSON format
- string **snort.--help-options**: [<option prefix>] output matching command line option quick help (same as -?) { (optional) }
- implied **snort.--help-plugins**: list all available plugins with brief help
- implied **snort.--help-signals**: dump available control signals
- int **snort.--id-offset** = 0: offset to add to instance IDs when logging to files { 0:65535 }
- implied **snort.--id-subdir**: create/use instance subdirectories in logdir instead of instance filename prefix
- implied **snort.--id-zero**: use id prefix / subdirectory even with one packet thread
- string **snort.--include-path**: <path> where to find Lua and rule included files; searched before current or config directories
- implied **snort.--list-buffers**: output available inspection buffers
- string **snort.--list-builtin**: [<module prefix>] output matching builtin rules { (optional) }
- string **snort.--list-gids**: [<module prefix>] output matching generators { (optional) }
- string **snort.--list-modules**: [<module type>] list all known modules of given type { (optional) }
- implied **snort.--list-plugins**: list all known plugins
- string **snort.--lua**: <chunk> extend/override conf with chunk; may be repeated
- string **snort.--lua-sandbox**: <file> file that contains the lua sandbox environment in which config will be loaded
- int **snort.--logid**: <0xid> log Identifier to uniquely id events for multiple snorts (same as -G) { 0:65535 }
- implied **snort.--markup**: output help in asciidoc compatible format

- int **snort.--max-packet-threads**: <count> configure maximum number of packet threads (same as -z) { 0:max32 }
- implied **snort.--mem-check**: like -T but also compile search engines
- string **snort.--metadata-filter**: <filter> load only rules containing filter string in metadata if set
- implied **snort.--nostamps**: don't include timestamps in log file names
- implied **snort.--nolock-pidfile**: do not try to lock Snort PID file
- implied **snort.--no-warn-flowbits**: ignore warnings about flowbits that are checked but not set and vice-versa
- implied **snort.--no-warn-rules**: ignore warnings about duplicate rules and rule parsing issues
- implied **snort.--pause**: wait for resume/quit command before processing packets/terminating
- string **snort.--pcap-file**: <file> file that contains a list of pcaps to read - read mode is implied
- string **snort.--pcap-list**: <list> a space separated list of pcaps to read - read mode is implied
- string **snort.--pcap-dir**: <dir> a directory to recurse to look for pcaps - read mode is implied
- string **snort.--pcap-filter**: <filter> filter to apply when getting pcaps from file or directory
- int **snort.--pcap-loop**: <count> read all pcaps <count> times; 0 will read until Snort is terminated { 0:max32 }
- implied **snort.--pcap-no-filter**: reset to use no filter when getting pcaps from file or directory
- implied **snort.--pcap-show**: print a line saying what pcap is currently being read
- implied **snort.--pedantic**: warnings are fatal
- string **snort.--plugin-path**: <path> a colon separated list of directories or plugin libraries
- implied **snort.--process-all-events**: process all action groups
- string **snort.--rule**: <rules> to be added to configuration; may be repeated
- string **snort.--rule-path**: <path> where to find rules files
- implied **snort.--rule-to-hex**: output so rule header to stdout for text rule on stdin
- string **snort.--rule-to-text**: output plain so rule header to stdout for text rule on stdin (specify delimiter or [Snort_SO_Rule] will be used) { 16 }
- string **snort.--run-prefix**: <pfx> prepend this to each output file
- string **snort.--script-path**: <path> to a luajit script or directory containing luajit scripts
- implied **snort.--shell**: enable the interactive command line
- implied **snort.--show-file-codes**: indicate how files are located: A=absolute and W, F, C which are relative to the working directory, including file, and config file respectively
- implied **snort.--show-plugins**: list module and plugin versions
- int **snort.--skip**: <n> skip 1st n packets { 0:max53 }
- int **snort.--snaplen**: <snap> set snaplen of packet (same as -s) { 0:65535 }
- implied **snort.--stdin-rules**: read rules from stdin until EOF or a line starting with END is read
- implied **snort.--talos**: enable Talos tweak (same as --tweaks talos)
- string **snort.--tweaks**: tune configuration
- implied **snort.--version**: show version number (same as -V)
- implied **snort.--warn-all**: enable all warnings

- implied **snort.--warn-conf**: warn about configuration issues
- implied **snort.--warn-conf-strict**: warn about unrecognized elements in configuration files
- implied **snort.--warn-daq**: warn about DAQ issues, usually related to mode
- implied **snort.--warn-flowbits**: warn about flowbits that are checked but not set and vice-versa
- implied **snort.--warn-hosts**: warn about host table issues
- implied **snort.--warn-plugins**: warn about issues that prevent plugins from loading
- implied **snort.--warn-rules**: warn about duplicate rules and rule parsing issues
- implied **snort.--warn-scripts**: warn about issues discovered while processing Lua scripts
- implied **snort.--warn-symbols**: warn about unknown symbols in your Lua config
- implied **snort.--warn-vars**: warn about variable definition and usage issues
- int **snort.--x2c**: output ASCII char for given hex (see also --c2x) { 0x00:0xFF }
- string **snort.--x2s**: output ASCII string for given byte code (see also --x2c)

Commands:

- **snort.set_watchdog_params**(timer, min_thread_count): set watchdog parameters
- **snort.show_plugins**(): show available plugins
- **snort.delete_inspector**(inspector): delete an inspector from the default policy
- **snort.dump_stats**(): show summary statistics
- **snort.dump_heap_stats**(): show heap statistics
- **snort.reset_stats**(type): clear summary statistics. Type can be: daq|module|appid|file_id|snort|ha|all. reset_stats() without a parameter clears all statistics.
- **snort.rotate_stats**(): roll perfmonitor log files
- **snort.reload_config**(filename): load new configuration
- **snort.reload_policy**(filename): reload part or all of the default policy
- **snort.reload_daq**(): reload daq module
- **snort.reload_hosts**(filename): load a new hosts table
- **snort.log_command**(command, logging): enable or disable command logging
- **snort.show_config_generation**(): show loaded configuration ID
- **snort.pause**(): suspend packet processing
- **snort.resume**(pkt_num): continue packet processing. If number of packets is specified, will resume for n packets and pause
- **snort.detach**(): detach from control shell (without shutting down)
- **snort.quit**(): shutdown and dump-stats
- **snort.help**(): this output

Peg counts:

- **snort.local_commands**: total local commands processed (sum)

- **snort.remote_commands**: total remote commands processed (sum)
- **snort.signals**: total signals processed (sum)
- **snort.conf_reloads**: number of times configuration was reloaded (sum)
- **snort.policy_reloads**: number of times policies were reloaded (sum)
- **snort.inspector_deletions**: number of times inspectors were deleted (sum)
- **snort.daq_reloads**: number of times daq configuration was reloaded (sum)
- **snort.attribute_table_reloads**: number of times hosts attribute table was reloaded (sum)
- **snort.attribute_table_hosts**: number of hosts added to the attribute table (sum)
- **snort.attribute_table_overflow**: number of host additions that failed due to attribute table full (sum)

2.32 suppress

Help: configure event suppressions

Type: basic

Usage: context

Configuration:

- int **`suppress[].gid`** = 0: rule generator ID { 0:8129 }
- int **`suppress[].sid`** = 0: rule signature ID { 0:max32 }
- enum **`suppress[].track`**: suppress only matching source or destination addresses { *by_src* | *by_dst* }
- string **`suppress[].ip`**: restrict suppression to these addresses according to track

2.33 trace

Help: configure trace log messages

Type: basic

Usage: global

Configuration:

- int **trace.modules.all**: enable trace for all modules { 0:255 }
- int **trace.modules.appid.all**: enable all trace options { 0:255 }
- int **trace.modules.dce_smb.all**: enable all trace options { 0:255 }
- int **trace.modules.dpx.all**: enable all trace options { 0:255 }
- int **trace.modules.file_id.all**: enable all trace options { 0:255 }
- int **trace.modules.js_norm.all**: enable all trace options { 0:255 }
- int **trace.modules.js_norm.proc**: enable processing logging { 0:255 }
- int **trace.modules.js_norm.dump**: enable data logging { 0:255 }
- int **trace.modules.memory.all**: enable all trace options { 0:255 }
- int **trace.modules.snort.all**: enable all trace options { 0:255 }

- int **trace.modules.snort.inspector_manager**: enable inspector manager trace logging { 0:255 }
- int **trace.modules.vba_data.all**: enable all trace options { 0:255 }
- int **trace.modules.wizard.all**: enable all trace options { 0:255 }
- int **trace.constraints.ip_proto**: numerical IP protocol ID filter { 0:255 }
- string **trace.constraints.src_ip**: source IP address filter
- int **trace.constraints.src_port**: source port filter { 0:65535 }
- string **trace.constraints.dst_ip**: destination IP address filter
- int **trace.constraints.dst_port**: destination port filter { 0:65535 }
- bool **trace.constraints.match** = true: use constraints to filter traces
- enum **trace.output**: output method for trace log messages { *stdout* | *syslog* }
- bool **trace.ntuple** = false: print packet n-tuple info with trace messages
- bool **trace.timestamp** = false: print message timestamps with trace messages

Commands:

- **trace.set**(modules, constraints, ntuple, timestamp): set modules traces, constraints, ntuple and timestamp options
- **trace.clear**(): clear modules traces and constraints

3 Codec Modules

Codec is short for coder / decoder. These modules are used for basic protocol decoding, anomaly detection, and construction of active responses.

3.1 arp

Help: support for address resolution protocol

Type: codec

Usage: context

Rules:

- **116:109** (arp) truncated ARP

3.2 auth

Help: support for IP authentication header

Type: codec

Usage: context

Rules:

- **116:465** (auth) truncated authentication header
- **116:466** (auth) bad authentication header length

3.3 ciscometadata

Help: support for cisco metadata

Type: codec

Usage: context

Rules:

- **116:468** (ciscometadata) truncated Cisco Metadata header
- **116:469** (ciscometadata) invalid Cisco Metadata option length
- **116:470** (ciscometadata) invalid Cisco Metadata option type
- **116:471** (ciscometadata) invalid Cisco Metadata security group tag

Peg counts:

- **ciscometadata.truncated_hdr**: total truncated Cisco Metadata headers (sum)
- **ciscometadata.invalid_hdr_ver**: total invalid Cisco Metadata header versions (sum)
- **ciscometadata.invalid_hdr_len**: total invalid Cisco Metadata header lengths (sum)
- **ciscometadata.invalid_opt_len**: total invalid Cisco Metadata option lengths (sum)
- **ciscometadata.invalid_opt_type**: total invalid Cisco Metadata option types (sum)
- **ciscometadata.invalid_sgt**: total invalid Cisco Metadata security group tags (sum)

3.4 eapol

Help: support for extensible authentication protocol over LAN

Type: codec

Usage: context

Rules:

- **116:110** (eapol) truncated EAP header
- **116:111** (eapol) EAP key truncated
- **116:112** (eapol) EAP header truncated

3.5 erspan2

Help: support for encapsulated remote switched port analyzer - type 2

Type: codec

Usage: context

Rules:

- **116:462** (erspan2) ERSpan header version mismatch
- **116:463** (erspan2) captured length < ERSpan type2 header length

3.6 erspan3

Help: support for encapsulated remote switched port analyzer - type 3

Type: codec

Usage: context

Rules:

- **116:464** (erspan3) captured < ERSpan type3 header length

3.7 esp

Help: support for encapsulating security payload

Type: codec

Usage: context

Configuration:

- bool **esp.decode_esp** = false: enable for inspection of esp traffic that has authentication but not encryption

Rules:

- **116:294** (esp) truncated encapsulated security payload header

3.8 eth

Help: support for ethernet protocol (DLT 1) (DLT 51)

Type: codec

Usage: context

Rules:

- **116:424** (eth) truncated ethernet header

3.9 fabricpath

Help: support for fabricpath

Type: codec

Usage: context

Rules:

- **116:467** (fabricpath) truncated FabricPath header

3.10 geneve

Help: support for Geneve: Generic Network Virtualization Encapsulation

Type: codec

Usage: context

Rules:

- **116:180** (geneve) insufficient room for geneve header
- **116:181** (geneve) invalid version
- **116:182** (geneve) invalid header
- **116:183** (geneve) invalid flags
- **116:184** (geneve) invalid options

3.11 gre

Help: support for generic routing encapsulation

Type: codec

Usage: context

Rules:

- **116:160** (gre) GRE header length > payload length
- **116:161** (gre) multiple encapsulations in packet
- **116:162** (gre) invalid GRE version
- **116:163** (gre) invalid GRE header
- **116:164** (gre) invalid GRE v.1 PPTP header
- **116:165** (gre) GRE trans header length > payload length

3.12 gtp

Help: support for general-packet-radio-service tunneling protocol

Type: codec

Usage: context

Rules:

- **116:297** (gtp) two or more GTP encapsulation layers present
- **116:298** (gtp) GTP header length is invalid

3.13 icmp4

Help: support for Internet control message protocol v4

Type: codec

Usage: context

Rules:

- **116:105** (icmp4) ICMP header truncated
- **116:106** (icmp4) ICMP timestamp header truncated
- **116:107** (icmp4) ICMP address header truncated
- **116:250** (icmp4) ICMP original IP header truncated
- **116:251** (icmp4) ICMP version and original IP header versions differ
- **116:252** (icmp4) ICMP original datagram length < original IP header length
- **116:253** (icmp4) ICMP original IP payload < 64 bits
- **116:254** (icmp4) ICMP original IP payload > 576 bytes
- **116:255** (icmp4) ICMP original IP fragmented and offset not 0
- **116:415** (icmp4) ICMP4 packet to multicast dest address
- **116:416** (icmp4) ICMP4 packet to broadcast dest address
- **116:418** (icmp4) ICMP4 type other
- **116:426** (icmp4) truncated ICMP4 header
- **116:434** (icmp4) ICMP ping Nmap
- **116:435** (icmp4) ICMP icmpenum v1.1.1
- **116:436** (icmp4) ICMP redirect host
- **116:437** (icmp4) ICMP redirect net
- **116:438** (icmp4) ICMP traceroute ipopts
- **116:439** (icmp4) ICMP source quench
- **116:440** (icmp4) broadscan smurf scanner
- **116:441** (icmp4) ICMP destination unreachable communication administratively prohibited
- **116:442** (icmp4) ICMP destination unreachable communication with destination host is administratively prohibited
- **116:443** (icmp4) ICMP destination unreachable communication with destination network is administratively prohibited
- **116:451** (icmp4) ICMP path MTU denial of service attempt
- **116:452** (icmp4) Linux ICMP header DOS attempt

Peg counts:

- **icmp4.bad_checksum**: non-zero icmp checksums (sum)
- **icmp4.checksum_bypassed**: checksum calculations bypassed (sum)

3.14 icmp6

Help: support for Internet control message protocol v6

Type: codec

Usage: context

Rules:

- **116:285** (icmp6) ICMPv6 packet of type 2 (message too big) with MTU field < 1280
- **116:286** (icmp6) ICMPv6 packet of type 1 (destination unreachable) with non-RFC 2463 code
- **116:287** (icmp6) ICMPv6 router solicitation packet with a code not equal to 0
- **116:288** (icmp6) ICMPv6 router advertisement packet with a code not equal to 0
- **116:289** (icmp6) ICMPv6 router solicitation packet with the reserved field not equal to 0
- **116:290** (icmp6) ICMPv6 router advertisement packet with the reachable time field set > 1 hour
- **116:427** (icmp6) truncated ICMPv6 header
- **116:431** (icmp6) ICMPv6 type not decoded
- **116:432** (icmp6) ICMPv6 packet to multicast address
- **116:457** (icmp6) ICMPv6 packet of type 1 (destination unreachable) with non-RFC 4443 code
- **116:460** (icmp6) ICMPv6 node info query/response packet with a code greater than 2
- **116:474** (icmp6) ICMPv6 not encapsulated in IPv6

Peg counts:

- **icmp6.bad_icmp6_checksum**: nonzero icmp6 checksums (sum)
- **icmp6.checksum_bypassed**: checksum calculations bypassed (sum)

3.15 igmp

Help: support for Internet group management protocol

Type: codec

Usage: context

Rules:

- **116:455** (igmp) DOS IGMP IP options validation attempt

3.16 ipv4

Help: support for Internet protocol v4 (DLT 228)

Type: codec

Usage: context

Rules:

- **116:1** (ipv4) not IPv4 datagram

- **116:2** (ipv4) IPv4 header length < minimum
- **116:3** (ipv4) IPv4 datagram length < header field
- **116:4** (ipv4) IPv4 options found with bad lengths
- **116:5** (ipv4) truncated IPv4 options
- **116:6** (ipv4) IPv4 datagram length > captured length
- **116:404** (ipv4) IPv4 packet with zero TTL
- **116:405** (ipv4) IPv4 packet with bad frag bits (both MF and DF set)
- **116:407** (ipv4) IPv4 packet frag offset + length exceed maximum
- **116:408** (ipv4) IPv4 packet from *current net* source address
- **116:409** (ipv4) IPv4 packet to *current net* dest address
- **116:410** (ipv4) IPv4 packet from multicast source address
- **116:411** (ipv4) IPv4 packet from reserved source address
- **116:412** (ipv4) IPv4 packet to reserved dest address
- **116:413** (ipv4) IPv4 packet from broadcast source address
- **116:414** (ipv4) IPv4 packet to broadcast dest address
- **116:425** (ipv4) truncated IPv4 header
- **116:428** (ipv4) IPv4 packet below TTL limit
- **116:430** (ipv4) IPv4 packet both DF and offset set
- **116:444** (ipv4) IPv4 option set
- **116:448** (ipv4) IPv4 reserved bit set

Peg counts:

- **ipv4.bad_checksum**: nonzero ip checksums (sum)
- **ipv4.checksum_bypassed**: checksum calculations bypassed (sum)

3.17 ipv6

Help: support for Internet protocol v6 (DLT 229)

Type: codec

Usage: context

Rules:

- **116:270** (ipv6) IPv6 packet below TTL limit
- **116:271** (ipv6) IPv6 header claims to not be IPv6
- **116:272** (ipv6) IPv6 truncated extension header
- **116:273** (ipv6) IPv6 truncated header
- **116:274** (ipv6) IPv6 datagram length < header field

- **116:275** (ipv6) IPv6 datagram length > captured length
- **116:276** (ipv6) IPv6 packet with destination address ::0
- **116:277** (ipv6) IPv6 packet with multicast source address
- **116:278** (ipv6) IPv6 packet with reserved multicast destination address
- **116:279** (ipv6) IPv6 header includes an undefined option type
- **116:280** (ipv6) IPv6 address includes an unassigned multicast scope value
- **116:281** (ipv6) IPv6 header includes an invalid value for the *next header* field
- **116:282** (ipv6) IPv6 header includes a routing extension header followed by a hop-by-hop header
- **116:283** (ipv6) IPv6 header includes two routing extension headers
- **116:291** (ipv6) IPV6 tunneled over IPv4, IPv6 header truncated, possible Linux kernel attack
- **116:292** (ipv6) IPv6 header has destination options followed by a routing header
- **116:295** (ipv6) IPv6 header includes an option which is too big for the containing header
- **116:296** (ipv6) IPv6 packet includes out-of-order extension headers
- **116:429** (ipv6) IPv6 packet has zero hop limit
- **116:453** (ipv6) ISATAP-addressed IPv6 traffic spoofing attempt
- **116:456** (ipv6) too many IPv6 extension headers
- **116:458** (ipv6) bogus fragmentation packet, possible BSD attack
- **116:461** (ipv6) IPv6 routing type 0 extension header
- **116:475** (ipv6) IPv6 mobility header includes an invalid value for the *payload protocol* field
- **116:476** (ipv6) IPv6 packet from reserved source address
- **116:477** (ipv6) IPv6 packet to reserved dest address

3.18 llc

Help: support for logical link control

Type: codec

Usage: context

Rules:

- **116:131** (llc) bad LLC header
- **116:132** (llc) bad extra LLC info

3.19 mpls

Help: support for multiprotocol label switching

Type: codec

Usage: context

Configuration:

- int **mpls.max_stack_depth** = -1: set maximum MPLS stack depth { -1:255 }
- enum **mpls.payload_type** = *auto*: force encapsulated payload type { *auto* | *eth* | *ip4* | *ip6* }

Rules:

- **116:170** (mpls) bad MPLS frame
- **116:171** (mpls) MPLS label 0 appears in bottom header when not decoding as ip4
- **116:172** (mpls) MPLS label 1 appears in bottom header
- **116:173** (mpls) MPLS label 2 appears in bottom header when not decoding as ip6
- **116:174** (mpls) MPLS label 3 appears in header
- **116:175** (mpls) MPLS label 4, 5,.. or 15 appears in header
- **116:176** (mpls) too many MPLS headers

3.20 pbb

Help: support for 802.1ah protocol

Type: codec

Usage: context

Rules:

- **116:424** (pbb) truncated ethernet header

3.21 pgm

Help: support for pragmatic general multicast

Type: codec

Usage: context

Rules:

- **116:454** (pgm) PGM nak list overflow attempt

3.22 pppoe

Help: support for point-to-point protocol over ethernet

Type: codec

Usage: context

Rules:

- **116:120** (pppoe) bad PPPOE frame detected

3.23 tcp

Help: support for transmission control protocol

Type: codec

Usage: context

Rules:

- **116:45** (tcp) TCP packet length is smaller than 20 bytes
- **116:46** (tcp) TCP data offset is less than 5
- **116:47** (tcp) TCP header length exceeds packet length
- **116:54** (tcp) TCP options found with bad lengths
- **116:55** (tcp) truncated TCP options
- **116:56** (tcp) T/TCP detected
- **116:57** (tcp) obsolete TCP options found
- **116:58** (tcp) experimental TCP options found
- **116:59** (tcp) TCP window scale option found with length > 14
- **116:400** (tcp) XMAS attack detected
- **116:401** (tcp) Nmap XMAS attack detected
- **116:402** (tcp) DOS NAPTHA vulnerability detected
- **116:403** (tcp) SYN to multicast address
- **116:419** (tcp) TCP urgent pointer exceeds payload length or no payload
- **116:420** (tcp) TCP SYN with FIN
- **116:421** (tcp) TCP SYN with RST
- **116:422** (tcp) TCP PDU missing ack for established session
- **116:423** (tcp) TCP has no SYN, ACK, or RST
- **116:433** (tcp) DDOS shaft SYN flood
- **116:446** (tcp) TCP port 0 traffic

Peg counts:

- **tcp.bad_tcp4_checksum**: nonzero tcp over ip checksums (sum)
- **tcp.bad_tcp6_checksum**: nonzero tcp over ipv6 checksums (sum)
- **tcp.checksum_bypassed**: checksum calculations bypassed (sum)

3.24 token_ring

Help: support for token ring decoding

Type: codec

Usage: context

Rules:

- **116:140** (token_ring) bad Token Ring header
- **116:141** (token_ring) bad Token Ring ETHLLC header
- **116:142** (token_ring) bad Token Ring MRLEN header
- **116:143** (token_ring) bad Token Ring MR header

3.25 udp

Help: support for user datagram protocol

Type: codec

Usage: context

Configuration:

- bool **udp.deep_teredo_inspection** = false: look for Teredo on all UDP ports (default is only 3544)
- bit_list **udp.gtp_ports** = *2152 3386*: set GTP ports { 65535 }
- bit_list **udp.vxlan_ports** = *4789*: set VXLAN ports { 65535 }
- bit_list **udp.geneve_ports** = *6081*: set Geneve ports { 65535 }

Rules:

- **116:95** (udp) truncated UDP header
- **116:96** (udp) invalid UDP header, length field < 8
- **116:97** (udp) short UDP packet, length field > payload length
- **116:98** (udp) long UDP packet, length field < payload length
- **116:406** (udp) invalid IPv6 UDP packet, checksum zero
- **116:445** (udp) large UDP packet (> 4000 bytes)
- **116:447** (udp) UDP port 0 traffic

Peg counts:

- **udp.bad_udp4_checksum**: nonzero udp over ipv4 checksums (sum)
- **udp.bad_udp6_checksum**: nonzero udp over ipv6 checksums (sum)
- **udp.checksum_bypassed**: checksum calculations bypassed (sum)

3.26 vlan

Help: support for local area network

Type: codec

Usage: context

Configuration:

- int_list **vlan.extra_tpid_ether_types** = *0x9100 0x9200*: set non-standard QinQ ether types { 65535 }

Rules:

- **116:130** (vlan) bad VLAN frame

3.27 wlan

Help: support for wireless local area network protocol (DLT 105)

Type: codec

Usage: context

Rules:

- **116:133** (wlan) bad 802.11 LLC header
- **116:134** (wlan) bad 802.11 extra LLC info

4 Connector Modules

Connectors support High Availability communication links.

4.1 file_connector

Help: implement the file based connector

Type: connector

Usage: global

Configuration:

- string **`file_connector[].connector`**: connector name
- string **`file_connector[].name`**: channel name
- enum **`file_connector[].format`**: file format { *binary* | *text* }
- enum **`file_connector[].direction`**: usage { *receive* | *transmit* | *duplex* }

Peg counts:

- **file_connector.messages**: total messages (sum)

4.2 tcp_connector

Help: implement the tcp stream connector

Type: connector

Usage: global

Configuration:

- string **`tcp_connector[].connector`**: connector name
- string **`tcp_connector[].address`**: address
- port **`tcp_connector[].base_port`**: base port number
- enum **`tcp_connector[].setup`**: stream establishment { *call* | *answer* }

Peg counts:

- **tcp_connector.messages**: total messages (sum)

5 Inspector Modules

These modules perform a variety of functions, including analysis of protocols beyond basic decoding.

5.1 appid

Help: application and service identification

Type: inspector (control)

Usage: global

Instance Type: global

Configuration:

- int **appid.memcap** = 1048576: max size of the service cache before we start pruning the cache { 1024:maxSZ }
- bool **appid.log_stats** = false: enable logging of appid statistics
- int **appid.app_stats_period** = 300: time period for collecting and logging appid statistics { 1:max32 }
- int **appid.app_stats_rollover_size** = 20971520: max file size for appid stats before rolling over the log file { 0:max32 }
- string **appid.app_detector_dir**: directory to load appid detectors from
- bool **appid.list_odp_detectors** = false: enable logging of odp detectors statistics
- string **appid.tp_appid_path**: path to third party appid dynamic library
- string **appid.tp_appid_config**: path to third party appid configuration file
- bool **appid.tp_appid_stats_enable**: enable collection of stats and print stats on exit in third party module
- bool **appid.tp_appid_config_dump**: print third party configuration on startup
- bool **appid.log_all_sessions** = false: enable logging of all appid sessions
- bool **appid.enable_rna_filter** = false: monitor only the networks specified in rna configuration
- string **appid.rna_conf_path**: path to rna configuration file

Commands:

- **appid.enable_debug**(proto, src_ip, src_port, dst_ip, dst_port): enable appid debugging
- **appid.disable_debug**(): disable appid debugging
- **appid.reload_third_party**(): reload appid third-party module
- **appid.reload_detectors**(): reload appid detectors

Peg counts:

- **appid.packets**: count of packets received (sum)
- **appid.processed_packets**: count of packets processed (sum)
- **appid.ignored_packets**: count of packets ignored (sum)
- **appid.total_sessions**: count of sessions created (sum)
- **appid.service_cache_prunes**: number of times the service cache was pruned (sum)
- **appid.service_cache_adds**: number of times an entry was added to the service cache (sum)
- **appid.service_cache_removes**: number of times an item was removed from the service cache (sum)
- **appid.odp_reload_ignored_pkts**: count of packets ignored after open detector package is reloaded (sum)
- **appid.tp_reload_ignored_pkts**: count of packets ignored after third-party module is reloaded (sum)
- **appid.bytes_in_use**: number of bytes in use in the cache (now)
- **appid.items_in_use**: items in use in the cache (now)

5.2 appid_listener

Help: log selected published data to appid_listener.log

Type: inspector (passive)

Usage: context

Instance Type: network

Configuration:

- bool **appid_listener.json_logging** = false: log appid data in json format
- string **appid_listener.file**: output data to given file

5.3 arp_spoof

Help: detect ARP attacks and anomalies

Type: inspector (network)

Usage: inspect

Instance Type: singleton

Configuration:

- ip4 **`arp_spoof.hosts[].ip`**: host ip address

- mac **arp_spoof.hosts[].mac**: host mac address

Rules:

- **112:1** (arp_spoof) unicast ARP request
- **112:2** (arp_spoof) ethernet/ARP mismatch for source hardware address
- **112:3** (arp_spoof) ethernet/ARP mismatch for destination hardware address in reply
- **112:4** (arp_spoof) attempted ARP cache overwrite attack

Peg counts:

- **arp_spoof.packets**: total packets (sum)

5.4 back_orifice

Help: back orifice detection

Type: inspector (network)

Usage: inspect

Instance Type: multiton

Rules:

- **105:1** (back_orifice) Back Orifice traffic detected, unknown direction
- **105:2** (back_orifice) Back Orifice client traffic detected
- **105:3** (back_orifice) Back Orifice server traffic detected
- **105:4** (back_orifice) Back Orifice length field >= 1024 bytes

Peg counts:

- **back_orifice.packets**: total packets (sum)

5.5 binder

Help: configure processing based on CIDRs, ports, services, etc.

Type: inspector (passive)

Usage: inspect

Instance Type: singleton

Configuration:

- int **binder[].when.ips_policy_id**: unique ID for selection of this config by external logic { 0:max32 }
- bit_list **binder[].when.vlans**: list of VLAN IDs { 4095 }
- addr_list **binder[].when.nets**: list of networks
- addr_list **binder[].when.src_nets**: list of source networks
- addr_list **binder[].when.dst_nets**: list of destination networks
- enum **binder[].when.proto**: protocol { *any* | *ip* | *icmp* | *tcp* | *udp* | *user* | *file* }

- bit_list **binder[].when.ports**: list of ports { 65535 }
- bit_list **binder[].when.src_ports**: list of source ports { 65535 }
- bit_list **binder[].when.dst_ports**: list of destination ports { 65535 }
- string **binder[].when.intfs**: list of interface IDs
- string **binder[].when.src_intfs**: list of source interface IDs
- string **binder[].when.dst_intfs**: list of destination interface IDs
- string **binder[].when.groups**: list of interface group IDs
- string **binder[].when.src_groups**: list of source interface group IDs
- string **binder[].when.dst_groups**: list of destination group IDs
- string **binder[].when.addr_spaces**: list of address space IDs
- string **binder[].when.tenants**: list of tenants
- enum **binder[].when.role** = *any*: use the given configuration on one or any end of a session { *client* | *server* | *any* }
- string **binder[].when.service**: override default configuration
- string **binder[].when.zones**: deprecated alias for groups
- string **binder[].when.src_zone**: deprecated alias for src_groups
- string **binder[].when.dst_zone**: deprecated alias for dst_groups
- enum **binder[].use.action** = *inspect*: what to do with matching traffic { *reset* | *block* | *allow* | *inspect* }
- string **binder[].use.file**: use configuration in given file
- string **binder[].use.inspection_policy**: use inspection policy from given file
- string **binder[].use.ips_policy**: use ips policy from given file
- string **binder[].use.service**: override automatic service identification
- string **binder[].use.type**: select module for binding
- string **binder[].use.name**: symbol name (defaults to type)

Peg counts:

- **binder.raw_packets**: raw packets evaluated (sum)
- **binder.new_flows**: new flows evaluated (sum)
- **binder.rebinds**: flows rebound (sum)
- **binder.service_changes**: flow service changes evaluated (sum)
- **binder.assistant_inspectors**: flow assistant inspector requests handled (sum)
- **binder.new_standby_flows**: new HA flows evaluated (sum)
- **binder.no_match**: binding evaluations that had no matches (sum)
- **binder.resets**: reset actions bound (sum)
- **binder.blocks**: block actions bound (sum)
- **binder.allows**: allow actions bound (sum)
- **binder.inspects**: inspect actions bound (sum)

5.6 cip

Help: cip inspection

Type: inspector (service)

Usage: inspect

Instance Type: multiton

Configuration:

- string **cip.embedded_cip_path** = *false*: check embedded CIP path
- int **cip.unconnected_timeout** = 300: unconnected timeout in seconds { 0:360 }
- int **cip.max_cip_connections** = 100: max cip connections { 1:10000 }
- int **cip.max_unconnected_messages** = 100: max unconnected cip messages { 1:10000 }

Rules:

- **148:1** (cip) CIP data is malformed
- **148:2** (cip) CIP data is non-conforming to ODVA standard
- **148:3** (cip) CIP connection limit exceeded. Least recently used connection removed
- **148:4** (cip) CIP unconnected request limit exceeded. Oldest request removed

Peg counts:

- **cip.packets**: total packets (sum)
- **cip.session**: total sessions (sum)
- **cip.concurrent_sessions**: total concurrent SIP sessions (now)
- **cip.max_concurrent_sessions**: maximum concurrent SIP sessions (max)

5.7 cpeos_test

Help: for testing CPE OS RNA event generation

Type: inspector (control)

Usage: context

Instance Type: network

5.8 data_log

Help: log selected published data to data.log

Type: inspector (passive)

Usage: inspect

Instance Type: singleton

Configuration:

- select **data_log.key** = 'http_request_header_event ': name of the event to log { http_request_header_event | http_response_header_even
  }
- int **data_log.limit** = 0: set maximum size in MB before rollover (0 is unlimited) { 0:max32 }

Peg counts:

- **data_log.packets**: total packets (sum)

5.9 dce_http_proxy

Help: dce over http inspection - client to/from proxy

Type: inspector (service)

Usage: inspect

Instance Type: multiton

Peg counts:

- **dce_http_proxy.http_proxy_sessions**: successful http proxy sessions (sum)
- **dce_http_proxy.http_proxy_session_failures**: failed http proxy sessions (sum)

5.10 dce_http_server

Help: dce over http inspection - proxy to/from server

Type: inspector (service)

Usage: inspect

Instance Type: multiton

Peg counts:

- **dce_http_server.http_server_sessions**: successful http server sessions (sum)
- **dce_http_server.http_server_session_failures**: failed http server sessions (sum)

5.11 dce_smb

Help: dce over smb inspection

Type: inspector (service)

Usage: inspect

Instance Type: multiton

Configuration:

- bool **dce_smb.limit_alerts** = true: limit DCE alert to at most one per signature per flow
- bool **dce_smb.disable_defrag** = false: disable DCE/RPC defragmentation
- int **dce_smb.max_frag_len** = 65535: maximum fragment size for defragmentation { 1514:65535 }
- int **dce_smb.reassemble_threshold** = 0: minimum bytes received before performing reassembly { 0:65535 }
- enum **dce_smb.smb_fingerprint_policy** = *none*: target based SMB policy to use { *none* | *client* | *server* | *both* }
- enum **dce_smb.policy** = *WinXP*: target based policy to use { *Win2000* | *WinXP* | *WinVista* | *Win2003* | *Win2008* | *Win7* | *Samba* | *Samba-3.0.37* | *Samba-3.0.22* | *Samba-3.0.20* }
- int **dce_smb.smb_max_chain** = 3: SMB max chain size { 0:255 }
- int **dce_smb.smb_max_compound** = 3: SMB max compound size { 0:255 }
- multi **dce_smb.valid_smb_versions** = *all*: valid SMB versions { v1 | v2 | all }
- enum **dce_smb.smb_file_inspection**: deprecated (not used): file inspection controlled by smb_file_depth { *off* | *on* | *only* }

- int **dce_smb.smb_file_depth** = 16384: SMB file depth for file data (-1 = disabled, 0 = unlimited) { -1:32767 }
- string **dce_smb.smb_invalid_shares**: SMB shares to alert on
- bool **dce_smb.smb_legacy_mode** = false: inspect only SMBv1
- int **dce_smb.smb_max_credit** = 8192: Maximum number of outstanding request { 1:65535 }
- int **dce_smb.memcap** = 8388608: Memory utilization limit on SMBv2 cache { 512:maxSZ }

Rules:

- **133:2** (dce_smb) SMB - bad NetBIOS session service session type
- **133:3** (dce_smb) SMB - bad SMB message type
- **133:4** (dce_smb) SMB - bad SMB Id (not \xffSMB for SMB1 or not \xfeSMB for SMB2)
- **133:5** (dce_smb) SMB - bad word count or structure size
- **133:6** (dce_smb) SMB - bad byte count
- **133:7** (dce_smb) SMB - bad format type
- **133:8** (dce_smb) SMB - bad offset
- **133:9** (dce_smb) SMB - zero total data count
- **133:10** (dce_smb) SMB - NetBIOS data length less than SMB header length
- **133:11** (dce_smb) SMB - remaining NetBIOS data length less than command length
- **133:12** (dce_smb) SMB - remaining NetBIOS data length less than command byte count
- **133:13** (dce_smb) SMB - remaining NetBIOS data length less than command data size
- **133:14** (dce_smb) SMB - remaining total data count less than this command data size
- **133:15** (dce_smb) SMB - total data sent (STDu64) greater than command total data expected
- **133:16** (dce_smb) SMB - byte count less than command data size (STDu64)
- **133:17** (dce_smb) SMB - invalid command data size for byte count
- **133:18** (dce_smb) SMB - excessive tree connect requests with pending tree connect responses
- **133:19** (dce_smb) SMB - excessive read requests with pending read responses
- **133:20** (dce_smb) SMB - excessive command chaining
- **133:21** (dce_smb) SMB - multiple chained tree connect requests
- **133:22** (dce_smb) SMB - multiple chained tree connect requests
- **133:23** (dce_smb) SMB - chained/compounded login followed by logoff
- **133:24** (dce_smb) SMB - chained/compounded tree connect followed by tree disconnect
- **133:25** (dce_smb) SMB - chained/compounded open pipe followed by close pipe
- **133:26** (dce_smb) SMB - invalid share access
- **133:44** (dce_smb) SMB - invalid SMB version 1 seen
- **133:45** (dce_smb) SMB - invalid SMB version 2 seen
- **133:46** (dce_smb) SMB - invalid user, tree connect, file binding

- **133:47** (dce_smb) SMB - excessive command compounding
- **133:48** (dce_smb) SMB - zero data count
- **133:50** (dce_smb) SMB - maximum number of outstanding requests exceeded
- **133:51** (dce_smb) SMB - outstanding requests with same MID
- **133:52** (dce_smb) SMB - deprecated dialect negotiated
- **133:53** (dce_smb) SMB - deprecated command used
- **133:54** (dce_smb) SMB - unusual command used
- **133:55** (dce_smb) SMB - invalid setup count for command
- **133:56** (dce_smb) SMB - client attempted multiple dialect negotiations on session
- **133:57** (dce_smb) SMB - client attempted to create or set a file's attributes to readonly/hidden/system
- **133:58** (dce_smb) SMB - file offset provided is greater than file size specified
- **133:59** (dce_smb) SMB - next command specified in SMB2 header is beyond payload boundary

Peg counts:

- **dce_smb.cache_adds**: smbv2 cache added new entry (sum)
- **dce_smb.cache_hits**: smbv2 cache found existing entry (sum)
- **dce_smb.cache_misses**: smbv2 cache did not find entry (sum)
- **dce_smb.cache_replaces**: smbv2 cache found entry and replaced its value (sum)
- **dce_smb.cache_max**: smbv2 cache's maximum byte usage (sum)
- **dce_smb.cache_prunes**: smbv2 cache pruned entry to make space for new entry (sum)
- **dce_smb.cache_removes**: smbv2 cache removed existing entry (sum)
- **dce_smb.events**: total events (sum)
- **dce_smb.pdus**: total connection-oriented PDUs (sum)
- **dce_smb.binds**: total connection-oriented binds (sum)
- **dce_smb.bind_acks**: total connection-oriented binds acks (sum)
- **dce_smb.alter_contexts**: total connection-oriented alter contexts (sum)
- **dce_smb.alter_context_responses**: total connection-oriented alter context responses (sum)
- **dce_smb.bind_naks**: total connection-oriented bind naks (sum)
- **dce_smb.requests**: total connection-oriented requests (sum)
- **dce_smb.responses**: total connection-oriented responses (sum)
- **dce_smb.cancels**: total connection-oriented cancels (sum)
- **dce_smb.orphaned**: total connection-oriented orphaned (sum)
- **dce_smb.faults**: total connection-oriented faults (sum)
- **dce_smb.auth3s**: total connection-oriented auth3s (sum)
- **dce_smb.shutdowns**: total connection-oriented shutdowns (sum)

- **dce_smb.rejects**: total connection-oriented rejects (sum)
- **dce_smb.ms_rpc_http_pdus**: total connection-oriented MS requests to send RPC over HTTP (sum)
- **dce_smb.other_requests**: total connection-oriented other requests (sum)
- **dce_smb.other_responses**: total connection-oriented other responses (sum)
- **dce_smb.request_fragments**: total connection-oriented request fragments (sum)
- **dce_smb.response_fragments**: total connection-oriented response fragments (sum)
- **dce_smb.client_max_fragment_size**: connection-oriented client maximum fragment size (sum)
- **dce_smb.client_min_fragment_size**: connection-oriented client minimum fragment size (sum)
- **dce_smb.client_segs_reassembled**: total connection-oriented client segments reassembled (sum)
- **dce_smb.client_frags_reassembled**: total connection-oriented client fragments reassembled (sum)
- **dce_smb.server_max_fragment_size**: connection-oriented server maximum fragment size (sum)
- **dce_smb.server_min_fragment_size**: connection-oriented server minimum fragment size (sum)
- **dce_smb.server_segs_reassembled**: total connection-oriented server segments reassembled (sum)
- **dce_smb.server_frags_reassembled**: total connection-oriented server fragments reassembled (sum)
- **dce_smb.sessions**: total smb sessions (sum)
- **dce_smb.packets**: total smb packets (sum)
- **dce_smb.ignored_bytes**: total ignored bytes (sum)
- **dce_smb.smb_client_segs_reassembled**: total smb client segments reassembled (sum)
- **dce_smb.smb_server_segs_reassembled**: total smb server segments reassembled (sum)
- **dce_smb.max_outstanding_requests**: maximum outstanding requests (max)
- **dce_smb.files_processed**: total smb files processed (sum)
- **dce_smb.v2_setup**: total number of SMBv2 setup packets seen (sum)
- **dce_smb.v2_setup_err_resp**: total number of SMBv2 setup error response packets seen (sum)
- **dce_smb.v2_setup_inv_str_sz**: total number of SMBv2 setup packets seen with invalid structure size (sum)
- **dce_smb.v2_setup_resp_hdr_err**: total number of SMBv2 setup response packets ignored due to corrupted header (sum)
- **dce_smb.v2_tree_cnct**: total number of SMBv2 tree connect packets seen (sum)
- **dce_smb.v2_tree_cnct_err_resp**: total number of SMBv2 tree connect error response packets seen (sum)
- **dce_smb.v2_tree_cnct_ignored**: total number of SMBv2 setup response packets ignored due to failure in creating tree tracker (sum)
- **dce_smb.v2_tree_cnct_inv_str_sz**: total number of SMBv2 tree connect packets seen with invalid structure size (sum)
- **dce_smb.v2_tree_cnct_resp_hdr_err**: total number of SMBv2 tree connect response packets ignored due to corrupted header (sum)
- **dce_smb.v2_crt**: total number of SMBv2 create packets seen (sum)
- **dce_smb.v2_crt_err_resp**: total number of SMBv2 create error response packets seen (sum)
- **dce_smb.v2_crt_inv_file_data**: total number of SMBv2 create request packets ignored due to error in getting file name (sum)
- **dce_smb.v2_crt_inv_str_sz**: total number of SMBv2 create packets seen with invalid structure size (sum)

- **dce_smb.v2_crt_resp_hdr_err**: total number of SMBv2 create response packets ignored due to corrupted header (sum)
- **dce_smb.v2_crt_req_hdr_err**: total number of SMBv2 create request packets ignored due to corrupted header (sum)
- **dce_smb.v2_crt_rtrkr_misng**: total number of SMBv2 create response packets ignored due to missing create request tracker (sum)
- **dce_smb.v2_crt_req_ipc**: total number of SMBv2 create request packets ignored as share type is IPC (sum)
- **dce_smb.v2_crt_tree_trkr_misng**: total number of SMBv2 create response packets ignored due to missing tree tracker (sum)
- **dce_smb.v2_wrt**: total number of SMBv2 write packets seen (sum)
- **dce_smb.v2_wrt_err_resp**: total number of SMBv2 write error response packets seen (sum)
- **dce_smb.v2_wrt_ignored**: total number of SMBv2 write packets ignored due to missing trackers or invalid share type (sum)
- **dce_smb.v2_wrt_inv_str_sz**: total number of SMBv2 write packets seen with invalid structure size (sum)
- **dce_smb.v2_wrt_req_hdr_err**: total number of SMBv2 write request packets ignored due to corrupted header (sum)
- **dce_smb.v2_read**: total number of SMBv2 read packets seen (sum)
- **dce_smb.v2_read_err_resp**: total number of SMBv2 read error response packets seen (sum)
- **dce_smb.v2_read_ignored**: total number of SMBv2 write packets ignored due to missing trackers or invalid share type (sum)
- **dce_smb.v2_read_inv_str_sz**: total number of SMBv2 read packets seen with invalid structure size (sum)
- **dce_smb.v2_read_rtrkr_misng**: total number of SMBv2 read response packets ignored due to missing read request tracker (sum)
- **dce_smb.v2_read_resp_hdr_err**: total number of SMBv2 read response packets ignored due to corrupted header (sum)
- **dce_smb.v2_read_req_hdr_err**: total number of SMBv2 read request packets ignored due to corrupted header (sum)
- **dce_smb.v2_setinfo**: total number of SMBv2 set info packets seen (sum)
- **dce_smb.v2_stinf_err_resp**: total number of SMBv2 set info error response packets seen (sum)
- **dce_smb.v2_stinf_ignored**: total number of SMBv2 set info packets ignored due to missing trackers or invalid share type (sum)
- **dce_smb.v2_stinf_inv_str_sz**: total number of SMBv2 set info packets seen with invalid structure size (sum)
- **dce_smb.v2_stinf_req_ftrkr_misng**: total number of SMBv2 set info request packets ignored due to missing file tracker (sum)
- **dce_smb.v2_stinf_req_hdr_err**: total number of SMBv2 set info request packets ignored due to corrupted header (sum)
- **dce_smb.v2_cls**: total number of SMBv2 close packets seen (sum)
- **dce_smb.v2_cls_err_resp**: total number of SMBv2 close error response packets seen (sum)
- **dce_smb.v2_cls_ignored**: total number of SMBv2 close packets ignored due to missing trackers or invalid share type (sum)
- **dce_smb.v2_cls_inv_str_sz**: total number of SMBv2 close packets seen with invalid structure size (sum)
- **dce_smb.v2_cls_req_ftrkr_misng**: total number of SMBv2 close request packets ignored due to missing file tracker (sum)
- **dce_smb.v2_cls_req_hdr_err**: total number of SMBv2 close request packets ignored due to corrupted header (sum)
- **dce_smb.v2_tree_discn**: total number of SMBv2 tree disconnect packets seen (sum)
- **dce_smb.v2_tree_discn_ignored**: total number of SMBv2 tree disconnect packets ignored due to missing trackers or invalid share type (sum)
- **dce_smb.v2_tree_discn_inv_str_sz**: total number of SMBv2 tree disconnect packets seen with invalid structure size (sum)

- **dce_smb.v2_tree_discn_req_hdr_err**: total number of SMBv2 tree disconnect request packets ignored due to corrupted header (sum)
- **dce_smb.v2_logoff**: total number of SMBv2 logoff (sum)
- **dce_smb.v2_logoff_inv_str_sz**: total number of SMBv2 logoff packets seen with invalid structure size (sum)
- **dce_smb.v2_hdr_err**: total number of SMBv2 packets seen with corrupted hdr (sum)
- **dce_smb.v2_bad_next_cmd_offset**: total number of SMBv2 packets seen with invalid next command offset (sum)
- **dce_smb.v2_extra_file_data_err**: total number of SMBv2 packets seen with where file data beyond file size is observed (sum)
- **dce_smb.v2_inv_file_ctx_err**: total number of times null file context are seen resulting in not being able to set file size (sum)
- **dce_smb.v2_msgs_uninspected**: total number of SMBv2 packets seen where command is not being inspected (sum)
- **dce_smb.v2_cmpnd_req_lt_crossed**: total number of SMBv2 packets seen where compound requests exceed the smb_max_compoun limit (sum)
- **dce_smb.concurrent_sessions**: total concurrent sessions (now)
- **dce_smb.max_concurrent_sessions**: maximum concurrent sessions (max)
- **dce_smb.total_smb1_sessions**: total smb1 sessions (sum)
- **dce_smb.total_smb2_sessions**: total smb2 sessions (sum)
- **dce_smb.total_encrypted_sessions**: total encrypted sessions (sum)
- **dce_smb.total_mc_sessions**: total multichannel sessions (sum)
- **dce_smb.v2_total_session_trackers**: total number of session trackers (sum)
- **dce_smb.v2_total_tree_trackers**: total number of tree trackers (sum)
- **dce_smb.v2_total_file_trackers**: total number of file trackers (sum)
- **dce_smb.v2_updated_file_flows**: total number of updated file flows due to parent flow cleanup (sum)
- **dce_smb.v2_ignored_file_processing**: total number of file processing ignored (sum)
- **dce_smb.v2_mc_file_transfers**: total multichannel files transferred (sum)
- **dce_smb.v2_ioctl**: total number of ioctl calls (sum)
- **dce_smb.v2_ioctl_err_resp**: total number of ioctl errors responses (sum)
- **dce_smb.v2_ioctl_inv_str_sz**: total number of ioctl invalid structure size (sum)
- **dce_smb.v2_ioctl_ignored**: total number of SMBv2 IOCTL packets ignored due to missing trackers or invalid share type (sum)
- **dce_smb.total_sessions**: total smb sessions (sum)

5.12 dce_tcp

Help: dce over tcp inspection

Type: inspector (service)

Usage: inspect

Instance Type: multiton

Configuration:

- bool **dce_tcp.limit_alerts** = true: limit DCE alert to at most one per signature per flow
- bool **dce_tcp.disable_defrag** = false: disable DCE/RPC defragmentation
- int **dce_tcp.max_frag_len** = 65535: maximum fragment size for defragmentation { 1514:65535 }
- int **dce_tcp.reassemble_threshold** = 0: minimum bytes received before performing reassembly { 0:65535 }
- enum **dce_tcp.policy** = *WinXP*: target based policy to use { *Win2000* | *WinXP* | *WinVista* | *Win2003* | *Win2008* | *Win7* | *Samba* | *Samba-3.0.37* | *Samba-3.0.22* | *Samba-3.0.20* }

Rules:

- **133:27** (dce_tcp) connection oriented DCE/RPC - invalid major version
- **133:28** (dce_tcp) connection oriented DCE/RPC - invalid minor version
- **133:29** (dce_tcp) connection-oriented DCE/RPC - invalid PDU type
- **133:30** (dce_tcp) connection-oriented DCE/RPC - fragment length less than header size
- **133:31** (dce_tcp) connection-oriented DCE/RPC - remaining fragment length less than size needed
- **133:32** (dce_tcp) connection-oriented DCE/RPC - no context items specified
- **133:33** (dce_tcp) connection-oriented DCE/RPC -no transfer syntaxes specified
- **133:34** (dce_tcp) connection-oriented DCE/RPC - fragment length on non-last fragment less than maximum negotiated fragment transmit size for client
- **133:35** (dce_tcp) connection-oriented DCE/RPC - fragment length greater than maximum negotiated fragment transmit size
- **133:36** (dce_tcp) connection-oriented DCE/RPC - alter context byte order different from bind
- **133:37** (dce_tcp) connection-oriented DCE/RPC - call id of non first/last fragment different from call id established for fragmented request
- **133:38** (dce_tcp) connection-oriented DCE/RPC - opnum of non first/last fragment different from opnum established for fragmented request
- **133:39** (dce_tcp) connection-oriented DCE/RPC - context id of non first/last fragment different from context id established for fragmented request

Peg counts:

- **dce_tcp.events**: total events (sum)
- **dce_tcp.pdus**: total connection-oriented PDUs (sum)
- **dce_tcp.binds**: total connection-oriented binds (sum)
- **dce_tcp.bind_acks**: total connection-oriented binds acks (sum)
- **dce_tcp.alter_contexts**: total connection-oriented alter contexts (sum)
- **dce_tcp.alter_context_responses**: total connection-oriented alter context responses (sum)
- **dce_tcp.bind_naks**: total connection-oriented bind naks (sum)
- **dce_tcp.requests**: total connection-oriented requests (sum)
- **dce_tcp.responses**: total connection-oriented responses (sum)
- **dce_tcp.cancels**: total connection-oriented cancels (sum)
- **dce_tcp.orphaned**: total connection-oriented orphaned (sum)

- **dce_tcp.faults**: total connection-oriented faults (sum)
- **dce_tcp.auth3s**: total connection-oriented auth3s (sum)
- **dce_tcp.shutdowns**: total connection-oriented shutdowns (sum)
- **dce_tcp.rejects**: total connection-oriented rejects (sum)
- **dce_tcp.ms_rpc_http_pdus**: total connection-oriented MS requests to send RPC over HTTP (sum)
- **dce_tcp.other_requests**: total connection-oriented other requests (sum)
- **dce_tcp.other_responses**: total connection-oriented other responses (sum)
- **dce_tcp.request_fragments**: total connection-oriented request fragments (sum)
- **dce_tcp.response_fragments**: total connection-oriented response fragments (sum)
- **dce_tcp.client_max_fragment_size**: connection-oriented client maximum fragment size (sum)
- **dce_tcp.client_min_fragment_size**: connection-oriented client minimum fragment size (sum)
- **dce_tcp.client_segs_reassembled**: total connection-oriented client segments reassembled (sum)
- **dce_tcp.client_frags_reassembled**: total connection-oriented client fragments reassembled (sum)
- **dce_tcp.server_max_fragment_size**: connection-oriented server maximum fragment size (sum)
- **dce_tcp.server_min_fragment_size**: connection-oriented server minimum fragment size (sum)
- **dce_tcp.server_segs_reassembled**: total connection-oriented server segments reassembled (sum)
- **dce_tcp.server_frags_reassembled**: total connection-oriented server fragments reassembled (sum)
- **dce_tcp.tcp_sessions**: total tcp sessions (sum)
- **dce_tcp.tcp_expected_sessions**: total tcp dynamic endpoint expected sessions (sum)
- **dce_tcp.tcp_expected_realized**: total tcp dynamic endpoint expected realized sessions (sum)
- **dce_tcp.tcp_packets**: total tcp packets (sum)
- **dce_tcp.concurrent_sessions**: total concurrent sessions (now)
- **dce_tcp.max_concurrent_sessions**: maximum concurrent sessions (max)

5.13 dce_udp

Help: dce over udp inspection

Type: inspector (service)

Usage: inspect

Instance Type: multiton

Configuration:

- bool **dce_udp.limit_alerts** = true: limit DCE alert to at most one per signature per flow
- bool **dce_udp.disable_defrag** = false: disable DCE/RPC defragmentation
- int **dce_udp.max_frag_len** = 65535: maximum fragment size for defragmentation { 1514:65535 }

Rules:

- **133:40** (dce_udp) connection-less DCE/RPC - invalid major version
- **133:41** (dce_udp) connection-less DCE/RPC - invalid PDU type
- **133:42** (dce_udp) connection-less DCE/RPC - data length less than header size
- **133:43** (dce_udp) connection-less DCE/RPC - bad sequence number

Peg counts:

- **dce_udp.events**: total events (sum)
- **dce_udp.udp_sessions**: total udp sessions (sum)
- **dce_udp.udp_packets**: total udp packets (sum)
- **dce_udp.requests**: total connection-less requests (sum)
- **dce_udp.acks**: total connection-less acks (sum)
- **dce_udp.cancels**: total connection-less cancels (sum)
- **dce_udp.client_facks**: total connection-less client facks (sum)
- **dce_udp.ping**: total connection-less ping (sum)
- **dce_udp.responses**: total connection-less responses (sum)
- **dce_udp.rejects**: total connection-less rejects (sum)
- **dce_udp.cancel_acks**: total connection-less cancel acks (sum)
- **dce_udp.server_facks**: total connection-less server facks (sum)
- **dce_udp.faults**: total connection-less faults (sum)
- **dce_udp.no_calls**: total connection-less no calls (sum)
- **dce_udp.working**: total connection-less working (sum)
- **dce_udp.other_requests**: total connection-less other requests (sum)
- **dce_udp.other_responses**: total connection-less other responses (sum)
- **dce_udp.fragments**: total connection-less fragments (sum)
- **dce_udp.max_fragment_size**: connection-less maximum fragment size (sum)
- **dce_udp.frags_reassembled**: total connection-less fragments reassembled (sum)
- **dce_udp.max_seqnum**: max connection-less seqnum (sum)
- **dce_udp.concurrent_sessions**: total concurrent sessions (now)
- **dce_udp.max_concurrent_sessions**: maximum concurrent sessions (max)

5.14 dnp3

Help: dnp3 inspection

Type: inspector (service)

Usage: inspect

Instance Type: multiton

Configuration:

- bool **dnp3.check_crc** = false: validate checksums in DNP3 link layer frames

Rules:

- **145:1** (dnp3) DNP3 link-layer frame contains bad CRC
- **145:2** (dnp3) DNP3 link-layer frame is truncated or frame length is invalid
- **145:3** (dnp3) DNP3 transport-layer segment sequence number is incorrect
- **145:4** (dnp3) DNP3 transport-layer segment flag violation is detected
- **145:5** (dnp3) DNP3 link-layer frame uses a reserved address
- **145:6** (dnp3) DNP3 application-layer fragment uses a reserved function code

Peg counts:

- **dnp3.total_packets**: total packets (sum)
- **dnp3.udp_packets**: total udp packets (sum)
- **dnp3.tcp_pdus**: total tcp pdus (sum)
- **dnp3.dnp3_link_layer_frames**: total dnp3 link layer frames (sum)
- **dnp3.dnp3_application_pdus**: total dnp3 application pdus (sum)
- **dnp3.concurrent_sessions**: total concurrent dnp3 sessions (now)
- **dnp3.max_concurrent_sessions**: maximum concurrent dnp3 sessions (max)

5.15 dns

Help: dns inspection

Type: inspector (service)

Usage: inspect

Instance Type: multiton

Configuration:

- bool **dns.publish_response** = false: parse and publish dns responses

Rules:

- **131:1** (dns) obsolete DNS RR types
- **131:2** (dns) experimental DNS RR types

- **131:3** (dns) DNS client rdata txt overflow

Peg counts:

- **dns.packets**: total packets processed (sum)
- **dns.requests**: total dns requests (sum)
- **dns.responses**: total dns responses (sum)
- **dns.concurrent_sessions**: total concurrent dns sessions (now)
- **dns.max_concurrent_sessions**: maximum concurrent dns sessions (max)

5.16 domain_filter

Help: alert on configured HTTP domains

Type: inspector (passive)

Usage: inspect

Instance Type: singleton

Configuration:

- string **domain_filter.file**: file with list of domains identifying hosts to be filtered
- string **domain_filter.hosts**: list of domains identifying hosts to be filtered

Rules:

- **175:1** (domain_filter) configured domain detected

Peg counts:

- **domain_filter.checked**: domains checked (sum)
- **domain_filter.filtered**: domains filtered (sum)

5.17 dpx

Help: dynamic inspector example

Type: inspector (network)

Usage: inspect

Instance Type: singleton

Configuration:

- port **dpx.port**: port to check
- int **dpx.max** = 0: maximum payload before alert { 0:65535 }

Rules:

- **256:1** (dpx) too much data sent to port

Peg counts:

- **dpx.packets**: total packets (sum)

5.18 file_id

Help: configure file identification

Type: inspector (file)

Usage: global

Instance Type: global

Configuration:

- int **file_id.type_depth** = 1460: stop type ID at this point { 0:max53 }
- int **file_id.signature_depth** = 10485760: stop signature at this point { 0:max53 }
- int **file_id.block_timeout** = 86400: stop blocking after this many seconds { 0:max31 }
- int **file_id.lookup_timeout** = 2: give up on lookup after this many seconds { 0:max31 }
- bool **file_id.block_timeout_lookup** = false: block if lookup times out
- int **file_id.capture_memcap** = 100: memcap for file capture in megabytes { 0:max53 }
- int **file_id.capture_max_size** = 1048576: stop file capture beyond this point { 0:max53 }
- int **file_id.capture_min_size** = 0: stop file capture if file size less than this { 0:max53 }
- int **file_id.capture_block_size** = 32768: file capture block size in bytes { 8:max53 }
- int **file_id.max_files_cached** = 65536: maximal number of files cached in memory { 8:max53 }
- int **file_id.max_files_per_flow** = 128: maximal number of files able to be concurrently processed per flow { 1:max53 }
- int **file_id.show_data_depth** = 100: print this many octets { 0:max53 }
- string **file_id.rules_file**: name of file with IPS rules for file identification
- bool **file_id.trace_type** = false: enable runtime dump of type info
- bool **file_id.trace_signature** = false: enable runtime dump of signature info
- bool **file_id.trace_stream** = false: enable runtime dump of file data
- int **file_id.decompress_buffer_size** = 100000: file decompression buffer size { 1024:max31 }

Rules:

- **150:1** (file_id) file not processed due to per flow limit

Peg counts:

- **file_id.total_files**: number of files processed (sum)
- **file_id.total_file_data**: number of file data bytes processed (sum)
- **file_id.cache_failures**: number of file cache add failures (sum)
- **file_id.files_not_processed**: number of files not processed due to per-flow limit (sum)
- **file_id.max_concurrent_files**: maximum files processed concurrently on a flow (max)

5.19 file_log

Help: log file event to file.log

Type: inspector (passive)

Usage: inspect

Instance Type: singleton

Configuration:

- bool **file_log.log_pkt_time** = true: log the packet time when event generated
- bool **file_log.log_sys_time** = false: log the system time when event generated

Peg counts:

- **file_log.total_events**: total file events (sum)

5.20 ftp_client

Help: FTP client configuration module for use with ftp_server

Type: inspector (passive)

Usage: inspect

Instance Type: multiton

Configuration:

- bool **ftp_client.bounce** = false: check for bounces
- addr **`ftp_client.bounce_to[].address`** = *1.0.0.0/32*: allowed IP address in CIDR format
- port **`ftp_client.bounce_to[].port`** = 20: allowed port
- port **`ftp_client.bounce_to[].last_port`**: optional allowed range from port to last_port inclusive
- bool **ftp_client.ignore_telnet_erase_cmds** = false: ignore erase character and erase line commands when normalizing
- int **ftp_client.max_resp_len** = 4294967295: maximum FTP response accepted by client { 0:max32 }
- bool **ftp_client.telnet_cmds** = false: detect Telnet escape sequences on FTP control channel

5.21 ftp_data

Help: FTP data channel handler

Type: inspector (service)

Usage: inspect

Instance Type: multiton

Peg counts:

- **ftp_data.packets**: total packets (sum)

5.22 ftp_server

Help: main FTP module; ftp_client should also be configured

Type: inspector (service)

Usage: inspect

Instance Type: multiton

Configuration:

- string **ftp_server.chk_str_fmt**: check the formatting of the given commands
- string **ftp_server.data_chan_cmds**: check the formatting of the given commands
- string **ftp_server.data_rest_cmds**: check the formatting of the given commands
- string **ftp_server.data_xfer_cmds**: check the formatting of the given commands
- string **`ftp_server.directory_cmds[].dir_cmd`**: directory command
- int **`ftp_server.directory_cmds[].rsp_code`** = 200: expected successful response code for command { 200:max32 }
- string **ftp_server.file_put_cmds**: check the formatting of the given commands
- string **ftp_server.file_get_cmds**: check the formatting of the given commands
- string **ftp_server.encr_cmds**: check the formatting of the given commands
- string **ftp_server.login_cmds**: check the formatting of the given commands
- bool **ftp_server.check_encrypted** = false: check for end of encryption
- string **`ftp_server.cmd_validity[].command`**: command string
- string **`ftp_server.cmd_validity[].format`**: format specification
- int **`ftp_server.cmd_validity[].length`** = 0: specify non-default maximum for command { 0:max32 }
- int **ftp_server.def_max_param_len** = 100: default maximum length of commands handled by server; 0 is unlimited { 1:max32 }
- bool **ftp_server.encrypted_traffic** = false: check for encrypted Telnet and FTP
- string **ftp_server.ftp_cmds**: specify additional commands supported by server beyond RFC 959
- bool **ftp_server.ignore_data_chan** = false: do not inspect FTP data channels
- bool **ftp_server.ignore_telnet_erase_cmds** = false: ignore erase character and erase line commands when normalizing
- bool **ftp_server.print_cmds** = false: print command configurations on start up
- bool **ftp_server.telnet_cmds** = false: detect Telnet escape sequences of FTP control channel

Rules:

- **125:1** (ftp_server) TELNET cmd on FTP command channel
- **125:2** (ftp_server) invalid FTP command
- **125:3** (ftp_server) FTP command parameters were too long
- **125:4** (ftp_server) FTP command parameters were malformed
- **125:5** (ftp_server) FTP command parameters contained potential string format

- **125:6** (ftp_server) FTP response message was too long
- **125:7** (ftp_server) FTP traffic encrypted
- **125:8** (ftp_server) FTP bounce attempt
- **125:9** (ftp_server) evasive (incomplete) TELNET cmd on FTP command channel

Peg counts:

- **ftp_server.total_packets**: total packets (sum)
- **ftp_server.total_bytes**: total number of bytes processed (sum)
- **ftp_server.concurrent_sessions**: total concurrent FTP sessions (now)
- **ftp_server.max_concurrent_sessions**: maximum concurrent FTP sessions (max)
- **ftp_server.start_tls**: total STARTTLS events generated (sum)
- **ftp_server.ssl_search_abandoned**: total SSL search abandoned (sum)
- **ftp_server.ssl_srch_abandoned_early**: total SSL search abandoned too soon (sum)
- **ftp_server.pkt_segment_size_changed**: total number of FTP data packets with segment size change (sum)
- **ftp_server.flow_segment_size_changed**: total number of FTP sessions with segment size change (sum)

5.23 gtp_inspect

Help: gtp control channel inspection

Type: inspector (service)

Usage: inspect

Instance Type: multiton

Configuration:

- int **`gtp_inspect[].version`** = 2: GTP version { 0:2 }
- int **`gtp_inspect[].messages[].type`** = 0: message type code { 0:255 }
- string **`gtp_inspect[].messages[].name`**: message name
- int **`gtp_inspect[].infos[].type`** = 0: information element type code { 0:255 }
- string **`gtp_inspect[].infos[].name`**: information element name
- int **`gtp_inspect[].infos[].length`** = 0: information element type code { 0:255 }

Rules:

- **143:1** (gtp_inspect) message length is invalid
- **143:2** (gtp_inspect) information element length is invalid
- **143:3** (gtp_inspect) information elements are out of order
- **143:4** (gtp_inspect) TEID is missing

Peg counts:

- **gtp_inspect.sessions**: total sessions processed (sum)
- **gtp_inspect.concurrent_sessions**: total concurrent gtp sessions (now)
- **gtp_inspect.max_concurrent_sessions**: maximum concurrent gtp sessions (max)
- **gtp_inspect.events**: requests (sum)
- **gtp_inspect.unknown_types**: unknown message types (sum)
- **gtp_inspect.unknown_infos**: unknown information elements (sum)

5.24 http2_inspect

Help: HTTP/2 inspector

Type: inspector (service)

Usage: inspect

Instance Type: multiton

Configuration:

- int **http2_inspect.concurrent_streams_limit** = 100: Maximum number of concurrent streams allowed in a single HTTP/2 flow { 100:1000 }

Rules:

- **121:1** (http2_inspect) invalid flag set on HTTP/2 frame
- **121:2** (http2_inspect) HPACK integer value has leading zeros
- **121:3** (http2_inspect) HTTP/2 stream initiated with invalid stream id
- **121:4** (http2_inspect) missing HTTP/2 continuation frame
- **121:5** (http2_inspect) unexpected HTTP/2 continuation frame
- **121:6** (http2_inspect) HTTP/2 headers HPACK decoding error
- **121:7** (http2_inspect) HTTP/2 connection preface does not match
- **121:8** (http2_inspect) HTTP/2 request missing required header field
- **121:9** (http2_inspect) HTTP/2 response has no status code
- **121:10** (http2_inspect) HTTP/2 CONNECT request with scheme or path
- **121:11** (http2_inspect) error in HTTP/2 settings frame
- **121:12** (http2_inspect) unknown parameter in HTTP/2 settings frame
- **121:13** (http2_inspect) invalid HTTP/2 frame sequence
- **121:14** (http2_inspect) HTTP/2 dynamic table has more than 512 entries
- **121:15** (http2_inspect) HTTP/2 push promise frame with promised stream ID already in use
- **121:16** (http2_inspect) HTTP/2 padding length is bigger than frame data size
- **121:17** (http2_inspect) HTTP/2 pseudo-header after regular header
- **121:18** (http2_inspect) HTTP/2 pseudo-header in trailers
- **121:19** (http2_inspect) invalid HTTP/2 pseudo-header

- **121:20** (http2_inspect) HTTP/2 trailers without END_STREAM bit
- **121:21** (http2_inspect) HTTP/2 push promise frame sent when prohibited by receiver
- **121:22** (http2_inspect) padding flag set on HTTP/2 frame with zero length
- **121:23** (http2_inspect) HTTP/2 push promise frame in client-to-server direction
- **121:24** (http2_inspect) invalid HTTP/2 push promise frame
- **121:25** (http2_inspect) HTTP/2 push promise frame sent at invalid time
- **121:26** (http2_inspect) invalid parameter value sent in HTTP/2 settings frame
- **121:27** (http2_inspect) excessive concurrent HTTP/2 streams
- **121:28** (http2_inspect) invalid HTTP/2 rst stream frame
- **121:29** (http2_inspect) HTTP/2 rst stream frame sent at invalid time
- **121:30** (http2_inspect) uppercase HTTP/2 header field name
- **121:31** (http2_inspect) invalid HTTP/2 window update frame
- **121:32** (http2_inspect) HTTP/2 window update frame with zero increment
- **121:33** (http2_inspect) HTTP/2 request without a method
- **121:34** (http2_inspect) HTTP/2 HPACK table size update not at the start of a header block
- **121:35** (http2_inspect) More than two HTTP/2 HPACK table size updates in a single header block
- **121:36** (http2_inspect) HTTP/2 HPACK table size update exceeds max value set by decoder in SETTINGS frame
- **121:37** (http2_inspect) Nonempty HTTP/2 Data frame where message body not expected
- **121:38** (http2_inspect) HTTP/2 non-Data frame longer than 63780 bytes
- **121:39** (http2_inspect) not HTTP/2 traffic or unrecoverable HTTP/2 protocol error
- **121:40** (http2_inspect) invalid HTTP/2 PRIORITY frame
- **121:41** (http2_inspect) invalid HTTP/2 GOAWAY frame
- **121:42** (http2_inspect) too many unacknowledged settings
- **121:43** (http2_inspect) setting acknowledgment without actual settings

Peg counts:

- **http2_inspect.flows**: HTTP/2 connections inspected (sum)
- **http2_inspect.concurrent_sessions**: total concurrent HTTP/2 sessions (now)
- **http2_inspect.max_concurrent_sessions**: maximum concurrent HTTP/2 sessions (max)
- **http2_inspect.max_table_entries**: maximum entries in an HTTP/2 dynamic table (max)
- **http2_inspect.max_concurrent_files**: maximum concurrent file transfers per HTTP/2 connection (max)
- **http2_inspect.total_bytes**: total HTTP/2 data bytes inspected (sum)
- **http2_inspect.max_concurrent_streams**: maximum concurrent streams per HTTP/2 connection (max)
- **http2_inspect.flows_over_stream_limit**: HTTP/2 flows exceeding 100 concurrent streams (sum)

5.25 http_inspect

Help: HTTP inspector

Type: inspector (service)

Usage: inspect

Instance Type: multiton

Configuration:

- int **http_inspect.request_depth** = -1: maximum request message body bytes to examine (-1 no limit) { -1:max53 }
- int **http_inspect.response_depth** = -1: maximum response message body bytes to examine (-1 no limit) { -1:max53 }
- bool **http_inspect.unzip** = true: decompress gzip and deflate message bodies
- int **http_inspect.maximum_host_length** = -1: maximum allowed length for Host header value (-1 no limit) { -1:max53 }
- int **http_inspect.maximum_chunk_length** = 4294967295: maximum allowed length for a message body chunk { 0:4294967295 }
- int **http_inspect.maximum_header_length** = 4096: alert when the length of a header exceeds this value { 0:65535 }
- int **http_inspect.maximum_headers** = 200: alert when the number of headers in a message exceeds this value { 0:65535 }
- int **http_inspect.maximum_pipelined_requests** = 99: alert when the number of pipelined requests exceeds this value { 0:99 }
- bool **http_inspect.normalize_utf** = true: normalize charset utf encodings in response bodies
- bool **http_inspect.decompress_pdf** = false: decompress pdf files in response bodies
- bool **http_inspect.decompress_swf** = false: decompress swf files in response bodies
- bool **http_inspect.decompress_zip** = false: decompress zip files in response bodies
- bool **http_inspect.decompress_vba** = false: decompress MS Office Visual Basic for Applications macro files in response bodies
- int **http_inspect.max_mime_attach** = 5: maximum number of mime attachments that will be inspected in a section of a request message { 1:65535 }
- bool **http_inspect.script_detection** = false: inspect JavaScript immediately upon script end
- bool **http_inspect.normalize_javascript** = false: use legacy normalizer to normalize JavaScript in response bodies
- int **http_inspect.max_javascript_whitespaces** = 200: maximum consecutive whitespaces allowed within the JavaScript obfuscated data { 1:65535 }
- bit_list **http_inspect.bad_characters**: alert when any of specified bytes are present in URI after percent decoding { 255 }
- string **http_inspect.ignore_unreserved**: do not alert when the specified unreserved characters are percent-encoded in a URI.Unreserved characters are 0-9, a-z, A-Z, period, underscore, tilde, and minus. { (optional) }
- bool **http_inspect.percent_u** = false: normalize %uNNNN and %UNNNN encodings
- bool **http_inspect.utf8** = true: normalize 2-byte and 3-byte UTF-8 characters to a single byte
- bool **http_inspect.utf8_bare_byte** = false: when doing UTF-8 character normalization include bytes that were not percent encoded
- bool **http_inspect.iis_unicode** = false: use IIS unicode code point mapping to normalize characters
- string **http_inspect.iis_unicode_map_file**: file containing code points for IIS unicode. { (optional) }

- int **http_inspect.iis_unicode_code_page** = 1252: code page to use from the IIS unicode map file { 0:65535 }
- bool **http_inspect.iis_double_decode** = true: perform double decoding of percent encodings to normalize characters
- int **http_inspect.oversize_dir_length** = 300: maximum length for URL directory { 1:65535 }
- bool **http_inspect.backslash_to_slash** = true: replace \ with / when normalizing URIs
- bool **http_inspect.plus_to_space** = true: replace + with <sp> when normalizing URIs
- bool **http_inspect.simplify_path** = true: reduce URI directory path to simplest form
- string **http_inspect.xff_headers** = *x-forwarded-for true-client-ip*: specifies the xff type headers to parse and consider in the same order of preference as defined
- bool **http_inspect.request_body_app_detection** = true: make HTTP request message bodies available for application detection (AppId) and other inspectors
- string **http_inspect.allowed_methods**: list of allowed methods
- string **http_inspect.disallowed_methods**: list of disallowed methods

Rules:

- **119:1** (http_inspect) URI has percent-encoding of an unreserved character
- **119:2** (http_inspect) URI contains double-encoded hexadecimal characters
- **119:3** (http_inspect) URI has non-standard %u-style Unicode encoding
- **119:4** (http_inspect) URI has Unicode encodings containing bytes that were not percent-encoded
- **119:6** (http_inspect) URI has two-byte or three-byte UTF-8 encoding
- **119:7** (http_inspect) URI has unicode map code point encoding
- **119:8** (http_inspect) URI path contains consecutive slash characters
- **119:9** (http_inspect) backslash character appears in the path portion of a URI
- **119:10** (http_inspect) URI path contains /./ pattern repeating the current directory
- **119:11** (http_inspect) URI path contains /../ pattern moving up a directory
- **119:12** (http_inspect) Tab character in HTTP start line
- **119:13** (http_inspect) HTTP start line or header line terminated by LF without a CR
- **119:14** (http_inspect) Normalized URI includes character from bad_characters list
- **119:15** (http_inspect) URI path contains a segment that is longer than the oversize_dir_length parameter
- **119:16** (http_inspect) chunk length exceeds configured maximum_chunk_length
- **119:18** (http_inspect) URI path includes /../ that goes above the root directory
- **119:19** (http_inspect) HTTP header line exceeds maximum_header_length option bytes
- **119:20** (http_inspect) HTTP message has more than maximum_headers option header fields
- **119:21** (http_inspect) HTTP message has more than one Content-Length header value
- **119:24** (http_inspect) Host header field appears more than once or has multiple values
- **119:25** (http_inspect) length of HTTP Host header field value exceeds maximum_host_length option
- **119:28** (http_inspect) HTTP POST or PUT request without content-length or chunks

- **119:31** (http_inspect) HTTP request method is not known to Snort
- **119:32** (http_inspect) HTTP request uses primitive HTTP format known as HTTP/0.9
- **119:33** (http_inspect) HTTP request URI has space character that is not percent-encoded
- **119:34** (http_inspect) HTTP connection has more than maximum_pipelined_requests simultaneous pipelined requests that have not been answered
- **119:102** (http_inspect) invalid status code in HTTP response
- **119:104** (http_inspect) HTTP response has UTF character set that failed to normalize
- **119:105** (http_inspect) HTTP response has UTF-7 character set
- **119:109** (http_inspect) more than one level of JavaScript obfuscation
- **119:110** (http_inspect) consecutive JavaScript whitespaces exceed maximum allowed
- **119:111** (http_inspect) multiple encodings within JavaScript obfuscated data
- **119:112** (http_inspect) SWF file zlib decompression failure
- **119:113** (http_inspect) SWF file LZMA decompression failure
- **119:114** (http_inspect) PDF file deflate decompression failure
- **119:115** (http_inspect) PDF file unsupported compression type
- **119:116** (http_inspect) PDF file with more than one compression applied
- **119:117** (http_inspect) PDF file parse failure
- **119:201** (http_inspect) not HTTP traffic or unrecoverable HTTP protocol error
- **119:202** (http_inspect) chunk length has excessive leading zeros
- **119:203** (http_inspect) white space before or between HTTP messages
- **119:204** (http_inspect) request message without URI
- **119:205** (http_inspect) control character in HTTP response reason phrase
- **119:206** (http_inspect) illegal extra whitespace in start line
- **119:207** (http_inspect) corrupted HTTP version
- **119:209** (http_inspect) format error in HTTP header
- **119:210** (http_inspect) chunk header options present
- **119:211** (http_inspect) URI badly formatted
- **119:212** (http_inspect) unrecognized type of percent encoding in URI
- **119:213** (http_inspect) HTTP chunk misformatted
- **119:214** (http_inspect) white space adjacent to chunk length
- **119:215** (http_inspect) white space within header name
- **119:216** (http_inspect) excessive gzip compression
- **119:217** (http_inspect) gzip decompression failed
- **119:218** (http_inspect) HTTP 0.9 requested followed by another request
- **119:219** (http_inspect) HTTP 0.9 request following a normal request

- **119:220** (http_inspect) message has both Content-Length and Transfer-Encoding
- **119:221** (http_inspect) status code implying no body combined with Transfer-Encoding or nonzero Content-Length
- **119:222** (http_inspect) Transfer-Encoding not ending with chunked
- **119:223** (http_inspect) Transfer-Encoding with encodings before chunked
- **119:225** (http_inspect) unsupported Content-Encoding used
- **119:226** (http_inspect) unknown Content-Encoding used
- **119:227** (http_inspect) multiple Content-Encodings applied
- **119:228** (http_inspect) server response before client request
- **119:229** (http_inspect) PDF/SWF/ZIP decompression of server response too big
- **119:230** (http_inspect) nonprinting character in HTTP message header name
- **119:231** (http_inspect) bad Content-Length value in HTTP header
- **119:232** (http_inspect) HTTP header line wrapped
- **119:233** (http_inspect) HTTP header line terminated by CR without a LF
- **119:234** (http_inspect) chunk terminated by nonstandard separator
- **119:235** (http_inspect) chunk length terminated by LF without CR
- **119:236** (http_inspect) more than one response with 100 status code
- **119:237** (http_inspect) 100 status code not in response to Expect header
- **119:238** (http_inspect) 1XX status code other than 100 or 101
- **119:239** (http_inspect) Expect header sent without a message body
- **119:240** (http_inspect) HTTP 1.0 message with Transfer-Encoding header
- **119:241** (http_inspect) Content-Transfer-Encoding used as HTTP header
- **119:242** (http_inspect) illegal field in chunked message trailers
- **119:243** (http_inspect) header field inappropriately appears twice or has two values
- **119:244** (http_inspect) invalid value chunked in Content-Encoding header
- **119:245** (http_inspect) 206 response sent to a request without a Range header
- **119:246** (http_inspect) *HTTP* in version field not all upper case
- **119:247** (http_inspect) white space embedded in critical header value
- **119:248** (http_inspect) gzip compressed data followed by unexpected non-gzip data
- **119:249** (http_inspect) excessive HTTP parameter key repeats
- **119:250** (http_inspect) HTTP/2 Transfer-Encoding header other than identity
- **119:251** (http_inspect) HTTP/2 message body overruns Content-Length header value
- **119:252** (http_inspect) HTTP/2 message body smaller than Content-Length header value
- **119:253** (http_inspect) HTTP CONNECT request with a message body
- **119:254** (http_inspect) HTTP client-to-server traffic after CONNECT request but before CONNECT response
- **119:255** (http_inspect) HTTP CONNECT 2XX response with Content-Length header

- **119:256** (http_inspect) HTTP CONNECT 2XX response with Transfer-Encoding header
- **119:257** (http_inspect) HTTP CONNECT response with 1XX status code
- **119:258** (http_inspect) HTTP CONNECT response before request message completed
- **119:259** (http_inspect) malformed HTTP Content-Disposition filename parameter
- **119:260** (http_inspect) HTTP Content-Length message body was truncated
- **119:261** (http_inspect) HTTP chunked message body was truncated
- **119:262** (http_inspect) HTTP URI scheme longer than 10 characters
- **119:263** (http_inspect) HTTP/1 client requested HTTP/2 upgrade
- **119:264** (http_inspect) HTTP/1 server granted HTTP/2 upgrade
- **119:268** (http_inspect) JavaScript code under the external script tags
- **119:269** (http_inspect) script opening tag in a short form
- **119:272** (http_inspect) Consecutive commas in HTTP Accept-Encoding header
- **119:275** (http_inspect) HTTP/1 version other than 1.0 or 1.1
- **119:276** (http_inspect) HTTP version in start line is 0
- **119:277** (http_inspect) HTTP version in start line is higher than 1
- **119:278** (http_inspect) HTTP gzip body with the FEXTRA flag set
- **119:279** (http_inspect) invalid status line
- **119:280** (http_inspect) HTTP message headers longer than 63780 bytes
- **119:281** (http_inspect) invalid request line
- **119:282** (http_inspect) too many white space characters when start line is expected
- **119:283** (http_inspect) HTTP message status line longer than 63780 bytes
- **119:284** (http_inspect) partial start line
- **119:285** (http_inspect) HTTP message request line longer than 63780 bytes
- **119:286** (http_inspect) HTTP/2 preface received instead of an HTTP/1 method
- **119:287** (http_inspect) HTTP request method is not on allowed methods list or is on disallowed methods list
- **119:288** (http_inspect) HTTP gzip body with reserved flag set

Peg counts:

- **http_inspect.flows**: HTTP connections inspected (sum)
- **http_inspect.scans**: TCP segments scanned looking for HTTP messages (sum)
- **http_inspect.reassembles**: TCP segments combined into HTTP messages (sum)
- **http_inspect.inspections**: total message sections inspected (sum)
- **http_inspect.requests**: HTTP request messages inspected (sum)
- **http_inspect.responses**: HTTP response messages inspected (sum)
- **http_inspect.get_requests**: GET requests inspected (sum)

- **http_inspect.head_requests**: HEAD requests inspected (sum)
- **http_inspect.post_requests**: POST requests inspected (sum)
- **http_inspect.put_requests**: PUT requests inspected (sum)
- **http_inspect.delete_requests**: DELETE requests inspected (sum)
- **http_inspect.connect_requests**: CONNECT requests inspected (sum)
- **http_inspect.options_requests**: OPTIONS requests inspected (sum)
- **http_inspect.trace_requests**: TRACE requests inspected (sum)
- **http_inspect.other_requests**: other request methods inspected (sum)
- **http_inspect.request_bodies**: POST, PUT, and other requests with message bodies (sum)
- **http_inspect.chunked**: chunked message bodies (sum)
- **http_inspect.uri_normalizations**: URIs needing to be normalization (sum)
- **http_inspect.uri_path**: URIs with path problems (sum)
- **http_inspect.uri_coding**: URIs with character coding problems (sum)
- **http_inspect.concurrent_sessions**: total concurrent http sessions (now)
- **http_inspect.max_concurrent_sessions**: maximum concurrent http sessions (max)
- **http_inspect.script_detections**: early inspections of scripts in HTTP responses (sum)
- **http_inspect.partial_inspections**: early inspections done for script detection (sum)
- **http_inspect.excess_parameters**: repeat parameters exceeding max (sum)
- **http_inspect.parameters**: HTTP parameters inspected (sum)
- **http_inspect.connect_tunnel_cutovers**: CONNECT tunnel flow cutovers to wizard (sum)
- **http_inspect.ssl_srch_abandoned_early**: total SSL search abandoned too soon (sum)
- **http_inspect.pipelined_flows**: total HTTP connections containing pipelined requests (sum)
- **http_inspect.pipelined_requests**: total requests placed in a pipeline (sum)
- **http_inspect.total_bytes**: total HTTP data bytes inspected (sum)
- **http_inspect.js_inline_scripts**: total number of inline JavaScripts processed (sum)
- **http_inspect.js_external_scripts**: total number of external JavaScripts processed (sum)
- **http_inspect.js_pdf_scripts**: total number of PDF files processed (sum)
- **http_inspect.skip_mime_attach**: total number of HTTP requests with too many MIME attachments to inspect (sum)

5.26 iec104

Help: iec104 inspection

Type: inspector (service)

Usage: inspect

Instance Type: multiton

Rules:

- **151:1** (iec104) Length in IEC104 APCI header does not match the length needed for the given IEC104 ASDU type id
- **151:2** (iec104) IEC104 Start byte does not match 0x68
- **151:3** (iec104) Reserved IEC104 ASDU type id in use
- **151:4** (iec104) IEC104 APCI U Reserved field contains a non-default value
- **151:5** (iec104) IEC104 APCI U message type was set to an invalid value
- **151:6** (iec104) IEC104 APCI S Reserved field contains a non-default value
- **151:7** (iec104) IEC104 APCI I number of elements set to zero
- **151:8** (iec104) IEC104 APCI I SQ bit set on an ASDU that does not support the feature
- **151:9** (iec104) IEC104 APCI I number of elements set to greater than one on an ASDU that does not support the feature
- **151:10** (iec104) IEC104 APCI I Cause of Initialization set to a reserved value
- **151:11** (iec104) IEC104 APCI I Qualifier of Interrogation Command set to a reserved value
- **151:12** (iec104) IEC104 APCI I Qualifier of Counter Interrogation Command request parameter set to a reserved value
- **151:13** (iec104) IEC104 APCI I Qualifier of Parameter of Measured Values kind of parameter set to a reserved value
- **151:14** (iec104) IEC104 APCI I Qualifier of Parameter of Measured Values local parameter change set to a technically valid but unused value
- **151:15** (iec104) IEC104 APCI I Qualifier of Parameter of Measured Values parameter option set to a technically valid but unused value
- **151:16** (iec104) IEC104 APCI I Qualifier of Parameter Activation set to a reserved value
- **151:17** (iec104) IEC104 APCI I Qualifier of Command set to a reserved value
- **151:18** (iec104) IEC104 APCI I Qualifier of Reset Process set to a reserved value
- **151:19** (iec104) IEC104 APCI I File Ready Qualifier set to a reserved value
- **151:20** (iec104) IEC104 APCI I Section Ready Qualifier set to a reserved value
- **151:21** (iec104) IEC104 APCI I Select and Call Qualifier set to a reserved value
- **151:22** (iec104) IEC104 APCI I Last Section or Segment Qualifier set to a reserved value
- **151:23** (iec104) IEC104 APCI I Acknowledge File or Section Qualifier set to a reserved value
- **151:24** (iec104) IEC104 APCI I Structure Qualifier set on a message where it should have no effect
- **151:25** (iec104) IEC104 APCI I Single Point Information Reserved field contains a non-default value
- **151:26** (iec104) IEC104 APCI I Double Point Information Reserved field contains a non-default value
- **151:27** (iec104) IEC104 APCI I Cause of Transmission set to a reserved value

- **151:28** (iec104) IEC104 APCI I Cause of Transmission set to a value not allowed for the ASDU
- **151:29** (iec104) IEC104 APCI I invalid two octet common address value detected
- **151:30** (iec104) IEC104 APCI I Quality Descriptor Structure Reserved field contains a non-default value
- **151:31** (iec104) IEC104 APCI I Quality Descriptor for Events of Protection Equipment Structure Reserved field contains a non-default value
- **151:32** (iec104) IEC104 APCI I IEEE STD 754 value results in NaN
- **151:33** (iec104) IEC104 APCI I IEEE STD 754 value results in infinity
- **151:34** (iec104) IEC104 APCI I Single Event of Protection Equipment Structure Reserved field contains a non-default value
- **151:35** (iec104) IEC104 APCI I Start Event of Protection Equipment Structure Reserved field contains a non-default value
- **151:36** (iec104) IEC104 APCI I Output Circuit Information Structure Reserved field contains a non-default value
- **151:37** (iec104) IEC104 APCI I Abnormal Fixed Test Bit Pattern detected
- **151:38** (iec104) IEC104 APCI I Single Command Structure Reserved field contains a non-default value
- **151:39** (iec104) IEC104 APCI I Double Command Structure contains an invalid value
- **151:40** (iec104) IEC104 APCI I Regulating Step Command Structure Reserved field contains a non-default value
- **151:41** (iec104) IEC104 APCI I Time2a Millisecond set outside of the allowable range
- **151:42** (iec104) IEC104 APCI I Time2a Minute set outside of the allowable range
- **151:43** (iec104) IEC104 APCI I Time2a Minute Reserved field contains a non-default value
- **151:44** (iec104) IEC104 APCI I Time2a Hours set outside of the allowable range
- **151:45** (iec104) IEC104 APCI I Time2a Hours Reserved field contains a non-default value
- **151:46** (iec104) IEC104 APCI I Time2a Day of Month set outside of the allowable range
- **151:47** (iec104) IEC104 APCI I Time2a Month set outside of the allowable range
- **151:48** (iec104) IEC104 APCI I Time2a Month Reserved field contains a non-default value
- **151:49** (iec104) IEC104 APCI I Time2a Year set outside of the allowable range
- **151:50** (iec104) IEC104 APCI I Time2a Year Reserved field contains a non-default value
- **151:51** (iec104) IEC104 APCI I a null Length of Segment value has been detected
- **151:52** (iec104) IEC104 APCI I an invalid Length of Segment value has been detected
- **151:53** (iec104) IEC104 APCI I Status of File set to a reserved value
- **151:54** (iec104) IEC104 APCI I Qualifier of Set Point Command ql field set to a reserved value

Peg counts:

- **iec104.sessions**: total sessions processed (sum)
- **iec104.frames**: total IEC104 messages (sum)
- **iec104.concurrent_sessions**: total concurrent IEC104 sessions (now)
- **iec104.max_concurrent_sessions**: maximum concurrent IEC104 sessions (max)

5.27 imap

Help: imap inspection

Type: inspector (service)

Usage: inspect

Instance Type: multiton

Configuration:

- int **imap.b64_decode_depth** = -1: base64 decoding depth (-1 no limit) { -1:65535 }
- int **imap.bitenc_decode_depth** = -1: non-Encoded MIME attachment extraction depth (-1 no limit) { -1:65535 }
- bool **imap.decompress_pdf** = false: decompress pdf files in MIME attachments
- bool **imap.decompress_swf** = false: decompress swf files in MIME attachments
- bool **imap.decompress_zip** = false: decompress zip files in MIME attachments
- bool **imap.decompress_vba** = false: decompress MS Office Visual Basic for Applications macro files in MIME attachments
- int **imap.qp_decode_depth** = -1: quoted Printable decoding depth (-1 no limit) { -1:65535 }
- int **imap.uu_decode_depth** = -1: Unix-to-Unix decoding depth (-1 no limit) { -1:65535 }

Rules:

- **141:1** (imap) unknown IMAP3 command
- **141:2** (imap) unknown IMAP3 response
- **141:4** (imap) base64 decoding failed
- **141:5** (imap) quoted-printable decoding failed
- **141:7** (imap) Unix-to-Unix decoding failed
- **141:8** (imap) file decompression failed

Peg counts:

- **imap.packets**: total packets processed (sum)
- **imap.sessions**: total imap sessions (sum)
- **imap.concurrent_sessions**: total concurrent imap sessions (now)
- **imap.max_concurrent_sessions**: maximum concurrent imap sessions (max)
- **imap.start_tls**: total STARTTLS events generated (sum)
- **imap.ssl_search_abandoned**: total SSL search abandoned (sum)
- **imap.ssl_srch_abandoned_early**: total SSL search abandoned too soon (sum)
- **imap.b64_attachments**: total base64 attachments decoded (sum)
- **imap.b64_decoded_bytes**: total base64 decoded bytes (sum)
- **imap.qp_attachments**: total quoted-printable attachments decoded (sum)
- **imap.qp_decoded_bytes**: total quoted-printable decoded bytes (sum)
- **imap.uu_attachments**: total uu attachments decoded (sum)
- **imap.uu_decoded_bytes**: total uu decoded bytes (sum)
- **imap.non_encoded_attachments**: total non-encoded attachments extracted (sum)
- **imap.non_encoded_bytes**: total non-encoded extracted bytes (sum)
- **imap.js_pdf_scripts**: total number of PDF files processed (sum)

5.28 mem_test

Help: for testing memory management

Type: inspector (service)

Usage: inspect

Instance Type: singleton

Peg counts:

- **mem_test.packets**: total packets (sum)

5.29 mms

Help: mms inspection

Type: inspector (service)

Usage: inspect

Instance Type: multiton

Rules:

no match

Peg counts:

- **mms.sessions**: total sessions processed (sum)
- **mms.frames**: total MMS messages (sum)
- **mms.concurrent_sessions**: total concurrent MMS sessions (now)
- **mms.max_concurrent_sessions**: maximum concurrent MMS sessions (max)

5.30 modbus

Help: modbus inspection

Type: inspector (service)

Usage: inspect

Instance Type: multiton

Rules:

- **144:1** (modbus) length in Modbus MBAP header does not match the length needed for the given function
- **144:2** (modbus) Modbus protocol ID is non-zero
- **144:3** (modbus) reserved Modbus function code in use

Peg counts:

- **modbus.sessions**: total sessions processed (sum)
- **modbus.frames**: total Modbus messages (sum)
- **modbus.concurrent_sessions**: total concurrent modbus sessions (now)
- **modbus.max_concurrent_sessions**: maximum concurrent modbus sessions (max)

5.31 netflow

Help: netflow inspection

Type: inspector (service)

Usage: inspect

Instance Type: multiton

Configuration:

- string **netflow.dump_file**: file name to dump netflow cache on shutdown; won't dump by default
- int **netflow.update_timeout** = 3600: the interval at which the system updates host cache information { 0:max32 }
- addr **`netflow.rules[].device_ip`**: restrict the NetFlow devices from which Snort will analyze packets
- bool **`netflow.rules[].exclude`** = false: exclude the NetFlow records that match this rule
- string **`netflow.rules[].zones`**: generate events only for NetFlow packets that originate from these zones
- string **`netflow.rules[].networks`**: generate events for NetFlow records that contain an initiator or responder IP from these networks
- bool **`netflow.rules[].create_host`** = false: generate a new host event
- bool **`netflow.rules[].create_service`** = false: generate a new or changed service event
- int **netflow.flow_memcap** = 0: maximum memory for flow record cache in bytes, 0 = unlimited { 0:maxSZ }
- int **netflow.template_memcap** = 0: maximum memory for template cache in bytes, 0 = unlimited { 0:maxSZ }
- string **netflow.netflow_service_id_path**: path to file containing service IDs for NetFlow

Peg counts:

- **netflow.cache_adds**: netflow cache added new entry (sum)
- **netflow.cache_hits**: netflow cache found existing entry (sum)
- **netflow.cache_misses**: netflow cache did not find entry (sum)
- **netflow.cache_replaces**: netflow cache found entry and replaced its value (sum)
- **netflow.cache_max**: netflow cache's maximum byte usage (sum)
- **netflow.cache_prunes**: netflow cache pruned entry to make space for new entry (sum)
- **netflow.cache_removes**: netflow cache removed existing entry (sum)
- **netflow.invalid_netflow_record**: count of invalid netflow records (sum)
- **netflow.packets**: total packets processed (sum)
- **netflow.records**: total records found in netflow data (sum)
- **netflow.unique_flows**: count of unique netflow flows (sum)
- **netflow.v9_missing_template**: count of data records that are missing templates (sum)
- **netflow.v9_options_template**: count of options template flowset (sum)
- **netflow.v9_templates**: count of total version 9 templates (sum)
- **netflow.version_5**: count of netflow version 5 packets received (sum)
- **netflow.version_9**: count of netflow version 9 packets received (sum)
- **netflow.netflow_cache_bytes_in_use**: number of bytes used in netflow cache (now)
- **netflow.template_cache_bytes_in_use**: number of bytes used in template cache (now)

5.32 normalizer

Help: packet scrubbing for inline mode

Type: inspector (packet)

Usage: context

Instance Type: network

Configuration:

- bool **normalizer.ip4.base** = false: clear options
- bool **normalizer.ip4.df** = false: clear don't frag flag
- bool **normalizer.ip4.rf** = false: clear reserved flag
- bool **normalizer.ip4.tos** = false: clear tos / differentiated services byte
- bool **normalizer.ip4.trim** = false: truncate excess payload beyond datagram length
- bool **normalizer.tcp.base** = false: clear reserved bits and option padding and fix urgent pointer / flags issues
- bool **normalizer.tcp.block** = false: allow packet drops during TCP normalization
- bool **normalizer.tcp.urp** = false: adjust urgent pointer if beyond segment length
- bool **normalizer.tcp.ips** = true: ensure consistency in retransmitted data
- select **normalizer.tcp.ecn** = *off*: clear ecn for all packets | sessions w/o ecn setup { off | packet | stream }
- bool **normalizer.tcp.pad** = false: clear any option padding bytes
- bool **normalizer.tcp.trim_syn** = false: remove data on SYN
- bool **normalizer.tcp.trim_rst** = false: remove any data from RST packet
- bool **normalizer.tcp.trim_win** = false: trim data to window
- bool **normalizer.tcp.trim_mss** = false: trim data to MSS
- bool **normalizer.tcp.opts** = false: clear all options except mss, wscale, timestamp, and any explicitly allowed
- bool **normalizer.tcp.req_urg** = false: clear the urgent pointer if the urgent flag is not set
- bool **normalizer.tcp.req_pay** = false: clear the urgent pointer and the urgent flag if there is no payload
- bool **normalizer.tcp.rsv** = false: clear the reserved bits in the TCP header
- bool **normalizer.tcp.req_urp** = false: clear the urgent flag if the urgent pointer is not set
- multi **normalizer.tcp.allow_names**: don't clear given option names { sack | echo | partial_order | conn_count | alt_checksum | md5 }
- string **normalizer.tcp.allow_codes**: don't clear given option codes
- bool **normalizer.ip6** = false: clear reserved flag
- bool **normalizer.icmp4** = false: clear reserved flag
- bool **normalizer.icmp6** = false: clear reserved flag

Peg counts:

- **normalizer.test_ip4_trim**: test eth packets trimmed to datagram size (sum)

- **normalizer.ip4_trim**: eth packets trimmed to datagram size (sum)
- **normalizer.test_ip4_tos**: test type of service normalizations (sum)
- **normalizer.ip4_tos**: type of service normalizations (sum)
- **normalizer.test_ip4_df**: test don't frag bit normalizations (sum)
- **normalizer.ip4_df**: don't frag bit normalizations (sum)
- **normalizer.test_ip4_rf**: test reserved flag bit clears (sum)
- **normalizer.ip4_rf**: reserved flag bit clears (sum)
- **normalizer.test_ip4_ttl**: test time-to-live normalizations (sum)
- **normalizer.ip4_ttl**: time-to-live normalizations (sum)
- **normalizer.test_ip4_opts**: test ip4 options cleared (sum)
- **normalizer.ip4_opts**: ip4 options cleared (sum)
- **normalizer.test_icmp4_echo**: test icmp4 ping normalizations (sum)
- **normalizer.icmp4_echo**: icmp4 ping normalizations (sum)
- **normalizer.test_ip6_hops**: test ip6 hop limit normalizations (sum)
- **normalizer.ip6_hops**: ip6 hop limit normalizations (sum)
- **normalizer.test_ip6_options**: test ip6 options cleared (sum)
- **normalizer.ip6_options**: ip6 options cleared (sum)
- **normalizer.test_icmp6_echo**: test icmp6 echo normalizations (sum)
- **normalizer.icmp6_echo**: icmp6 echo normalizations (sum)
- **normalizer.test_tcp_syn_options**: test SYN only options cleared from non-SYN packets (sum)
- **normalizer.tcp_syn_options**: SYN only options cleared from non-SYN packets (sum)
- **normalizer.test_tcp_options**: test packets with options cleared (sum)
- **normalizer.tcp_options**: packets with options cleared (sum)
- **normalizer.test_tcp_padding**: test packets with padding cleared (sum)
- **normalizer.tcp_padding**: packets with padding cleared (sum)
- **normalizer.test_tcp_reserved**: test packets with reserved bits cleared (sum)
- **normalizer.tcp_reserved**: packets with reserved bits cleared (sum)
- **normalizer.test_tcp_nonce**: test packets with nonce bit cleared (sum)
- **normalizer.tcp_nonce**: packets with nonce bit cleared (sum)
- **normalizer.test_tcp_urgent_ptr**: test packets without data with urgent pointer cleared (sum)
- **normalizer.tcp_urgent_ptr**: packets without data with urgent pointer cleared (sum)
- **normalizer.test_tcp_ecn_pkt**: test packets with ECN bits cleared (sum)
- **normalizer.tcp_ecn_pkt**: packets with ECN bits cleared (sum)
- **normalizer.test_tcp_ts_ecr**: test timestamp cleared on non-ACKs (sum)
- **normalizer.tcp_ts_ecr**: timestamp cleared on non-ACKs (sum)

- **normalizer.test_tcp_req_urg**: test cleared urgent pointer when urgent flag is not set (sum)
- **normalizer.tcp_req_urg**: cleared urgent pointer when urgent flag is not set (sum)
- **normalizer.test_tcp_req_pay**: test cleared urgent pointer and urgent flag when there is no payload (sum)
- **normalizer.tcp_req_pay**: cleared urgent pointer and urgent flag when there is no payload (sum)
- **normalizer.test_tcp_req_urp**: test cleared the urgent flag if the urgent pointer is not set (sum)
- **normalizer.tcp_req_urp**: cleared the urgent flag if the urgent pointer is not set (sum)
- **normalizer.test_tcp_trim_syn**: test tcp segments trimmed on SYN (sum)
- **normalizer.tcp_trim_syn**: tcp segments trimmed on SYN (sum)
- **normalizer.test_tcp_trim_rst**: test RST packets with data trimmed (sum)
- **normalizer.tcp_trim_rst**: RST packets with data trimmed (sum)
- **normalizer.test_tcp_trim_win**: test data trimmed to window (sum)
- **normalizer.tcp_trim_win**: data trimmed to window (sum)
- **normalizer.test_tcp_trim_mss**: test data trimmed to MSS (sum)
- **normalizer.tcp_trim_mss**: data trimmed to MSS (sum)
- **normalizer.test_tcp_ecn_session**: test ECN bits cleared (sum)
- **normalizer.tcp_ecn_session**: ECN bits cleared (sum)
- **normalizer.test_tcp_ts_nop**: test timestamp options cleared (sum)
- **normalizer.tcp_ts_nop**: timestamp options cleared (sum)
- **normalizer.test_tcp_ips_data**: test normalized segments (sum)
- **normalizer.tcp_ips_data**: normalized segments (sum)
- **normalizer.test_tcp_block**: test blocked segments (sum)
- **normalizer.tcp_block**: blocked segments (sum)

5.33 null_trace_logger

Help: trace logger with a null printout

Type: inspector (passive)

Usage: global

Instance Type: global

5.34 packet_capture

Help: raw packet dumping facility

Type: inspector (probe)

Usage: global

Instance Type: global

Configuration:

- bool **packet_capture.enable** = false: initially enable packet dumping
- string **packet_capture.filter**: bpf filter to use for packet dump
- int **packet_capture.group** = -1: group filter to use for the packet dump { -1:32767 }

Commands:

- **packet_capture.enable**(filter, group): dump raw packets
- **packet_capture.disable**(): stop packet dump

Peg counts:

- **packet_capture.processed**: packets processed against filter (sum)
- **packet_capture.captured**: packets matching dumped after matching filter (sum)

5.35 perf_monitor

Help: performance monitoring and flow statistics collection

Type: inspector (probe)

Usage: global

Instance Type: global

Configuration:

- bool **perf_monitor.base** = true: enable base statistics
- bool **perf_monitor.cpu** = false: enable cpu statistics
- bool **perf_monitor.flow** = false: enable traffic statistics
- bool **perf_monitor.flow_ip** = false: enable statistics on host pairs
- int **perf_monitor.packets** = 10000: minimum packets to report { 0:max32 }
- int **perf_monitor.seconds** = 60: report interval { 0:max32 }
- int **perf_monitor.flow_ip_memcap** = 52428800: maximum memory in bytes for flow tracking { 236:maxSZ }
- int **perf_monitor.max_file_size** = 1073741824: files will be rolled over if they exceed this size { 4096:max53 }
- int **perf_monitor.flow_ports** = 1023: maximum ports to track { 0:65535 }
- enum **perf_monitor.output** = *file*: output location for stats { *file* | *console* }
- string **`perf_monitor.modules[].name`**: name of the module
- string **`perf_monitor.modules[].pegs`**: list of statistics to track or empty for all counters
- enum **perf_monitor.format** = *csv*: output format for stats { *csv* | *text* | *json* }
- bool **perf_monitor.summary** = false: output summary at shutdown

Commands:

- **perf_monitor.enable_flow_ip_profiling**(seconds, packets): enable statistics on host pairs
- **perf_monitor.disable_flow_ip_profiling**(): disable statistics on host pairs

- **perf_monitor.show_flow_ip_profiling**(): show status of statistics on host pairs

Peg counts:

- **perf_monitor.packets**: total packets processed by performance monitor (sum)
- **perf_monitor.flow_tracker_creates**: total number of flow trackers created (sum)
- **perf_monitor.flow_tracker_total_deletes**: flow trackers deleted to stay below memcap limit (sum)
- **perf_monitor.flow_tracker_reload_deletes**: flow trackers deleted due to memcap change on config reload (sum)
- **perf_monitor.flow_tracker_prunes**: flow trackers pruned for reuse by new flows (sum)

5.36 pop

Help: pop inspection

Type: inspector (service)

Usage: inspect

Instance Type: multiton

Configuration:

- int **pop.b64_decode_depth** = -1: base64 decoding depth (-1 no limit) { -1:65535 }
- int **pop.bitenc_decode_depth** = -1: Non-Encoded MIME attachment extraction depth (-1 no limit) { -1:65535 }
- bool **pop.decompress_pdf** = false: decompress pdf files in MIME attachments
- bool **pop.decompress_swf** = false: decompress swf files in MIME attachments
- bool **pop.decompress_zip** = false: decompress zip files in MIME attachments
- bool **pop.decompress_vba** = false: decompress MS Office Visual Basic for Applications macro files in MIME attachments
- int **pop.qp_decode_depth** = -1: Quoted Printable decoding depth (-1 no limit) { -1:65535 }
- int **pop.uu_decode_depth** = -1: Unix-to-Unix decoding depth (-1 no limit) { -1:65535 }

Rules:

- **142:1** (pop) unknown POP3 command
- **142:2** (pop) unknown POP3 response
- **142:4** (pop) base64 decoding failed
- **142:5** (pop) quoted-printable decoding failed
- **142:7** (pop) Unix-to-Unix decoding failed
- **142:8** (pop) file decompression failed

Peg counts:

- **pop.packets**: total packets processed (sum)
- **pop.total_bytes**: total number of bytes processed (sum)
- **pop.sessions**: total pop sessions (sum)

- **pop.concurrent_sessions**: total concurrent pop sessions (now)
- **pop.max_concurrent_sessions**: maximum concurrent pop sessions (max)
- **pop.start_tls**: total STARTTLS events generated (sum)
- **pop.ssl_search_abandoned**: total SSL search abandoned (sum)
- **pop.ssl_srch_abandoned_early**: total SSL search abandoned too soon (sum)
- **pop.b64_attachments**: total base64 attachments decoded (sum)
- **pop.b64_decoded_bytes**: total base64 decoded bytes (sum)
- **pop.qp_attachments**: total quoted-printable attachments decoded (sum)
- **pop.qp_decoded_bytes**: total quoted-printable decoded bytes (sum)
- **pop.uu_attachments**: total uu attachments decoded (sum)
- **pop.uu_decoded_bytes**: total uu decoded bytes (sum)
- **pop.non_encoded_attachments**: total non-encoded attachments extracted (sum)
- **pop.non_encoded_bytes**: total non-encoded extracted bytes (sum)
- **pop.js_pdf_scripts**: total number of PDF files processed (sum)

5.37 port_scan

Help: detect various ip, icmp, tcp, and udp port or protocol scans

Type: inspector (probe)

Usage: global

Instance Type: global

Configuration:

- int **port_scan.memcap** = 10485760: maximum tracker memory in bytes { 1024:maxSZ }
- multi **port_scan.protos** = *all*: choose the protocols to monitor { tcp | udp | icmp | ip | all }
- multi **port_scan.scan_types** = *all*: choose type of scans to look for { portscan | portsweep | decoy_portscan | distributed_portscan | all }
- string **port_scan.watch_ip**: list of CIDRs with optional ports to watch
- string **port_scan.ignore_scanners**: list of CIDRs with optional ports to ignore if the source of scan alerts
- string **port_scan.ignore_scanned**: list of CIDRs with optional ports to ignore if the destination of scan alerts
- bool **port_scan.alert_all** = false: alert on all events over threshold within window if true; else alert on first only
- bool **port_scan.include_midstream** = false: list of CIDRs with optional ports
- int **port_scan.tcp_ports.scans** = 100: scan attempts { 0:65535 }
- int **port_scan.tcp_ports.rejects** = 15: scan attempts with negative response { 0:65535 }
- int **port_scan.tcp_ports.nets** = 25: number of times address changed from prior attempt { 0:65535 }
- int **port_scan.tcp_ports.ports** = 25: number of times port (or proto) changed from prior attempt { 0:65535 }
- int **port_scan.tcp_decoy.scans** = 100: scan attempts { 0:65535 }

- int **port_scan.tcp_decoy.rejects** = 15: scan attempts with negative response { 0:65535 }
- int **port_scan.tcp_decoy.nets** = 25: number of times address changed from prior attempt { 0:65535 }
- int **port_scan.tcp_decoy.ports** = 25: number of times port (or proto) changed from prior attempt { 0:65535 }
- int **port_scan.tcp_sweep.scans** = 100: scan attempts { 0:65535 }
- int **port_scan.tcp_sweep.rejects** = 15: scan attempts with negative response { 0:65535 }
- int **port_scan.tcp_sweep.nets** = 25: number of times address changed from prior attempt { 0:65535 }
- int **port_scan.tcp_sweep.ports** = 25: number of times port (or proto) changed from prior attempt { 0:65535 }
- int **port_scan.tcp_dist.scans** = 100: scan attempts { 0:65535 }
- int **port_scan.tcp_dist.rejects** = 15: scan attempts with negative response { 0:65535 }
- int **port_scan.tcp_dist.nets** = 25: number of times address changed from prior attempt { 0:65535 }
- int **port_scan.tcp_dist.ports** = 25: number of times port (or proto) changed from prior attempt { 0:65535 }
- int **port_scan.udp_ports.scans** = 100: scan attempts { 0:65535 }
- int **port_scan.udp_ports.rejects** = 15: scan attempts with negative response { 0:65535 }
- int **port_scan.udp_ports.nets** = 25: number of times address changed from prior attempt { 0:65535 }
- int **port_scan.udp_ports.ports** = 25: number of times port (or proto) changed from prior attempt { 0:65535 }
- int **port_scan.udp_decoy.scans** = 100: scan attempts { 0:65535 }
- int **port_scan.udp_decoy.rejects** = 15: scan attempts with negative response { 0:65535 }
- int **port_scan.udp_decoy.nets** = 25: number of times address changed from prior attempt { 0:65535 }
- int **port_scan.udp_decoy.ports** = 25: number of times port (or proto) changed from prior attempt { 0:65535 }
- int **port_scan.udp_sweep.scans** = 100: scan attempts { 0:65535 }
- int **port_scan.udp_sweep.rejects** = 15: scan attempts with negative response { 0:65535 }
- int **port_scan.udp_sweep.nets** = 25: number of times address changed from prior attempt { 0:65535 }
- int **port_scan.udp_sweep.ports** = 25: number of times port (or proto) changed from prior attempt { 0:65535 }
- int **port_scan.udp_dist.scans** = 100: scan attempts { 0:65535 }
- int **port_scan.udp_dist.rejects** = 15: scan attempts with negative response { 0:65535 }
- int **port_scan.udp_dist.nets** = 25: number of times address changed from prior attempt { 0:65535 }
- int **port_scan.udp_dist.ports** = 25: number of times port (or proto) changed from prior attempt { 0:65535 }
- int **port_scan.ip_proto.scans** = 100: scan attempts { 0:65535 }
- int **port_scan.ip_proto.rejects** = 15: scan attempts with negative response { 0:65535 }
- int **port_scan.ip_proto.nets** = 25: number of times address changed from prior attempt { 0:65535 }
- int **port_scan.ip_proto.ports** = 25: number of times port (or proto) changed from prior attempt { 0:65535 }
- int **port_scan.ip_decoy.scans** = 100: scan attempts { 0:65535 }
- int **port_scan.ip_decoy.rejects** = 15: scan attempts with negative response { 0:65535 }
- int **port_scan.ip_decoy.nets** = 25: number of times address changed from prior attempt { 0:65535 }
- int **port_scan.ip_decoy.ports** = 25: number of times port (or proto) changed from prior attempt { 0:65535 }

- int **port_scan.ip_sweep.scans** = 100: scan attempts { 0:65535 }
- int **port_scan.ip_sweep.rejects** = 15: scan attempts with negative response { 0:65535 }
- int **port_scan.ip_sweep.nets** = 25: number of times address changed from prior attempt { 0:65535 }
- int **port_scan.ip_sweep.ports** = 25: number of times port (or proto) changed from prior attempt { 0:65535 }
- int **port_scan.ip_dist.scans** = 100: scan attempts { 0:65535 }
- int **port_scan.ip_dist.rejects** = 15: scan attempts with negative response { 0:65535 }
- int **port_scan.ip_dist.nets** = 25: number of times address changed from prior attempt { 0:65535 }
- int **port_scan.ip_dist.ports** = 25: number of times port (or proto) changed from prior attempt { 0:65535 }
- int **port_scan.icmp_sweep.scans** = 100: scan attempts { 0:65535 }
- int **port_scan.icmp_sweep.rejects** = 15: scan attempts with negative response { 0:65535 }
- int **port_scan.icmp_sweep.nets** = 25: number of times address changed from prior attempt { 0:65535 }
- int **port_scan.icmp_sweep.ports** = 25: number of times port (or proto) changed from prior attempt { 0:65535 }
- int **port_scan.tcp_window** = 0: detection interval for all TCP scans { 0:max32 }
- int **port_scan.udp_window** = 0: detection interval for all UDP scans { 0:max32 }
- int **port_scan.ip_window** = 0: detection interval for all IP scans { 0:max32 }
- int **port_scan.icmp_window** = 0: detection interval for all ICMP scans { 0:max32 }

Rules:

- **122:1** (port_scan) TCP portscan
- **122:2** (port_scan) TCP decoy portscan
- **122:3** (port_scan) TCP portsweep
- **122:4** (port_scan) TCP distributed portscan
- **122:5** (port_scan) TCP filtered portscan
- **122:6** (port_scan) TCP filtered decoy portscan
- **122:7** (port_scan) TCP filtered portsweep
- **122:8** (port_scan) TCP filtered distributed portscan
- **122:9** (port_scan) IP protocol scan
- **122:10** (port_scan) IP decoy protocol scan
- **122:11** (port_scan) IP protocol sweep
- **122:12** (port_scan) IP distributed protocol scan
- **122:13** (port_scan) IP filtered protocol scan
- **122:14** (port_scan) IP filtered decoy protocol scan
- **122:15** (port_scan) IP filtered protocol sweep
- **122:16** (port_scan) IP filtered distributed protocol scan
- **122:17** (port_scan) UDP portscan

- **122:18** (port_scan) UDP decoy portscan
- **122:19** (port_scan) UDP portsweep
- **122:20** (port_scan) UDP distributed portscan
- **122:21** (port_scan) UDP filtered portscan
- **122:22** (port_scan) UDP filtered decoy portscan
- **122:23** (port_scan) UDP filtered portsweep
- **122:24** (port_scan) UDP filtered distributed portscan
- **122:25** (port_scan) ICMP sweep
- **122:26** (port_scan) ICMP filtered sweep
- **122:27** (port_scan) open port

Peg counts:

- **port_scan.packets**: number of packets processed by port scan (sum)
- **port_scan.trackers**: number of trackers allocated by port scan (sum)
- **port_scan.alloc_prunes**: number of trackers pruned on allocation of new tracking (sum)
- **port_scan.reload_prunes**: number of trackers pruned on reload due to reduced memcap (sum)
- **port_scan.bytes_in_use**: number of bytes currently used by portscan (now)

5.38 reputation

Help: reputation inspection

Type: inspector (passive)

Usage: context

Instance Type: network

Configuration:

- string **reputation.blocklist**: blocklist file name with IP lists
- string **reputation.list_dir**: directory for IP lists and manifest file
- int **reputation.memcap** = 500: maximum total MB of memory allocated { 1:4095 }
- enum **reputation.nested_ip** = *inner*: IP to use when there is IP encapsulation { *inner*|*outer*|*all* }
- enum **reputation.priority** = *allowlist*: defines priority when there is a decision conflict during run-time { *blocklist*|*allowlist* }
- bool **reputation.scan_local** = false: inspect local address defined in RFC 1918
- enum **reputation.allow** = *do_not_block*: specify the meaning of allowlist { *do_not_block*|*trust* }
- string **reputation.allowlist**: allowlist file name with IP lists

Commands:

- **reputation.reload**(): reload reputation data

Rules:

- **136:1** (reputation) packets blocked based on source
- **136:2** (reputation) packets trusted based on source
- **136:3** (reputation) packets monitored based on source
- **136:4** (reputation) packets blocked based on destination
- **136:5** (reputation) packets trusted based on destination
- **136:6** (reputation) packets monitored based on destination

Peg counts:

- **reputation.packets**: total packets processed (sum)
- **reputation.blocked**: number of packets blocked (sum)
- **reputation.trusted**: number of packets trusted (sum)
- **reputation.monitored**: number of packets monitored (sum)
- **reputation.memory_allocated**: total memory allocated (sum)
- **reputation.aux_ip_blocked**: number of auxiliary ip packets blocked (sum)
- **reputation.aux_ip_trusted**: number of auxiliary ip packets trusted (sum)
- **reputation.aux_ip_monitored**: number of auxiliary ip packets monitored (sum)

5.39 rna

Help: Real-time network awareness and OS fingerprinting (experimental)

Type: inspector (control)

Usage: context

Instance Type: network

Configuration:

- string **rna.rna_conf_path**: path to rna configuration
- bool **rna.enable_logger** = true: enable or disable writing discovery events into logger
- bool **rna.log_when_idle** = false: enable host update logging when snort is idle
- string **rna.dump_file**: file name to dump RNA mac cache on shutdown; won't dump by default
- int **`rna.tcp_fingerprints[].fpid`** = 0: fingerprint id { 0:max32 }
- int **`rna.tcp_fingerprints[].type`** = 0: fingerprint type { 0:max32 }
- string **`rna.tcp_fingerprints[].uuid`**: fingerprint uuid
- int **`rna.tcp_fingerprints[].ttl`** = 0: fingerprint ttl { 0:255 }
- string **`rna.tcp_fingerprints[].tcp_window`**: fingerprint tcp window
- string **`rna.tcp_fingerprints[].mss`** = *X*: fingerprint mss
- string **`rna.tcp_fingerprints[].id`** = *X*: id
- string **`rna.tcp_fingerprints[].topts`**: fingerprint tcp options

- string **rna.tcp_fingerprints[].ws** = *X*: fingerprint window size
- bool **rna.tcp_fingerprints[].df** = false: fingerprint don't fragment flag
- enum **rna.tcp_fingerprints[].ua_type** = *os*: type of user agent fingerprints { *os* | *device* | *jail-broken* | *jail-broken-host* }
- string **rna.tcp_fingerprints[].user_agent[].substring**: a substring of user agent string
- string **rna.tcp_fingerprints[].host_name**: host name information
- string **rna.tcp_fingerprints[].device**: device information
- string **rna.tcp_fingerprints[].dhcp55**: dhcp option 55 values
- string **rna.tcp_fingerprints[].dhcp60**: dhcp option 60 values
- int **rna.tcp_fingerprints[].major**: smb major version { 0:max31 }
- int **rna.tcp_fingerprints[].minor**: smb minor version { 0:max31 }
- int **rna.tcp_fingerprints[].flags**: smb flags { 0:max32 }
- int **rna.ua_fingerprints[].fpid** = 0: fingerprint id { 0:max32 }
- int **rna.ua_fingerprints[].type** = 0: fingerprint type { 0:max32 }
- string **rna.ua_fingerprints[].uuid**: fingerprint uuid
- int **rna.ua_fingerprints[].ttl** = 0: fingerprint ttl { 0:255 }
- string **rna.ua_fingerprints[].tcp_window**: fingerprint tcp window
- string **rna.ua_fingerprints[].mss** = *X*: fingerprint mss
- string **rna.ua_fingerprints[].id** = *X*: id
- string **rna.ua_fingerprints[].topts**: fingerprint tcp options
- string **rna.ua_fingerprints[].ws** = *X*: fingerprint window size
- bool **rna.ua_fingerprints[].df** = false: fingerprint don't fragment flag
- enum **rna.ua_fingerprints[].ua_type** = *os*: type of user agent fingerprints { *os* | *device* | *jail-broken* | *jail-broken-host* }
- string **rna.ua_fingerprints[].user_agent[].substring**: a substring of user agent string
- string **rna.ua_fingerprints[].host_name**: host name information
- string **rna.ua_fingerprints[].device**: device information
- string **rna.ua_fingerprints[].dhcp55**: dhcp option 55 values
- string **rna.ua_fingerprints[].dhcp60**: dhcp option 60 values
- int **rna.ua_fingerprints[].major**: smb major version { 0:max31 }
- int **rna.ua_fingerprints[].minor**: smb minor version { 0:max31 }
- int **rna.ua_fingerprints[].flags**: smb flags { 0:max32 }
- int **rna.udp_fingerprints[].fpid** = 0: fingerprint id { 0:max32 }
- int **rna.udp_fingerprints[].type** = 0: fingerprint type { 0:max32 }
- string **rna.udp_fingerprints[].uuid**: fingerprint uuid
- int **rna.udp_fingerprints[].ttl** = 0: fingerprint ttl { 0:255 }

- string **rna.udp_fingerprints[].tcp_window**: fingerprint tcp window
- string **rna.udp_fingerprints[].mss** = *X*: fingerprint mss
- string **rna.udp_fingerprints[].id** = *X*: id
- string **rna.udp_fingerprints[].topts**: fingerprint tcp options
- string **rna.udp_fingerprints[].ws** = *X*: fingerprint window size
- bool **rna.udp_fingerprints[].df** = false: fingerprint don't fragment flag
- enum **rna.udp_fingerprints[].ua_type** = *os*: type of user agent fingerprints { *os* | *device* | *jail-broken* | *jail-broken-host* }
- string **rna.udp_fingerprints[].user_agent[].substring**: a substring of user agent string
- string **rna.udp_fingerprints[].host_name**: host name information
- string **rna.udp_fingerprints[].device**: device information
- string **rna.udp_fingerprints[].dhcp55**: dhcp option 55 values
- string **rna.udp_fingerprints[].dhcp60**: dhcp option 60 values
- int **rna.udp_fingerprints[].major**: smb major version { 0:max31 }
- int **rna.udp_fingerprints[].minor**: smb minor version { 0:max31 }
- int **rna.udp_fingerprints[].flags**: smb flags { 0:max32 }
- int **rna.smb_fingerprints[].fpid** = 0: fingerprint id { 0:max32 }
- int **rna.smb_fingerprints[].type** = 0: fingerprint type { 0:max32 }
- string **rna.smb_fingerprints[].uuid**: fingerprint uuid
- int **rna.smb_fingerprints[].ttl** = 0: fingerprint ttl { 0:255 }
- string **rna.smb_fingerprints[].tcp_window**: fingerprint tcp window
- string **rna.smb_fingerprints[].mss** = *X*: fingerprint mss
- string **rna.smb_fingerprints[].id** = *X*: id
- string **rna.smb_fingerprints[].topts**: fingerprint tcp options
- string **rna.smb_fingerprints[].ws** = *X*: fingerprint window size
- bool **rna.smb_fingerprints[].df** = false: fingerprint don't fragment flag
- enum **rna.smb_fingerprints[].ua_type** = *os*: type of user agent fingerprints { *os* | *device* | *jail-broken* | *jail-broken-host* }
- string **rna.smb_fingerprints[].user_agent[].substring**: a substring of user agent string
- string **rna.smb_fingerprints[].host_name**: host name information
- string **rna.smb_fingerprints[].device**: device information
- string **rna.smb_fingerprints[].dhcp55**: dhcp option 55 values
- string **rna.smb_fingerprints[].dhcp60**: dhcp option 60 values
- int **rna.smb_fingerprints[].major**: smb major version { 0:max31 }
- int **rna.smb_fingerprints[].minor**: smb minor version { 0:max31 }
- int **rna.smb_fingerprints[].flags**: smb flags { 0:max32 }

Commands:

- **rna.dump_macs**(): dump rna's internal MAC trackers
- **rna.delete_mac_host**(mac): delete a MAC from rna's MAC cache
- **rna.delete_mac_host_proto**(mac, proto): delete a protocol associated with a MAC host
- **rna.purge_data**(): purge all host cache and mac cache data

Peg counts:

- **rna.appid_change**: count of appid change events received (sum)
- **rna.cpe_os**: count of CPE OS events received (sum)
- **rna.icmp_bidirectional**: count of bidirectional ICMP flows received (sum)
- **rna.icmp_new**: count of new ICMP flows received (sum)
- **rna.ip_bidirectional**: count of bidirectional IP received (sum)
- **rna.ip_new**: count of new IP flows received (sum)
- **rna.udp_bidirectional**: count of bidirectional UDP flows received (sum)
- **rna.udp_new**: count of new UDP flows received (sum)
- **rna.tcp_syn**: count of TCP SYN packets received (sum)
- **rna.tcp_syn_ack**: count of TCP SYN-ACK packets received (sum)
- **rna.tcp_midstream**: count of TCP midstream packets received (sum)
- **rna.other_packets**: count of packets received without session tracking (sum)
- **rna.change_host_update**: count number of change host update events (sum)
- **rna.dhcp_data**: count of DHCP data events received (sum)
- **rna.dhcp_info**: count of new DHCP lease events received (sum)
- **rna.smb**: count of new SMB events received (sum)
- **rna.netflow_record**: count of netflow record events received (sum)
- **rna.total_events_in_interval**: count of RNA events generated (sum)
- **rna.total_packets_in_interval**: count of packets processed (sum)
- **rna.total_bytes_in_interval**: count of bytes processed (sum)

5.40 rpc_decode

Help: RPC inspector

Type: inspector (service)

Usage: inspect

Instance Type: multiton

Rules:

- **106:1** (rpc_decode) fragmented RPC records

- **106:2** (rpc_decode) multiple RPC records
- **106:3** (rpc_decode) large RPC record fragment
- **106:4** (rpc_decode) incomplete RPC segment
- **106:5** (rpc_decode) zero-length RPC fragment

Peg counts:

- **rpc_decode.total_packets**: total packets (sum)
- **rpc_decode.concurrent_sessions**: total concurrent rpc sessions (now)
- **rpc_decode.max_concurrent_sessions**: maximum concurrent rpc sessions (max)

5.41 s7commplus

Help: s7commplus inspection

Type: inspector (service)

Usage: inspect

Instance Type: multiton

Rules:

- **149:1** (s7commplus) length in S7commplus MBAP header does not match the length needed for the given S7commplus function
- **149:2** (s7commplus) S7commplus protocol ID is non-zero
- **149:3** (s7commplus) reserved S7commplus function code in use

Peg counts:

- **s7commplus.sessions**: total sessions processed (sum)
- **s7commplus.frames**: total S7commplus messages (sum)
- **s7commplus.concurrent_sessions**: total concurrent s7commplus sessions (now)
- **s7commplus.max_concurrent_sessions**: maximum concurrent s7commplus sessions (max)

5.42 sip

Help: sip inspection

Type: inspector (service)

Usage: inspect

Instance Type: multiton

Configuration:

- bool **sip.ignore_call_channel** = false: enables the support for ignoring audio/video data channel
- int **sip.max_call_id_len** = 256: maximum call id field size { 0:65535 }
- int **sip.max_contact_len** = 256: maximum contact field size { 0:65535 }

- int **sip.max_content_len** = 1024: maximum content length of the message body { 0:65535 }
- int **sip.max_dialogs** = 4: maximum number of dialogs within one stream session { 1:max32 }
- int **sip.max_from_len** = 256: maximum from field size { 0:65535 }
- int **sip.max_request_name_len** = 20: maximum request name field size { 0:65535 }
- int **sip.max_requestName_len** = 20: deprecated - use max_request_name_len instead { 0:65535 }
- int **sip.max_to_len** = 256: maximum to field size { 0:65535 }
- int **sip.max_uri_len** = 256: maximum request uri field size { 0:65535 }
- int **sip.max_via_len** = 1024: maximum via field size { 0:65535 }
- string **sip.methods** = *invite cancel ack bye register options*: list of methods to check in SIP messages

Rules:

- **140:2** (sip) empty request URI
- **140:3** (sip) URI is too long
- **140:4** (sip) empty call-Id
- **140:5** (sip) Call-Id is too long
- **140:6** (sip) CSeq number is too large or negative
- **140:7** (sip) request name in CSeq is too long
- **140:8** (sip) empty From header
- **140:9** (sip) From header is too long
- **140:10** (sip) empty To header
- **140:11** (sip) To header is too long
- **140:12** (sip) empty Via header
- **140:13** (sip) Via header is too long
- **140:14** (sip) empty Contact
- **140:15** (sip) contact is too long
- **140:16** (sip) content length is too large or negative
- **140:17** (sip) multiple SIP messages in a packet
- **140:18** (sip) content length mismatch
- **140:19** (sip) request name is invalid
- **140:20** (sip) Invite replay attack
- **140:21** (sip) illegal session information modification
- **140:22** (sip) response status code is not a 3 digit number
- **140:23** (sip) empty Content-type header
- **140:24** (sip) SIP version is invalid
- **140:25** (sip) mismatch in METHOD of request and the CSEQ header

- **140:26** (sip) method is unknown
- **140:27** (sip) maximum dialogs within a session reached

Peg counts:

- **sip.packets**: total packets (sum)
- **sip.sessions**: total sessions (sum)
- **sip.concurrent_sessions**: total concurrent SIP sessions (now)
- **sip.max_concurrent_sessions**: maximum concurrent SIP sessions (max)
- **sip.events**: events generated (sum)
- **sip.dialogs**: total dialogs (sum)
- **sip.ignored_channels**: total channels ignored (sum)
- **sip.ignored_sessions**: total sessions ignored (sum)
- **sip.total_requests**: total requests (sum)
- **sip.invite**: invite (sum)
- **sip.cancel**: cancel (sum)
- **sip.ack**: ack (sum)
- **sip.bye**: bye (sum)
- **sip.register**: register (sum)
- **sip.options**: options (sum)
- **sip.refer**: refer (sum)
- **sip.subscribe**: subscribe (sum)
- **sip.update**: update (sum)
- **sip.join**: join (sum)
- **sip.info**: info (sum)
- **sip.message**: message (sum)
- **sip.notify**: notify (sum)
- **sip.prack**: prack (sum)
- **sip.total_responses**: total responses (sum)
- **sip.code_1xx**: 1xx (sum)
- **sip.code_2xx**: 2xx (sum)
- **sip.code_3xx**: 3xx (sum)
- **sip.code_4xx**: 4xx (sum)
- **sip.code_5xx**: 5xx (sum)
- **sip.code_6xx**: 6xx (sum)
- **sip.code_7xx**: 7xx (sum)
- **sip.code_8xx**: 8xx (sum)
- **sip.code_9xx**: 9xx (sum)

5.43 smtp

Help: smtp inspection

Type: inspector (service)

Usage: inspect

Instance Type: multiton

Configuration:

- string **`smtp.alt_max_command_line_len[].command`**: command string
- int **`smtp.alt_max_command_line_len[].length`** = 0: specify non-default maximum for command { 0:max32 }
- string **smtp.auth_cmds**: commands that initiate an authentication exchange
- int **smtp.b64_decode_depth** = -1: depth used to decode the base64 encoded MIME attachments (-1 no limit) { -1:65535 }
- string **smtp.binary_data_cmds**: commands that initiate sending of data and use a length value after the command
- int **smtp.bitenc_decode_depth** = -1: depth used to extract the non-encoded MIME attachments (-1 no limit) { -1:65535 }
- string **smtp.data_cmds**: commands that initiate sending of data with an end of data delimiter
- bool **smtp.decompress_pdf** = false: decompress pdf files in MIME attachments
- bool **smtp.decompress_swf** = false: decompress swf files in MIME attachments
- bool **smtp.decompress_zip** = false: decompress zip files in MIME attachments
- bool **smtp.decompress_vba** = false: decompress MS Office Visual Basic for Applications macro files in MIME attachments
- int **smtp.email_hdrs_log_depth** = 1464: depth for logging email headers { 0:20480 }
- bool **smtp.ignore_data** = false: ignore data section of mail
- bool **smtp.ignore_tls_data** = false: ignore TLS-encrypted data when processing rules
- string **smtp.invalid_cmds**: alert if this command is sent from client side
- bool **smtp.log_email_hdrs** = false: log the SMTP email headers extracted from SMTP data
- bool **smtp.log_filename** = false: log the MIME attachment filenames extracted from the Content-Disposition header within the MIME body
- bool **smtp.log_mailfrom** = false: log the sender's email address extracted from the MAIL FROM command
- bool **smtp.log_rcptto** = false: log the recipient's email address extracted from the RCPT TO command
- int **smtp.max_auth_command_line_len** = 1000: max auth command Line Length { 0:65535 }
- int **smtp.max_command_line_len** = 512: max Command Line Length { 0:65535 }
- int **smtp.max_header_line_len** = 1000: max SMTP DATA header line { 0:65535 }
- int **smtp.max_response_line_len** = 512: max SMTP response line { 0:65535 }
- enum **smtp.normalize** = *none*: turns on/off normalization { *none* | *cmds* | *all* }
- string **smtp.normalize_cmds**: list of commands to normalize
- int **smtp.qp_decode_depth** = -1: quoted-Printable decoding depth (-1 no limit) { -1:65535 }
- int **smtp.uu_decode_depth** = -1: Unix-to-Unix decoding depth (-1 no limit) { -1:65535 }
- string **smtp.valid_cmds**: list of valid commands

- enum **smtp.xlink2state** = *alert*: enable/disable xlink2state alert { *disable* | *alert* | *drop* }

Rules:

- **124:1** (smtp) attempted command buffer overflow
- **124:2** (smtp) attempted data header buffer overflow
- **124:3** (smtp) attempted response buffer overflow
- **124:4** (smtp) attempted specific command buffer overflow
- **124:5** (smtp) unknown command
- **124:6** (smtp) illegal command
- **124:7** (smtp) attempted header name buffer overflow
- **124:8** (smtp) attempted X-Link2State command buffer overflow
- **124:10** (smtp) base64 decoding failed
- **124:11** (smtp) quoted-printable decoding failed
- **124:13** (smtp) Unix-to-Unix decoding failed
- **124:14** (smtp) Cyrus SASL authentication attack
- **124:15** (smtp) attempted authentication command buffer overflow
- **124:16** (smtp) file decompression failed
- **124:17** (smtp) STARTTLS command injection attempt
- **124:18** (smtp) mix of LF and CRLF as end of line

Peg counts:

- **smtp.packets**: total packets processed (sum)
- **smtp.total_bytes**: total number of bytes processed (sum)
- **smtp.sessions**: total smtp sessions (sum)
- **smtp.concurrent_sessions**: total concurrent smtp sessions (now)
- **smtp.max_concurrent_sessions**: maximum concurrent smtp sessions (max)
- **smtp.start_tls**: total STARTTLS events generated (sum)
- **smtp.ssl_search_abandoned**: total SSL search abandoned (sum)
- **smtp.ssl_srch_abandoned_early**: total SSL search abandoned too soon (sum)
- **smtp.b64_attachments**: total base64 attachments decoded (sum)
- **smtp.b64_decoded_bytes**: total base64 decoded bytes (sum)
- **smtp.qp_attachments**: total quoted-printable attachments decoded (sum)
- **smtp.qp_decoded_bytes**: total quoted-printable decoded bytes (sum)
- **smtp.uu_attachments**: total uu attachments decoded (sum)
- **smtp.uu_decoded_bytes**: total uu decoded bytes (sum)
- **smtp.non_encoded_attachments**: total non-encoded attachments extracted (sum)
- **smtp.non_encoded_bytes**: total non-encoded extracted bytes (sum)
- **smtp.js_pdf_scripts**: total number of PDF files processed (sum)

5.44 so_proxy

Help: a proxy inspector to track flow data from SO rules (internal use only)

Type: inspector (passive)

Usage: global

Instance Type: global

5.45 ssh

Help: ssh inspection

Type: inspector (service)

Usage: inspect

Instance Type: multiton

Configuration:

- int **ssh.max_encrypted_packets** = 25: ignore session after this many encrypted packets { 0:65535 }
- int **ssh.max_client_bytes** = 19600: number of unanswered bytes before alerting on challenge-response overflow or CRC32 { 0:65535 }
- int **ssh.max_server_version_len** = 80: limit before alerting on secure CRT server version string overflow { 0:255 }

Rules:

- **128:1** (ssh) challenge-response overflow exploit
- **128:2** (ssh) SSH1 CRC32 exploit
- **128:3** (ssh) server version string overflow
- **128:5** (ssh) bad message direction
- **128:6** (ssh) payload size incorrect for the given payload
- **128:7** (ssh) failed to detect SSH version string

Peg counts:

- **ssh.packets**: total packets (sum)
- **ssh.total_bytes**: total number of bytes processed (sum)
- **ssh.concurrent_sessions**: total concurrent ssh sessions (now)
- **ssh.max_concurrent_sessions**: maximum concurrent ssh sessions (max)

5.46 ssl

Help: ssl inspection

Type: inspector (service)

Usage: inspect

Instance Type: multiton

Configuration:

- bool **ssl.trust_servers** = false: disables requirement that application (encrypted) data must be observed on both sides
- int **ssl.max_heartbeat_length** = 0: maximum length of heartbeat record allowed { 0:65535 }

Rules:

- **137:1** (ssl) invalid client HELLO after server HELLO detected
- **137:2** (ssl) invalid server HELLO without client HELLO detected
- **137:3** (ssl) heartbeat read overrun attempt detected
- **137:4** (ssl) large heartbeat response detected

Peg counts:

- **ssl.packets**: total packets processed (sum)
- **ssl.decoded**: ssl packets decoded (sum)
- **ssl.client_hello**: total client hellos (sum)
- **ssl.server_hello**: total server hellos (sum)
- **ssl.certificate**: total ssl certificates (sum)
- **ssl.server_done**: total server done (sum)
- **ssl.client_key_exchange**: total client key exchanges (sum)
- **ssl.server_key_exchange**: total server key exchanges (sum)
- **ssl.change_cipher**: total change cipher records (sum)
- **ssl.finished**: total handshakes finished (sum)
- **ssl.client_application**: total client application records (sum)
- **ssl.server_application**: total server application records (sum)
- **ssl.alert**: total ssl alert records (sum)
- **ssl.unrecognized_records**: total unrecognized records (sum)
- **ssl.handshakes_completed**: total completed ssl handshakes (sum)
- **ssl.bad_handshakes**: total bad handshakes (sum)
- **ssl.sessions_ignored**: total sessions ignore (sum)
- **ssl.detection_disabled**: total detection disabled (sum)
- **ssl.concurrent_sessions**: total concurrent ssl sessions (now)
- **ssl.max_concurrent_sessions**: maximum concurrent ssl sessions (max)

5.47 stream

Help: common flow tracking

Type: inspector (stream)

Usage: global

Instance Type: global

Configuration:

- bool **stream.ip_frags_only** = false: don't process non-frag flows
- int **stream.max_flows** = 476288: maximum simultaneous flows tracked before pruning { 2:max32 }
- int **stream.prune_flows** = 10: maximum flows to prune at one time { 1:max32 }
- int **stream.pruning_timeout** = 30: minimum inactive time before being eligible for pruning { 1:max32 }
- int **stream.held_packet_timeout** = 1000: timeout in milliseconds for held packets { 1:max32 }
- int **stream.ip_cache.idle_timeout** = 180: maximum inactive time before retiring session tracker { 1:max32 }
- int **stream.icmp_cache.idle_timeout** = 180: maximum inactive time before retiring session tracker { 1:max32 }
- int **stream.tcp_cache.idle_timeout** = 3600: maximum inactive time before retiring session tracker { 1:max32 }
- int **stream.udp_cache.idle_timeout** = 180: maximum inactive time before retiring session tracker { 1:max32 }
- int **stream.user_cache.idle_timeout** = 180: maximum inactive time before retiring session tracker { 1:max32 }
- int **stream.file_cache.idle_timeout** = 180: maximum inactive time before retiring session tracker { 1:max32 }

Rules:

- **135:1** (stream) TCP SYN received
- **135:2** (stream) TCP session established
- **135:3** (stream) TCP session cleared

Peg counts:

- **stream.flows**: total sessions (sum)
- **stream.total_prunes**: total sessions pruned (sum)
- **stream.idle_prunes_max_flows**: sessions pruned due to pruning timeout since max flows is reached (sum)
- **stream.idle_prunes_proto_timeout**: sessions pruned due to protocol timeout (sum)
- **stream.excess_prunes**: sessions pruned due to excess (sum)
- **stream.uni_prunes**: uni sessions pruned (sum)
- **stream.memcap_prunes**: sessions pruned due to memcap (sum)
- **stream.ha_prunes**: sessions pruned by high availability sync (sum)
- **stream.stale_prunes**: sessions pruned due to stale connection (sum)
- **stream.expected_flows**: total expected flows created within snort (sum)
- **stream.expected_realized**: number of expected flows realized (sum)
- **stream.expected_pruned**: number of expected flows pruned (sum)

- **stream.expected_overflows**: number of expected cache overflows (sum)
- **stream.reload_tuning_idle**: number of times stream resource tuner called while idle (sum)
- **stream.reload_tuning_packets**: number of times stream resource tuner called while processing packets (sum)
- **stream.reload_total_adds**: number of flows added by config reloads (sum)
- **stream.reload_total_deletes**: number of flows deleted by config reloads (sum)
- **stream.reload_freelist_deletes**: number of flows deleted from the free list by config reloads (sum)
- **stream.reload_allowed_deletes**: number of allowed flows deleted by config reloads (sum)
- **stream.reload_blocked_deletes**: number of blocked flows deleted by config reloads (sum)
- **stream.reload_offloaded_deletes**: number of offloaded flows deleted by config reloads (sum)
- **stream.ip_timeout_prunes**: number of IP flows pruned due to timeout (sum)
- **stream.tcp_timeout_prunes**: number of TCP flows pruned due to timeout (sum)
- **stream.udp_timeout_prunes**: number of UDP flows pruned due to timeout (sum)
- **stream.icmp_timeout_prunes**: number of ICMP flows pruned due to timeout (sum)
- **stream.user_timeout_prunes**: number of USER flows pruned due to timeout (sum)
- **stream.file_timeout_prunes**: number of FILE flows pruned due to timeout (sum)
- **stream.pdu_timeout_prunes**: number of PDU flows pruned due to timeout (sum)
- **stream.ip_memcap_prunes**: number of IP flows pruned due to memcap (sum)
- **stream.tcp_memcap_prunes**: number of TCP flows pruned due to memcap (sum)
- **stream.udp_memcap_prunes**: number of UDP flows pruned due to memcap (sum)
- **stream.icmp_memcap_prunes**: number of ICMP flows pruned due to memcap (sum)
- **stream.user_memcap_prunes**: number of USER flows pruned due to memcap (sum)
- **stream.file_memcap_prunes**: number of FILE flows pruned due to memcap (sum)
- **stream.pdu_memcap_prunes**: number of PDU flows pruned due to memcap (sum)
- **stream.current_flows**: current number of flows in cache (now)
- **stream.uni_flows**: number of uni flows in cache (now)
- **stream.uni_ip_flows**: number of uni ip flows in cache (now)

5.48 stream_file

Help: stream inspector for file flow tracking and processing

Type: inspector (stream)

Usage: inspect

Instance Type: multiton

Configuration:

- bool **stream_file.upload** = false: indicate file transfer direction

5.49 stream_icmp

Help: stream inspector for ICMP flow tracking

Type: inspector (stream)

Usage: inspect

Instance Type: multiton

Configuration:

- int **stream_icmp.session_timeout** = 60: session tracking timeout { 1:max31 }

Peg counts:

- **stream_icmp.sessions**: total icmp sessions (sum)
- **stream_icmp.max**: max icmp sessions (max)
- **stream_icmp.created**: icmp session trackers created (sum)
- **stream_icmp.released**: icmp session trackers released (sum)
- **stream_icmp.timeouts**: icmp session timeouts (sum)
- **stream_icmp.prunes**: icmp session prunes (sum)

5.50 stream_ip

Help: stream inspector for IP flow tracking and defragmentation

Type: inspector (stream)

Usage: inspect

Instance Type: multiton

Configuration:

- int **stream_ip.max_frags** = 8192: maximum number of simultaneous fragments being tracked { 1:max32 }
- int **stream_ip.max_overlaps** = 0: maximum allowed overlaps per datagram; 0 is unlimited { 0:max32 }
- int **stream_ip.min_frag_length** = 0: alert if fragment length is below this limit before or after trimming { 0:65535 }
- int **stream_ip.min_ttl** = 1: discard fragments with TTL below the minimum { 1:255 }
- enum **stream_ip.policy** = *linux*: fragment reassembly policy { *first* | *linux* | *bsd* | *bsd_right* | *last* | *windows* | *solaris* }
- int **stream_ip.session_timeout** = 60: session tracking timeout { 1:max31 }

Rules:

- **123:1** (stream_ip) inconsistent IP options on fragmented packets
- **123:2** (stream_ip) teardrop attack
- **123:3** (stream_ip) short fragment, possible DOS attempt
- **123:4** (stream_ip) fragment packet ends after defragmented packet
- **123:5** (stream_ip) zero-byte fragment packet
- **123:6** (stream_ip) bad fragment size, packet size is negative

- **123:7** (stream_ip) bad fragment size, packet size is greater than 65536
- **123:8** (stream_ip) fragmentation overlap
- **123:11** (stream_ip) TTL value less than configured minimum, not using for reassembly
- **123:12** (stream_ip) excessive fragment overlap
- **123:13** (stream_ip) tiny fragment

Peg counts:

- **stream_ip.sessions**: total ip sessions (sum)
- **stream_ip.max**: max ip sessions (max)
- **stream_ip.created**: ip session trackers created (sum)
- **stream_ip.released**: ip session trackers released (sum)
- **stream_ip.timeouts**: ip session timeouts (sum)
- **stream_ip.prunes**: ip session prunes (sum)
- **stream_ip.total_bytes**: total number of bytes processed (sum)
- **stream_ip.total_frags**: total fragments (sum)
- **stream_ip.current_frags**: current fragments (now)
- **stream_ip.max_frags**: max fragments (sum)
- **stream_ip.reassembled**: reassembled datagrams (sum)
- **stream_ip.discards**: fragments discarded (sum)
- **stream_ip.frag_timeouts**: datagrams abandoned (sum)
- **stream_ip.overlaps**: overlapping fragments (sum)
- **stream_ip.anomalies**: anomalies detected (sum)
- **stream_ip.alerts**: alerts generated (sum)
- **stream_ip.drops**: fragments dropped (sum)
- **stream_ip.trackers_added**: datagram trackers created (sum)
- **stream_ip.trackers_freed**: datagram trackers released (sum)
- **stream_ip.trackers_cleared**: datagram trackers cleared (sum)
- **stream_ip.trackers_completed**: datagram trackers completed (sum)
- **stream_ip.nodes_inserted**: fragments added to tracker (sum)
- **stream_ip.nodes_deleted**: fragments deleted from tracker (sum)
- **stream_ip.reassembled_bytes**: total reassembled bytes (sum)
- **stream_ip.fragmented_bytes**: total fragmented bytes (sum)

5.51 stream_tcp

Help: stream inspector for TCP flow tracking and stream normalization and reassembly

Type: inspector (stream)

Usage: inspect

Instance Type: multiton

Configuration:

- int **stream_tcp.flush_factor** = 0: flush upon seeing a drop in segment size after given number of non-decreasing segments { 0:65535 }
- int **stream_tcp.max_window** = 0: maximum allowed TCP window { 0:1073725440 }
- int **stream_tcp.overlap_limit** = 0: maximum number of allowed overlapping segments per session { 0:max32 }
- int **stream_tcp.max_pdu** = 16384: maximum reassembled PDU size { 1460:32768 }
- bool **stream_tcp.no_ack** = false: received data is implicitly acked immediately
- enum **stream_tcp.policy** = *bsd*: determines operating system characteristics like reassembly { *first* | *last* | *linux* | *old_linux* | *bsd* | *macos* | *solaris* | *irix* | *hpux11* | *hpux10* | *windows* | *win_2003* | *vista* | *proxy* }
- bool **stream_tcp.reassemble_async** = true: queue data for reassembly before traffic is seen in both directions
- int **stream_tcp.require_3whs** = -1: don't track midstream sessions after given seconds from start up; -1 tracks all { -1:max31 }
- bool **stream_tcp.show_rebuilt_packets** = false: enable cmg like output of reassembled packets
- int **stream_tcp.queue_limit.max_bytes** = 4194304: don't queue more than given bytes per session and direction, 0 = unlimited { 0:max32 }
- int **stream_tcp.queue_limit.max_segments** = 3072: don't queue more than given segments per session and direction, 0 = unlimited { 0:max32 }
- int **stream_tcp.small_segments.count** = 0: number of consecutive (in the received order) TCP small segments considered to be excessive (129:12) { 0:2048 }
- int **stream_tcp.small_segments.maximum_size** = 0: minimum bytes for a TCP segment not to be considered small (129:12) { 0:2048 }
- int **stream_tcp.session_timeout** = 180: session tracking timeout { 1:max31 }
- bool **stream_tcp.track_only** = false: disable reassembly if true
- int **stream_tcp.embryonic_timeout** = 30: Non-established connection timeout { 1:max31 }
- int **stream_tcp.idle_timeout** = 3600: session deletion on idle { 1:max31 }

Rules:

- **129:1** (stream_tcp) SYN on established session
- **129:2** (stream_tcp) data on SYN packet
- **129:3** (stream_tcp) data sent on stream not accepting data
- **129:4** (stream_tcp) TCP timestamp is outside of PAWS window
- **129:5** (stream_tcp) bad segment, adjusted size ⇐ 0 (deprecated)
- **129:6** (stream_tcp) window size (after scaling) larger than policy allows

- **129:7** (stream_tcp) limit on number of overlapping TCP packets reached
- **129:8** (stream_tcp) data sent on stream after TCP reset sent
- **129:9** (stream_tcp) TCP client possibly hijacked, different ethernet address
- **129:10** (stream_tcp) TCP server possibly hijacked, different ethernet address
- **129:11** (stream_tcp) TCP data with no TCP flags set
- **129:12** (stream_tcp) consecutive TCP small segments exceeding threshold
- **129:13** (stream_tcp) 4-way handshake detected
- **129:14** (stream_tcp) TCP timestamp is missing
- **129:15** (stream_tcp) reset outside window
- **129:16** (stream_tcp) FIN number is greater than prior FIN
- **129:17** (stream_tcp) ACK number is greater than prior FIN
- **129:18** (stream_tcp) data sent on stream after TCP reset received
- **129:19** (stream_tcp) TCP window closed before receiving data
- **129:20** (stream_tcp) TCP session without 3-way handshake

Peg counts:

- **stream_tcp.sessions**: total tcp sessions (sum)
- **stream_tcp.max**: max tcp sessions (max)
- **stream_tcp.created**: tcp session trackers created (sum)
- **stream_tcp.released**: tcp session trackers released (sum)
- **stream_tcp.timeouts**: tcp session timeouts (sum)
- **stream_tcp.prunes**: tcp session prunes (sum)
- **stream_tcp.instantiated**: new sessions instantiated (sum)
- **stream_tcp.setups**: session initializations (sum)
- **stream_tcp.restarts**: sessions restarted (sum)
- **stream_tcp.resyns**: SYN received on established session (sum)
- **stream_tcp.discards**: tcp packets discarded (sum)
- **stream_tcp.discards_skipped**: tcp packet discards skipped due to normalization disabled (sum)
- **stream_tcp.invalid_seq_num**: tcp packets received with an invalid sequence number (sum)
- **stream_tcp.invalid_ack**: tcp packets received with an invalid ack number (sum)
- **stream_tcp.no_flags_set**: tcp packets received with no TCP flags set (sum)
- **stream_tcp.events**: events generated (sum)
- **stream_tcp.ignored**: tcp packets ignored (sum)
- **stream_tcp.untracked**: tcp packets not tracked (sum)
- **stream_tcp.syn_trackers**: tcp session tracking started on syn (sum)

- **stream_tcp.syn_ack_trackers**: tcp session tracking started on syn-ack (sum)
- **stream_tcp.three_way_trackers**: tcp session tracking started on ack (sum)
- **stream_tcp.data_trackers**: tcp session tracking started on data (sum)
- **stream_tcp.segs_queued**: total segments queued (sum)
- **stream_tcp.segs_released**: total segments released (sum)
- **stream_tcp.segs_split**: tcp segments split when reassembling PDUs (sum)
- **stream_tcp.segs_used**: queued tcp segments applied to reassembled PDUs (sum)
- **stream_tcp.rebuilt_packets**: total reassembled PDUs (sum)
- **stream_tcp.rebuilt_buffers**: rebuilt PDU sections (sum)
- **stream_tcp.rebuilt_bytes**: total rebuilt bytes (sum)
- **stream_tcp.overlaps**: overlapping segments queued (sum)
- **stream_tcp.gaps**: missing data between PDUs (sum)
- **stream_tcp.exceeded_max_segs**: number of times the maximum queued segment limit was reached (sum)
- **stream_tcp.exceeded_max_bytes**: number of times the maximum queued byte limit was reached (sum)
- **stream_tcp.payload_fully_trimmed**: segments with no data after trimming (sum)
- **stream_tcp.internal_events**: 135:X events generated (sum)
- **stream_tcp.client_cleanups**: number of times data from server was flushed when session released (sum)
- **stream_tcp.server_cleanups**: number of times data from client was flushed when session released (sum)
- **stream_tcp.memory**: current memory in use (now)
- **stream_tcp.initializing**: number of sessions currently initializing (now)
- **stream_tcp.established**: number of sessions currently established (now)
- **stream_tcp.closing**: number of sessions currently closing (now)
- **stream_tcp.syns**: number of syn packets (sum)
- **stream_tcp.syn_acks**: number of syn-ack packets (sum)
- **stream_tcp.resets**: number of reset packets (sum)
- **stream_tcp.fins**: number of fin packets (sum)
- **stream_tcp.meta_acks**: number of meta acks processed (sum)
- **stream_tcp.packets_held**: number of packets held (sum)
- **stream_tcp.held_packet_rexmits**: number of retransmits of held packets (sum)
- **stream_tcp.held_packets_dropped**: number of held packets dropped (sum)
- **stream_tcp.held_packets_passed**: number of held packets passed (sum)
- **stream_tcp.held_packet_timeouts**: number of held packets that timed out (sum)
- **stream_tcp.held_packet_purges**: number of held packets that were purged without flushing (sum)
- **stream_tcp.held_packet_retries**: number of held packets that were added to the retry queue (sum)
- **stream_tcp.cur_packets_held**: number of packets currently held (now)

- **stream_tcp.max_packets_held**: maximum number of packets held simultaneously (max)
- **stream_tcp.partial_flushes**: number of partial flushes initiated (sum)
- **stream_tcp.partial_flush_bytes**: partial flush total bytes (sum)
- **stream_tcp.inspector_fallbacks**: count of fallbacks from assigned service inspector (sum)
- **stream_tcp.partial_fallbacks**: count of fallbacks from assigned service stream splitter (sum)
- **stream_tcp.max_segs**: maximum number of segments queued in any flow (max)
- **stream_tcp.max_bytes**: maximum number of bytes queued in any flow (max)
- **stream_tcp.zero_len_tcp_opt**: number of zero length tcp options (sum)
- **stream_tcp.zero_win_probes**: number of tcp zero window probes (sum)

5.52 stream_udp

Help: stream inspector for UDP flow tracking

Type: inspector (stream)

Usage: inspect

Instance Type: multiton

Configuration:

- int **stream_udp.session_timeout** = 30: session tracking timeout { 1:max31 }

Peg counts:

- **stream_udp.sessions**: total udp sessions (sum)
- **stream_udp.max**: max udp sessions (max)
- **stream_udp.created**: udp session trackers created (sum)
- **stream_udp.released**: udp session trackers released (sum)
- **stream_udp.timeouts**: udp session timeouts (sum)
- **stream_udp.prunes**: udp session prunes (sum)
- **stream_udp.total_bytes**: total number of bytes processed (sum)
- **stream_udp.ignored**: udp packets ignored (sum)

5.53 stream_user

Help: stream inspector for user flow tracking and reassembly

Type: inspector (stream)

Usage: inspect

Instance Type: multiton

Configuration:

- int **stream_user.session_timeout** = 60: session tracking timeout { 1:max31 }

5.54 telnet

Help: telnet inspection and normalization

Type: inspector (service)

Usage: inspect

Instance Type: multiton

Configuration:

- int **telnet.ayt_attack_thresh** = -1: alert beyond this number of consecutive Telnet AYT commands (-1 is disabled) { -1:max31 }
- bool **telnet.check_encrypted** = false: check for end of encryption
- bool **telnet.encrypted_traffic** = false: check for encrypted Telnet
- bool **telnet.normalize** = false: eliminate escape sequences

Rules:

- **126:1** (telnet) consecutive Telnet AYT commands beyond threshold
- **126:2** (telnet) Telnet traffic encrypted
- **126:3** (telnet) Telnet subnegotiation begin command without subnegotiation end

Peg counts:

- **telnet.total_packets**: total packets (sum)
- **telnet.concurrent_sessions**: total concurrent Telnet sessions (now)
- **telnet.max_concurrent_sessions**: maximum concurrent Telnet sessions (max)

5.55 wizard

Help: inspector that implements port-independent protocol identification

Type: inspector (wizard)

Usage: inspect

Instance Type: multiton

Configuration:

- string **`wizard.hexes[].service`**: name of service
- select **`wizard.hexes[].proto`** = *any*: protocol to scan { tcp | udp | any }
- string **`wizard.hexes[].to_server[].hex`**: sequence of data with wild chars (?)
- string **`wizard.hexes[].to_client[].hex`**: sequence of data with wild chars (?)
- string **`wizard.spells[].service`**: name of service
- select **`wizard.spells[].proto`** = *any*: protocol to scan { tcp | udp | any }
- string **`wizard.spells[].to_server[].spell`**: sequence of data with wild cards (*)
- string **`wizard.spells[].to_client[].spell`**: sequence of data with wild cards (*)

- multi **wizard.curses**: enable service identification based on internal algorithm { dce_smb | dce_udp | dce_tcp | mms | s7commplus | sslv2 }
- int **wizard.max_search_depth** = 8192: maximum scan depth per flow { 0:65535 }

Peg counts:

- **wizard.tcp_scans**: tcp payload scans (sum)
- **wizard.tcp_hits**: tcp identifications (sum)
- **wizard.tcp_misses**: tcp searches abandoned (sum)
- **wizard.udp_scans**: udp payload scans (sum)
- **wizard.udp_hits**: udp identifications (sum)
- **wizard.udp_misses**: udp searches abandoned (sum)
- **wizard.user_scans**: user payload scans (sum)
- **wizard.user_hits**: user identifications (sum)
- **wizard.user_misses**: user searches abandoned (sum)

6 IPS Action Modules

IPS actions allow you to perform custom actions when events are generated. Unlike loggers, these are invoked before thresholding and can be used to control external agents.

Externally defined actions must be configured to become available to the parser. For the reject rule, you can set reject = { } to get the rule to parse.

6.1 react

Help: send response to client and terminate session

Type: ips_action

Usage: detect

Configuration:

- string **react.page**: file containing HTTP response body

6.2 reject

Help: terminate session with TCP reset or ICMP unreachable

Type: ips_action

Usage: detect

Configuration:

- enum **reject.reset** = *both*: send TCP reset to one or both ends { *none|source|dest|both* }
- enum **reject.control** = *none*: send ICMP unreachable(s) { *none|network|host|port|forward|all* }

7 IPS Option Modules

IPS options are the building blocks of IPS rules.

7.1 ack

Help: rule option to match on TCP ack numbers

Type: ips_option

Usage: detect

Configuration:

- interval **ack.~range**: check if TCP ack value is *value* | *min*<>*max* | <*max* | >*min* { 0: }

7.2 appids

Help: detection option for application ids

Type: ips_option

Usage: detect

Configuration:

- string **appids.~**: comma separated list of application names

7.3 base64_decode

Help: rule option to decode base64 data - must be used with base64_data option

Type: ips_option

Usage: detect

Configuration:

- int **base64_decode.bytes**: number of base64 encoded bytes to decode { 1:max32 }
- int **base64_decode.offset** = 0: bytes past start of buffer to start decoding { 0:max32 }
- implied **base64_decode.relative**: apply offset to cursor instead of start of buffer

7.4 ber_data

Help: rule option to move to the data for a specified BER element

Type: ips_option

Usage: detect

Configuration:

- int **ber_data.~type**: move to the data for the specified BER element type { 0:255 }

7.5 ber_skip

Help: rule option to skip BER element

Type: ips_option

Usage: detect

Configuration:

- int **ber_skip.~type**: BER element type to skip { 0:255 }
- implied **ber_skip.optional**: match even if the specified BER type is not found

7.6 bufferlen

Help: rule option to check length of current buffer

Type: ips_option

Usage: detect

Configuration:

- interval **bufferlen.~range**: check that total length of current buffer is in given range { 0:65535 }
- implied **bufferlen.relative**: use remaining length (from current position) instead of total length

7.7 byte_extract

Help: rule option to convert data to an integer variable

Type: ips_option

Usage: detect

Configuration:

- int **byte_extract.~count**: number of bytes to pick up from the buffer (string can pick less) { 1:10 }
- int **byte_extract.~offset**: number of bytes into the buffer to start processing { -65535:65535 }
- string **byte_extract.~name**: name of the variable that will be used in other rule options
- implied **byte_extract.relative**: offset from cursor instead of start of buffer
- int **byte_extract.multiplier** = 1: scale extracted value by given amount { 1:65535 }
- int **byte_extract.align** = 0: round the number of converted bytes up to the next 2- or 4-byte boundary { 0:4 }
- implied **byte_extract.big**: big endian
- implied **byte_extract.little**: little endian
- implied **byte_extract.dce**: dcerpc2 determines endianness
- implied **byte_extract.string**: convert from string
- implied **byte_extract.hex**: convert from hex string
- implied **byte_extract.oct**: convert from octal string
- implied **byte_extract.dec**: convert from decimal string
- int **byte_extract.bitmask**: applies as an AND to the extracted value before storage in *name* { 0x1:0xFFFFFFFF }

7.8 byte_jump

Help: rule option to move the detection cursor

Type: ips_option

Usage: detect

Configuration:

- int **byte_jump.~count**: number of bytes to pick up from the buffer (string can pick less) { 0:10 }
- string **byte_jump.~offset**: variable name or number of bytes into the buffer to start processing
- implied **byte_jump.relative**: offset from cursor instead of start of buffer
- implied **byte_jump.from_beginning**: jump from start of buffer instead of cursor
- implied **byte_jump.from_end**: jump backward from end of buffer
- int **byte_jump.multiplier** = 1: scale extracted value by given amount { 1:65535 }
- int **byte_jump.align** = 0: round the number of converted bytes up to the next 2- or 4-byte boundary { 0:4 }
- string **byte_jump.post_offset**: skip forward or backward (positive or negative value) by variable name or number of bytes after the other jump options have been applied
- implied **byte_jump.big**: big endian
- implied **byte_jump.little**: little endian
- implied **byte_jump.dce**: dcerpc2 determines endianness
- implied **byte_jump.string**: convert from string
- implied **byte_jump.hex**: convert from hex string
- implied **byte_jump.oct**: convert from octal string
- implied **byte_jump.dec**: convert from decimal string
- int **byte_jump.bitmask**: applies as an AND prior to evaluation { 0x1:0xFFFFFFFF }

7.9 byte_math

Help: rule option to perform mathematical operations on extracted value and a specified value or existing variable

Type: ips_option

Usage: detect

Configuration:

- int **byte_math.bytes**: number of bytes to pick up from the buffer (string can pick less) { 1:10 }
- string **byte_math.offset**: number of bytes into the buffer to start processing
- enum **byte_math.oper**: mathematical operation to perform { +|-|*|/|<<|>> }
- string **byte_math.rvalue**: value to use mathematical operation against
- string **byte_math.result**: name of the variable to store the result
- implied **byte_math.relative**: offset from cursor instead of start of buffer
- enum **byte_math.endian**: specify big/little endian { *big*|*little* }
- implied **byte_math.dce**: dcerpc2 determines endianness
- enum **byte_math.string**: convert extracted string to dec/hex/oct { *hex*|*dec*|*oct* }
- int **byte_math.bitmask**: applies as bitwise AND to the extracted value before storage in *name* { 0x1:0xFFFFFFFF }

7.10 byte_test

Help: rule option to convert data to integer and compare

Type: ips_option

Usage: detect

Configuration:

- int **byte_test.~count**: number of bytes to pick up from the buffer (string can pick less) { 1:10 }
- string **byte_test.~operator**: operation to perform to test the value
- string **byte_test.~compare**: variable name or value to test the converted result against
- string **byte_test.~offset**: variable name or number of bytes into the payload to start processing
- implied **byte_test.relative**: offset from cursor instead of start of buffer
- implied **byte_test.big**: big endian
- implied **byte_test.little**: little endian
- implied **byte_test.dce**: dcerpc2 determines endianness
- implied **byte_test.string**: convert from string
- implied **byte_test.hex**: convert from hex string
- implied **byte_test.oct**: convert from octal string
- implied **byte_test.dec**: convert from decimal string
- int **byte_test.bitmask**: applies as an AND prior to evaluation { 0x1:0xFFFFFFFF }

7.11 cip_attribute

Help: detection option to match CIP attribute

Type: ips_option

Usage: detect

Configuration:

- interval **cip_attribute.~range**: match CIP attribute { 0:65535 }

7.12 cip_class

Help: detection option to match CIP class

Type: ips_option

Usage: detect

Configuration:

- interval **cip_class.~range**: match CIP class { 0:65535 }

7.13 cip_conn_path_class

Help: detection option to match CIP Connection Path Class

Type: ips_option

Usage: detect

Configuration:

- interval **cip_conn_path_class.~range**: match CIP Connection Path Class { 0:65535 }

7.14 cip_instance

Help: detection option to match CIP instance

Type: ips_option

Usage: detect

Configuration:

- interval **cip_instance.~range**: match CIP instance { 0:4294967295 }

7.15 cip_req

Help: detection option to match CIP request

Type: ips_option

Usage: detect

7.16 cip_rsp

Help: detection option to match CIP response

Type: ips_option

Usage: detect

7.17 cip_service

Help: detection option to match CIP service

Type: ips_option

Usage: detect

Configuration:

- interval **cip_service.~range**: match CIP service { 0:127 }

7.18 cip_status

Help: detection option to match CIP response status

Type: ips_option

Usage: detect

Configuration:

- interval **cip_status.~range**: match CIP response status { 0:255 }

7.19 classtype

Help: general rule option for rule classification

Type: ips_option

Usage: detect

Configuration:

- string **classtype.~**: classification for this rule

7.20 content

Help: payload rule option for basic pattern matching

Type: ips_option

Usage: detect

Configuration:

- string **content.~data**: data to match
- implied **content.nocase**: case insensitive match
- implied **content.fast_pattern**: use this content in the fast pattern matcher instead of the content selected by default
- int **content.fast_pattern_offset** = 0: number of leading characters of this content the fast pattern matcher should exclude { 0:65535 }
- int **content.fast_pattern_length**: maximum number of characters from this content the fast pattern matcher should use { 1:65535 }
- string **content.offset**: var or number of bytes from start of buffer to start search
- string **content.depth**: var or maximum number of bytes to search from beginning of buffer
- string **content.distance**: var or number of bytes from cursor to start search
- string **content.within**: var or maximum number of bytes to search from cursor

7.21 cvs

Help: payload rule option for detecting specific attacks

Type: ips_option

Usage: detect

Configuration:

- implied **cvs.invalid-entry**: looks for an invalid Entry string

7.22 dce_iface

Help: detection option to check dcerpc interface

Type: ips_option

Usage: detect

Configuration:

- string **dce_iface.uuid**: match given dcerpc uuid
- interval **dce_iface.version**: interface version { 0: }
- implied **dce_iface.any_frag**: match on any fragment

7.23 dce_opnum

Help: detection option to check dcerpc operation number

Type: ips_option

Usage: detect

Configuration:

- string **dce_opnum.~**: match given dcerpc operation number, range or list

7.24 dce_stub_data

Help: sets the cursor to dcerpc stub data

Type: ips_option

Usage: detect

7.25 detection_filter

Help: rule option to require multiple hits before a rule generates an event

Type: ips_option

Usage: detect

Configuration:

- enum **detection_filter.track**: track hits by source or destination IP address { *by_src* | *by_dst* }
- int **detection_filter.count**: hits in interval before allowing the rule to fire { 1:max32 }
- int **detection_filter.seconds**: length of interval to count hits { 1:max32 }

7.26 dnp3_data

Help: sets the cursor to dnp3 data

Type: ips_option

Usage: detect

7.27 dnp3_func

Help: detection option to check DNP3 function code

Type: ips_option

Usage: detect

Configuration:

- string **dnp3_func.~**: match DNP3 function code or name

7.28 dnp3_ind

Help: detection option to check DNP3 indicator flags

Type: ips_option

Usage: detect

Configuration:

- string **dnp3_ind.~**: match given DNP3 indicator flags

7.29 dnp3_obj

Help: detection option to check DNP3 object headers

Type: ips_option

Usage: detect

Configuration:

- int **dnp3_obj.group** = 0: match given DNP3 object header group { 0:255 }
- int **dnp3_obj.var** = 0: match given DNP3 object header var { 0:255 }

7.30 dsize

Help: rule option to test payload size

Type: ips_option

Usage: detect

Configuration:

- interval **dsize.~range**: check if packet payload size is in the given range { 0:65535 }

7.31 enable

Help: stub rule option to enable or disable full rule

Type: ips_option

Usage: detect

Configuration:

- enum **enable.~enable** = *yes*: enable or disable rule in current ips policy or use default defined by ips policy { *no* | *yes* | *inherit* }

7.32 enip_command

Help: detection option to match CIP Enip Command

Type: ips_option

Usage: detect

Configuration:

- interval **enip_command.~range**: match CIP Enip Command { 0:65535 }

7.33 enip_req

Help: detection option to match ENIP Request

Type: ips_option

Usage: detect

7.34 enip_rsp

Help: detection option to match ENIP response

Type: ips_option

Usage: detect

7.35 file_data

Help: rule option to set detection cursor to file data

Type: ips_option

Usage: detect

7.36 file_meta

Help: rule option to set file metadata (file type and id)

Type: ips_option

Usage: detect

Configuration:

- string **file_meta.type**: file type to set
- int **file_meta.id**: file type id { 1:1023 }
- string **file_meta.category**: file type category
- string **file_meta.group**: comma separated list of groups associated with file type
- string **file_meta.version**: file type version

7.37 file_type

Help: rule option to check file type

Type: ips_option

Usage: detect

Configuration:

- string **file_type.~**: list of file type IDs to match

7.38 flags

Help: rule option to test TCP control flags

Type: ips_option

Usage: detect

Configuration:

- string **flags.~test_flags**: these flags are tested
- string **flags.~mask_flags**: these flags are don't cares

7.39 flow

Help: rule option to check session properties

Type: ips_option

Usage: detect

Configuration:

- implied **flow.to_client**: match on server responses
- implied **flow.to_server**: match on client requests
- implied **flow.from_client**: same as to_server
- implied **flow.from_server**: same as to_client
- implied **flow.established**: match only during data transfer phase
- implied **flow.not_established**: match only outside data transfer phase
- implied **flow.stateless**: match regardless of stream state
- implied **flow.no_stream**: match on raw packets only
- implied **flow.only_stream**: match on reassembled packets only
- implied **flow.no_frag**: match on raw packets only
- implied **flow.only_frag**: match on defragmented packets only

7.40 flowbits

Help: rule option to set and test arbitrary boolean flags

Type: ips_option

Usage: detect

Configuration:

- enum **flowbits.~op**: bit operation or noalert (no bits) { *set* | *unset* | *isset* | *isnotset* | *noalert* }
- string **flowbits.~bits**: bit [|bit]* or bit [&bit]*

7.41 fragbits

Help: rule option to test IP frag flags

Type: ips_option

Usage: detect

Configuration:

- string **fragbits.~flags**: these flags are tested

7.42 fragoffset

Help: rule option to test IP frag offset

Type: ips_option

Usage: detect

Configuration:

- interval **fragoffset.~range**: check if ip fragment offset is in given range { 0:8192 }

7.43 gid

Help: rule option specifying rule generator

Type: ips_option

Usage: detect

Configuration:

- int **gid.~**: generator id { 1:8129 }

7.44 gtp_info

Help: rule option to check gtp info element

Type: ips_option

Usage: detect

Configuration:

- string **gtp_info.~**: info element to match

7.45 gtp_type

Help: rule option to check gtp types

Type: ips_option

Usage: detect

Configuration:

- string **gtp_type.~**: list of types to match

7.46 gtp_version

Help: rule option to check GTP version

Type: ips_option

Usage: detect

Configuration:

- int **gtp_version.~**: version to match { 0:2 }

7.47 http_client_body

Help: rule option to set the detection cursor to the request body

Type: ips_option

Usage: detect

7.48 http_cookie

Help: rule option to set the detection cursor to the HTTP cookie

Type: ips_option

Usage: detect

Configuration:

- implied **http_cookie.request**: match against the cookie from the request message even when examining the response
- implied **http_cookie.with_header**: option is no longer used and will be removed in a future release
- implied **http_cookie.with_body**: option is no longer used and will be removed in a future release
- implied **http_cookie.with_trailer**: option is no longer used and will be removed in a future release

7.49 http_header

Help: rule option to set the detection cursor to the normalized headers

Type: ips_option

Usage: detect

Configuration:

- string **http_header.field**: restrict to given header. Header name is case insensitive.
- implied **http_header.request**: match against the headers from the request message even when examining the response
- implied **http_header.with_header**: option is no longer used and will be removed in a future release
- implied **http_header.with_body**: option is no longer used and will be removed in a future release
- implied **http_header.with_trailer**: option is no longer used and will be removed in a future release

7.50 http_header_test

Help: rule option to perform range check on specified header field, check whether it is a number, or check if the field is absent

Type: ips_option

Usage: detect

Configuration:

- string **http_header_test.field**: Header to perform check on. Header name is case insensitive.
- implied **http_header_test.request**: match against the headers from the request message even when examining the response
- implied **http_header_test.with_header**: option is no longer used and will be removed in a future release
- implied **http_header_test.with_body**: option is no longer used and will be removed in a future release
- implied **http_header_test.with_trailer**: option is no longer used and will be removed in a future release
- interval **http_header_test.check**: range check to perform on header value { 0:999999999999999999 }
- bool **http_header_test.numeric**: header value is a number
- implied **http_header_test.absent**: header is absent

7.51 http_max_header_line

Help: rule option to perform range check on longest header line

Type: ips_option

Usage: detect

Configuration:

- interval **http_max_header_line.~range**: check that longest line of current header is in given range { 0:65535 }
- implied **http_max_header_line.request**: match against the version from the request message even when examining the response

7.52 http_max_trailer_line

Help: rule option to perform range check on longest trailer line

Type: ips_option

Usage: detect

Configuration:

- interval **http_max_trailer_line.~range**: check that longest line of current trailer is in given range { 0:65535 }
- implied **http_max_trailer_line.request**: match against the version from the request message even when examining the response

7.53 http_method

Help: rule option to set the detection cursor to the HTTP request method

Type: ips_option

Usage: detect

Configuration:

- implied **http_method.with_header**: option is no longer used and will be removed in a future release
- implied **http_method.with_body**: option is no longer used and will be removed in a future release
- implied **http_method.with_trailer**: option is no longer used and will be removed in a future release

7.54 http_num_cookies

Help: rule option to perform range check on number of cookies

Type: ips_option

Usage: detect

Configuration:

- interval **http_num_cookies.~range**: check that number of cookies of current header are in given range { 0:65535 }
- implied **http_num_cookies.request**: match against the version from the request message even when examining the response

7.55 http_num_headers

Help: rule option to perform range check on number of headers

Type: ips_option

Usage: detect

Configuration:

- interval **http_num_headers.~range**: check that number of headers of current buffer are in given range { 0:65535 }
- implied **http_num_headers.request**: match against the version from the request message even when examining the response
- implied **http_num_headers.with_header**: option is no longer used and will be removed in a future release
- implied **http_num_headers.with_body**: option is no longer used and will be removed in a future release
- implied **http_num_headers.with_trailer**: option is no longer used and will be removed in a future release

7.56 http_num_trailers

Help: rule option to perform range check on number of trailers

Type: ips_option

Usage: detect

Configuration:

- interval **http_num_trailers.~range**: check that number of headers of current buffer are in given range { 0:65535 }
- implied **http_num_trailers.request**: match against the version from the request message even when examining the response
- implied **http_num_trailers.with_header**: option is no longer used and will be removed in a future release
- implied **http_num_trailers.with_body**: option is no longer used and will be removed in a future release
- implied **http_num_trailers.with_trailer**: option is no longer used and will be removed in a future release

7.57 http_param

Help: rule option to set the detection cursor to the value of the specified HTTP parameter key which may be in the query or body

Type: ips_option

Usage: detect

Configuration:

- string **http_param.~param**: parameter to match
- implied **http_param.nocase**: case insensitive match

7.58 http_raw_body

Help: rule option to set the detection cursor to the unnormalized message body

Type: ips_option

Usage: detect

7.59 http_raw_cookie

Help: rule option to set the detection cursor to the unnormalized cookie

Type: ips_option

Usage: detect

Configuration:

- implied **http_raw_cookie.request**: match against the cookie from the request message even when examining the response
- implied **http_raw_cookie.with_header**: option is no longer used and will be removed in a future release
- implied **http_raw_cookie.with_body**: option is no longer used and will be removed in a future release
- implied **http_raw_cookie.with_trailer**: option is no longer used and will be removed in a future release

7.60 http_raw_header

Help: rule option to set the detection cursor to the unnormalized headers

Type: ips_option

Usage: detect

Configuration:

- string **http_raw_header.field**: restrict to given header. Header name is case insensitive.
- implied **http_raw_header.request**: match against the headers from the request message even when examining the response
- implied **http_raw_header.with_header**: option is no longer used and will be removed in a future release
- implied **http_raw_header.with_body**: option is no longer used and will be removed in a future release
- implied **http_raw_header.with_trailer**: option is no longer used and will be removed in a future release

7.61 http_raw_request

Help: rule option to set the detection cursor to the unnormalized request line

Type: ips_option

Usage: detect

Configuration:

- implied **http_raw_request.with_header**: option is no longer used and will be removed in a future release
- implied **http_raw_request.with_body**: option is no longer used and will be removed in a future release
- implied **http_raw_request.with_trailer**: option is no longer used and will be removed in a future release

7.62 http_raw_status

Help: rule option to set the detection cursor to the unnormalized status line

Type: ips_option

Usage: detect

Configuration:

- implied **http_raw_status.with_body**: option is no longer used and will be removed in a future release
- implied **http_raw_status.with_trailer**: option is no longer used and will be removed in a future release

7.63 http_raw_trailer

Help: rule option to set the detection cursor to the unnormalized trailers

Type: ips_option

Usage: detect

Configuration:

- string **http_raw_trailer.field**: restrict to given trailer. Trailer name is case insensitive.
- implied **http_raw_trailer.request**: match against the trailers from the request message even when examining the response
- implied **http_raw_trailer.with_header**: option is no longer used and will be removed in a future release
- implied **http_raw_trailer.with_body**: option is no longer used and will be removed in a future release

7.64 http_raw_uri

Help: rule option to set the detection cursor to the unnormalized URI

Type: ips_option

Usage: detect

Configuration:

- implied **http_raw_uri.with_header**: option is no longer used and will be removed in a future release
- implied **http_raw_uri.with_body**: option is no longer used and will be removed in a future release
- implied **http_raw_uri.with_trailer**: option is no longer used and will be removed in a future release

- implied **http_raw_uri.scheme**: match against scheme section of URI only
- implied **http_raw_uri.host**: match against host section of URI only
- implied **http_raw_uri.port**: match against port section of URI only
- implied **http_raw_uri.path**: match against path section of URI only
- implied **http_raw_uri.query**: match against query section of URI only
- implied **http_raw_uri.fragment**: match against fragment section of URI only

7.65 http_stat_code

Help: rule option to set the detection cursor to the HTTP status code

Type: ips_option

Usage: detect

Configuration:

- implied **http_stat_code.with_body**: option is no longer used and will be removed in a future release
- implied **http_stat_code.with_trailer**: option is no longer used and will be removed in a future release

7.66 http_stat_msg

Help: rule option to set the detection cursor to the HTTP status message

Type: ips_option

Usage: detect

Configuration:

- implied **http_stat_msg.with_body**: option is no longer used and will be removed in a future release
- implied **http_stat_msg.with_trailer**: option is no longer used and will be removed in a future release

7.67 http_trailer

Help: rule option to set the detection cursor to the normalized trailers

Type: ips_option

Usage: detect

Configuration:

- string **http_trailer.field**: restrict to given trailer
- implied **http_trailer.request**: match against the trailers from the request message even when examining the response
- implied **http_trailer.with_header**: option is no longer used and will be removed in a future release
- implied **http_trailer.with_body**: option is no longer used and will be removed in a future release

7.68 http_trailer_test

Help: rule option to perform range check on specified trailer field, check whether it is a number, or check if the field is absent

Type: ips_option

Usage: detect

Configuration:

- string **http_trailer_test.field**: Trailer to perform check on. Trailer name is case insensitive.
- implied **http_trailer_test.request**: match against the trailers from the request message even when examining the response
- implied **http_trailer_test.with_header**: option is no longer used and will be removed in a future release
- implied **http_trailer_test.with_body**: option is no longer used and will be removed in a future release
- interval **http_trailer_test.check**: range check to perform on trailer value { 0:999999999999999999 }
- bool **http_trailer_test.numeric**: trailer value is a number
- implied **http_trailer_test.absent**: trailer is absent

7.69 http_true_ip

Help: rule option to set the detection cursor to the final client IP address

Type: ips_option

Usage: detect

Configuration:

- implied **http_true_ip.with_header**: option is no longer used and will be removed in a future release
- implied **http_true_ip.with_body**: option is no longer used and will be removed in a future release
- implied **http_true_ip.with_trailer**: option is no longer used and will be removed in a future release

7.70 http_uri

Help: rule option to set the detection cursor to the normalized URI buffer

Type: ips_option

Usage: detect

Configuration:

- implied **http_uri.with_header**: option is no longer used and will be removed in a future release
- implied **http_uri.with_body**: option is no longer used and will be removed in a future release
- implied **http_uri.with_trailer**: option is no longer used and will be removed in a future release
- implied **http_uri.scheme**: match against scheme section of URI only
- implied **http_uri.host**: match against host section of URI only
- implied **http_uri.port**: match against port section of URI only
- implied **http_uri.path**: match against path section of URI only
- implied **http_uri.query**: match against query section of URI only
- implied **http_uri.fragment**: match against fragment section of URI only

7.71 http_version

Help: rule option to set the detection cursor to the version buffer

Type: ips_option

Usage: detect

Configuration:

- implied **http_version.request**: match against the version from the request message even when examining the response
- implied **http_version.with_header**: option is no longer used and will be removed in a future release
- implied **http_version.with_body**: option is no longer used and will be removed in a future release
- implied **http_version.with_trailer**: option is no longer used and will be removed in a future release

7.72 http_version_match

Help: rule option to match version to listed values

Type: ips_option

Usage: detect

Configuration:

- string **http_version_match.~version_list**: space-separated list of versions to match
- implied **http_version_match.request**: match against the version from the request message even when examining the response
- implied **http_version_match.with_header**: option is no longer used and will be removed in a future release
- implied **http_version_match.with_body**: option is no longer used and will be removed in a future release
- implied **http_version_match.with_trailer**: option is no longer used and will be removed in a future release

7.73 icmp_id

Help: rule option to check ICMP ID

Type: ips_option

Usage: detect

Configuration:

- interval **icmp_id.~range**: check if ICMP ID is in given range { 0:65535 }

7.74 icmp_seq

Help: rule option to check ICMP sequence number

Type: ips_option

Usage: detect

Configuration:

- interval **icmp_seq.~range**: check if ICMP sequence number is in given range { 0:65535 }

7.75 icode

Help: rule option to check ICMP code

Type: ips_option

Usage: detect

Configuration:

- interval **icode.~range**: check if ICMP code is in given range is { 0:255 }

7.76 id

Help: rule option to check the IP ID field

Type: ips_option

Usage: detect

Configuration:

- interval **id.~range**: check if the IP ID is in the given range { 0: }

7.77 iec104_apci_type

Help: rule option to check iec104 apci type

Type: ips_option

Usage: detect

Configuration:

- string **iec104_apci_type.~**: APCI type to match

7.78 iec104_asdu_func

Help: rule option to check iec104 function code

Type: ips_option

Usage: detect

Configuration:

- string **iec104_asdu_func.~**: function code to match

7.79 ip_proto

Help: rule option to check the IP protocol number

Type: ips_option

Usage: detect

Configuration:

- string **ip_proto.~proto**: [!|>|<] name or number

7.80 ipopts

Help: rule option to check for IP options

Type: ips_option

Usage: detect

Configuration:

- select **ipopts.~opt**: output format { rr|eol|nop|ts|sec|esec|lsrr|lsrre|ssrr|satid|any }

7.81 isdataat

Help: rule option to check for the presence of payload data

Type: ips_option

Usage: detect

Configuration:

- string **isdataat.~length**: num | !num
- implied **isdataat.relative**: offset from cursor instead of start of buffer

7.82 itype

Help: rule option to check ICMP type

Type: ips_option

Usage: detect

Configuration:

- interval **itype.~range**: check if ICMP type is in given range { 0:255 }

7.83 js_data

Help: rule option to set detection cursor to normalized JavaScript data

Type: ips_option

Usage: detect

7.84 md5

Help: payload rule option for hash matching

Type: ips_option

Usage: detect

Configuration:

- string **md5.~hash**: data to match
- int **md5.length**: number of octets in plain text { 1:65535 }
- string **md5.offset**: var or number of bytes from start of buffer to start search
- implied **md5.relative** = *false*: offset from cursor instead of start of buffer

7.85 metadata

Help: rule option for conveying arbitrary comma-separated name, value data within the rule text

Type: ips_option

Usage: detect

Configuration:

- string **`metadata.*`**: comma-separated list of arbitrary name value pairs

7.86 mms_data

Help: rule option to set cursor to MMS data

Type: ips_option

Usage: detect

7.87 mms_func

Help: rule option to check MMS function

Type: ips_option

Usage: detect

Configuration:

- string **mms_func.~**: func to match

7.88 modbus_data

Help: rule option to set cursor to modbus data

Type: ips_option

Usage: detect

7.89 modbus_func

Help: rule option to check modbus function code

Type: ips_option

Usage: detect

Configuration:

- string **modbus_func.~**: function code to match

7.90 modbus_unit

Help: rule option to check Modbus unit ID

Type: ips_option

Usage: detect

Configuration:

- int **modbus_unit.~**: Modbus unit ID { 0:255 }

7.91 msg

Help: rule option summarizing rule purpose output with events

Type: ips_option

Usage: detect

Configuration:

- string **msg.~**: message describing rule

7.92 mss

Help: detection for TCP maximum segment size

Type: ips_option

Usage: detect

Configuration:

- interval **mss.~range**: check if TCP MSS is in given range { 0:65535 }

7.93 pcre

Help: rule option for matching payload data with pcre

Type: ips_option

Usage: detect

Configuration:

- string **pcre.~re**: Snort regular expression

Peg counts:

- **pcre.pcre_rules**: total rules processed with pcre option (sum)
- **pcre.pcre_to_hyper**: total pcre rules by hyperscan engine (sum)
- **pcre.pcre_native**: total pcre rules compiled by pcre engine (sum)
- **pcre.pcre_negated**: total pcre rules using negation syntax (sum)

7.94 pkt_data

Help: rule option to set the detection cursor to the normalized packet data

Type: ips_option

Usage: detect

7.95 pkt_num

Help: alert on raw packet number

Type: ips_option

Usage: detect

Configuration:

- interval **pkt_num.~range**: check if packet number is in given range { 1: }

7.96 priority

Help: rule option for prioritizing events

Type: ips_option

Usage: detect

Configuration:

- int **priority.~**: relative severity level; 1 is highest priority { 1:max31 }

7.97 raw_data

Help: rule option to set the detection cursor to the raw packet data

Type: ips_option

Usage: detect

7.98 reference

Help: rule option to indicate relevant attack identification system

Type: ips_option

Usage: detect

Configuration:

- string **reference.~ref**: reference: <scheme>,<id>

7.99 regex

Help: rule option for matching payload data with hyperscan regex; uses pcre syntax

Type: ips_option

Usage: detect

Configuration:

- string **regex.~re**: hyperscan regular expression
- implied **regex.dotall**: matching a . will not exclude newlines
- implied **regex.fast_pattern**: use this content in the fast pattern matcher instead of the content selected by default
- implied **regex.multiline**: ^ and $ anchors match any newlines in data
- implied **regex.nocase**: case insensitive match
- implied **regex.relative**: start search from end of last match instead of start of buffer

7.100 rem

Help: rule option to convey an arbitrary comment in the rule body

Type: ips_option

Usage: detect

Configuration:

- string **rem.~**: comment

7.101 replace

Help: rule option to overwrite payload data; use with "rewrite" action; works for raw packets only

Type: ips_option

Usage: detect

Configuration:

- string **replace.~**: byte code to replace with

7.102 rev

Help: rule option to indicate current revision of signature

Type: ips_option

Usage: detect

Configuration:

- int **rev.~**: revision { 1:max32 }

7.103 rpc

Help: rule option to check SUNRPC CALL parameters

Type: ips_option

Usage: detect

Configuration:

- int **rpc.~app**: application number { 0:max32 }
- string **rpc.~ver**: version number or * for any
- string **rpc.~proc**: procedure number or * for any

7.104 s7commplus_content

Help: rule option to set cursor to s7commplus content

Type: ips_option

Usage: detect

7.105 s7commplus_func

Help: rule option to check s7commplus function code

Type: ips_option

Usage: detect

Configuration:

- string **s7commplus_func.~**: function code to match

7.106 s7commplus_opcode

Help: rule option to check s7commplus opcode code

Type: ips_option

Usage: detect

Configuration:

- string **s7commplus_opcode.~**: opcode code to match

7.107 sd_pattern

Help: rule option for detecting sensitive data

Type: ips_option

Usage: detect

Configuration:

- string **sd_pattern.~pattern**: The pattern to search for
- int **sd_pattern.threshold** = 1: number of matches before alerting { 1:max32 }

Peg counts:

- **sd_pattern.below_threshold**: sd_pattern matched but missed threshold (sum)
- **sd_pattern.pattern_not_found**: sd_pattern did not not match (sum)
- **sd_pattern.terminated**: hyperscan terminated (sum)

7.108 seq

Help: rule option to check TCP sequence number

Type: ips_option

Usage: detect

Configuration:

- interval **seq.~range**: check if TCP sequence number is in given range { 0: }

7.109 service

Help: rule option to specify list of services for grouping rules

Type: ips_option

Usage: detect

Configuration:

- string **`service.*`**: one or more comma-separated service names

7.110 sha256

Help: payload rule option for hash matching

Type: ips_option

Usage: detect

Configuration:

- string **sha256.~hash**: data to match
- int **sha256.length**: number of octets in plain text { 1:65535 }
- string **sha256.offset**: var or number of bytes from start of buffer to start search
- implied **sha256.relative** = *false*: offset from cursor instead of start of buffer

7.111 sha512

Help: payload rule option for hash matching

Type: ips_option

Usage: detect

Configuration:

- string **sha512.~hash**: data to match
- int **sha512.length**: number of octets in plain text { 1:65535 }
- string **sha512.offset**: var or number of bytes from start of buffer to start search
- implied **sha512.relative** = *false*: offset from cursor instead of start of buffer

7.112 sid

Help: rule option to indicate signature number

Type: ips_option

Usage: detect

Configuration:

- int **sid.~**: signature id { 1:max32 }

7.113 sip_body

Help: rule option to set the detection cursor to the request body

Type: ips_option

Usage: detect

7.114 sip_header

Help: rule option to set the detection cursor to the SIP header buffer

Type: ips_option

Usage: detect

7.115 sip_method

Help: detection option for sip method

Type: ips_option

Usage: detect

Configuration:

- string **`sip_method.*method`**: sip method

7.116 sip_stat_code

Help: detection option for sip stat code

Type: ips_option

Usage: detect

Configuration:

- int **`sip_stat_code.*code`**: status code { 1:999 }

7.117 so

Help: rule option to call custom eval function

Type: ips_option

Usage: detect

Configuration:

- string **so.~func**: name of eval function
- implied **so.relative**: offset from cursor instead of start of buffer

7.118 soid

Help: rule option to specify a shared object rule ID

Type: ips_option

Usage: detect

Configuration:

- string **soid.~**: SO rule ID is unique key, eg <gid>_<sid>_<rev> like 3_45678_9

7.119 ssl_state

Help: detection option for ssl state

Type: ips_option

Usage: detect

Configuration:

- implied **ssl_state.client_hello**: check for client hello

- implied **ssl_state.server_hello**: check for server hello
- implied **ssl_state.client_keyx**: check for client keyx
- implied **ssl_state.server_keyx**: check for server keyx
- implied **ssl_state.unknown**: check for unknown record
- implied **ssl_state.!client_hello**: check for records that are not client hello
- implied **ssl_state.!server_hello**: check for records that are not server hello
- implied **ssl_state.!client_keyx**: check for records that are not client keyx
- implied **ssl_state.!server_keyx**: check for records that are not server keyx
- implied **ssl_state.!unknown**: check for records that are not unknown

7.120 ssl_version

Help: detection option for ssl version

Type: ips_option

Usage: detect

Configuration:

- implied **ssl_version.sslv2**: check for sslv2
- implied **ssl_version.sslv3**: check for sslv3
- implied **ssl_version.tls1.0**: check for tls1.0
- implied **ssl_version.tls1.1**: check for tls1.1
- implied **ssl_version.tls1.2**: check for tls1.2
- implied **ssl_version.!sslv2**: check for records that are not sslv2
- implied **ssl_version.!sslv3**: check for records that are not sslv3
- implied **ssl_version.!tls1.0**: check for records that are not tls1.0
- implied **ssl_version.!tls1.1**: check for records that are not tls1.1
- implied **ssl_version.!tls1.2**: check for records that are not tls1.2

7.121 stream_reassemble

Help: detection option for stream reassembly control

Type: ips_option

Usage: detect

Configuration:

- enum **stream_reassemble.action**: stop or start stream reassembly { *disable|enable* }
- enum **stream_reassemble.direction**: action applies to the given direction(s) { *client|server|both* }
- implied **stream_reassemble.noalert**: don't alert when rule matches
- implied **stream_reassemble.fastpath**: optionally trust the remainder of the session

7.122 stream_size

Help: detection option for stream size checking

Type: ips_option

Usage: detect

Configuration:

- interval **stream_size.~range**: check if the stream size is in the given range { 0: }
- enum **stream_size.~direction**: compare applies to the given direction(s) { *either|to_server|to_client|both* }

7.123 tag

Help: rule option to log additional packets

Type: ips_option

Usage: detect

Configuration:

- enum **tag.~**: log all packets in session or all packets to or from host { *session|host_src|host_dst* }
- int **tag.packets**: tag this many packets { 1:max32 }
- int **tag.seconds**: tag for this many seconds { 1:max32 }
- int **tag.bytes**: tag for this many bytes { 1:max32 }

7.124 target

Help: rule option to indicate target of attack

Type: ips_option

Usage: detect

Configuration:

- enum **target.~**: indicate the target of the attack { *src_ip | dst_ip* }

7.125 tos

Help: rule option to check type of service field

Type: ips_option

Usage: detect

Configuration:

- interval **tos.~range**: check if IP TOS is in given range { 0:255 }

7.126 ttl

Help: rule option to check time to live field

Type: ips_option

Usage: detect

Configuration:

- interval **ttl.~range**: check if IP TTL is in the given range { 0:255 }

7.127 urg

Help: detection for TCP urgent pointer

Type: ips_option

Usage: detect

Configuration:

- interval **urg.~range**: check if tcp urgent offset is in given range { 0:65535 }

7.128 vba_data

Help: rule option to set the detection cursor to the MS Office Visual Basic for Applications macros buffer

Type: ips_option

Usage: detect

7.129 window

Help: rule option to check TCP window field

Type: ips_option

Usage: detect

Configuration:

- interval **window.~range**: check if TCP window size is in given range { 0:65535 }

7.130 wscale

Help: detection for TCP window scale

Type: ips_option

Usage: detect

Configuration:

- interval **wscale.~range**: check if TCP window scale is in given range { 0:65535 }

8 Search Engine Modules

Search engines perform multipattern searching of packets and payload to find rules that should be evaluated. There are currently no specific modules, although there are several search engine plugins. Related configuration is done with the basic detection module.

9 SO Rule Modules

SO rules are dynamic rules that require custom coding to perform detection not possible with the existing rule options. These rules typically do not have associated modules.

10 Logger Modules

All output of events and packets is done by Loggers.

10.1 alert_csv

Help: output event in csv format

Type: logger

Usage: global

Configuration:

- bool **alert_csv.file** = false: output to alert_csv.txt instead of stdout
- multi **alert_csv.fields** = *timestamp pkt_num proto pkt_gen pkt_len dir src_ap dst_ap rule action*: selected fields will be output in given order left to right { action | class | b64_data | client_bytes | client_pkts | dir | dst_addr | dst_ap | dst_port | eth_dst | eth_len | eth_src | eth_type | flowstart_time | geneve_vni | gid | icmp_code | icmp_id | icmp_seq | icmp_type | iface | ip_id | ip_len | msg | mpls | pkt_gen | pkt_len | pkt_num | priority | proto | rev | rule | seconds | server_bytes | server_pkts | service | sgt| sid | src_addr | src_ap | src_port | target | tcp_ack | tcp_flags | tcp_len | tcp_seq | tcp_win | timestamp | tos | ttl | udp_len | vlan }
- int **alert_csv.limit** = 0: set maximum size in MB before rollover (0 is unlimited) { 0:maxSZ }
- string **alert_csv.separator** = ', ': separate fields with this character sequence

10.2 alert_ex

Help: output gid:sid:rev for alerts

Type: logger

Usage: context

Configuration:

- bool **alert_ex.upper** = false: true/false → convert to upper/lower case

10.3 alert_fast

Help: output event with brief text format

Type: logger

Usage: global

Configuration:

- bool **alert_fast.file** = false: output to alert_fast.txt instead of stdout
- bool **alert_fast.packet** = false: output packet dump with alert
- bool **alert_fast.buffers** = false: output IPS buffer dump
- int **alert_fast.buffers_depth** = 0: number of IPS buffer bytes to dump per buffer (0 is unlimited) { 0:maxSZ }
- int **alert_fast.limit** = 0: set maximum size in MB before rollover (0 is unlimited) { 0:maxSZ }

10.4 alert_full

Help: output event with full packet dump

Type: logger

Usage: global

Configuration:

- bool **alert_full.file** = false: output to alert_full.txt instead of stdout
- int **alert_full.limit** = 0: set maximum size in MB before rollover (0 is unlimited) { 0:maxSZ }

10.5 alert_json

Help: output event in json format

Type: logger

Usage: global

Configuration:

- bool **alert_json.file** = false: output to alert_json.txt instead of stdout
- multi **alert_json.fields** = *timestamp pkt_num proto pkt_gen pkt_len dir src_ap dst_ap rule action*: selected fields will be output in given order left to right { action | class | b64_data | client_bytes | client_pkts | dir | dst_addr | dst_ap | dst_port | eth_dst | eth_len | eth_src | eth_type | flowstart_time | geneve_vni | gid | icmp_code | icmp_id | icmp_seq | icmp_type | iface | ip_id | ip_len | msg | mpls | pkt_gen | pkt_len | pkt_num | priority | proto | rev | rule | seconds | server_bytes | server_pkts | service | sgt| sid | src_addr | src_ap | src_port | target | tcp_ack | tcp_flags | tcp_len | tcp_seq | tcp_win | timestamp | tos | ttl | udp_len | vlan }
- int **alert_json.limit** = 0: set maximum size in MB before rollover (0 is unlimited) { 0:maxSZ }
- string **alert_json.separator** = ', ': separate fields with this character sequence

10.6 alert_syslog

Help: output event to syslog

Type: logger

Usage: global

Configuration:

- enum **alert_syslog.facility** = *auth*: part of priority applied to each message { *auth* | *authpriv* | *daemon* | *user* | *local0* | *local1* | *local2* | *local3* | *local4* | *local5* | *local6* | *local7* }
- enum **alert_syslog.level** = *info*: part of priority applied to each message { *emerg* | *alert* | *crit* | *err* | *warning* | *notice* | *info* | *debug* }
- multi **alert_syslog.options**: used to open the syslog connection { cons | ndelay | perror | pid }

10.7 alert_talos

Help: output event in Talos alert format

Type: logger

Usage: global

10.8 alert_unixsock

Help: output event over unix socket

Type: logger

Usage: global

10.9 log_codecs

Help: log protocols in packet by layer

Type: logger

Usage: global

Configuration:

- bool **log_codecs.file** = false: output to log_codecs.txt instead of stdout
- bool **log_codecs.msg** = false: include alert msg

10.10 log_hext

Help: output payload suitable for daq hext

Type: logger

Usage: global

Configuration:

- bool **log_hext.file** = false: output to log_hext.txt instead of stdout
- bool **log_hext.raw** = false: output all full packets if true, else just TCP payload
- int **log_hext.limit** = 0: set maximum size in MB before rollover (0 is unlimited) { 0:maxSZ }
- int **log_hext.width** = 20: set line width (0 is unlimited) { 0:max32 }

10.11 log_pcap

Help: log packet in pcap format

Type: logger

Usage: global

Configuration:

- int **log_pcap.limit** = 0: set maximum size in MB before rollover (0 is unlimited) { 0:maxSZ }

10.12 unified2

Help: output event and packet in unified2 format file

Type: logger

Usage: global

Configuration:

- bool **unified2.legacy_events** = false: generate Snort 2.X style events for barnyard2 compatibility
- int **unified2.limit** = 0: set maximum size in MB before rollover (0 is unlimited) { 0:maxSZ }
- bool **unified2.nostamp** = true: append file creation time to name (in Unix Epoch format)

11 Appendix

11.1 Build Options

The options listed below must be explicitly enabled so they are built into the Snort binary. For a full list of build options, run ./configure --help.

- **--enable-shell**: enable building local and remote command line shell support.
- **--enable-tsc-clock**: use the TSC register on x86 systems for improved performance of latency and profiler features.

These options are built only if the required libraries and headers are present. There is no need to explicitly enable.

- **hyperscan** >= 4.4.0: for the regex and sd_pattern rule options and the hyperscan search engine.
- **iconv**: for converting UTF16-LE filenames to UTF8 (usually included in glibc)
- **libunwind**: for printing a backtrace when a fatal signal is received.
- **lzma**: for decompression of SWF and PDF files.
- **safec**: for additional runtime error checking of some memory copy operations.

If you need to use headers and/or libraries in non-standard locations, you can use these options:

- **--with-pkg-includes**: specify the directory containing the package headers.
- **--with-pkg-libraries**: specify the directory containing the package libraries.

These can be used for pcap, luajit, pcre, dnet, daq, lzma, openssl, iconv, and hyperscan packages. For more information on these libraries see the Getting Started section of the manual.

11.2 Environment Variables

- **HOSTTYPE**: optional string that is output with the version at end of line.
- **SNORT_IGNORE**: the list of symbols Snort should ignore when parsing the Lua conf. Unknown symbols not in SNORT_IGNORE will cause warnings with --warn-unknown or fatals with --warn-unknown --pedantic.
- **SNORT_PROMPT**: the character sequence that is printed at startup, shutdown, and in the shell. The default is the mini-pig: o")~ .
- **SNORT_PLUGIN_PATH**: an optional path where Snort can find supplemental shared libraries. This is only used when Snort is building manuals. Modules in supplemental shared libraries will be added to the manuals.

11.3 Command Line Options

- **-?** <option prefix> output matching command line option quick help (same as --help-options) (optional)
- **-A** <mode> set alert mode: none, cmg, or alert_*
- **-B** <mask> obfuscated IP addresses in alerts and packet dumps using CIDR mask
- **-C** print out payloads with character data only (no hex)
- **-c** <conf> use this configuration
- **-D** run Snort in background (daemon) mode

- **-d** dump the Application Layer
- **-e** display the second layer header info
- **-f** turn off fflush() calls after binary log writes
- **-G** <0xid> (same as --logid) (0:65535)
- **-g** <gname> run snort gid as <gname> group (or gid) after initialization
- **-H** make hash tables deterministic
- **-h** show help overview (same as --help)
- **-i** <iface>… list of interfaces
- **-j** <port> to listen for Telnet connections
- **-k** <mode> checksum mode; default is all (all|noip|notcp|noudp|noicmp|none)
- **-L** <mode> logging mode (none, dump, pcap, or log_*)
- **-l** <logdir> log to this directory instead of current directory
- **-M** log messages to syslog (not alerts)
- **-m** <umask> set the process file mode creation mask (0x000:0x1FF)
- **-n** <count> stop after count packets (0:max53)
- **-O** obfuscate the logged IP addresses
- **-Q** enable inline mode operation
- **-q** quiet mode - suppress normal logging on stdout
- **-R** <rules> include this rules file in the default policy
- **-r** <pcap>… (same as --pcap-list)
- **-s** <snap> (same as --snaplen); default is 1518 (0:65535)
- **-T** test and report on the current Snort configuration
- **-t** <dir> chroots process to <dir> after initialization
- **-U** use UTC for timestamps
- **-u** <uname> run snort as <uname> or <uid> after initialization
- **-V** (same as --version)
- **-v** be verbose
- **-X** dump the raw packet data starting at the link layer
- **-x** same as --pedantic
- **-y** include year in timestamp in the alert and log files
- **-z** <count> maximum number of packet threads (same as --max-packet-threads); 0 gets the number of CPU cores reported by the system; default is 1 (0:max32)
- **--alert-before-pass** evaluate alert rules before pass rules; default is pass rules first
- **--bpf** <filter options> are standard BPF options, as seen in TCPDump
- **--c2x** output hex for given char (see also --x2c)

- **--control-socket** <file> to create unix socket
- **--create-pidfile** create PID file, even when not in Daemon mode
- **--daq** <type> select packet acquisition module (default is pcap)
- **--daq-batch-size** <size> set the DAQ receive batch size; default is 64 (1:)
- **--daq-dir** <dir> tell snort where to find desired DAQ
- **--daq-list** list packet acquisition modules available in optional dir, default is static modules only
- **--daq-mode** <mode> select DAQ module operating mode (overrides automatic selection) (passive | inline | read-file)
- **--daq-var** <name=value> specify extra DAQ configuration variable
- **--dirty-pig** don't flush packets on shutdown
- **--dump-builtin-options** additional options to include with --dump-builtin-rules stubs
- **--dump-builtin-rules** [<module prefix>] output stub rules for selected modules (optional)
- **--dump-config** dump config in json format (all | top)
- **--dump-config-text** dump config in text format
- **--dump-dynamic-rules** output stub rules for all loaded rules libraries
- **--dump-defaults** [<module prefix>] output module defaults in Lua format (optional)
- **--dump-rule-databases** dump rule databases to given directory (hyperscan only)
- **--dump-rule-deps** dump rule dependencies in json format for use by other tools
- **--dump-rule-meta** dump configured rule info in json format for use by other tools
- **--dump-rule-state** dump configured rule state in json format for use by other tools
- **--dump-version** output the version, the whole version, and only the version
- **--enable-inline-test** enable Inline-Test Mode Operation
- **--enable-test-features** enable features used in testing
- **--gen-msg-map** dump configured rules in gen-msg.map format for use by other tools
- **--help** show help overview
- **--help-commands** [<module prefix>] output matching commands (optional)
- **--help-config** [<module prefix>] output matching config options (optional)
- **--help-counts** [<module prefix>] output matching peg counts (optional)
- **--help-limits** print the int upper bounds denoted by max*
- **--help-module** <module> output description of given module
- **--help-modules** list all available modules with brief help
- **--help-modules-json** dump description of all available modules in JSON format
- **--help-options** [<option prefix>] output matching command line option quick help (same as -?) (optional)
- **--help-plugins** list all available plugins with brief help
- **--help-signals** dump available control signals
- **--id-offset** offset to add to instance IDs when logging to files (0:65535)

- **--id-subdir** create/use instance subdirectories in logdir instead of instance filename prefix
- **--id-zero** use id prefix / subdirectory even with one packet thread
- **--include-path** <path> where to find Lua and rule included files; searched before current or config directories
- **--list-buffers** output available inspection buffers
- **--list-builtin** [<module prefix>] output matching builtin rules (optional)
- **--list-gids** [<module prefix>] output matching generators (optional)
- **--list-modules** [<module type>] list all known modules of given type (optional)
- **--list-plugins** list all known plugins
- **--lua** <chunk> extend/override conf with chunk; may be repeated
- **--lua-sandbox** <file> file that contains the lua sandbox environment in which config will be loaded
- **--logid** <0xid> log Identifier to uniquely id events for multiple snorts (same as -G) (0:65535)
- **--markup** output help in asciidoc compatible format
- **--max-packet-threads** <count> configure maximum number of packet threads (same as -z) (0:max32)
- **--mem-check** like -T but also compile search engines
- **--metadata-filter** <filter> load only rules containing filter string in metadata if set
- **--nostamps** don't include timestamps in log file names
- **--nolock-pidfile** do not try to lock Snort PID file
- **--no-warn-flowbits** ignore warnings about flowbits that are checked but not set and vice-versa
- **--no-warn-rules** ignore warnings about duplicate rules and rule parsing issues
- **--pause** wait for resume/quit command before processing packets/terminating
- **--pcap-file** <file> file that contains a list of pcaps to read - read mode is implied
- **--pcap-list** <list> a space separated list of pcaps to read - read mode is implied
- **--pcap-dir** <dir> a directory to recurse to look for pcaps - read mode is implied
- **--pcap-filter** <filter> filter to apply when getting pcaps from file or directory
- **--pcap-loop** <count> read all pcaps <count> times; 0 will read until Snort is terminated (0:max32)
- **--pcap-no-filter** reset to use no filter when getting pcaps from file or directory
- **--pcap-show** print a line saying what pcap is currently being read
- **--pedantic** warnings are fatal
- **--plugin-path** <path> a colon separated list of directories or plugin libraries
- **--process-all-events** process all action groups
- **--rule** <rules> to be added to configuration; may be repeated
- **--rule-path** <path> where to find rules files
- **--rule-to-hex** output so rule header to stdout for text rule on stdin
- **--rule-to-text** output plain so rule header to stdout for text rule on stdin (specify delimiter or [Snort_SO_Rule] will be used) (16)

- **--run-prefix** <pfx> prepend this to each output file
- **--script-path** <path> to a luajit script or directory containing luajit scripts
- **--shell** enable the interactive command line
- **--show-file-codes** indicate how files are located: A=absolute and W, F, C which are relative to the working directory, including file, and config file respectively
- **--show-plugins** list module and plugin versions
- **--skip** <n> skip 1st n packets (0:max53)
- **--snaplen** <snap> set snaplen of packet (same as -s) (0:65535)
- **--stdin-rules** read rules from stdin until EOF or a line starting with END is read
- **--talos** enable Talos tweak (same as --tweaks talos)
- **--tweaks** tune configuration
- **--version** show version number (same as -V)
- **--warn-all** enable all warnings
- **--warn-conf** warn about configuration issues
- **--warn-conf-strict** warn about unrecognized elements in configuration files
- **--warn-daq** warn about DAQ issues, usually related to mode
- **--warn-flowbits** warn about flowbits that are checked but not set and vice-versa
- **--warn-hosts** warn about host table issues
- **--warn-plugins** warn about issues that prevent plugins from loading
- **--warn-rules** warn about duplicate rules and rule parsing issues
- **--warn-scripts** warn about issues discovered while processing Lua scripts
- **--warn-symbols** warn about unknown symbols in your Lua config
- **--warn-vars** warn about variable definition and usage issues
- **--x2c** output ASCII char for given hex (see also --c2x) (0x00:0xFF)
- **--x2s** output ASCII string for given byte code (see also --x2c)

11.4 Configuration

- interval **ack.~range**: check if TCP ack value is *value* | *min*<>*max* | <*max* | >*min* { 0: }
- int **active.attempts** = 0: number of TCP packets sent per response (with varying sequence numbers) { 0:255 }
- string **active.device**: use *ip* for network layer responses or *eth0* etc for link layer
- string **active.dst_mac**: use format *01:23:45:67:89:ab*
- int **active.max_responses** = 0: maximum number of responses { 0:255 }
- int **active.min_interval** = 255: minimum number of seconds between responses { 1:255 }
- string **`address_space_selector[].addr_spaces`**: list of address space IDs to match
- string **`address_space_selector[].file`**: use configuration in given file

- multi **alert_csv.fields** = *timestamp pkt_num proto pkt_gen pkt_len dir src_ap dst_ap rule action*: selected fields will be output in given order left to right { action | class | b64_data | client_bytes | client_pkts | dir | dst_addr | dst_ap | dst_port | eth_dst | eth_len | eth_src | eth_type | flowstart_time | geneve_vni | gid | icmp_code | icmp_id | icmp_seq | icmp_type | iface | ip_id | ip_len | msg | mpls | pkt_gen | pkt_len | pkt_num | priority | proto | rev | rule | seconds | server_bytes | server_pkts | service | sgt| sid | src_addr | src_ap | src_port | target | tcp_ack | tcp_flags | tcp_len | tcp_seq | tcp_win | timestamp | tos | ttl | udp_len | vlan }
- bool **alert_csv.file** = false: output to alert_csv.txt instead of stdout
- int **alert_csv.limit** = 0: set maximum size in MB before rollover (0 is unlimited) { 0:maxSZ }
- string **alert_csv.separator** = ', ': separate fields with this character sequence
- bool **alert_ex.upper** = false: true/false → convert to upper/lower case
- int **alert_fast.buffers_depth** = 0: number of IPS buffer bytes to dump per buffer (0 is unlimited) { 0:maxSZ }
- bool **alert_fast.buffers** = false: output IPS buffer dump
- bool **alert_fast.file** = false: output to alert_fast.txt instead of stdout
- int **alert_fast.limit** = 0: set maximum size in MB before rollover (0 is unlimited) { 0:maxSZ }
- bool **alert_fast.packet** = false: output packet dump with alert
- bool **alert_full.file** = false: output to alert_full.txt instead of stdout
- int **alert_full.limit** = 0: set maximum size in MB before rollover (0 is unlimited) { 0:maxSZ }
- multi **alert_json.fields** = *timestamp pkt_num proto pkt_gen pkt_len dir src_ap dst_ap rule action*: selected fields will be output in given order left to right { action | class | b64_data | client_bytes | client_pkts | dir | dst_addr | dst_ap | dst_port | eth_dst | eth_len | eth_src | eth_type | flowstart_time | geneve_vni | gid | icmp_code | icmp_id | icmp_seq | icmp_type | iface | ip_id | ip_len | msg | mpls | pkt_gen | pkt_len | pkt_num | priority | proto | rev | rule | seconds | server_bytes | server_pkts | service | sgt| sid | src_addr | src_ap | src_port | target | tcp_ack | tcp_flags | tcp_len | tcp_seq | tcp_win | timestamp | tos | ttl | udp_len | vlan }
- bool **alert_json.file** = false: output to alert_json.txt instead of stdout
- int **alert_json.limit** = 0: set maximum size in MB before rollover (0 is unlimited) { 0:maxSZ }
- string **alert_json.separator** = ', ': separate fields with this character sequence
- bool **alerts.alert_with_interface_name** = false: include interface in alert info (fast, full, or syslog only)
- int **alerts.detection_filter_memcap** = 1048576: set available MB of memory for detection_filters { 0:max32 }
- int **alerts.event_filter_memcap** = 1048576: set available MB of memory for event_filters { 0:max32 }
- bool **alerts.log_references** = false: include rule references in alert info (full only)
- string **alerts.order**: change the order of rule action application
- int **alerts.rate_filter_memcap** = 1048576: set available MB of memory for rate_filters { 0:max32 }
- string **alerts.reference_net**: set the CIDR for homenet (for use with -l or -B, does NOT change $HOME_NET in IDS mode)
- string **alerts.tunnel_verdicts**: let DAQ handle non-allow verdicts for gtp|teredo|6in4|4in6|4in4|6in6|gre|mpls|vxlan traffic
- enum **alert_syslog.facility** = *auth*: part of priority applied to each message { *auth* | *authpriv* | *daemon* | *user* | *local0* | *local1* | *local2* | *local3* | *local4* | *local5* | *local6* | *local7* }
- enum **alert_syslog.level** = *info*: part of priority applied to each message { *emerg* | *alert* | *crit* | *err* | *warning* | *notice* | *info* | *debug* }
- multi **alert_syslog.options**: used to open the syslog connection { cons | ndelay | perror | pid }
- string **appid.app_detector_dir**: directory to load appid detectors from
- int **appid.app_stats_period** = 300: time period for collecting and logging appid statistics { 1:max32 }

- int **appid.app_stats_rollover_size** = 20971520: max file size for appid stats before rolling over the log file { 0:max32 }
- bool **appid.enable_rna_filter** = false: monitor only the networks specified in rna configuration
- string **appid_listener.file**: output data to given file
- bool **appid_listener.json_logging** = false: log appid data in json format
- bool **appid.list_odp_detectors** = false: enable logging of odp detectors statistics
- bool **appid.log_all_sessions** = false: enable logging of all appid sessions
- bool **appid.log_stats** = false: enable logging of appid statistics
- int **appid.memcap** = 1048576: max size of the service cache before we start pruning the cache { 1024:maxSZ }
- string **appid.rna_conf_path**: path to rna configuration file
- string **appids.~**: comma separated list of application names
- bool **appid.tp_appid_config_dump**: print third party configuration on startup
- string **appid.tp_appid_config**: path to third party appid configuration file
- string **appid.tp_appid_path**: path to third party appid dynamic library
- bool **appid.tp_appid_stats_enable**: enable collection of stats and print stats on exit in third party module
- ip4 **`arp_spoof.hosts[].ip`**: host ip address
- mac **`arp_spoof.hosts[].mac`**: host mac address
- string **attribute_table.hosts_file**: filename to load attribute host table from
- int **attribute_table.max_hosts** = 1024: maximum number of hosts in attribute table { 32:max53 }
- int **attribute_table.max_metadata_services** = 9: maximum number of services in rule { 1:255 }
- int **attribute_table.max_services_per_host** = 8: maximum number of services per host entry in attribute table { 1:65535 }
- int **attribute_table.segments** = 4: number of segments of hosts attribute table. It must be power of 2. { 1:32 }
- int **base64_decode.bytes**: number of base64 encoded bytes to decode { 1:max32 }
- int **base64_decode.offset** = 0: bytes past start of buffer to start decoding { 0:max32 }
- implied **base64_decode.relative**: apply offset to cursor instead of start of buffer
- int **ber_data.~type**: move to the data for the specified BER element type { 0:255 }
- implied **ber_skip.optional**: match even if the specified BER type is not found
- int **ber_skip.~type**: BER element type to skip { 0:255 }
- enum **`binder[].use.action`** = *inspect*: what to do with matching traffic { *reset* | *block* | *allow* | *inspect* }
- string **`binder[].use.file`**: use configuration in given file
- string **`binder[].use.inspection_policy`**: use inspection policy from given file
- string **`binder[].use.ips_policy`**: use ips policy from given file
- string **`binder[].use.name`**: symbol name (defaults to type)
- string **`binder[].use.service`**: override automatic service identification
- string **`binder[].use.type`**: select module for binding
- string **`binder[].when.addr_spaces`**: list of address space IDs

- string **binder[].when.dst_groups**: list of destination group IDs
- string **binder[].when.dst_intfs**: list of destination interface IDs
- addr_list **binder[].when.dst_nets**: list of destination networks
- bit_list **binder[].when.dst_ports**: list of destination ports { 65535 }
- string **binder[].when.dst_zone**: deprecated alias for dst_groups
- string **binder[].when.groups**: list of interface group IDs
- string **binder[].when.intfs**: list of interface IDs
- int **binder[].when.ips_policy_id**: unique ID for selection of this config by external logic { 0:max32 }
- addr_list **binder[].when.nets**: list of networks
- bit_list **binder[].when.ports**: list of ports { 65535 }
- enum **binder[].when.proto**: protocol { *any* | *ip* | *icmp* | *tcp* | *udp* | *user* | *file* }
- enum **binder[].when.role** = *any*: use the given configuration on one or any end of a session { *client* | *server* | *any* }
- string **binder[].when.service**: override default configuration
- string **binder[].when.src_groups**: list of source interface group IDs
- string **binder[].when.src_intfs**: list of source interface IDs
- addr_list **binder[].when.src_nets**: list of source networks
- bit_list **binder[].when.src_ports**: list of source ports { 65535 }
- string **binder[].when.src_zone**: deprecated alias for src_groups
- string **binder[].when.tenants**: list of tenants
- bit_list **binder[].when.vlans**: list of VLAN IDs { 4095 }
- string **binder[].when.zones**: deprecated alias for groups
- interval **bufferlen.~range**: check that total length of current buffer is in given range { 0:65535 }
- implied **bufferlen.relative**: use remaining length (from current position) instead of total length
- int **byte_extract.align** = 0: round the number of converted bytes up to the next 2- or 4-byte boundary { 0:4 }
- implied **byte_extract.big**: big endian
- int **byte_extract.bitmask**: applies as an AND to the extracted value before storage in *name* { 0x1:0xFFFFFFFF }
- int **byte_extract.~count**: number of bytes to pick up from the buffer (string can pick less) { 1:10 }
- implied **byte_extract.dce**: dcerpc2 determines endianness
- implied **byte_extract.dec**: convert from decimal string
- implied **byte_extract.hex**: convert from hex string
- implied **byte_extract.little**: little endian
- int **byte_extract.multiplier** = 1: scale extracted value by given amount { 1:65535 }
- string **byte_extract.~name**: name of the variable that will be used in other rule options
- implied **byte_extract.oct**: convert from octal string
- int **byte_extract.~offset**: number of bytes into the buffer to start processing { -65535:65535 }

- implied **byte_extract.relative**: offset from cursor instead of start of buffer
- implied **byte_extract.string**: convert from string
- int **byte_jump.align** = 0: round the number of converted bytes up to the next 2- or 4-byte boundary { 0:4 }
- implied **byte_jump.big**: big endian
- int **byte_jump.bitmask**: applies as an AND prior to evaluation { 0x1:0xFFFFFFFF }
- int **byte_jump.~count**: number of bytes to pick up from the buffer (string can pick less) { 0:10 }
- implied **byte_jump.dce**: dcerpc2 determines endianness
- implied **byte_jump.dec**: convert from decimal string
- implied **byte_jump.from_beginning**: jump from start of buffer instead of cursor
- implied **byte_jump.from_end**: jump backward from end of buffer
- implied **byte_jump.hex**: convert from hex string
- implied **byte_jump.little**: little endian
- int **byte_jump.multiplier** = 1: scale extracted value by given amount { 1:65535 }
- implied **byte_jump.oct**: convert from octal string
- string **byte_jump.~offset**: variable name or number of bytes into the buffer to start processing
- string **byte_jump.post_offset**: skip forward or backward (positive or negative value) by variable name or number of bytes after the other jump options have been applied
- implied **byte_jump.relative**: offset from cursor instead of start of buffer
- implied **byte_jump.string**: convert from string
- int **byte_math.bitmask**: applies as bitwise AND to the extracted value before storage in *name* { 0x1:0xFFFFFFFF }
- int **byte_math.bytes**: number of bytes to pick up from the buffer (string can pick less) { 1:10 }
- implied **byte_math.dce**: dcerpc2 determines endianness
- enum **byte_math.endian**: specify big/little endian { *big|little* }
- string **byte_math.offset**: number of bytes into the buffer to start processing
- enum **byte_math.oper**: mathematical operation to perform { +|-|*|/|<<|>> }
- implied **byte_math.relative**: offset from cursor instead of start of buffer
- string **byte_math.result**: name of the variable to store the result
- string **byte_math.rvalue**: value to use mathematical operation against
- enum **byte_math.string**: convert extracted string to dec/hex/oct { *hex|dec|oct* }
- implied **byte_test.big**: big endian
- int **byte_test.bitmask**: applies as an AND prior to evaluation { 0x1:0xFFFFFFFF }
- string **byte_test.~compare**: variable name or value to test the converted result against
- int **byte_test.~count**: number of bytes to pick up from the buffer (string can pick less) { 1:10 }
- implied **byte_test.dce**: dcerpc2 determines endianness
- implied **byte_test.dec**: convert from decimal string

- implied **byte_test.hex**: convert from hex string
- implied **byte_test.little**: little endian
- implied **byte_test.oct**: convert from octal string
- string **byte_test.~offset**: variable name or number of bytes into the payload to start processing
- string **byte_test.~operator**: operation to perform to test the value
- implied **byte_test.relative**: offset from cursor instead of start of buffer
- implied **byte_test.string**: convert from string
- interval **cip_attribute.~range**: match CIP attribute { 0:65535 }
- interval **cip_class.~range**: match CIP class { 0:65535 }
- interval **cip_conn_path_class.~range**: match CIP Connection Path Class { 0:65535 }
- string **cip.embedded_cip_path** = *false*: check embedded CIP path
- interval **cip_instance.~range**: match CIP instance { 0:4294967295 }
- int **cip.max_cip_connections** = 100: max cip connections { 1:10000 }
- int **cip.max_unconnected_messages** = 100: max unconnected cip messages { 1:10000 }
- interval **cip_service.~range**: match CIP service { 0:127 }
- interval **cip_status.~range**: match CIP response status { 0:255 }
- int **cip.unconnected_timeout** = 300: unconnected timeout in seconds { 0:360 }
- string **`classifications[].name`**: name used with classtype rule option
- int **`classifications[].priority`** = 1: default priority for class { 0:max32 }
- string **`classifications[].text`**: description of class
- string **classtype.~**: classification for this rule
- string **content.~data**: data to match
- string **content.depth**: var or maximum number of bytes to search from beginning of buffer
- string **content.distance**: var or number of bytes from cursor to start search
- int **content.fast_pattern_length**: maximum number of characters from this content the fast pattern matcher should use { 1:65535 }
- int **content.fast_pattern_offset** = 0: number of leading characters of this content the fast pattern matcher should exclude { 0:65535 }
- implied **content.fast_pattern**: use this content in the fast pattern matcher instead of the content selected by default
- implied **content.nocase**: case insensitive match
- string **content.offset**: var or number of bytes from start of buffer to start search
- string **content.within**: var or maximum number of bytes to search from cursor
- implied **cvs.invalid-entry**: looks for an invalid Entry string
- int **daq.batch_size** = 64: set receive batch size (same as --daq-batch-size) { 1: }
- string **`daq.inputs[].input`**: input source
- string **`daq.module_dirs[].path`**: directory path

- enum **daq.modules[].mode** = *passive*: DAQ module mode { *passive* | *inline* | *read-file* }
- string **daq.modules[].name**: DAQ module name (required)
- string **daq.modules[].variables[].variable**: DAQ module variable (foo[=bar])
- int **daq.snaplen** = 1518: set snap length (same as -s) { 0:65535 }
- select **data_log.key** = 'http_request_header_event ': name of the event to log { http_request_header_event | http_response_header_ever
  }
- int **data_log.limit** = 0: set maximum size in MB before rollover (0 is unlimited) { 0:max32 }
- implied **dce_iface.any_frag**: match on any fragment
- string **dce_iface.uuid**: match given dcerpc uuid
- interval **dce_iface.version**: interface version { 0: }
- string **dce_opnum.~**: match given dcerpc operation number, range or list
- bool **dce_smb.disable_defrag** = false: disable DCE/RPC defragmentation
- bool **dce_smb.limit_alerts** = true: limit DCE alert to at most one per signature per flow
- int **dce_smb.max_frag_len** = 65535: maximum fragment size for defragmentation { 1514:65535 }
- int **dce_smb.memcap** = 8388608: Memory utilization limit on SMBv2 cache { 512:maxSZ }
- enum **dce_smb.policy** = *WinXP*: target based policy to use { *Win2000* | *WinXP* | *WinVista* | *Win2003* | *Win2008* | *Win7* | *Samba* | *Samba-3.0.37* | *Samba-3.0.22* | *Samba-3.0.20* }
- int **dce_smb.reassemble_threshold** = 0: minimum bytes received before performing reassembly { 0:65535 }
- int **dce_smb.smb_file_depth** = 16384: SMB file depth for file data (-1 = disabled, 0 = unlimited) { -1:32767 }
- enum **dce_smb.smb_file_inspection**: deprecated (not used): file inspection controlled by smb_file_depth { *off* | *on* | *only* }
- enum **dce_smb.smb_fingerprint_policy** = *none*: target based SMB policy to use { *none* | *client* | *server* | *both* }
- string **dce_smb.smb_invalid_shares**: SMB shares to alert on
- bool **dce_smb.smb_legacy_mode** = false: inspect only SMBv1
- int **dce_smb.smb_max_chain** = 3: SMB max chain size { 0:255 }
- int **dce_smb.smb_max_compound** = 3: SMB max compound size { 0:255 }
- int **dce_smb.smb_max_credit** = 8192: Maximum number of outstanding request { 1:65535 }
- multi **dce_smb.valid_smb_versions** = *all*: valid SMB versions { v1 | v2 | all }
- bool **dce_tcp.disable_defrag** = false: disable DCE/RPC defragmentation
- bool **dce_tcp.limit_alerts** = true: limit DCE alert to at most one per signature per flow
- int **dce_tcp.max_frag_len** = 65535: maximum fragment size for defragmentation { 1514:65535 }
- enum **dce_tcp.policy** = *WinXP*: target based policy to use { *Win2000* | *WinXP* | *WinVista* | *Win2003* | *Win2008* | *Win7* | *Samba* | *Samba-3.0.37* | *Samba-3.0.22* | *Samba-3.0.20* }
- int **dce_tcp.reassemble_threshold** = 0: minimum bytes received before performing reassembly { 0:65535 }
- bool **dce_udp.disable_defrag** = false: disable DCE/RPC defragmentation
- bool **dce_udp.limit_alerts** = true: limit DCE alert to at most one per signature per flow
- int **dce_udp.max_frag_len** = 65535: maximum fragment size for defragmentation { 1514:65535 }

- bool **detection.allow_missing_so_rules** = false: warn (true) or error (false) when an SO rule stub refers to an SO rule that isn't loaded
- bool **detection.enable_address_anomaly_checks** = false: enable check and alerting of address anomalies
- bool **detection.enable_strict_reduction** = false: enable strict deduplication of rule headers by ports (saves memory, but loses some speed during config reading)
- int **detection_filter.count**: hits in interval before allowing the rule to fire { 1:max32 }
- int **detection_filter.seconds**: length of interval to count hits { 1:max32 }
- enum **detection_filter.track**: track hits by source or destination IP address { *by_src* | *by_dst* }
- bool **detection.global_default_rule_state** = true: enable or disable rules by default (overridden by ips policy settings)
- bool **detection.global_rule_state** = false: apply rule_state against all policies
- bool **detection.hyperscan_literals** = false: use hyperscan for content literal searches instead of boyer-moore
- int **detection.max_continuations_per_flow** = 1024: maximum number of continuations stored simultaneously on the flow { 0:65535 }
- int **detection.offload_limit** = 99999: minimum sizeof PDU to offload fast pattern search (defaults to disabled) { 0:max32 }
- int **detection.offload_threads** = 0: maximum number of simultaneous offloads (defaults to disabled) { 0:max32 }
- bool **detection.pcre_enable** = true: enable pcre pattern matching
- int **detection.pcre_match_limit** = 1500: limit pcre backtracking, 0 = off { 0:max32 }
- int **detection.pcre_match_limit_recursion** = 1500: limit pcre stack consumption, 0 = off { 0:max32 }
- bool **detection.pcre_override** = true: enable pcre match limit overrides when pattern matching (ie ignore /O)
- bool **detection.pcre_to_regex** = false: enable the use of regex instead of pcre for compatible expressions
- string **`detection.service_extension[].extend_to[].extend_to_service`**: service to extend to
- string **`detection.service_extension[].service`**: service to perform extension for
- bool **dnp3.check_crc** = false: validate checksums in DNP3 link layer frames
- string **dnp3_func.~**: match DNP3 function code or name
- string **dnp3_ind.~**: match given DNP3 indicator flags
- int **dnp3_obj.group** = 0: match given DNP3 object header group { 0:255 }
- int **dnp3_obj.var** = 0: match given DNP3 object header var { 0:255 }
- bool **dns.publish_response** = false: parse and publish dns responses
- string **domain_filter.file**: file with list of domains identifying hosts to be filtered
- string **domain_filter.hosts**: list of domains identifying hosts to be filtered
- int **dpx.max** = 0: maximum payload before alert { 0:65535 }
- port **dpx.port**: port to check
- interval **dsize.~range**: check if packet payload size is in the given range { 0:65535 }
- enum **enable.~enable** = *yes*: enable or disable rule in current ips policy or use default defined by ips policy { *no* | *yes* | *inherit* }
- interval **enip_command.~range**: match CIP Enip Command { 0:65535 }

- bool **esp.decode_esp** = false: enable for inspection of esp traffic that has authentication but not encryption
- int **`event_filter[].count`** = 0: number of events in interval before tripping; -1 to disable { -1:max31 }
- int **`event_filter[].gid`** = 1: rule generator ID { 0:8129 }
- string **`event_filter[].ip`**: restrict filter to these addresses according to track
- int **`event_filter[].seconds`** = 0: count interval { 0:max32 }
- int **`event_filter[].sid`** = 1: rule signature ID { 0:max32 }
- enum **`event_filter[].track`**: filter only matching source or destination addresses { *by_src* | *by_dst* }
- enum **`event_filter[].type`**: 1st count events | every count events | once after count events { *limit* | *threshold* | *both* }
- int **event_queue.log** = 3: maximum events to log { 1:max32 }
- int **event_queue.max_queue** = 8: maximum events to queue { 1:max32 }
- enum **event_queue.order_events** = *content_length*: criteria for ordering incoming events { *priority*|*content_length* }
- bool **event_queue.process_all_events** = false: process just first action group or all action groups
- string **`file_connector[].connector`**: connector name
- enum **`file_connector[].direction`**: usage { *receive* | *transmit* | *duplex* }
- enum **`file_connector[].format`**: file format { *binary* | *text* }
- string **`file_connector[].name`**: channel name
- int **file_id.block_timeout** = 86400: stop blocking after this many seconds { 0:max31 }
- bool **file_id.block_timeout_lookup** = false: block if lookup times out
- int **file_id.capture_block_size** = 32768: file capture block size in bytes { 8:max53 }
- int **file_id.capture_max_size** = 1048576: stop file capture beyond this point { 0:max53 }
- int **file_id.capture_memcap** = 100: memcap for file capture in megabytes { 0:max53 }
- int **file_id.capture_min_size** = 0: stop file capture if file size less than this { 0:max53 }
- int **file_id.decompress_buffer_size** = 100000: file decompression buffer size { 1024:max31 }
- int **file_id.lookup_timeout** = 2: give up on lookup after this many seconds { 0:max31 }
- int **file_id.max_files_cached** = 65536: maximal number of files cached in memory { 8:max53 }
- int **file_id.max_files_per_flow** = 128: maximal number of files able to be concurrently processed per flow { 1:max53 }
- string **file_id.rules_file**: name of file with IPS rules for file identification
- int **file_id.show_data_depth** = 100: print this many octets { 0:max53 }
- int **file_id.signature_depth** = 10485760: stop signature at this point { 0:max53 }
- bool **file_id.trace_signature** = false: enable runtime dump of signature info
- bool **file_id.trace_stream** = false: enable runtime dump of file data
- bool **file_id.trace_type** = false: enable runtime dump of type info
- int **file_id.type_depth** = 1460: stop type ID at this point { 0:max53 }
- bool **file_log.log_pkt_time** = true: log the packet time when event generated
- bool **file_log.log_sys_time** = false: log the system time when event generated

- string **file_meta.category**: file type category
- string **file_meta.group**: comma separated list of groups associated with file type
- int **file_meta.id**: file type id { 1:1023 }
- string **file_meta.type**: file type to set
- string **file_meta.version**: file type version
- bool **file_policy.enable_capture** = false: enable file capture
- bool **file_policy.enable_signature** = false: enable signature calculation
- bool **file_policy.enable_type** = true: enable type ID
- bool **`file_policy.rules[].use.enable_file_capture`** = false: true/false → enable/disable file capture
- bool **`file_policy.rules[].use.enable_file_signature`** = false: true/false → enable/disable file signature
- bool **`file_policy.rules[].use.enable_file_type`** = false: true/false → enable/disable file type identification
- enum **`file_policy.rules[].use.verdict`** = *unknown*: what to do with matching traffic { *unknown* | *log* | *stop* | *block* | *reset* }
- int **`file_policy.rules[].when.file_type_id`** = 0: unique ID for file type in file magic rule { 0:max32 }
- string **`file_policy.rules[].when.sha256`**: SHA 256
- int **file_policy.verdict_delay** = 0: number of queries to return final verdict { 0:max53 }
- string **file_type.~**: list of file type IDs to match
- string **flags.~mask_flags**: these flags are don't cares
- string **flags.~test_flags**: these flags are tested
- string **flowbits.~bits**: bit [|bit]* or bit [&bit]*
- enum **flowbits.~op**: bit operation or noalert (no bits) { *set* | *unset* | *isset* | *isnotset* | *noalert* }
- implied **flow.established**: match only during data transfer phase
- implied **flow.from_client**: same as to_server
- implied **flow.from_server**: same as to_client
- implied **flow.no_frag**: match on raw packets only
- implied **flow.no_stream**: match on raw packets only
- implied **flow.not_established**: match only outside data transfer phase
- implied **flow.only_frag**: match on defragmented packets only
- implied **flow.only_stream**: match on reassembled packets only
- implied **flow.stateless**: match regardless of stream state
- implied **flow.to_client**: match on server responses
- implied **flow.to_server**: match on client requests
- string **fragbits.~flags**: these flags are tested
- interval **fragoffset.~range**: check if ip fragment offset is in given range { 0:8192 }
- bool **ftp_client.bounce** = false: check for bounces

- addr **ftp_client.bounce_to[].address** = *1.0.0.0/32*: allowed IP address in CIDR format
- port **ftp_client.bounce_to[].last_port**: optional allowed range from port to last_port inclusive
- port **ftp_client.bounce_to[].port** = 20: allowed port
- bool **ftp_client.ignore_telnet_erase_cmds** = false: ignore erase character and erase line commands when normalizing
- int **ftp_client.max_resp_len** = 4294967295: maximum FTP response accepted by client { 0:max32 }
- bool **ftp_client.telnet_cmds** = false: detect Telnet escape sequences on FTP control channel
- bool **ftp_server.check_encrypted** = false: check for end of encryption
- string **ftp_server.chk_str_fmt**: check the formatting of the given commands
- string **ftp_server.cmd_validity[].command**: command string
- string **ftp_server.cmd_validity[].format**: format specification
- int **ftp_server.cmd_validity[].length** = 0: specify non-default maximum for command { 0:max32 }
- string **ftp_server.data_chan_cmds**: check the formatting of the given commands
- string **ftp_server.data_rest_cmds**: check the formatting of the given commands
- string **ftp_server.data_xfer_cmds**: check the formatting of the given commands
- int **ftp_server.def_max_param_len** = 100: default maximum length of commands handled by server; 0 is unlimited { 1:max32 }
- string **ftp_server.directory_cmds[].dir_cmd**: directory command
- int **ftp_server.directory_cmds[].rsp_code** = 200: expected successful response code for command { 200:max32 }
- string **ftp_server.encr_cmds**: check the formatting of the given commands
- bool **ftp_server.encrypted_traffic** = false: check for encrypted Telnet and FTP
- string **ftp_server.file_get_cmds**: check the formatting of the given commands
- string **ftp_server.file_put_cmds**: check the formatting of the given commands
- string **ftp_server.ftp_cmds**: specify additional commands supported by server beyond RFC 959
- bool **ftp_server.ignore_data_chan** = false: do not inspect FTP data channels
- bool **ftp_server.ignore_telnet_erase_cmds** = false: ignore erase character and erase line commands when normalizing
- string **ftp_server.login_cmds**: check the formatting of the given commands
- bool **ftp_server.print_cmds** = false: print command configurations on start up
- bool **ftp_server.telnet_cmds** = false: detect Telnet escape sequences of FTP control channel
- int **gid.~**: generator id { 1:8129 }
- string **gtp_info.~**: info element to match
- int **gtp_inspect[].infos[].length** = 0: information element type code { 0:255 }
- string **gtp_inspect[].infos[].name**: information element name
- int **gtp_inspect[].infos[].type** = 0: information element type code { 0:255 }
- string **gtp_inspect[].messages[].name**: message name
- int **gtp_inspect[].messages[].type** = 0: message type code { 0:255 }

- int **gtp_inspect[].version** = 2: GTP version { 0:2 }
- string **gtp_type.~**: list of types to match
- int **gtp_version.~**: version to match { 0:2 }
- bool **high_availability.daq_channel** = false: enable use of daq data plane channel
- bool **high_availability.enable** = false: enable high availability
- int **high_availability.min_age** = 0: minimum session life in milliseconds before HA updates { 0:max32 }
- int **high_availability.min_sync** = 0: minimum interval in milliseconds between HA updates { 0:max32 }
- bit_list **high_availability.ports**: side channel message port list { 65535 }
- string **host_cache.dump_file**: file name to dump host cache on shutdown; won't dump by default
- int **host_cache.memcap** = 8388608: maximum host cache size in bytes { 512:maxSZ }
- int **host_cache.segments** = 4: number of host cache segments. It must be power of 2. { 1:32 }
- enum **hosts[].frag_policy**: defragmentation policy { *first* | *linux* | *bsd* | *bsd_right* | *last* | *windows* | *solaris* }
- addr **hosts[].ip** = *0.0.0.0/32*: hosts address / CIDR
- string **hosts[].services[].name**: service identifier
- port **hosts[].services[].port**: port number
- enum **hosts[].services[].proto** = *tcp*: IP protocol { *tcp* | *udp* }
- enum **hosts[].tcp_policy**: TCP reassembly policy { *first* | *last* | *linux* | *old_linux* | *bsd* | *macos* | *solaris* | *irix* | *hpux11* | *hpux10* | *windows* | *win_2003* | *vista* | *proxy* }
- addr **host_tracker[].ip**: hosts address / cidr
- port **host_tracker[].services[].port**: port number
- enum **host_tracker[].services[].proto**: IP protocol { *ip* | *tcp* | *udp* }
- int **http2_inspect.concurrent_streams_limit** = 100: Maximum number of concurrent streams allowed in a single HTTP/2 flow { 100:1000 }
- implied **http_cookie.request**: match against the cookie from the request message even when examining the response
- implied **http_cookie.with_body**: option is no longer used and will be removed in a future release
- implied **http_cookie.with_header**: option is no longer used and will be removed in a future release
- implied **http_cookie.with_trailer**: option is no longer used and will be removed in a future release
- string **http_header.field**: restrict to given header. Header name is case insensitive.
- implied **http_header.request**: match against the headers from the request message even when examining the response
- implied **http_header_test.absent**: header is absent
- interval **http_header_test.check**: range check to perform on header value { 0:999999999999999999 }
- string **http_header_test.field**: Header to perform check on. Header name is case insensitive.
- bool **http_header_test.numeric**: header value is a number
- implied **http_header_test.request**: match against the headers from the request message even when examining the response
- implied **http_header_test.with_body**: option is no longer used and will be removed in a future release
- implied **http_header_test.with_header**: option is no longer used and will be removed in a future release

- implied **http_header_test.with_trailer**: option is no longer used and will be removed in a future release
- implied **http_header.with_body**: option is no longer used and will be removed in a future release
- implied **http_header.with_header**: option is no longer used and will be removed in a future release
- implied **http_header.with_trailer**: option is no longer used and will be removed in a future release
- string **http_inspect.allowed_methods**: list of allowed methods
- bool **http_inspect.backslash_to_slash** = true: replace \ with / when normalizing URIs
- bit_list **http_inspect.bad_characters**: alert when any of specified bytes are present in URI after percent decoding { 255 }
- bool **http_inspect.decompress_pdf** = false: decompress pdf files in response bodies
- bool **http_inspect.decompress_swf** = false: decompress swf files in response bodies
- bool **http_inspect.decompress_vba** = false: decompress MS Office Visual Basic for Applications macro files in response bodies
- bool **http_inspect.decompress_zip** = false: decompress zip files in response bodies
- string **http_inspect.disallowed_methods**: list of disallowed methods
- string **http_inspect.ignore_unreserved**: do not alert when the specified unreserved characters are percent-encoded in a URI.Unreserved characters are 0-9, a-z, A-Z, period, underscore, tilde, and minus. { (optional) }
- bool **http_inspect.iis_double_decode** = true: perform double decoding of percent encodings to normalize characters
- int **http_inspect.iis_unicode_code_page** = 1252: code page to use from the IIS unicode map file { 0:65535 }
- bool **http_inspect.iis_unicode** = false: use IIS unicode code point mapping to normalize characters
- string **http_inspect.iis_unicode_map_file**: file containing code points for IIS unicode. { (optional) }
- int **http_inspect.maximum_chunk_length** = 4294967295: maximum allowed length for a message body chunk { 0:4294967295 }
- int **http_inspect.maximum_header_length** = 4096: alert when the length of a header exceeds this value { 0:65535 }
- int **http_inspect.maximum_headers** = 200: alert when the number of headers in a message exceeds this value { 0:65535 }
- int **http_inspect.maximum_host_length** = -1: maximum allowed length for Host header value (-1 no limit) { -1:max53 }
- int **http_inspect.maximum_pipelined_requests** = 99: alert when the number of pipelined requests exceeds this value { 0:99 }
- int **http_inspect.max_javascript_whitespaces** = 200: maximum consecutive whitespaces allowed within the JavaScript obfuscated data { 1:65535 }
- int **http_inspect.max_mime_attach** = 5: maximum number of mime attachments that will be inspected in a section of a request message { 1:65535 }
- bool **http_inspect.normalize_javascript** = false: use legacy normalizer to normalize JavaScript in response bodies
- bool **http_inspect.normalize_utf** = true: normalize charset utf encodings in response bodies
- int **http_inspect.oversize_dir_length** = 300: maximum length for URL directory { 1:65535 }
- bool **http_inspect.percent_u** = false: normalize %uNNNN and %UNNNN encodings
- bool **http_inspect.plus_to_space** = true: replace + with <sp> when normalizing URIs
- bool **http_inspect.request_body_app_detection** = true: make HTTP request message bodies available for application detection (AppId) and other inspectors

- int **http_inspect.request_depth** = -1: maximum request message body bytes to examine (-1 no limit) { -1:max53 }
- int **http_inspect.response_depth** = -1: maximum response message body bytes to examine (-1 no limit) { -1:max53 }
- bool **http_inspect.script_detection** = false: inspect JavaScript immediately upon script end
- bool **http_inspect.simplify_path** = true: reduce URI directory path to simplest form
- bool **http_inspect.unzip** = true: decompress gzip and deflate message bodies
- bool **http_inspect.utf8_bare_byte** = false: when doing UTF-8 character normalization include bytes that were not percent encoded
- bool **http_inspect.utf8** = true: normalize 2-byte and 3-byte UTF-8 characters to a single byte
- string **http_inspect.xff_headers** = *x-forwarded-for true-client-ip*: specifies the xff type headers to parse and consider in the same order of preference as defined
- interval **http_max_header_line.~range**: check that longest line of current header is in given range { 0:65535 }
- implied **http_max_header_line.request**: match against the version from the request message even when examining the response
- interval **http_max_trailer_line.~range**: check that longest line of current trailer is in given range { 0:65535 }
- implied **http_max_trailer_line.request**: match against the version from the request message even when examining the response
- implied **http_method.with_body**: option is no longer used and will be removed in a future release
- implied **http_method.with_header**: option is no longer used and will be removed in a future release
- implied **http_method.with_trailer**: option is no longer used and will be removed in a future release
- interval **http_num_cookies.~range**: check that number of cookies of current header are in given range { 0:65535 }
- implied **http_num_cookies.request**: match against the version from the request message even when examining the response
- interval **http_num_headers.~range**: check that number of headers of current buffer are in given range { 0:65535 }
- implied **http_num_headers.request**: match against the version from the request message even when examining the response
- implied **http_num_headers.with_body**: option is no longer used and will be removed in a future release
- implied **http_num_headers.with_header**: option is no longer used and will be removed in a future release
- implied **http_num_headers.with_trailer**: option is no longer used and will be removed in a future release
- interval **http_num_trailers.~range**: check that number of headers of current buffer are in given range { 0:65535 }
- implied **http_num_trailers.request**: match against the version from the request message even when examining the response
- implied **http_num_trailers.with_body**: option is no longer used and will be removed in a future release
- implied **http_num_trailers.with_header**: option is no longer used and will be removed in a future release
- implied **http_num_trailers.with_trailer**: option is no longer used and will be removed in a future release
- implied **http_param.nocase**: case insensitive match
- string **http_param.~param**: parameter to match
- implied **http_raw_cookie.request**: match against the cookie from the request message even when examining the response
- implied **http_raw_cookie.with_body**: option is no longer used and will be removed in a future release
- implied **http_raw_cookie.with_header**: option is no longer used and will be removed in a future release

- implied **http_raw_cookie.with_trailer**: option is no longer used and will be removed in a future release
- string **http_raw_header.field**: restrict to given header. Header name is case insensitive.
- implied **http_raw_header.request**: match against the headers from the request message even when examining the response
- implied **http_raw_header.with_body**: option is no longer used and will be removed in a future release
- implied **http_raw_header.with_header**: option is no longer used and will be removed in a future release
- implied **http_raw_header.with_trailer**: option is no longer used and will be removed in a future release
- implied **http_raw_request.with_body**: option is no longer used and will be removed in a future release
- implied **http_raw_request.with_header**: option is no longer used and will be removed in a future release
- implied **http_raw_request.with_trailer**: option is no longer used and will be removed in a future release
- implied **http_raw_status.with_body**: option is no longer used and will be removed in a future release
- implied **http_raw_status.with_trailer**: option is no longer used and will be removed in a future release
- string **http_raw_trailer.field**: restrict to given trailer. Trailer name is case insensitive.
- implied **http_raw_trailer.request**: match against the trailers from the request message even when examining the response
- implied **http_raw_trailer.with_body**: option is no longer used and will be removed in a future release
- implied **http_raw_trailer.with_header**: option is no longer used and will be removed in a future release
- implied **http_raw_uri.fragment**: match against fragment section of URI only
- implied **http_raw_uri.host**: match against host section of URI only
- implied **http_raw_uri.path**: match against path section of URI only
- implied **http_raw_uri.port**: match against port section of URI only
- implied **http_raw_uri.query**: match against query section of URI only
- implied **http_raw_uri.scheme**: match against scheme section of URI only
- implied **http_raw_uri.with_body**: option is no longer used and will be removed in a future release
- implied **http_raw_uri.with_header**: option is no longer used and will be removed in a future release
- implied **http_raw_uri.with_trailer**: option is no longer used and will be removed in a future release
- implied **http_stat_code.with_body**: option is no longer used and will be removed in a future release
- implied **http_stat_code.with_trailer**: option is no longer used and will be removed in a future release
- implied **http_stat_msg.with_body**: option is no longer used and will be removed in a future release
- implied **http_stat_msg.with_trailer**: option is no longer used and will be removed in a future release
- string **http_trailer.field**: restrict to given trailer
- implied **http_trailer.request**: match against the trailers from the request message even when examining the response
- implied **http_trailer_test.absent**: trailer is absent
- interval **http_trailer_test.check**: range check to perform on trailer value { 0:999999999999999999 }
- string **http_trailer_test.field**: Trailer to perform check on. Trailer name is case insensitive.
- bool **http_trailer_test.numeric**: trailer value is a number
- implied **http_trailer_test.request**: match against the trailers from the request message even when examining the response

- implied **http_trailer_test.with_body**: option is no longer used and will be removed in a future release
- implied **http_trailer_test.with_header**: option is no longer used and will be removed in a future release
- implied **http_trailer.with_body**: option is no longer used and will be removed in a future release
- implied **http_trailer.with_header**: option is no longer used and will be removed in a future release
- implied **http_true_ip.with_body**: option is no longer used and will be removed in a future release
- implied **http_true_ip.with_header**: option is no longer used and will be removed in a future release
- implied **http_true_ip.with_trailer**: option is no longer used and will be removed in a future release
- implied **http_uri.fragment**: match against fragment section of URI only
- implied **http_uri.host**: match against host section of URI only
- implied **http_uri.path**: match against path section of URI only
- implied **http_uri.port**: match against port section of URI only
- implied **http_uri.query**: match against query section of URI only
- implied **http_uri.scheme**: match against scheme section of URI only
- implied **http_uri.with_body**: option is no longer used and will be removed in a future release
- implied **http_uri.with_header**: option is no longer used and will be removed in a future release
- implied **http_uri.with_trailer**: option is no longer used and will be removed in a future release
- implied **http_version_match.request**: match against the version from the request message even when examining the response
- string **http_version_match.~version_list**: space-separated list of versions to match
- implied **http_version_match.with_body**: option is no longer used and will be removed in a future release
- implied **http_version_match.with_header**: option is no longer used and will be removed in a future release
- implied **http_version_match.with_trailer**: option is no longer used and will be removed in a future release
- implied **http_version.request**: match against the version from the request message even when examining the response
- implied **http_version.with_body**: option is no longer used and will be removed in a future release
- implied **http_version.with_header**: option is no longer used and will be removed in a future release
- implied **http_version.with_trailer**: option is no longer used and will be removed in a future release
- interval **icmp_id.~range**: check if ICMP ID is in given range { 0:65535 }
- interval **icmp_seq.~range**: check if ICMP sequence number is in given range { 0:65535 }
- interval **icode.~range**: check if ICMP code is in given range is { 0:255 }
- interval **id.~range**: check if the IP ID is in the given range { 0: }
- string **iec104_apci_type.~**: APCI type to match
- string **iec104_asdu_func.~**: function code to match
- int **imap.b64_decode_depth** = -1: base64 decoding depth (-1 no limit) { -1:65535 }
- int **imap.bitenc_decode_depth** = -1: non-Encoded MIME attachment extraction depth (-1 no limit) { -1:65535 }
- bool **imap.decompress_pdf** = false: decompress pdf files in MIME attachments
- bool **imap.decompress_swf** = false: decompress swf files in MIME attachments

- bool **imap.decompress_vba** = false: decompress MS Office Visual Basic for Applications macro files in MIME attachments
- bool **imap.decompress_zip** = false: decompress zip files in MIME attachments
- int **imap.qp_decode_depth** = -1: quoted Printable decoding depth (-1 no limit) { -1:65535 }
- int **imap.uu_decode_depth** = -1: Unix-to-Unix decoding depth (-1 no limit) { -1:65535 }
- int **inspection.id** = 0: correlate policy and events with other items in configuration { 0:max64 }
- int **inspection.max_aux_ip** = 16: maximum number of auxiliary IPs per flow to detect and save (-1 = disable, 0 = detect but don't save, 1+ = save in FIFO manner) { -1:127 }
- enum **inspection.mode** = *inline-test*: set policy mode { *inline* | *inline-test* }
- string **inspection.uuid**: correlate events by uuid
- select **ipopts.~opt**: output format { rr|eol|nop|ts|sec|esec|lsrr|lsrre|ssrr|satid|any }
- string **ip_proto.~proto**: [!|>|<] name or number
- string **`ips.action_map[].replace`**: action you want to change
- string **`ips.action_map[].with`**: action you want to use instead
- string **ips.action_override**: use this action for all rules (applied before action_map)
- enum **ips.default_rule_state** = *inherit*: enable or disable ips rules { *no* | *yes* | *inherit* }
- bool **ips.enable_builtin_rules** = false: enable events from builtin rules w/o stubs
- int **ips.id** = 0: correlate unified2 events with configuration { 0:max64 }
- string **ips.include**: snort rules and includes
- enum **ips.mode**: set policy mode { *tap* | *inline* | *inline-test* }
- bool **ips.obfuscate_pii** = true: mask all but the last 4 characters of credit card, SSN, phone number, and email
- string **ips.rules**: snort rules and includes (may contain states too)
- string **ips.states**: snort rule states and includes (may contain rules too)
- string **ips.uuid** = *00000000-0000-0000-0000-000000000000*: IPS policy uuid
- string **ips.variables.nets.$var**: IPS policy variable
- string **ips.variables.paths.$var**: IPS policy variable
- string **ips.variables.ports.$var**: IPS policy variable
- string **isdataat.~length**: num | !num
- implied **isdataat.relative**: offset from cursor instead of start of buffer
- interval **itype.~range**: check if ICMP type is in given range { 0:255 }
- int **js_norm.bytes_depth** = -1: number of input JavaScript bytes to normalize (-1 unlimited) { -1:max53 }
- int **js_norm.identifier_depth** = 65536: max number of unique JavaScript identifiers to normalize { 0:65536 }
- string **`js_norm.ident_ignore[].ident_name`**: name of the identifier to ignore
- int **js_norm.max_bracket_depth** = 256: maximum depth of bracket nesting that enhanced JavaScript normalizer will process { 1:65535 }
- int **js_norm.max_scope_depth** = 256: maximum depth of scope nesting that enhanced JavaScript normalizer will process { 1:65535 }

- int **js_norm.max_tmpl_nest** = 32: maximum depth of template literal nesting that enhanced JavaScript normalizer will process { 0:255 }
- string **`js_norm.prop_ignore[].prop_name`**: name of the object property to ignore
- bool **latency.packet.fastpath** = false: fastpath expensive packets (max_time exceeded)
- int **latency.packet.max_time** = 500: set timeout for packet latency thresholding (usec) { 0:max53 }
- int **latency.rule.max_suspend_time** = 30000: set max time for suspending a rule (ms, 0 means permanently disable rule) { 0:max32 }
- int **latency.rule.max_time** = 500: set timeout for rule evaluation (usec) { 0:max53 }
- bool **latency.rule.suspend** = false: temporarily suspend expensive rules
- int **latency.rule.suspend_threshold** = 5: set threshold for number of timeouts before suspending a rule { 1:max32 }
- bool **log_codecs.file** = false: output to log_codecs.txt instead of stdout
- bool **log_codecs.msg** = false: include alert msg
- bool **log_hext.file** = false: output to log_hext.txt instead of stdout
- int **log_hext.limit** = 0: set maximum size in MB before rollover (0 is unlimited) { 0:maxSZ }
- bool **log_hext.raw** = false: output all full packets if true, else just TCP payload
- int **log_hext.width** = 20: set line width (0 is unlimited) { 0:max32 }
- int **log_pcap.limit** = 0: set maximum size in MB before rollover (0 is unlimited) { 0:maxSZ }
- string **md5.~hash**: data to match
- int **md5.length**: number of octets in plain text { 1:65535 }
- string **md5.offset**: var or number of bytes from start of buffer to start search
- implied **md5.relative** = *false*: offset from cursor instead of start of buffer
- int **memory.cap** = 0: set the process cap on memory in bytes (0 to disable) { 0:maxSZ }
- int **memory.interval** = 50: approximate ms between memory epochs (0 to disable) { 0:max32 }
- int **memory.prune_target** = 1048576: bytes to prune per packet thread prune cycle { 1:max32 }
- int **memory.threshold** = 100: scale cap to account for heap overhead { 1:100 }
- string **`metadata.*`**: comma-separated list of arbitrary name value pairs
- string **mms_func.~**: func to match
- string **modbus_func.~**: function code to match
- int **modbus_unit.~**: Modbus unit ID { 0:255 }
- int **mpls.max_stack_depth** = -1: set maximum MPLS stack depth { -1:255 }
- enum **mpls.payload_type** = *auto*: force encapsulated payload type { *auto* | *eth* | *ip4* | *ip6* }
- string **msg.~**: message describing rule
- interval **mss.~range**: check if TCP MSS is in given range { 0:65535 }
- string **netflow.dump_file**: file name to dump netflow cache on shutdown; won't dump by default
- int **netflow.flow_memcap** = 0: maximum memory for flow record cache in bytes, 0 = unlimited { 0:maxSZ }
- string **netflow.netflow_service_id_path**: path to file containing service IDs for NetFlow

- bool **`netflow.rules[].create_host`** = false: generate a new host event
- bool **`netflow.rules[].create_service`** = false: generate a new or changed service event
- addr **`netflow.rules[].device_ip`**: restrict the NetFlow devices from which Snort will analyze packets
- bool **`netflow.rules[].exclude`** = false: exclude the NetFlow records that match this rule
- string **`netflow.rules[].networks`**: generate events for NetFlow records that contain an initiator or responder IP from these networks
- string **`netflow.rules[].zones`**: generate events only for NetFlow packets that originate from these zones
- int **netflow.template_memcap** = 0: maximum memory for template cache in bytes, 0 = unlimited { 0:maxSZ }
- int **netflow.update_timeout** = 3600: the interval at which the system updates host cache information { 0:max32 }
- multi **network.checksum_drop** = *none*: drop if checksum is bad { all | ip | noip | tcp | notcp | udp | noudp | icmp | noicmp | none }
- multi **network.checksum_eval** = *all*: checksums to verify { all | ip | noip | tcp | notcp | udp | noudp | icmp | noicmp | none }
- int **network.id** = 0: correlate unified2 events with configuration { 0:18446744073709551614 }
- int **network.layers** = 40: the maximum number of protocols that Snort can correctly decode { 3:255 }
- int **network.max_ip6_extensions** = 0: the maximum number of IP6 options Snort will process for a given IPv6 layer before raising 116:456 (0 = unlimited) { 0:255 }
- int **network.max_ip_layers** = 0: the maximum number of IP layers Snort will process for a given packet before raising 116:293 (0 = unlimited) { 0:255 }
- int **network.min_ttl** = 1: alert / normalize packets with lower TTL / hop limit (you must enable rules and / or normalization also) { 1:255 }
- int **network.new_ttl** = 1: use this value for responses and when normalizing { 1:255 }
- bool **normalizer.icmp4** = false: clear reserved flag
- bool **normalizer.icmp6** = false: clear reserved flag
- bool **normalizer.ip4.base** = false: clear options
- bool **normalizer.ip4.df** = false: clear don't frag flag
- bool **normalizer.ip4.rf** = false: clear reserved flag
- bool **normalizer.ip4.tos** = false: clear tos / differentiated services byte
- bool **normalizer.ip4.trim** = false: truncate excess payload beyond datagram length
- bool **normalizer.ip6** = false: clear reserved flag
- string **normalizer.tcp.allow_codes**: don't clear given option codes
- multi **normalizer.tcp.allow_names**: don't clear given option names { sack | echo | partial_order | conn_count | alt_checksum | md5 }
- bool **normalizer.tcp.base** = false: clear reserved bits and option padding and fix urgent pointer / flags issues
- bool **normalizer.tcp.block** = false: allow packet drops during TCP normalization
- select **normalizer.tcp.ecn** = *off*: clear ecn for all packets | sessions w/o ecn setup { off | packet | stream }
- bool **normalizer.tcp.ips** = true: ensure consistency in retransmitted data
- bool **normalizer.tcp.opts** = false: clear all options except mss, wscale, timestamp, and any explicitly allowed

- bool **normalizer.tcp.pad** = false: clear any option padding bytes
- bool **normalizer.tcp.req_pay** = false: clear the urgent pointer and the urgent flag if there is no payload
- bool **normalizer.tcp.req_urg** = false: clear the urgent pointer if the urgent flag is not set
- bool **normalizer.tcp.req_urp** = false: clear the urgent flag if the urgent pointer is not set
- bool **normalizer.tcp.rsv** = false: clear the reserved bits in the TCP header
- bool **normalizer.tcp.trim_mss** = false: trim data to MSS
- bool **normalizer.tcp.trim_rst** = false: remove any data from RST packet
- bool **normalizer.tcp.trim_syn** = false: remove data on SYN
- bool **normalizer.tcp.trim_win** = false: trim data to window
- bool **normalizer.tcp.urp** = false: adjust urgent pointer if beyond segment length
- bool **output.dump_chars_only** = false: turns on character dumps (same as -C)
- bool **output.dump_payload** = false: dumps application layer (same as -d)
- bool **output.dump_payload_verbose** = false: dumps raw packet starting at link layer (same as -X)
- int **output.event_trace.max_data** = 0: maximum amount of packet data to capture { 0:65535 }
- string **output.logdir** = .: where to put log files (same as -l)
- bool **output.obfuscate** = false: obfuscate the logged IP addresses (same as -O)
- bool **output.quiet** = false: suppress normal logging on stdout (same as -q)
- bool **output.show_year** = false: include year in timestamp in the alert and log files (same as -y)
- int **output.tagged_packet_limit** = 256: maximum number of packets tagged for non-packet metrics { 0:max32 }
- bool **output.verbose** = false: be verbose (same as -v)
- bool **output.wide_hex_dump** = false: output 20 bytes per lines instead of 16 when dumping buffers
- bool **packet_capture.enable** = false: initially enable packet dumping
- string **packet_capture.filter**: bpf filter to use for packet dump
- int **packet_capture.group** = -1: group filter to use for the packet dump { -1:32767 }
- bool **packets.address_space_agnostic** = false: determines whether DAQ address space info is used to track fragments and connections
- string **packets.bpf_file**: file with BPF to select traffic for Snort
- int **packets.limit** = 0: maximum number of packets to process before stopping (0 is unlimited) { 0:max53 }
- bool **packets.mpls_agnostic** = true: determines whether MPLS labels are used to track fragments and connections
- int **packets.skip** = 0: number of packets to skip before before processing { 0:max53 }
- bool **packets.vlan_agnostic** = false: determines whether VLAN tags are used to track fragments and connections
- bool **packet_tracer.enable** = false: enable summary output of state that determined packet verdict
- enum **packet_tracer.output** = *console*: select where to send packet trace { *console* | *file* }
- string **pcre.~re**: Snort regular expression
- bool **perf_monitor.base** = true: enable base statistics

- bool **perf_monitor.cpu** = false: enable cpu statistics
- bool **perf_monitor.flow** = false: enable traffic statistics
- bool **perf_monitor.flow_ip** = false: enable statistics on host pairs
- int **perf_monitor.flow_ip_memcap** = 52428800: maximum memory in bytes for flow tracking { 236:maxSZ }
- int **perf_monitor.flow_ports** = 1023: maximum ports to track { 0:65535 }
- enum **perf_monitor.format** = *csv*: output format for stats { *csv* | *text* | *json* }
- int **perf_monitor.max_file_size** = 1073741824: files will be rolled over if they exceed this size { 4096:max53 }
- string **`perf_monitor.modules[].name`**: name of the module
- string **`perf_monitor.modules[].pegs`**: list of statistics to track or empty for all counters
- enum **perf_monitor.output** = *file*: output location for stats { *file* | *console* }
- int **perf_monitor.packets** = 10000: minimum packets to report { 0:max32 }
- int **perf_monitor.seconds** = 60: report interval { 0:max32 }
- bool **perf_monitor.summary** = false: output summary at shutdown
- interval **pkt_num.~range**: check if packet number is in given range { 1: }
- int **pop.b64_decode_depth** = -1: base64 decoding depth (-1 no limit) { -1:65535 }
- int **pop.bitenc_decode_depth** = -1: Non-Encoded MIME attachment extraction depth (-1 no limit) { -1:65535 }
- bool **pop.decompress_pdf** = false: decompress pdf files in MIME attachments
- bool **pop.decompress_swf** = false: decompress swf files in MIME attachments
- bool **pop.decompress_vba** = false: decompress MS Office Visual Basic for Applications macro files in MIME attachments
- bool **pop.decompress_zip** = false: decompress zip files in MIME attachments
- int **pop.qp_decode_depth** = -1: Quoted Printable decoding depth (-1 no limit) { -1:65535 }
- int **pop.uu_decode_depth** = -1: Unix-to-Unix decoding depth (-1 no limit) { -1:65535 }
- bool **port_scan.alert_all** = false: alert on all events over threshold within window if true; else alert on first only
- int **port_scan.icmp_sweep.nets** = 25: number of times address changed from prior attempt { 0:65535 }
- int **port_scan.icmp_sweep.ports** = 25: number of times port (or proto) changed from prior attempt { 0:65535 }
- int **port_scan.icmp_sweep.rejects** = 15: scan attempts with negative response { 0:65535 }
- int **port_scan.icmp_sweep.scans** = 100: scan attempts { 0:65535 }
- int **port_scan.icmp_window** = 0: detection interval for all ICMP scans { 0:max32 }
- string **port_scan.ignore_scanned**: list of CIDRs with optional ports to ignore if the destination of scan alerts
- string **port_scan.ignore_scanners**: list of CIDRs with optional ports to ignore if the source of scan alerts
- bool **port_scan.include_midstream** = false: list of CIDRs with optional ports
- int **port_scan.ip_decoy.nets** = 25: number of times address changed from prior attempt { 0:65535 }
- int **port_scan.ip_decoy.ports** = 25: number of times port (or proto) changed from prior attempt { 0:65535 }
- int **port_scan.ip_decoy.rejects** = 15: scan attempts with negative response { 0:65535 }
- int **port_scan.ip_decoy.scans** = 100: scan attempts { 0:65535 }

- int **port_scan.ip_dist.nets** = 25: number of times address changed from prior attempt { 0:65535 }
- int **port_scan.ip_dist.ports** = 25: number of times port (or proto) changed from prior attempt { 0:65535 }
- int **port_scan.ip_dist.rejects** = 15: scan attempts with negative response { 0:65535 }
- int **port_scan.ip_dist.scans** = 100: scan attempts { 0:65535 }
- int **port_scan.ip_proto.nets** = 25: number of times address changed from prior attempt { 0:65535 }
- int **port_scan.ip_proto.ports** = 25: number of times port (or proto) changed from prior attempt { 0:65535 }
- int **port_scan.ip_proto.rejects** = 15: scan attempts with negative response { 0:65535 }
- int **port_scan.ip_proto.scans** = 100: scan attempts { 0:65535 }
- int **port_scan.ip_sweep.nets** = 25: number of times address changed from prior attempt { 0:65535 }
- int **port_scan.ip_sweep.ports** = 25: number of times port (or proto) changed from prior attempt { 0:65535 }
- int **port_scan.ip_sweep.rejects** = 15: scan attempts with negative response { 0:65535 }
- int **port_scan.ip_sweep.scans** = 100: scan attempts { 0:65535 }
- int **port_scan.ip_window** = 0: detection interval for all IP scans { 0:max32 }
- int **port_scan.memcap** = 10485760: maximum tracker memory in bytes { 1024:maxSZ }
- multi **port_scan.protos** = *all*: choose the protocols to monitor { tcp | udp | icmp | ip | all }
- multi **port_scan.scan_types** = *all*: choose type of scans to look for { portscan | portsweep | decoy_portscan | distributed_portscan | all }
- int **port_scan.tcp_decoy.nets** = 25: number of times address changed from prior attempt { 0:65535 }
- int **port_scan.tcp_decoy.ports** = 25: number of times port (or proto) changed from prior attempt { 0:65535 }
- int **port_scan.tcp_decoy.rejects** = 15: scan attempts with negative response { 0:65535 }
- int **port_scan.tcp_decoy.scans** = 100: scan attempts { 0:65535 }
- int **port_scan.tcp_dist.nets** = 25: number of times address changed from prior attempt { 0:65535 }
- int **port_scan.tcp_dist.ports** = 25: number of times port (or proto) changed from prior attempt { 0:65535 }
- int **port_scan.tcp_dist.rejects** = 15: scan attempts with negative response { 0:65535 }
- int **port_scan.tcp_dist.scans** = 100: scan attempts { 0:65535 }
- int **port_scan.tcp_ports.nets** = 25: number of times address changed from prior attempt { 0:65535 }
- int **port_scan.tcp_ports.ports** = 25: number of times port (or proto) changed from prior attempt { 0:65535 }
- int **port_scan.tcp_ports.rejects** = 15: scan attempts with negative response { 0:65535 }
- int **port_scan.tcp_ports.scans** = 100: scan attempts { 0:65535 }
- int **port_scan.tcp_sweep.nets** = 25: number of times address changed from prior attempt { 0:65535 }
- int **port_scan.tcp_sweep.ports** = 25: number of times port (or proto) changed from prior attempt { 0:65535 }
- int **port_scan.tcp_sweep.rejects** = 15: scan attempts with negative response { 0:65535 }
- int **port_scan.tcp_sweep.scans** = 100: scan attempts { 0:65535 }
- int **port_scan.tcp_window** = 0: detection interval for all TCP scans { 0:max32 }
- int **port_scan.udp_decoy.nets** = 25: number of times address changed from prior attempt { 0:65535 }

- int **port_scan.udp_decoy.ports** = 25: number of times port (or proto) changed from prior attempt { 0:65535 }
- int **port_scan.udp_decoy.rejects** = 15: scan attempts with negative response { 0:65535 }
- int **port_scan.udp_decoy.scans** = 100: scan attempts { 0:65535 }
- int **port_scan.udp_dist.nets** = 25: number of times address changed from prior attempt { 0:65535 }
- int **port_scan.udp_dist.ports** = 25: number of times port (or proto) changed from prior attempt { 0:65535 }
- int **port_scan.udp_dist.rejects** = 15: scan attempts with negative response { 0:65535 }
- int **port_scan.udp_dist.scans** = 100: scan attempts { 0:65535 }
- int **port_scan.udp_ports.nets** = 25: number of times address changed from prior attempt { 0:65535 }
- int **port_scan.udp_ports.ports** = 25: number of times port (or proto) changed from prior attempt { 0:65535 }
- int **port_scan.udp_ports.rejects** = 15: scan attempts with negative response { 0:65535 }
- int **port_scan.udp_ports.scans** = 100: scan attempts { 0:65535 }
- int **port_scan.udp_sweep.nets** = 25: number of times address changed from prior attempt { 0:65535 }
- int **port_scan.udp_sweep.ports** = 25: number of times port (or proto) changed from prior attempt { 0:65535 }
- int **port_scan.udp_sweep.rejects** = 15: scan attempts with negative response { 0:65535 }
- int **port_scan.udp_sweep.scans** = 100: scan attempts { 0:65535 }
- int **port_scan.udp_window** = 0: detection interval for all UDP scans { 0:max32 }
- string **port_scan.watch_ip**: list of CIDRs with optional ports to watch
- int **priority.~**: relative severity level; 1 is highest priority { 1:max31 }
- string **process.chroot**: set chroot directory (same as -t)
- bool **process.daemon** = false: fork as a daemon (same as -D)
- bool **process.dirty_pig** = false: shutdown without internal cleanup
- string **process.set_gid**: set group ID (same as -g)
- string **process.set_uid**: set user ID (same as -u)
- string **`process.threads[].cpuset`**: pin the associated thread to this cpuset
- string **`process.threads[].name`**: define which threads will have specified affinity, by thread name
- int **`process.threads[].thread`**: set cpu affinity for the <cur_thread_num> thread that runs { 0:65535 }
- enum **`process.threads[].type`**: define which threads will have specified affinity, by their type { *other|packet|main* }
- int **process.umask**: set process umask (same as -m) { 0x000:0x1FF }
- bool **process.utc** = false: use UTC instead of local time for timestamps
- int **process.watchdog_min_thread_count** = 1: minimum unresponsive threads for watchdog to trigger { 1:65535 }
- int **process.watchdog_timer** = 0: watchdog timer for packet threads (seconds, 0 to disable) { 0:60 }
- int **profiler.memory.count** = 0: limit results to count items per level (0 = no limit) { 0:max32 }
- int **profiler.memory.dump_file_size** = 1073741824: files will be rolled over if they exceed this size { 4096:max53 }
- int **profiler.memory.max_depth** = -1: limit depth to max_depth (-1 = no limit) { -1:255 }
- bool **profiler.memory.show** = true: show module memory profile stats

- enum **profiler.memory.sort** = *total_used*: sort by given field { *none* | *allocations* | *total_used* | *avg_allocation* }
- int **profiler.modules.count** = 0: limit results to count items per level (0 = no limit) { 0:max32 }
- int **profiler.modules.max_depth** = -1: limit depth to max_depth (-1 = no limit) { -1:255 }
- bool **profiler.modules.show** = true: show module time profile stats
- enum **profiler.modules.sort** = *total_time*: sort by given field { *none* | *checks* | *avg_check* | *total_time* }
- int **profiler.rules.count** = 0: print results to given level (0 = all) { 0:max32 }
- bool **profiler.rules.show** = true: show rule time profile stats
- enum **profiler.rules.sort** = *total_time*: sort by given field { *none* | *checks* | *avg_check* | *total_time* | *matches* | *no_matches* | *avg_match* | *avg_no_match* }
- string **`rate_filter[].apply_to`**: restrict filter to these addresses according to track
- int **`rate_filter[].count`** = 1: number of events in interval before tripping { 0:max32 }
- int **`rate_filter[].gid`** = 1: rule generator ID { 0:8129 }
- dynamic **`rate_filter[].new_action`** = alert: take this action on future hits until timeout { alert | block | drop | file_id | log | pass | react | reject | rewrite }
- int **`rate_filter[].seconds`** = 1: count interval { 0:max32 }
- int **`rate_filter[].sid`** = 1: rule signature ID { 0:max32 }
- int **`rate_filter[].timeout`** = 1: count interval { 0:max32 }
- enum **`rate_filter[].track`** = *by_src*: filter only matching source or destination addresses { *by_src* | *by_dst* | *by_rule* }
- string **react.page**: file containing HTTP response body
- string **reference.~ref**: reference: <scheme>,<id>
- string **`references[].name`**: name used with reference rule option
- string **`references[].url`**: where this reference is defined
- implied **regex.dotall**: matching a . will not exclude newlines
- implied **regex.fast_pattern**: use this content in the fast pattern matcher instead of the content selected by default
- implied **regex.multiline**: ^ and $ anchors match any newlines in data
- implied **regex.nocase**: case insensitive match
- string **regex.~re**: hyperscan regular expression
- implied **regex.relative**: start search from end of last match instead of start of buffer
- enum **reject.control** = *none*: send ICMP unreachable(s) { *none|network|host|port|forward|all* }
- enum **reject.reset** = *both*: send TCP reset to one or both ends { *none|source|dest|both* }
- string **rem.~**: comment
- string **replace.~**: byte code to replace with
- enum **reputation.allow** = *do_not_block*: specify the meaning of allowlist { *do_not_block|trust* }
- string **reputation.allowlist**: allowlist file name with IP lists
- string **reputation.blocklist**: blocklist file name with IP lists
- string **reputation.list_dir**: directory for IP lists and manifest file

- int **reputation.memcap** = 500: maximum total MB of memory allocated { 1:4095 }
- enum **reputation.nested_ip** = *inner*: IP to use when there is IP encapsulation { *inner|outer|all* }
- enum **reputation.priority** = *allowlist*: defines priority when there is a decision conflict during run-time { *blocklist|allowlist* }
- bool **reputation.scan_local** = false: inspect local address defined in RFC 1918
- int **rev.~**: revision { 1:max32 }
- string **rna.dump_file**: file name to dump RNA mac cache on shutdown; won't dump by default
- bool **rna.enable_logger** = true: enable or disable writing discovery events into logger
- bool **rna.log_when_idle** = false: enable host update logging when snort is idle
- string **rna.rna_conf_path**: path to rna configuration
- string **rna.smb_fingerprints[].device**: device information
- bool **rna.smb_fingerprints[].df** = false: fingerprint don't fragment flag
- string **rna.smb_fingerprints[].dhcp55**: dhcp option 55 values
- string **rna.smb_fingerprints[].dhcp60**: dhcp option 60 values
- int **rna.smb_fingerprints[].flags**: smb flags { 0:max32 }
- int **rna.smb_fingerprints[].fpid** = 0: fingerprint id { 0:max32 }
- string **rna.smb_fingerprints[].host_name**: host name information
- string **rna.smb_fingerprints[].id** = *X*: id
- int **rna.smb_fingerprints[].major**: smb major version { 0:max31 }
- int **rna.smb_fingerprints[].minor**: smb minor version { 0:max31 }
- string **rna.smb_fingerprints[].mss** = *X*: fingerprint mss
- string **rna.smb_fingerprints[].tcp_window**: fingerprint tcp window
- string **rna.smb_fingerprints[].topts**: fingerprint tcp options
- int **rna.smb_fingerprints[].ttl** = 0: fingerprint ttl { 0:255 }
- int **rna.smb_fingerprints[].type** = 0: fingerprint type { 0:max32 }
- enum **rna.smb_fingerprints[].ua_type** = *os*: type of user agent fingerprints { *os* | *device* | *jail-broken* | *jail-broken-host* }
- string **rna.smb_fingerprints[].user_agent[].substring**: a substring of user agent string
- string **rna.smb_fingerprints[].uuid**: fingerprint uuid
- string **rna.smb_fingerprints[].ws** = *X*: fingerprint window size
- string **rna.tcp_fingerprints[].device**: device information
- bool **rna.tcp_fingerprints[].df** = false: fingerprint don't fragment flag
- string **rna.tcp_fingerprints[].dhcp55**: dhcp option 55 values
- string **rna.tcp_fingerprints[].dhcp60**: dhcp option 60 values
- int **rna.tcp_fingerprints[].flags**: smb flags { 0:max32 }
- int **rna.tcp_fingerprints[].fpid** = 0: fingerprint id { 0:max32 }

- string **rna.tcp_fingerprints[].host_name**: host name information
- string **rna.tcp_fingerprints[].id** = *X*: id
- int **rna.tcp_fingerprints[].major**: smb major version { 0:max31 }
- int **rna.tcp_fingerprints[].minor**: smb minor version { 0:max31 }
- string **rna.tcp_fingerprints[].mss** = *X*: fingerprint mss
- string **rna.tcp_fingerprints[].tcp_window**: fingerprint tcp window
- string **rna.tcp_fingerprints[].topts**: fingerprint tcp options
- int **rna.tcp_fingerprints[].ttl** = 0: fingerprint ttl { 0:255 }
- int **rna.tcp_fingerprints[].type** = 0: fingerprint type { 0:max32 }
- enum **rna.tcp_fingerprints[].ua_type** = *os*: type of user agent fingerprints { *os* | *device* | *jail-broken* | *jail-broken-host* }
- string **rna.tcp_fingerprints[].user_agent[].substring**: a substring of user agent string
- string **rna.tcp_fingerprints[].uuid**: fingerprint uuid
- string **rna.tcp_fingerprints[].ws** = *X*: fingerprint window size
- string **rna.ua_fingerprints[].device**: device information
- bool **rna.ua_fingerprints[].df** = false: fingerprint don't fragment flag
- string **rna.ua_fingerprints[].dhcp55**: dhcp option 55 values
- string **rna.ua_fingerprints[].dhcp60**: dhcp option 60 values
- int **rna.ua_fingerprints[].flags**: smb flags { 0:max32 }
- int **rna.ua_fingerprints[].fpid** = 0: fingerprint id { 0:max32 }
- string **rna.ua_fingerprints[].host_name**: host name information
- string **rna.ua_fingerprints[].id** = *X*: id
- int **rna.ua_fingerprints[].major**: smb major version { 0:max31 }
- int **rna.ua_fingerprints[].minor**: smb minor version { 0:max31 }
- string **rna.ua_fingerprints[].mss** = *X*: fingerprint mss
- string **rna.ua_fingerprints[].tcp_window**: fingerprint tcp window
- string **rna.ua_fingerprints[].topts**: fingerprint tcp options
- int **rna.ua_fingerprints[].ttl** = 0: fingerprint ttl { 0:255 }
- int **rna.ua_fingerprints[].type** = 0: fingerprint type { 0:max32 }
- enum **rna.ua_fingerprints[].ua_type** = *os*: type of user agent fingerprints { *os* | *device* | *jail-broken* | *jail-broken-host* }
- string **rna.ua_fingerprints[].user_agent[].substring**: a substring of user agent string
- string **rna.ua_fingerprints[].uuid**: fingerprint uuid
- string **rna.ua_fingerprints[].ws** = *X*: fingerprint window size
- string **rna.udp_fingerprints[].device**: device information
- bool **rna.udp_fingerprints[].df** = false: fingerprint don't fragment flag

- string **rna.udp_fingerprints[].dhcp55**: dhcp option 55 values
- string **rna.udp_fingerprints[].dhcp60**: dhcp option 60 values
- int **rna.udp_fingerprints[].flags**: smb flags { 0:max32 }
- int **rna.udp_fingerprints[].fpid** = 0: fingerprint id { 0:max32 }
- string **rna.udp_fingerprints[].host_name**: host name information
- string **rna.udp_fingerprints[].id** = *X*: id
- int **rna.udp_fingerprints[].major**: smb major version { 0:max31 }
- int **rna.udp_fingerprints[].minor**: smb minor version { 0:max31 }
- string **rna.udp_fingerprints[].mss** = *X*: fingerprint mss
- string **rna.udp_fingerprints[].tcp_window**: fingerprint tcp window
- string **rna.udp_fingerprints[].topts**: fingerprint tcp options
- int **rna.udp_fingerprints[].ttl** = 0: fingerprint ttl { 0:255 }
- int **rna.udp_fingerprints[].type** = 0: fingerprint type { 0:max32 }
- enum **rna.udp_fingerprints[].ua_type** = *os*: type of user agent fingerprints { *os* | *device* | *jail-broken* | *jail-broken-host* }
- string **rna.udp_fingerprints[].user_agent[].substring**: a substring of user agent string
- string **rna.udp_fingerprints[].uuid**: fingerprint uuid
- string **rna.udp_fingerprints[].ws** = *X*: fingerprint window size
- int **rpc.~app**: application number { 0:max32 }
- string **rpc.~proc**: procedure number or * for any
- string **rpc.~ver**: version number or * for any
- string **s7commplus_func.~**: function code to match
- string **s7commplus_opcode.~**: opcode code to match
- string **sd_pattern.~pattern**: The pattern to search for
- int **sd_pattern.threshold** = 1: number of matches before alerting { 1:max32 }
- int **search_engine.bleedover_port_limit** = 1024: maximum ports in rule before demotion to any-any port group { 1:max32 }
- bool **search_engine.bleedover_warnings_enabled** = false: print warning if a rule is demoted to any-any port group
- bool **search_engine.debug** = false: print verbose fast pattern info
- bool **search_engine.debug_print_nocontent_rule_tests** = false: print rule group info during packet evaluation
- bool **search_engine.debug_print_rule_group_build_details** = false: print rule group info during compilation
- bool **search_engine.debug_print_rule_groups_compiled** = false: prints compiled rule group information
- bool **search_engine.debug_print_rule_groups_uncompiled** = false: prints uncompiled rule group information
- bool **search_engine.detect_raw_tcp** = false: detect on TCP payload before reassembly
- bool **search_engine.enable_single_rule_group** = false: put all rules into one group
- int **search_engine.max_pattern_len** = 0: truncate patterns when compiling into state machine (0 means no maximum) { 0:max32 }

- int **search_engine.max_queue_events** = 5: maximum number of matching fast pattern states to queue per packet { 2:100 }
- dynamic **search_engine.offload_search_method**: set fast pattern offload algorithm - choose available search engine { ac_bnfa | ac_full | hyperscan | lowmem }
- int **search_engine.queue_limit** = 0: maximum number of fast pattern matches to queue per packet (0 is unlimited) { 0:max32 }
- string **search_engine.rule_db_dir**: deserialize rule databases from given directory
- dynamic **search_engine.search_method** = ac_bnfa: set fast pattern algorithm - choose available search engine { ac_bnfa | ac_full | hyperscan | lowmem }
- bool **search_engine.show_fast_patterns** = false: print fast pattern info for each rule
- bool **search_engine.split_any_any** = true: evaluate any-any rules separately to save memory
- interval **seq.~range**: check if TCP sequence number is in given range { 0: }
- string **`service.*`**: one or more comma-separated service names
- string **sha256.~hash**: data to match
- int **sha256.length**: number of octets in plain text { 1:65535 }
- string **sha256.offset**: var or number of bytes from start of buffer to start search
- implied **sha256.relative** = *false*: offset from cursor instead of start of buffer
- string **sha512.~hash**: data to match
- int **sha512.length**: number of octets in plain text { 1:65535 }
- string **sha512.offset**: var or number of bytes from start of buffer to start search
- implied **sha512.relative** = *false*: offset from cursor instead of start of buffer
- string **`side_channel[].connector`**: connector handle
- string **`side_channel[].connectors[].connector`**: connector handle
- bit_list **`side_channel[].ports`**: side channel message port list { 65535 }
- int **sid.~**: signature id { 1:max32 }
- bool **sip.ignore_call_channel** = false: enables the support for ignoring audio/video data channel
- int **sip.max_call_id_len** = 256: maximum call id field size { 0:65535 }
- int **sip.max_contact_len** = 256: maximum contact field size { 0:65535 }
- int **sip.max_content_len** = 1024: maximum content length of the message body { 0:65535 }
- int **sip.max_dialogs** = 4: maximum number of dialogs within one stream session { 1:max32 }
- int **sip.max_from_len** = 256: maximum from field size { 0:65535 }
- int **sip.max_requestName_len** = 20: deprecated - use max_request_name_len instead { 0:65535 }
- int **sip.max_request_name_len** = 20: maximum request name field size { 0:65535 }
- int **sip.max_to_len** = 256: maximum to field size { 0:65535 }
- int **sip.max_uri_len** = 256: maximum request uri field size { 0:65535 }
- int **sip.max_via_len** = 1024: maximum via field size { 0:65535 }
- string **`sip_method.*method`**: sip method

- string **sip.methods** = *invite cancel ack bye register options*: list of methods to check in SIP messages
- int **sip_stat_code.*code**: status code { 1:999 }
- string **smtp.alt_max_command_line_len[].command**: command string
- int **smtp.alt_max_command_line_len[].length** = 0: specify non-default maximum for command { 0:max32 }
- string **smtp.auth_cmds**: commands that initiate an authentication exchange
- int **smtp.b64_decode_depth** = -1: depth used to decode the base64 encoded MIME attachments (-1 no limit) { -1:65535 }
- string **smtp.binary_data_cmds**: commands that initiate sending of data and use a length value after the command
- int **smtp.bitenc_decode_depth** = -1: depth used to extract the non-encoded MIME attachments (-1 no limit) { -1:65535 }
- string **smtp.data_cmds**: commands that initiate sending of data with an end of data delimiter
- bool **smtp.decompress_pdf** = false: decompress pdf files in MIME attachments
- bool **smtp.decompress_swf** = false: decompress swf files in MIME attachments
- bool **smtp.decompress_vba** = false: decompress MS Office Visual Basic for Applications macro files in MIME attachments
- bool **smtp.decompress_zip** = false: decompress zip files in MIME attachments
- int **smtp.email_hdrs_log_depth** = 1464: depth for logging email headers { 0:20480 }
- bool **smtp.ignore_data** = false: ignore data section of mail
- bool **smtp.ignore_tls_data** = false: ignore TLS-encrypted data when processing rules
- string **smtp.invalid_cmds**: alert if this command is sent from client side
- bool **smtp.log_email_hdrs** = false: log the SMTP email headers extracted from SMTP data
- bool **smtp.log_filename** = false: log the MIME attachment filenames extracted from the Content-Disposition header within the MIME body
- bool **smtp.log_mailfrom** = false: log the sender's email address extracted from the MAIL FROM command
- bool **smtp.log_rcptto** = false: log the recipient's email address extracted from the RCPT TO command
- int **smtp.max_auth_command_line_len** = 1000: max auth command Line Length { 0:65535 }
- int **smtp.max_command_line_len** = 512: max Command Line Length { 0:65535 }
- int **smtp.max_header_line_len** = 1000: max SMTP DATA header line { 0:65535 }
- int **smtp.max_response_line_len** = 512: max SMTP response line { 0:65535 }
- string **smtp.normalize_cmds**: list of commands to normalize
- enum **smtp.normalize** = *none*: turns on/off normalization { *none* | *cmds* | *all* }
- int **smtp.qp_decode_depth** = -1: quoted-Printable decoding depth (-1 no limit) { -1:65535 }
- int **smtp.uu_decode_depth** = -1: Unix-to-Unix decoding depth (-1 no limit) { -1:65535 }
- string **smtp.valid_cmds**: list of valid commands
- enum **smtp.xlink2state** = *alert*: enable/disable xlink2state alert { *disable* | *alert* | *drop* }
- implied **snort.--alert-before-pass**: evaluate alert rules before pass rules; default is pass rules first
- string **snort.-A**: <mode> set alert mode: none, cmg, or alert_*
- addr **snort.-B** = *255.255.255.255/32*: <mask> obfuscated IP addresses in alerts and packet dumps using CIDR mask

- string **snort.--bpf**: <filter options> are standard BPF options, as seen in TCPDump
- string **snort.--c2x**: output hex for given char (see also --x2c)
- string **snort.-c**: <conf> use this configuration
- string **snort.--control-socket**: <file> to create unix socket
- implied **snort.-C**: print out payloads with character data only (no hex)
- implied **snort.--create-pidfile**: create PID file, even when not in Daemon mode
- int **snort.--daq-batch-size**: <size> set the DAQ receive batch size; default is 64 { 1: }
- string **snort.--daq-dir**: <dir> tell snort where to find desired DAQ
- implied **snort.--daq-list**: list packet acquisition modules available in optional dir, default is static modules only
- enum **snort.--daq-mode**: <mode> select DAQ module operating mode (overrides automatic selection) { *passive* | *inline* | *read-file* }
- string **snort.--daq**: <type> select packet acquisition module (default is pcap)
- string **snort.--daq-var**: <name=value> specify extra DAQ configuration variable
- implied **snort.-d**: dump the Application Layer
- implied **snort.--dirty-pig**: don't flush packets on shutdown
- implied **snort.-D**: run Snort in background (daemon) mode
- string **snort.--dump-builtin-options**: additional options to include with --dump-builtin-rules stubs
- string **snort.--dump-builtin-rules**: [<module prefix>] output stub rules for selected modules { (optional) }
- select **snort.--dump-config**: dump config in json format { all | top }
- implied **snort.--dump-config-text**: dump config in text format
- string **snort.--dump-defaults**: [<module prefix>] output module defaults in Lua format { (optional) }
- implied **snort.--dump-dynamic-rules**: output stub rules for all loaded rules libraries
- string **snort.--dump-rule-databases**: dump rule databases to given directory (hyperscan only)
- implied **snort.--dump-rule-deps**: dump rule dependencies in json format for use by other tools
- implied **snort.--dump-rule-meta**: dump configured rule info in json format for use by other tools
- implied **snort.--dump-rule-state**: dump configured rule state in json format for use by other tools
- implied **snort.--dump-version**: output the version, the whole version, and only the version
- implied **snort.-e**: display the second layer header info
- implied **snort.--enable-inline-test**: enable Inline-Test Mode Operation
- implied **snort.--enable-test-features**: enable features used in testing
- implied **snort.-f**: turn off fflush() calls after binary log writes
- int **snort.-G**: <0xid> (same as --logid) { 0:65535 }
- implied **snort.--gen-msg-map**: dump configured rules in gen-msg.map format for use by other tools
- string **snort.-g**: <gname> run snort gid as <gname> group (or gid) after initialization
- string **snort.--help-commands**: [<module prefix>] output matching commands { (optional) }

- string **snort.--help-config**: [<module prefix>] output matching config options { (optional) }
- string **snort.--help-counts**: [<module prefix>] output matching peg counts { (optional) }
- implied **snort.--help-limits**: print the int upper bounds denoted by max*
- string **snort.--help-module**: <module> output description of given module
- implied **snort.--help-modules-json**: dump description of all available modules in JSON format
- implied **snort.--help-modules**: list all available modules with brief help
- string **snort.--help-options**: [<option prefix>] output matching command line option quick help (same as -?) { (optional) }
- implied **snort.--help-plugins**: list all available plugins with brief help
- implied **snort.--help**: show help overview
- implied **snort.--help-signals**: dump available control signals
- implied **snort.-H**: make hash tables deterministic
- implied **snort.-h**: show help overview (same as --help)
- int **snort.--id-offset** = 0: offset to add to instance IDs when logging to files { 0:65535 }
- implied **snort.--id-subdir**: create/use instance subdirectories in logdir instead of instance filename prefix
- implied **snort.--id-zero**: use id prefix / subdirectory even with one packet thread
- string **snort.-i**: <iface>... list of interfaces
- string **snort.--include-path**: <path> where to find Lua and rule included files; searched before current or config directories
- port **snort.-j**: <port> to listen for Telnet connections
- enum **snort.-k** = *all*: <mode> checksum mode; default is all { *all*|*noip*|*notcp*|*noudp*|*noicmp*|*none* }
- implied **snort.--list-buffers**: output available inspection buffers
- string **snort.--list-builtin**: [<module prefix>] output matching builtin rules { (optional) }
- string **snort.--list-gids**: [<module prefix>] output matching generators { (optional) }
- string **snort.--list-modules**: [<module type>] list all known modules of given type { (optional) }
- implied **snort.--list-plugins**: list all known plugins
- string **snort.-l**: <logdir> log to this directory instead of current directory
- string **snort.-L**: <mode> logging mode (none, dump, pcap, or log_*)
- int **snort.--logid**: <0xid> log Identifier to uniquely id events for multiple snorts (same as -G) { 0:65535 }
- string **snort.--lua**: <chunk> extend/override conf with chunk; may be repeated
- string **snort.--lua-sandbox**: <file> file that contains the lua sandbox environment in which config will be loaded
- implied **snort.--markup**: output help in asciidoc compatible format
- int **snort.--max-packet-threads**: <count> configure maximum number of packet threads (same as -z) { 0:max32 }
- implied **snort.--mem-check**: like -T but also compile search engines
- string **snort.--metadata-filter**: <filter> load only rules containing filter string in metadata if set
- implied **snort.-M**: log messages to syslog (not alerts)
- int **snort.-m**: <umask> set the process file mode creation mask { 0x000:0x1FF }

- int **snort.-n**: <count> stop after count packets { 0:max53 }
- implied **snort.--nolock-pidfile**: do not try to lock Snort PID file
- implied **snort.--nostamps**: don't include timestamps in log file names
- implied **snort.--no-warn-flowbits**: ignore warnings about flowbits that are checked but not set and vice-versa
- implied **snort.--no-warn-rules**: ignore warnings about duplicate rules and rule parsing issues
- implied **snort.-O**: obfuscate the logged IP addresses
- string **snort.-?**: <option prefix> output matching command line option quick help (same as --help-options) { (optional) }
- implied **snort.--pause**: wait for resume/quit command before processing packets/terminating
- string **snort.--pcap-dir**: <dir> a directory to recurse to look for pcaps - read mode is implied
- string **snort.--pcap-file**: <file> file that contains a list of pcaps to read - read mode is implied
- string **snort.--pcap-filter**: <filter> filter to apply when getting pcaps from file or directory
- string **snort.--pcap-list**: <list> a space separated list of pcaps to read - read mode is implied
- int **snort.--pcap-loop**: <count> read all pcaps <count> times; 0 will read until Snort is terminated { 0:max32 }
- implied **snort.--pcap-no-filter**: reset to use no filter when getting pcaps from file or directory
- implied **snort.--pcap-show**: print a line saying what pcap is currently being read
- implied **snort.--pedantic**: warnings are fatal
- string **snort.--plugin-path**: <path> a colon separated list of directories or plugin libraries
- implied **snort.--process-all-events**: process all action groups
- implied **snort.-Q**: enable inline mode operation
- implied **snort.-q**: quiet mode - suppress normal logging on stdout
- string **snort.-r**: <pcap>... (same as --pcap-list)
- string **snort.-R**: <rules> include this rules file in the default policy
- string **snort.--rule-path**: <path> where to find rules files
- string **snort.--rule**: <rules> to be added to configuration; may be repeated
- implied **snort.--rule-to-hex**: output so rule header to stdout for text rule on stdin
- string **snort.--rule-to-text**: output plain so rule header to stdout for text rule on stdin (specify delimiter or [Snort_SO_Rule] will be used) { 16 }
- string **snort.--run-prefix**: <pfx> prepend this to each output file
- string **snort.--script-path**: <path> to a luajit script or directory containing luajit scripts
- implied **snort.--shell**: enable the interactive command line
- implied **snort.--show-file-codes**: indicate how files are located: A=absolute and W, F, C which are relative to the working directory, including file, and config file respectively
- implied **snort.--show-plugins**: list module and plugin versions
- int **snort.--skip**: <n> skip 1st n packets { 0:max53 }
- int **snort.--snaplen**: <snap> set snaplen of packet (same as -s) { 0:65535 }
- int **snort.-s**: <snap> (same as --snaplen); default is 1518 { 0:65535 }

- implied **snort.--stdin-rules**: read rules from stdin until EOF or a line starting with END is read
- implied **snort.--talos**: enable Talos tweak (same as --tweaks talos)
- string **snort.-t**: <dir> chroots process to <dir> after initialization
- implied **snort.-T**: test and report on the current Snort configuration
- string **snort.--tweaks**: tune configuration
- string **snort.-u**: <uname> run snort as <uname> or <uid> after initialization
- implied **snort.-U**: use UTC for timestamps
- implied **snort.-v**: be verbose
- implied **snort.--version**: show version number (same as -V)
- implied **snort.-V**: (same as --version)
- implied **snort.--warn-all**: enable all warnings
- implied **snort.--warn-conf-strict**: warn about unrecognized elements in configuration files
- implied **snort.--warn-conf**: warn about configuration issues
- implied **snort.--warn-daq**: warn about DAQ issues, usually related to mode
- implied **snort.--warn-flowbits**: warn about flowbits that are checked but not set and vice-versa
- implied **snort.--warn-hosts**: warn about host table issues
- implied **snort.--warn-plugins**: warn about issues that prevent plugins from loading
- implied **snort.--warn-rules**: warn about duplicate rules and rule parsing issues
- implied **snort.--warn-scripts**: warn about issues discovered while processing Lua scripts
- implied **snort.--warn-symbols**: warn about unknown symbols in your Lua config
- implied **snort.--warn-vars**: warn about variable definition and usage issues
- int **snort.--x2c**: output ASCII char for given hex (see also --c2x) { 0x00:0xFF }
- string **snort.--x2s**: output ASCII string for given byte code (see also --x2c)
- implied **snort.-X**: dump the raw packet data starting at the link layer
- implied **snort.-x**: same as --pedantic
- implied **snort.-y**: include year in timestamp in the alert and log files
- int **snort.-z**: <count> maximum number of packet threads (same as --max-packet-threads); 0 gets the number of CPU cores reported by the system; default is 1 { 0:max32 }
- string **so.~func**: name of eval function
- string **soid.~**: SO rule ID is unique key, eg <gid>_<sid>_<rev> like 3_45678_9
- implied **so.relative**: offset from cursor instead of start of buffer
- int **ssh.max_client_bytes** = 19600: number of unanswered bytes before alerting on challenge-response overflow or CRC32 { 0:65535 }
- int **ssh.max_encrypted_packets** = 25: ignore session after this many encrypted packets { 0:65535 }
- int **ssh.max_server_version_len** = 80: limit before alerting on secure CRT server version string overflow { 0:255 }
- int **ssl.max_heartbeat_length** = 0: maximum length of heartbeat record allowed { 0:65535 }

- implied **ssl_state.client_hello**: check for client hello
- implied **ssl_state.!client_hello**: check for records that are not client hello
- implied **ssl_state.client_keyx**: check for client keyx
- implied **ssl_state.!client_keyx**: check for records that are not client keyx
- implied **ssl_state.!server_hello**: check for records that are not server hello
- implied **ssl_state.server_hello**: check for server hello
- implied **ssl_state.!server_keyx**: check for records that are not server keyx
- implied **ssl_state.server_keyx**: check for server keyx
- implied **ssl_state.!unknown**: check for records that are not unknown
- implied **ssl_state.unknown**: check for unknown record
- bool **ssl.trust_servers** = false: disables requirement that application (encrypted) data must be observed on both sides
- implied **ssl_version.!sslv2**: check for records that are not sslv2
- implied **ssl_version.sslv2**: check for sslv2
- implied **ssl_version.!sslv3**: check for records that are not sslv3
- implied **ssl_version.sslv3**: check for sslv3
- implied **ssl_version.!tls1.0**: check for records that are not tls1.0
- implied **ssl_version.tls1.0**: check for tls1.0
- implied **ssl_version.!tls1.1**: check for records that are not tls1.1
- implied **ssl_version.tls1.1**: check for tls1.1
- implied **ssl_version.!tls1.2**: check for records that are not tls1.2
- implied **ssl_version.tls1.2**: check for tls1.2
- int **stream.file_cache.idle_timeout** = 180: maximum inactive time before retiring session tracker { 1:max32 }
- bool **stream_file.upload** = false: indicate file transfer direction
- int **stream.held_packet_timeout** = 1000: timeout in milliseconds for held packets { 1:max32 }
- int **stream.icmp_cache.idle_timeout** = 180: maximum inactive time before retiring session tracker { 1:max32 }
- int **stream_icmp.session_timeout** = 60: session tracking timeout { 1:max31 }
- int **stream.ip_cache.idle_timeout** = 180: maximum inactive time before retiring session tracker { 1:max32 }
- bool **stream.ip_frags_only** = false: don't process non-frag flows
- int **stream_ip.max_frags** = 8192: maximum number of simultaneous fragments being tracked { 1:max32 }
- int **stream_ip.max_overlaps** = 0: maximum allowed overlaps per datagram; 0 is unlimited { 0:max32 }
- int **stream_ip.min_frag_length** = 0: alert if fragment length is below this limit before or after trimming { 0:65535 }
- int **stream_ip.min_ttl** = 1: discard fragments with TTL below the minimum { 1:255 }
- enum **stream_ip.policy** = *linux*: fragment reassembly policy { *first* | *linux* | *bsd* | *bsd_right* | *last* | *windows* | *solaris* }
- int **stream_ip.session_timeout** = 60: session tracking timeout { 1:max31 }
- int **stream.max_flows** = 476288: maximum simultaneous flows tracked before pruning { 2:max32 }

- int **stream.prune_flows** = 10: maximum flows to prune at one time { 1:max32 }
- int **stream.pruning_timeout** = 30: minimum inactive time before being eligible for pruning { 1:max32 }
- enum **stream_reassemble.action**: stop or start stream reassembly { *disable|enable* }
- enum **stream_reassemble.direction**: action applies to the given direction(s) { *client|server|both* }
- implied **stream_reassemble.fastpath**: optionally trust the remainder of the session
- implied **stream_reassemble.noalert**: don't alert when rule matches
- enum **stream_size.~direction**: compare applies to the given direction(s) { *either|to_server|to_client|both* }
- interval **stream_size.~range**: check if the stream size is in the given range { 0: }
- int **stream.tcp_cache.idle_timeout** = 3600: maximum inactive time before retiring session tracker { 1:max32 }
- int **stream_tcp.embryonic_timeout** = 30: Non-established connection timeout { 1:max31 }
- int **stream_tcp.flush_factor** = 0: flush upon seeing a drop in segment size after given number of non-decreasing segments { 0:65535 }
- int **stream_tcp.idle_timeout** = 3600: session deletion on idle { 1:max31 }
- int **stream_tcp.max_pdu** = 16384: maximum reassembled PDU size { 1460:32768 }
- int **stream_tcp.max_window** = 0: maximum allowed TCP window { 0:1073725440 }
- bool **stream_tcp.no_ack** = false: received data is implicitly acked immediately
- int **stream_tcp.overlap_limit** = 0: maximum number of allowed overlapping segments per session { 0:max32 }
- enum **stream_tcp.policy** = *bsd*: determines operating system characteristics like reassembly { *first* | *last* | *linux* | *old_linux* | *bsd* | *macos* | *solaris* | *irix* | *hpux11* | *hpux10* | *windows* | *win_2003* | *vista* | *proxy* }
- int **stream_tcp.queue_limit.max_bytes** = 4194304: don't queue more than given bytes per session and direction, 0 = unlimited { 0:max32 }
- int **stream_tcp.queue_limit.max_segments** = 3072: don't queue more than given segments per session and direction, 0 = unlimited { 0:max32 }
- bool **stream_tcp.reassemble_async** = true: queue data for reassembly before traffic is seen in both directions
- int **stream_tcp.require_3whs** = -1: don't track midstream sessions after given seconds from start up; -1 tracks all { -1:max31 }
- int **stream_tcp.session_timeout** = 180: session tracking timeout { 1:max31 }
- bool **stream_tcp.show_rebuilt_packets** = false: enable cmg like output of reassembled packets
- int **stream_tcp.small_segments.count** = 0: number of consecutive (in the received order) TCP small segments considered to be excessive (129:12) { 0:2048 }
- int **stream_tcp.small_segments.maximum_size** = 0: minimum bytes for a TCP segment not to be considered small (129:12) { 0:2048 }
- bool **stream_tcp.track_only** = false: disable reassembly if true
- int **stream.udp_cache.idle_timeout** = 180: maximum inactive time before retiring session tracker { 1:max32 }
- int **stream_udp.session_timeout** = 30: session tracking timeout { 1:max31 }
- int **stream.user_cache.idle_timeout** = 180: maximum inactive time before retiring session tracker { 1:max32 }
- int **stream_user.session_timeout** = 60: session tracking timeout { 1:max31 }

- int **`suppress[].gid`** = 0: rule generator ID { 0:8129 }
- string **`suppress[].ip`**: restrict suppression to these addresses according to track
- int **`suppress[].sid`** = 0: rule signature ID { 0:max32 }
- enum **`suppress[].track`**: suppress only matching source or destination addresses { *by_src* | *by_dst* }
- int **tag.bytes**: tag for this many bytes { 1:max32 }
- enum **tag.~**: log all packets in session or all packets to or from host { *session|host_src|host_dst* }
- int **tag.packets**: tag this many packets { 1:max32 }
- int **tag.seconds**: tag for this many seconds { 1:max32 }
- enum **target.~**: indicate the target of the attack { *src_ip* | *dst_ip* }
- string **`tcp_connector[].address`**: address
- port **`tcp_connector[].base_port`**: base port number
- string **`tcp_connector[].connector`**: connector name
- enum **`tcp_connector[].setup`**: stream establishment { *call* | *answer* }
- int **telnet.ayt_attack_thresh** = -1: alert beyond this number of consecutive Telnet AYT commands (-1 is disabled) { -1:max31 }
- bool **telnet.check_encrypted** = false: check for end of encryption
- bool **telnet.encrypted_traffic** = false: check for encrypted Telnet
- bool **telnet.normalize** = false: eliminate escape sequences
- string **`tenant_selector[].file`**: use configuration in given file
- string **`tenant_selector[].tenants`**: list of tenants to match
- interval **tos.~range**: check if IP TOS is in given range { 0:255 }
- string **trace.constraints.dst_ip**: destination IP address filter
- int **trace.constraints.dst_port**: destination port filter { 0:65535 }
- int **trace.constraints.ip_proto**: numerical IP protocol ID filter { 0:255 }
- bool **trace.constraints.match** = true: use constraints to filter traces
- string **trace.constraints.src_ip**: source IP address filter
- int **trace.constraints.src_port**: source port filter { 0:65535 }
- int **trace.modules.all**: enable trace for all modules { 0:255 }
- int **trace.modules.appid.all**: enable all trace options { 0:255 }
- int **trace.modules.dce_smb.all**: enable all trace options { 0:255 }
- int **trace.modules.dpx.all**: enable all trace options { 0:255 }
- int **trace.modules.file_id.all**: enable all trace options { 0:255 }
- int **trace.modules.js_norm.all**: enable all trace options { 0:255 }
- int **trace.modules.js_norm.dump**: enable data logging { 0:255 }
- int **trace.modules.js_norm.proc**: enable processing logging { 0:255 }

- int **trace.modules.memory.all**: enable all trace options { 0:255 }
- int **trace.modules.snort.all**: enable all trace options { 0:255 }
- int **trace.modules.snort.inspector_manager**: enable inspector manager trace logging { 0:255 }
- int **trace.modules.vba_data.all**: enable all trace options { 0:255 }
- int **trace.modules.wizard.all**: enable all trace options { 0:255 }
- bool **trace.ntuple** = false: print packet n-tuple info with trace messages
- enum **trace.output**: output method for trace log messages { *stdout* | *syslog* }
- bool **trace.timestamp** = false: print message timestamps with trace messages
- interval **ttl.~range**: check if IP TTL is in the given range { 0:255 }
- bool **udp.deep_teredo_inspection** = false: look for Teredo on all UDP ports (default is only 3544)
- bit_list **udp.geneve_ports** = *6081*: set Geneve ports { 65535 }
- bit_list **udp.gtp_ports** = *2152 3386*: set GTP ports { 65535 }
- bit_list **udp.vxlan_ports** = *4789*: set VXLAN ports { 65535 }
- bool **unified2.legacy_events** = false: generate Snort 2.X style events for barnyard2 compatibility
- int **unified2.limit** = 0: set maximum size in MB before rollover (0 is unlimited) { 0:maxSZ }
- bool **unified2.nostamp** = true: append file creation time to name (in Unix Epoch format)
- interval **urg.~range**: check if tcp urgent offset is in given range { 0:65535 }
- int_list **vlan.extra_tpid_ether_types** = *0x9100 0x9200*: set non-standard QinQ ether types { 65535 }
- interval **window.~range**: check if TCP window size is in given range { 0:65535 }
- multi **wizard.curses**: enable service identification based on internal algorithm { dce_smb | dce_udp | dce_tcp | mms | s7commplus | sslv2 }
- select **`wizard.hexes[].proto`** = *any*: protocol to scan { tcp | udp | any }
- string **`wizard.hexes[].service`**: name of service
- string **`wizard.hexes[].to_client[].hex`**: sequence of data with wild chars (?)
- string **`wizard.hexes[].to_server[].hex`**: sequence of data with wild chars (?)
- int **wizard.max_search_depth** = 8192: maximum scan depth per flow { 0:65535 }
- select **`wizard.spells[].proto`** = *any*: protocol to scan { tcp | udp | any }
- string **`wizard.spells[].service`**: name of service
- string **`wizard.spells[].to_client[].spell`**: sequence of data with wild cards (*)
- string **`wizard.spells[].to_server[].spell`**: sequence of data with wild cards (*)
- interval **wscale.~range**: check if TCP window scale is in given range { 0:65535 }

11.5 Counts

- **active.direct_injects**: total crafted packets directly injected (sum)
- **active.failed_direct_injects**: total crafted packet direct injects that failed (sum)
- **active.failed_injects**: total crafted packet encode + injects that failed (sum)
- **active.holds_allowed**: total number of packet hold requests allowed (sum)
- **active.holds_canceled**: total number of packet hold requests canceled (sum)
- **active.holds_denied**: total number of packet hold requests denied (sum)
- **active.injects**: total crafted packets encoded and injected (sum)
- **address_space_selector.no_match**: selection evaluations that had no matches (sum)
- **address_space_selector.packets**: packets evaluated (sum)
- **appid.bytes_in_use**: number of bytes in use in the cache (now)
- **appid.ignored_packets**: count of packets ignored (sum)
- **appid.items_in_use**: items in use in the cache (now)
- **appid.odp_reload_ignored_pkts**: count of packets ignored after open detector package is reloaded (sum)
- **appid.packets**: count of packets received (sum)
- **appid.processed_packets**: count of packets processed (sum)
- **appid.service_cache_adds**: number of times an entry was added to the service cache (sum)
- **appid.service_cache_prunes**: number of times the service cache was pruned (sum)
- **appid.service_cache_removes**: number of times an item was removed from the service cache (sum)
- **appid.total_sessions**: count of sessions created (sum)
- **appid.tp_reload_ignored_pkts**: count of packets ignored after third-party module is reloaded (sum)
- **arp_spoof.packets**: total packets (sum)
- **back_orifice.packets**: total packets (sum)
- **binder.allows**: allow actions bound (sum)
- **binder.assistant_inspectors**: flow assistant inspector requests handled (sum)
- **binder.blocks**: block actions bound (sum)
- **binder.inspects**: inspect actions bound (sum)
- **binder.new_flows**: new flows evaluated (sum)
- **binder.new_standby_flows**: new HA flows evaluated (sum)
- **binder.no_match**: binding evaluations that had no matches (sum)
- **binder.raw_packets**: raw packets evaluated (sum)
- **binder.rebinds**: flows rebound (sum)
- **binder.resets**: reset actions bound (sum)
- **binder.service_changes**: flow service changes evaluated (sum)
- **cip.concurrent_sessions**: total concurrent SIP sessions (now)

- **cip.max_concurrent_sessions**: maximum concurrent SIP sessions (max)
- **cip.packets**: total packets (sum)
- **cip.session**: total sessions (sum)
- **ciscometadata.invalid_hdr_len**: total invalid Cisco Metadata header lengths (sum)
- **ciscometadata.invalid_hdr_ver**: total invalid Cisco Metadata header versions (sum)
- **ciscometadata.invalid_opt_len**: total invalid Cisco Metadata option lengths (sum)
- **ciscometadata.invalid_opt_type**: total invalid Cisco Metadata option types (sum)
- **ciscometadata.invalid_sgt**: total invalid Cisco Metadata security group tags (sum)
- **ciscometadata.truncated_hdr**: total truncated Cisco Metadata headers (sum)
- **daq.allow**: total allow verdicts (sum)
- **daq.analyzed**: total packets analyzed from DAQ (sum)
- **daq.blacklist**: total blacklist verdicts (sum)
- **daq.block**: total block verdicts (sum)
- **daq.dropped**: packets dropped (sum)
- **daq.eof_messages**: end of flow messages received from DAQ (sum)
- **daq.expected_flows**: expected flows created in DAQ (sum)
- **daq.filtered**: packets filtered out (sum)
- **daq.idle**: attempts to acquire from DAQ without available packets (sum)
- **daq.ignore**: total ignore verdicts (sum)
- **daq.injected**: active responses or replacements (sum)
- **daq.internal_blacklist**: packets blacklisted internally due to lack of DAQ support (sum)
- **daq.internal_whitelist**: packets whitelisted internally due to lack of DAQ support (sum)
- **daq.other_messages**: messages received from DAQ with unrecognized message type (sum)
- **daq.outstanding_max**: maximum of packets unprocessed (max)
- **daq.outstanding**: packets unprocessed (now)
- **daq.pcaps**: total files and interfaces processed (max)
- **daq.received**: total packets received from DAQ (sum)
- **daq.replace**: total replace verdicts (sum)
- **daq.retries_discarded**: messages discarded when purging the retry queue (sum)
- **daq.retries_dropped**: messages dropped when overrunning the retry queue (sum)
- **daq.retries_processed**: messages processed from the retry queue (sum)
- **daq.retries_queued**: messages queued for retry (sum)
- **daq.rx_bytes**: total bytes received (sum)
- **daq.skipped**: packets skipped at startup (sum)
- **daq.sof_messages**: start of flow messages received from DAQ (sum)

- **daq.whitelist**: total whitelist verdicts (sum)
- **data_log.packets**: total packets (sum)
- **dce_http_proxy.http_proxy_session_failures**: failed http proxy sessions (sum)
- **dce_http_proxy.http_proxy_sessions**: successful http proxy sessions (sum)
- **dce_http_server.http_server_session_failures**: failed http server sessions (sum)
- **dce_http_server.http_server_sessions**: successful http server sessions (sum)
- **dce_smb.alter_context_responses**: total connection-oriented alter context responses (sum)
- **dce_smb.alter_contexts**: total connection-oriented alter contexts (sum)
- **dce_smb.auth3s**: total connection-oriented auth3s (sum)
- **dce_smb.bind_acks**: total connection-oriented binds acks (sum)
- **dce_smb.bind_naks**: total connection-oriented bind naks (sum)
- **dce_smb.binds**: total connection-oriented binds (sum)
- **dce_smb.cache_adds**: smbv2 cache added new entry (sum)
- **dce_smb.cache_hits**: smbv2 cache found existing entry (sum)
- **dce_smb.cache_max**: smbv2 cache's maximum byte usage (sum)
- **dce_smb.cache_misses**: smbv2 cache did not find entry (sum)
- **dce_smb.cache_prunes**: smbv2 cache pruned entry to make space for new entry (sum)
- **dce_smb.cache_removes**: smbv2 cache removed existing entry (sum)
- **dce_smb.cache_replaces**: smbv2 cache found entry and replaced its value (sum)
- **dce_smb.cancels**: total connection-oriented cancels (sum)
- **dce_smb.client_frags_reassembled**: total connection-oriented client fragments reassembled (sum)
- **dce_smb.client_max_fragment_size**: connection-oriented client maximum fragment size (sum)
- **dce_smb.client_min_fragment_size**: connection-oriented client minimum fragment size (sum)
- **dce_smb.client_segs_reassembled**: total connection-oriented client segments reassembled (sum)
- **dce_smb.concurrent_sessions**: total concurrent sessions (now)
- **dce_smb.events**: total events (sum)
- **dce_smb.faults**: total connection-oriented faults (sum)
- **dce_smb.files_processed**: total smb files processed (sum)
- **dce_smb.ignored_bytes**: total ignored bytes (sum)
- **dce_smb.max_concurrent_sessions**: maximum concurrent sessions (max)
- **dce_smb.max_outstanding_requests**: maximum outstanding requests (max)
- **dce_smb.ms_rpc_http_pdus**: total connection-oriented MS requests to send RPC over HTTP (sum)
- **dce_smb.orphaned**: total connection-oriented orphaned (sum)
- **dce_smb.other_requests**: total connection-oriented other requests (sum)
- **dce_smb.other_responses**: total connection-oriented other responses (sum)

- **dce_smb.packets**: total smb packets (sum)
- **dce_smb.pdus**: total connection-oriented PDUs (sum)
- **dce_smb.rejects**: total connection-oriented rejects (sum)
- **dce_smb.request_fragments**: total connection-oriented request fragments (sum)
- **dce_smb.requests**: total connection-oriented requests (sum)
- **dce_smb.response_fragments**: total connection-oriented response fragments (sum)
- **dce_smb.responses**: total connection-oriented responses (sum)
- **dce_smb.server_frags_reassembled**: total connection-oriented server fragments reassembled (sum)
- **dce_smb.server_max_fragment_size**: connection-oriented server maximum fragment size (sum)
- **dce_smb.server_min_fragment_size**: connection-oriented server minimum fragment size (sum)
- **dce_smb.server_segs_reassembled**: total connection-oriented server segments reassembled (sum)
- **dce_smb.sessions**: total smb sessions (sum)
- **dce_smb.shutdowns**: total connection-oriented shutdowns (sum)
- **dce_smb.smb_client_segs_reassembled**: total smb client segments reassembled (sum)
- **dce_smb.smb_server_segs_reassembled**: total smb server segments reassembled (sum)
- **dce_smb.total_encrypted_sessions**: total encrypted sessions (sum)
- **dce_smb.total_mc_sessions**: total multichannel sessions (sum)
- **dce_smb.total_sessions**: total smb sessions (sum)
- **dce_smb.total_smb1_sessions**: total smb1 sessions (sum)
- **dce_smb.total_smb2_sessions**: total smb2 sessions (sum)
- **dce_smb.v2_bad_next_cmd_offset**: total number of SMBv2 packets seen with invalid next command offset (sum)
- **dce_smb.v2_cls_err_resp**: total number of SMBv2 close error response packets seen (sum)
- **dce_smb.v2_cls_ignored**: total number of SMBv2 close packets ignored due to missing trackers or invalid share type (sum)
- **dce_smb.v2_cls_inv_str_sz**: total number of SMBv2 close packets seen with invalid structure size (sum)
- **dce_smb.v2_cls_req_ftrkr_misng**: total number of SMBv2 close request packets ignored due to missing file tracker (sum)
- **dce_smb.v2_cls_req_hdr_err**: total number of SMBv2 close request packets ignored due to corrupted header (sum)
- **dce_smb.v2_cls**: total number of SMBv2 close packets seen (sum)
- **dce_smb.v2_cmpnd_req_lt_crossed**: total number of SMBv2 packets seen where compound requests exceed the smb_max_compoun limit (sum)
- **dce_smb.v2_crt_err_resp**: total number of SMBv2 create error response packets seen (sum)
- **dce_smb.v2_crt_inv_file_data**: total number of SMBv2 create request packets ignored due to error in getting file name (sum)
- **dce_smb.v2_crt_inv_str_sz**: total number of SMBv2 create packets seen with invalid structure size (sum)
- **dce_smb.v2_crt_req_hdr_err**: total number of SMBv2 create request packets ignored due to corrupted header (sum)
- **dce_smb.v2_crt_req_ipc**: total number of SMBv2 create request packets ignored as share type is IPC (sum)
- **dce_smb.v2_crt_resp_hdr_err**: total number of SMBv2 create response packets ignored due to corrupted header (sum)

- **dce_smb.v2_crt_rtrkr_misng**: total number of SMBv2 create response packets ignored due to missing create request tracker (sum)
- **dce_smb.v2_crt**: total number of SMBv2 create packets seen (sum)
- **dce_smb.v2_crt_tree_trkr_misng**: total number of SMBv2 create response packets ignored due to missing tree tracker (sum)
- **dce_smb.v2_extra_file_data_err**: total number of SMBv2 packets seen with where file data beyond file size is observed (sum)
- **dce_smb.v2_hdr_err**: total number of SMBv2 packets seen with corrupted hdr (sum)
- **dce_smb.v2_ignored_file_processing**: total number of file processing ignored (sum)
- **dce_smb.v2_inv_file_ctx_err**: total number of times null file context are seen resulting in not being able to set file size (sum)
- **dce_smb.v2_ioctl_err_resp**: total number of ioctl errors responses (sum)
- **dce_smb.v2_ioctl_ignored**: total number of SMBv2 IOCTL packets ignored due to missing trackers or invalid share type (sum)
- **dce_smb.v2_ioctl_inv_str_sz**: total number of ioctl invalid structure size (sum)
- **dce_smb.v2_ioctl**: total number of ioctl calls (sum)
- **dce_smb.v2_logoff_inv_str_sz**: total number of SMBv2 logoff packets seen with invalid structure size (sum)
- **dce_smb.v2_logoff**: total number of SMBv2 logoff (sum)
- **dce_smb.v2_mc_file_transfers**: total multichannel files transferred (sum)
- **dce_smb.v2_msgs_uninspected**: total number of SMBv2 packets seen where command is not being inspected (sum)
- **dce_smb.v2_read_err_resp**: total number of SMBv2 read error response packets seen (sum)
- **dce_smb.v2_read_ignored**: total number of SMBv2 write packets ignored due to missing trackers or invalid share type (sum)
- **dce_smb.v2_read_inv_str_sz**: total number of SMBv2 read packets seen with invalid structure size (sum)
- **dce_smb.v2_read_req_hdr_err**: total number of SMBv2 read request packets ignored due to corrupted header (sum)
- **dce_smb.v2_read_resp_hdr_err**: total number of SMBv2 read response packets ignored due to corrupted header (sum)
- **dce_smb.v2_read_rtrkr_misng**: total number of SMBv2 read response packets ignored due to missing read request tracker (sum)
- **dce_smb.v2_read**: total number of SMBv2 read packets seen (sum)
- **dce_smb.v2_setinfo**: total number of SMBv2 set info packets seen (sum)
- **dce_smb.v2_setup_err_resp**: total number of SMBv2 setup error response packets seen (sum)
- **dce_smb.v2_setup_inv_str_sz**: total number of SMBv2 setup packets seen with invalid structure size (sum)
- **dce_smb.v2_setup_resp_hdr_err**: total number of SMBv2 setup response packets ignored due to corrupted header (sum)
- **dce_smb.v2_setup**: total number of SMBv2 setup packets seen (sum)
- **dce_smb.v2_stinf_err_resp**: total number of SMBv2 set info error response packets seen (sum)
- **dce_smb.v2_stinf_ignored**: total number of SMBv2 set info packets ignored due to missing trackers or invalid share type (sum)
- **dce_smb.v2_stinf_inv_str_sz**: total number of SMBv2 set info packets seen with invalid structure size (sum)
- **dce_smb.v2_stinf_req_ftrkr_misng**: total number of SMBv2 set info request packets ignored due to missing file tracker (sum)
- **dce_smb.v2_stinf_req_hdr_err**: total number of SMBv2 set info request packets ignored due to corrupted header (sum)

- **dce_smb.v2_total_file_trackers**: total number of file trackers (sum)
- **dce_smb.v2_total_session_trackers**: total number of session trackers (sum)
- **dce_smb.v2_total_tree_trackers**: total number of tree trackers (sum)
- **dce_smb.v2_tree_cnct_err_resp**: total number of SMBv2 tree connect error response packets seen (sum)
- **dce_smb.v2_tree_cnct_ignored**: total number of SMBv2 setup response packets ignored due to failure in creating tree tracker (sum)
- **dce_smb.v2_tree_cnct_inv_str_sz**: total number of SMBv2 tree connect packets seen with invalid structure size (sum)
- **dce_smb.v2_tree_cnct_resp_hdr_err**: total number of SMBv2 tree connect response packets ignored due to corrupted header (sum)
- **dce_smb.v2_tree_cnct**: total number of SMBv2 tree connect packets seen (sum)
- **dce_smb.v2_tree_discn_ignored**: total number of SMBv2 tree disconnect packets ignored due to missing trackers or invalid share type (sum)
- **dce_smb.v2_tree_discn_inv_str_sz**: total number of SMBv2 tree disconnect packets seen with invalid structure size (sum)
- **dce_smb.v2_tree_discn_req_hdr_err**: total number of SMBv2 tree disconnect request packets ignored due to corrupted header (sum)
- **dce_smb.v2_tree_discn**: total number of SMBv2 tree disconnect packets seen (sum)
- **dce_smb.v2_updated_file_flows**: total number of updated file flows due to parent flow cleanup (sum)
- **dce_smb.v2_wrt_err_resp**: total number of SMBv2 write error response packets seen (sum)
- **dce_smb.v2_wrt_ignored**: total number of SMBv2 write packets ignored due to missing trackers or invalid share type (sum)
- **dce_smb.v2_wrt_inv_str_sz**: total number of SMBv2 write packets seen with invalid structure size (sum)
- **dce_smb.v2_wrt_req_hdr_err**: total number of SMBv2 write request packets ignored due to corrupted header (sum)
- **dce_smb.v2_wrt**: total number of SMBv2 write packets seen (sum)
- **dce_tcp.alter_context_responses**: total connection-oriented alter context responses (sum)
- **dce_tcp.alter_contexts**: total connection-oriented alter contexts (sum)
- **dce_tcp.auth3s**: total connection-oriented auth3s (sum)
- **dce_tcp.bind_acks**: total connection-oriented binds acks (sum)
- **dce_tcp.bind_naks**: total connection-oriented bind naks (sum)
- **dce_tcp.binds**: total connection-oriented binds (sum)
- **dce_tcp.cancels**: total connection-oriented cancels (sum)
- **dce_tcp.client_frags_reassembled**: total connection-oriented client fragments reassembled (sum)
- **dce_tcp.client_max_fragment_size**: connection-oriented client maximum fragment size (sum)
- **dce_tcp.client_min_fragment_size**: connection-oriented client minimum fragment size (sum)
- **dce_tcp.client_segs_reassembled**: total connection-oriented client segments reassembled (sum)
- **dce_tcp.concurrent_sessions**: total concurrent sessions (now)
- **dce_tcp.events**: total events (sum)
- **dce_tcp.faults**: total connection-oriented faults (sum)

- **dce_tcp.max_concurrent_sessions**: maximum concurrent sessions (max)
- **dce_tcp.ms_rpc_http_pdus**: total connection-oriented MS requests to send RPC over HTTP (sum)
- **dce_tcp.orphaned**: total connection-oriented orphaned (sum)
- **dce_tcp.other_requests**: total connection-oriented other requests (sum)
- **dce_tcp.other_responses**: total connection-oriented other responses (sum)
- **dce_tcp.pdus**: total connection-oriented PDUs (sum)
- **dce_tcp.rejects**: total connection-oriented rejects (sum)
- **dce_tcp.request_fragments**: total connection-oriented request fragments (sum)
- **dce_tcp.requests**: total connection-oriented requests (sum)
- **dce_tcp.response_fragments**: total connection-oriented response fragments (sum)
- **dce_tcp.responses**: total connection-oriented responses (sum)
- **dce_tcp.server_frags_reassembled**: total connection-oriented server fragments reassembled (sum)
- **dce_tcp.server_max_fragment_size**: connection-oriented server maximum fragment size (sum)
- **dce_tcp.server_min_fragment_size**: connection-oriented server minimum fragment size (sum)
- **dce_tcp.server_segs_reassembled**: total connection-oriented server segments reassembled (sum)
- **dce_tcp.shutdowns**: total connection-oriented shutdowns (sum)
- **dce_tcp.tcp_expected_realized**: total tcp dynamic endpoint expected realized sessions (sum)
- **dce_tcp.tcp_expected_sessions**: total tcp dynamic endpoint expected sessions (sum)
- **dce_tcp.tcp_packets**: total tcp packets (sum)
- **dce_tcp.tcp_sessions**: total tcp sessions (sum)
- **dce_udp.acks**: total connection-less acks (sum)
- **dce_udp.cancel_acks**: total connection-less cancel acks (sum)
- **dce_udp.cancels**: total connection-less cancels (sum)
- **dce_udp.client_facks**: total connection-less client facks (sum)
- **dce_udp.concurrent_sessions**: total concurrent sessions (now)
- **dce_udp.events**: total events (sum)
- **dce_udp.faults**: total connection-less faults (sum)
- **dce_udp.fragments**: total connection-less fragments (sum)
- **dce_udp.frags_reassembled**: total connection-less fragments reassembled (sum)
- **dce_udp.max_concurrent_sessions**: maximum concurrent sessions (max)
- **dce_udp.max_fragment_size**: connection-less maximum fragment size (sum)
- **dce_udp.max_seqnum**: max connection-less seqnum (sum)
- **dce_udp.no_calls**: total connection-less no calls (sum)
- **dce_udp.other_requests**: total connection-less other requests (sum)
- **dce_udp.other_responses**: total connection-less other responses (sum)

- **dce_udp.ping**: total connection-less ping (sum)
- **dce_udp.rejects**: total connection-less rejects (sum)
- **dce_udp.requests**: total connection-less requests (sum)
- **dce_udp.responses**: total connection-less responses (sum)
- **dce_udp.server_facks**: total connection-less server facks (sum)
- **dce_udp.udp_packets**: total udp packets (sum)
- **dce_udp.udp_sessions**: total udp sessions (sum)
- **dce_udp.working**: total connection-less working (sum)
- **detection.alert_limit**: events previously triggered on same PDU (sum)
- **detection.alerts**: alerts not including IP reputation (sum)
- **detection.alt_searches**: alt fast pattern searches in packet data (sum)
- **detection.analyzed**: total packets processed (now)
- **detection.buf_dumps**: total number of IPS buffers collected from matched rules (sum)
- **detection.cont_creations**: total number of continuations created (sum)
- **detection.cont_evals**: total number of condition-met continuations (sum)
- **detection.context_stalls**: times processing stalled to wait for an available context (sum)
- **detection.cont_flows**: total number of flows using continuation (sum)
- **detection.cont_match_distance**: total number of bytes jumped over by matched continuations (sum)
- **detection.cont_matches**: total number of continuations matched (sum)
- **detection.cont_max_num**: peak number of simultaneous continuations per flow (max)
- **detection.cont_mismatch_distance**: total number of bytes jumped over by mismatched continuations (sum)
- **detection.cont_mismatches**: total number of continuations mismatched (sum)
- **detection.cont_recalls**: total number of continuations recalled (sum)
- **detection.cooked_searches**: fast pattern searches in cooked packet data (sum)
- **detection.event_limit**: events filtered (sum)
- **detection.file_searches**: fast pattern searches in file buffer (sum)
- **detection.hard_evals**: non-fast pattern rule evaluations (sum)
- **detection.logged**: logged packets (sum)
- **detection.log_limit**: events queued but not logged (sum)
- **detection.match_limit**: fast pattern matches not processed (sum)
- **detection.offload_busy**: times offload was not available (sum)
- **detection.offload_failures**: fast pattern offload search failures (sum)
- **detection.offload_fallback**: fast pattern offload search fallback attempts (sum)
- **detection.offloads**: fast pattern searches that were offloaded (sum)
- **detection.offload_suspends**: fast pattern search suspends due to offload context chains (sum)

- **detection.onload_waits**: times processing waited for onload to complete (sum)
- **detection.passed**: passed packets (sum)
- **detection.pcre_error**: total number of times pcre returns error (sum)
- **detection.pcre_match_limit**: total number of times pcre hit the match limit (sum)
- **detection.pcre_recursion_limit**: total number of times pcre hit the recursion limit (sum)
- **detection.pdu_searches**: fast pattern searches in service buffers (sum)
- **detection.pkt_searches**: fast pattern searches in packet data (sum)
- **detection.queue_limit**: events not queued because queue full (sum)
- **detection.raw_searches**: fast pattern searches in raw packet data (sum)
- **detection.total_alerts**: alerts including IP reputation (sum)
- **dnp3.concurrent_sessions**: total concurrent dnp3 sessions (now)
- **dnp3.dnp3_application_pdus**: total dnp3 application pdus (sum)
- **dnp3.dnp3_link_layer_frames**: total dnp3 link layer frames (sum)
- **dnp3.max_concurrent_sessions**: maximum concurrent dnp3 sessions (max)
- **dnp3.tcp_pdus**: total tcp pdus (sum)
- **dnp3.total_packets**: total packets (sum)
- **dnp3.udp_packets**: total udp packets (sum)
- **dns.concurrent_sessions**: total concurrent dns sessions (now)
- **dns.max_concurrent_sessions**: maximum concurrent dns sessions (max)
- **dns.packets**: total packets processed (sum)
- **dns.requests**: total dns requests (sum)
- **dns.responses**: total dns responses (sum)
- **domain_filter.checked**: domains checked (sum)
- **domain_filter.filtered**: domains filtered (sum)
- **dpx.packets**: total packets (sum)
- **event_filter.no_memory_global**: number of times event filter ran out of global memory (sum)
- **event_filter.no_memory_local**: number of times event filter ran out of local memory (sum)
- **file_connector.messages**: total messages (sum)
- **file_id.cache_failures**: number of file cache add failures (sum)
- **file_id.files_not_processed**: number of files not processed due to per-flow limit (sum)
- **file_id.max_concurrent_files**: maximum files processed concurrently on a flow (max)
- **file_id.total_file_data**: number of file data bytes processed (sum)
- **file_id.total_files**: number of files processed (sum)
- **file_log.total_events**: total file events (sum)
- **ftp_data.packets**: total packets (sum)

- **ftp_server.concurrent_sessions**: total concurrent FTP sessions (now)
- **ftp_server.flow_segment_size_changed**: total number of FTP sessions with segment size change (sum)
- **ftp_server.max_concurrent_sessions**: maximum concurrent FTP sessions (max)
- **ftp_server.pkt_segment_size_changed**: total number of FTP data packets with segment size change (sum)
- **ftp_server.ssl_search_abandoned**: total SSL search abandoned (sum)
- **ftp_server.ssl_srch_abandoned_early**: total SSL search abandoned too soon (sum)
- **ftp_server.start_tls**: total STARTTLS events generated (sum)
- **ftp_server.total_bytes**: total number of bytes processed (sum)
- **ftp_server.total_packets**: total packets (sum)
- **gtp_inspect.concurrent_sessions**: total concurrent gtp sessions (now)
- **gtp_inspect.events**: requests (sum)
- **gtp_inspect.max_concurrent_sessions**: maximum concurrent gtp sessions (max)
- **gtp_inspect.sessions**: total sessions processed (sum)
- **gtp_inspect.unknown_infos**: unknown information elements (sum)
- **gtp_inspect.unknown_types**: unknown message types (sum)
- **high_availability.client_consume_errors**: client data consume failure count (sum)
- **high_availability.daq_imports**: states imported via daq (sum)
- **high_availability.daq_stores**: states stored via daq (sum)
- **high_availability.delete_msgs_consumed**: deletion messages consumed (sum)
- **high_availability.key_mismatch**: messages received with a flow key mismatch (sum)
- **high_availability.msg_length_mismatch**: messages received with an inconsistent total length (sum)
- **high_availability.msgs_recv**: total messages received (sum)
- **high_availability.msg_version_mismatch**: messages received with a version mismatch (sum)
- **high_availability.truncated_msgs**: truncated messages received (sum)
- **high_availability.unknown_client_idx**: messages received with an unknown client index (sum)
- **high_availability.unknown_key_type**: messages received with an unknown flow key type (sum)
- **high_availability.update_msgs_consumed**: update messages fully consumed (sum)
- **high_availability.update_msgs_recv_no_flow**: update messages received without a local flow (sum)
- **high_availability.update_msgs_recv**: update messages received (sum)
- **host_cache.adds**: lru cache added new entry (sum)
- **host_cache.alloc_prunes**: lru cache pruned entry to make space for new entry (sum)
- **host_cache.bytes_in_use**: current number of bytes in use (now)
- **host_cache.find_hits**: lru cache found entry in cache (sum)
- **host_cache.find_misses**: lru cache did not find entry in cache (sum)
- **host_cache.items_in_use**: current number of items in the cache (now)

- **host_cache.reload_prunes**: lru cache pruned entry for lower memcap during reload (sum)
- **host_cache.removes**: lru cache found entry and removed it (sum)
- **host_cache.replaced**: lru cache found entry and replaced it (sum)
- **hosts.dynamic_host_adds**: number of host additions after initial host file load (sum)
- **hosts.dynamic_service_adds**: number of service additions after initial host file load (sum)
- **hosts.dynamic_service_updates**: number of service updates after initial host file load (sum)
- **hosts.hosts_pruned**: number of LRU hosts pruned due to configured resource limits (sum)
- **hosts.service_list_overflows**: number of service additions that failed due to configured resource limits (sum)
- **hosts.total_hosts**: maximum number of entries in the host attribute table (max)
- **host_tracker.service_adds**: host service adds (sum)
- **host_tracker.service_finds**: host service finds (sum)
- **http2_inspect.concurrent_sessions**: total concurrent HTTP/2 sessions (now)
- **http2_inspect.flows**: HTTP/2 connections inspected (sum)
- **http2_inspect.flows_over_stream_limit**: HTTP/2 flows exceeding 100 concurrent streams (sum)
- **http2_inspect.max_concurrent_files**: maximum concurrent file transfers per HTTP/2 connection (max)
- **http2_inspect.max_concurrent_sessions**: maximum concurrent HTTP/2 sessions (max)
- **http2_inspect.max_concurrent_streams**: maximum concurrent streams per HTTP/2 connection (max)
- **http2_inspect.max_table_entries**: maximum entries in an HTTP/2 dynamic table (max)
- **http2_inspect.total_bytes**: total HTTP/2 data bytes inspected (sum)
- **http_inspect.chunked**: chunked message bodies (sum)
- **http_inspect.concurrent_sessions**: total concurrent http sessions (now)
- **http_inspect.connect_requests**: CONNECT requests inspected (sum)
- **http_inspect.connect_tunnel_cutovers**: CONNECT tunnel flow cutovers to wizard (sum)
- **http_inspect.delete_requests**: DELETE requests inspected (sum)
- **http_inspect.excess_parameters**: repeat parameters exceeding max (sum)
- **http_inspect.flows**: HTTP connections inspected (sum)
- **http_inspect.get_requests**: GET requests inspected (sum)
- **http_inspect.head_requests**: HEAD requests inspected (sum)
- **http_inspect.inspections**: total message sections inspected (sum)
- **http_inspect.js_external_scripts**: total number of external JavaScripts processed (sum)
- **http_inspect.js_inline_scripts**: total number of inline JavaScripts processed (sum)
- **http_inspect.js_pdf_scripts**: total number of PDF files processed (sum)
- **http_inspect.max_concurrent_sessions**: maximum concurrent http sessions (max)
- **http_inspect.options_requests**: OPTIONS requests inspected (sum)
- **http_inspect.other_requests**: other request methods inspected (sum)

- **http_inspect.parameters**: HTTP parameters inspected (sum)
- **http_inspect.partial_inspections**: early inspections done for script detection (sum)
- **http_inspect.pipelined_flows**: total HTTP connections containing pipelined requests (sum)
- **http_inspect.pipelined_requests**: total requests placed in a pipeline (sum)
- **http_inspect.post_requests**: POST requests inspected (sum)
- **http_inspect.put_requests**: PUT requests inspected (sum)
- **http_inspect.reassembles**: TCP segments combined into HTTP messages (sum)
- **http_inspect.request_bodies**: POST, PUT, and other requests with message bodies (sum)
- **http_inspect.requests**: HTTP request messages inspected (sum)
- **http_inspect.responses**: HTTP response messages inspected (sum)
- **http_inspect.scans**: TCP segments scanned looking for HTTP messages (sum)
- **http_inspect.script_detections**: early inspections of scripts in HTTP responses (sum)
- **http_inspect.skip_mime_attach**: total number of HTTP requests with too many MIME attachments to inspect (sum)
- **http_inspect.ssl_srch_abandoned_early**: total SSL search abandoned too soon (sum)
- **http_inspect.total_bytes**: total HTTP data bytes inspected (sum)
- **http_inspect.trace_requests**: TRACE requests inspected (sum)
- **http_inspect.uri_coding**: URIs with character coding problems (sum)
- **http_inspect.uri_normalizations**: URIs needing to be normalization (sum)
- **http_inspect.uri_path**: URIs with path problems (sum)
- **icmp4.bad_checksum**: non-zero icmp checksums (sum)
- **icmp4.checksum_bypassed**: checksum calculations bypassed (sum)
- **icmp6.bad_icmp6_checksum**: nonzero icmp6 checksums (sum)
- **icmp6.checksum_bypassed**: checksum calculations bypassed (sum)
- **iec104.concurrent_sessions**: total concurrent IEC104 sessions (now)
- **iec104.frames**: total IEC104 messages (sum)
- **iec104.max_concurrent_sessions**: maximum concurrent IEC104 sessions (max)
- **iec104.sessions**: total sessions processed (sum)
- **imap.b64_attachments**: total base64 attachments decoded (sum)
- **imap.b64_decoded_bytes**: total base64 decoded bytes (sum)
- **imap.concurrent_sessions**: total concurrent imap sessions (now)
- **imap.js_pdf_scripts**: total number of PDF files processed (sum)
- **imap.max_concurrent_sessions**: maximum concurrent imap sessions (max)
- **imap.non_encoded_attachments**: total non-encoded attachments extracted (sum)
- **imap.non_encoded_bytes**: total non-encoded extracted bytes (sum)
- **imap.packets**: total packets processed (sum)

- **imap.qp_attachments**: total quoted-printable attachments decoded (sum)
- **imap.qp_decoded_bytes**: total quoted-printable decoded bytes (sum)
- **imap.sessions**: total imap sessions (sum)
- **imap.ssl_search_abandoned**: total SSL search abandoned (sum)
- **imap.ssl_srch_abandoned_early**: total SSL search abandoned too soon (sum)
- **imap.start_tls**: total STARTTLS events generated (sum)
- **imap.uu_attachments**: total uu attachments decoded (sum)
- **imap.uu_decoded_bytes**: total uu decoded bytes (sum)
- **ipv4.bad_checksum**: nonzero ip checksums (sum)
- **ipv4.checksum_bypassed**: checksum calculations bypassed (sum)
- **js_norm.bytes**: total number of bytes processed (sum)
- **js_norm.identifier_overflows**: total number of unique identifier limit overflows (sum)
- **js_norm.identifiers**: total number of unique identifiers processed (sum)
- **latency.max_usecs**: maximum usecs elapsed (sum)
- **latency.packet_timeouts**: packets that timed out (sum)
- **latency.rule_eval_timeouts**: rule evals that timed out (sum)
- **latency.rule_tree_enables**: rule tree re-enables (sum)
- **latency.total_packets**: total packets monitored (sum)
- **latency.total_rule_evals**: total rule evals monitored (sum)
- **latency.total_usecs**: total usecs elapsed (sum)
- **memory.active**: total bytes allocated in active pages (now)
- **memory.allocated**: total amount of memory allocated by packet threads (now)
- **memory.app_all**: total bytes allocated by application (now)
- **memory.cur_in_use**: current memory used (now)
- **memory.deallocated**: total amount of memory deallocated by packet threads (now)
- **memory.epochs**: number of memory updates (sum)
- **memory.max_in_use**: maximum memory used (max)
- **memory.reap_aborts**: abort pruning before target due to process under limit (sum)
- **memory.reap_attempts**: attempts to reclaim memory (sum)
- **memory.reap_cycles**: number of actionable over-limit conditions (sum)
- **memory.reap_decrease**: total amount of the decrease in thread memory while process over limit (sum)
- **memory.reap_failures**: failures to reclaim memory (sum)
- **memory.reap_increase**: total amount of the increase in thread memory while process over limit (sum)
- **memory.resident**: maximum bytes physically resident (now)
- **memory.retained**: total bytes not returned to OS (now)

- **memory.start_up_use**: memory used before packet processing (now)
- **mem_test.packets**: total packets (sum)
- **mms.concurrent_sessions**: total concurrent MMS sessions (now)
- **mms.frames**: total MMS messages (sum)
- **mms.max_concurrent_sessions**: maximum concurrent MMS sessions (max)
- **mms.sessions**: total sessions processed (sum)
- **modbus.concurrent_sessions**: total concurrent modbus sessions (now)
- **modbus.frames**: total Modbus messages (sum)
- **modbus.max_concurrent_sessions**: maximum concurrent modbus sessions (max)
- **modbus.sessions**: total sessions processed (sum)
- **netflow.cache_adds**: netflow cache added new entry (sum)
- **netflow.cache_hits**: netflow cache found existing entry (sum)
- **netflow.cache_max**: netflow cache's maximum byte usage (sum)
- **netflow.cache_misses**: netflow cache did not find entry (sum)
- **netflow.cache_prunes**: netflow cache pruned entry to make space for new entry (sum)
- **netflow.cache_removes**: netflow cache removed existing entry (sum)
- **netflow.cache_replaces**: netflow cache found entry and replaced its value (sum)
- **netflow.invalid_netflow_record**: count of invalid netflow records (sum)
- **netflow.netflow_cache_bytes_in_use**: number of bytes used in netflow cache (now)
- **netflow.packets**: total packets processed (sum)
- **netflow.records**: total records found in netflow data (sum)
- **netflow.template_cache_bytes_in_use**: number of bytes used in template cache (now)
- **netflow.unique_flows**: count of unique netflow flows (sum)
- **netflow.v9_missing_template**: count of data records that are missing templates (sum)
- **netflow.v9_options_template**: count of options template flowset (sum)
- **netflow.v9_templates**: count of total version 9 templates (sum)
- **netflow.version_5**: count of netflow version 5 packets received (sum)
- **netflow.version_9**: count of netflow version 9 packets received (sum)
- **normalizer.icmp4_echo**: icmp4 ping normalizations (sum)
- **normalizer.icmp6_echo**: icmp6 echo normalizations (sum)
- **normalizer.ip4_df**: don't frag bit normalizations (sum)
- **normalizer.ip4_opts**: ip4 options cleared (sum)
- **normalizer.ip4_rf**: reserved flag bit clears (sum)
- **normalizer.ip4_tos**: type of service normalizations (sum)
- **normalizer.ip4_trim**: eth packets trimmed to datagram size (sum)

- **normalizer.ip4_ttl**: time-to-live normalizations (sum)
- **normalizer.ip6_hops**: ip6 hop limit normalizations (sum)
- **normalizer.ip6_options**: ip6 options cleared (sum)
- **normalizer.tcp_block**: blocked segments (sum)
- **normalizer.tcp_ecn_pkt**: packets with ECN bits cleared (sum)
- **normalizer.tcp_ecn_session**: ECN bits cleared (sum)
- **normalizer.tcp_ips_data**: normalized segments (sum)
- **normalizer.tcp_nonce**: packets with nonce bit cleared (sum)
- **normalizer.tcp_options**: packets with options cleared (sum)
- **normalizer.tcp_padding**: packets with padding cleared (sum)
- **normalizer.tcp_req_pay**: cleared urgent pointer and urgent flag when there is no payload (sum)
- **normalizer.tcp_req_urg**: cleared urgent pointer when urgent flag is not set (sum)
- **normalizer.tcp_req_urp**: cleared the urgent flag if the urgent pointer is not set (sum)
- **normalizer.tcp_reserved**: packets with reserved bits cleared (sum)
- **normalizer.tcp_syn_options**: SYN only options cleared from non-SYN packets (sum)
- **normalizer.tcp_trim_mss**: data trimmed to MSS (sum)
- **normalizer.tcp_trim_rst**: RST packets with data trimmed (sum)
- **normalizer.tcp_trim_syn**: tcp segments trimmed on SYN (sum)
- **normalizer.tcp_trim_win**: data trimmed to window (sum)
- **normalizer.tcp_ts_ecr**: timestamp cleared on non-ACKs (sum)
- **normalizer.tcp_ts_nop**: timestamp options cleared (sum)
- **normalizer.tcp_urgent_ptr**: packets without data with urgent pointer cleared (sum)
- **normalizer.test_icmp4_echo**: test icmp4 ping normalizations (sum)
- **normalizer.test_icmp6_echo**: test icmp6 echo normalizations (sum)
- **normalizer.test_ip4_df**: test don't frag bit normalizations (sum)
- **normalizer.test_ip4_opts**: test ip4 options cleared (sum)
- **normalizer.test_ip4_rf**: test reserved flag bit clears (sum)
- **normalizer.test_ip4_tos**: test type of service normalizations (sum)
- **normalizer.test_ip4_trim**: test eth packets trimmed to datagram size (sum)
- **normalizer.test_ip4_ttl**: test time-to-live normalizations (sum)
- **normalizer.test_ip6_hops**: test ip6 hop limit normalizations (sum)
- **normalizer.test_ip6_options**: test ip6 options cleared (sum)
- **normalizer.test_tcp_block**: test blocked segments (sum)
- **normalizer.test_tcp_ecn_pkt**: test packets with ECN bits cleared (sum)
- **normalizer.test_tcp_ecn_session**: test ECN bits cleared (sum)

- **normalizer.test_tcp_ips_data**: test normalized segments (sum)
- **normalizer.test_tcp_nonce**: test packets with nonce bit cleared (sum)
- **normalizer.test_tcp_options**: test packets with options cleared (sum)
- **normalizer.test_tcp_padding**: test packets with padding cleared (sum)
- **normalizer.test_tcp_req_pay**: test cleared urgent pointer and urgent flag when there is no payload (sum)
- **normalizer.test_tcp_req_urg**: test cleared urgent pointer when urgent flag is not set (sum)
- **normalizer.test_tcp_req_urp**: test cleared the urgent flag if the urgent pointer is not set (sum)
- **normalizer.test_tcp_reserved**: test packets with reserved bits cleared (sum)
- **normalizer.test_tcp_syn_options**: test SYN only options cleared from non-SYN packets (sum)
- **normalizer.test_tcp_trim_mss**: test data trimmed to MSS (sum)
- **normalizer.test_tcp_trim_rst**: test RST packets with data trimmed (sum)
- **normalizer.test_tcp_trim_syn**: test tcp segments trimmed on SYN (sum)
- **normalizer.test_tcp_trim_win**: test data trimmed to window (sum)
- **normalizer.test_tcp_ts_ecr**: test timestamp cleared on non-ACKs (sum)
- **normalizer.test_tcp_ts_nop**: test timestamp options cleared (sum)
- **normalizer.test_tcp_urgent_ptr**: test packets without data with urgent pointer cleared (sum)
- **packet_capture.captured**: packets matching dumped after matching filter (sum)
- **packet_capture.processed**: packets processed against filter (sum)
- **payload_injector.http2_injects**: total number of http2 injections (sum)
- **payload_injector.http2_mid_frame**: total number of attempts to inject mid-frame (sum)
- **payload_injector.http2_translate_err**: total number of http2 page translation errors (sum)
- **payload_injector.http_injects**: total number of http injections (sum)
- **pcre.pcre_native**: total pcre rules compiled by pcre engine (sum)
- **pcre.pcre_negated**: total pcre rules using negation syntax (sum)
- **pcre.pcre_rules**: total rules processed with pcre option (sum)
- **pcre.pcre_to_hyper**: total pcre rules by hyperscan engine (sum)
- **perf_monitor.flow_tracker_creates**: total number of flow trackers created (sum)
- **perf_monitor.flow_tracker_prunes**: flow trackers pruned for reuse by new flows (sum)
- **perf_monitor.flow_tracker_reload_deletes**: flow trackers deleted due to memcap change on config reload (sum)
- **perf_monitor.flow_tracker_total_deletes**: flow trackers deleted to stay below memcap limit (sum)
- **perf_monitor.packets**: total packets processed by performance monitor (sum)
- **pop.b64_attachments**: total base64 attachments decoded (sum)
- **pop.b64_decoded_bytes**: total base64 decoded bytes (sum)
- **pop.concurrent_sessions**: total concurrent pop sessions (now)
- **pop.js_pdf_scripts**: total number of PDF files processed (sum)

- **pop.max_concurrent_sessions**: maximum concurrent pop sessions (max)
- **pop.non_encoded_attachments**: total non-encoded attachments extracted (sum)
- **pop.non_encoded_bytes**: total non-encoded extracted bytes (sum)
- **pop.packets**: total packets processed (sum)
- **pop.qp_attachments**: total quoted-printable attachments decoded (sum)
- **pop.qp_decoded_bytes**: total quoted-printable decoded bytes (sum)
- **pop.sessions**: total pop sessions (sum)
- **pop.ssl_search_abandoned**: total SSL search abandoned (sum)
- **pop.ssl_srch_abandoned_early**: total SSL search abandoned too soon (sum)
- **pop.start_tls**: total STARTTLS events generated (sum)
- **pop.total_bytes**: total number of bytes processed (sum)
- **pop.uu_attachments**: total uu attachments decoded (sum)
- **pop.uu_decoded_bytes**: total uu decoded bytes (sum)
- **port_scan.alloc_prunes**: number of trackers pruned on allocation of new tracking (sum)
- **port_scan.bytes_in_use**: number of bytes currently used by portscan (now)
- **port_scan.packets**: number of packets processed by port scan (sum)
- **port_scan.reload_prunes**: number of trackers pruned on reload due to reduced memcap (sum)
- **port_scan.trackers**: number of trackers allocated by port scan (sum)
- **rate_filter.no_memory**: number of times rate filter ran out of memory (sum)
- **reputation.aux_ip_blocked**: number of auxiliary ip packets blocked (sum)
- **reputation.aux_ip_monitored**: number of auxiliary ip packets monitored (sum)
- **reputation.aux_ip_trusted**: number of auxiliary ip packets trusted (sum)
- **reputation.blocked**: number of packets blocked (sum)
- **reputation.memory_allocated**: total memory allocated (sum)
- **reputation.monitored**: number of packets monitored (sum)
- **reputation.packets**: total packets processed (sum)
- **reputation.trusted**: number of packets trusted (sum)
- **rna.appid_change**: count of appid change events received (sum)
- **rna.change_host_update**: count number of change host update events (sum)
- **rna.cpe_os**: count of CPE OS events received (sum)
- **rna.dhcp_data**: count of DHCP data events received (sum)
- **rna.dhcp_info**: count of new DHCP lease events received (sum)
- **rna.icmp_bidirectional**: count of bidirectional ICMP flows received (sum)
- **rna.icmp_new**: count of new ICMP flows received (sum)
- **rna.ip_bidirectional**: count of bidirectional IP received (sum)

- **rna.ip_new**: count of new IP flows received (sum)
- **rna.netflow_record**: count of netflow record events received (sum)
- **rna.other_packets**: count of packets received without session tracking (sum)
- **rna.smb**: count of new SMB events received (sum)
- **rna.tcp_midstream**: count of TCP midstream packets received (sum)
- **rna.tcp_syn_ack**: count of TCP SYN-ACK packets received (sum)
- **rna.tcp_syn**: count of TCP SYN packets received (sum)
- **rna.total_bytes_in_interval**: count of bytes processed (sum)
- **rna.total_events_in_interval**: count of RNA events generated (sum)
- **rna.total_packets_in_interval**: count of packets processed (sum)
- **rna.udp_bidirectional**: count of bidirectional UDP flows received (sum)
- **rna.udp_new**: count of new UDP flows received (sum)
- **rpc_decode.concurrent_sessions**: total concurrent rpc sessions (now)
- **rpc_decode.max_concurrent_sessions**: maximum concurrent rpc sessions (max)
- **rpc_decode.total_packets**: total packets (sum)
- **s7commplus.concurrent_sessions**: total concurrent s7commplus sessions (now)
- **s7commplus.frames**: total S7commplus messages (sum)
- **s7commplus.max_concurrent_sessions**: maximum concurrent s7commplus sessions (max)
- **s7commplus.sessions**: total sessions processed (sum)
- **sd_pattern.below_threshold**: sd_pattern matched but missed threshold (sum)
- **sd_pattern.pattern_not_found**: sd_pattern did not not match (sum)
- **sd_pattern.terminated**: hyperscan terminated (sum)
- **search_engine.max_queued**: maximum fast pattern matches queued for further evaluation (max)
- **search_engine.non_qualified_events**: total non-qualified events (sum)
- **search_engine.qualified_events**: total qualified events (sum)
- **search_engine.searched_bytes**: total bytes searched (sum)
- **search_engine.total_flushed**: total fast pattern matches processed (sum)
- **search_engine.total_inserts**: total fast pattern hits (sum)
- **search_engine.total_overruns**: fast pattern matches discarded due to overflow (sum)
- **search_engine.total_unique**: total unique fast pattern hits (sum)
- **side_channel.packets**: total packets (sum)
- **sip.ack**: ack (sum)
- **sip.bye**: bye (sum)
- **sip.cancel**: cancel (sum)
- **sip.code_1xx**: 1xx (sum)

- **sip.code_2xx**: 2xx (sum)
- **sip.code_3xx**: 3xx (sum)
- **sip.code_4xx**: 4xx (sum)
- **sip.code_5xx**: 5xx (sum)
- **sip.code_6xx**: 6xx (sum)
- **sip.code_7xx**: 7xx (sum)
- **sip.code_8xx**: 8xx (sum)
- **sip.code_9xx**: 9xx (sum)
- **sip.concurrent_sessions**: total concurrent SIP sessions (now)
- **sip.dialogs**: total dialogs (sum)
- **sip.events**: events generated (sum)
- **sip.ignored_channels**: total channels ignored (sum)
- **sip.ignored_sessions**: total sessions ignored (sum)
- **sip.info**: info (sum)
- **sip.invite**: invite (sum)
- **sip.join**: join (sum)
- **sip.max_concurrent_sessions**: maximum concurrent SIP sessions (max)
- **sip.message**: message (sum)
- **sip.notify**: notify (sum)
- **sip.options**: options (sum)
- **sip.packets**: total packets (sum)
- **sip.prack**: prack (sum)
- **sip.refer**: refer (sum)
- **sip.register**: register (sum)
- **sip.sessions**: total sessions (sum)
- **sip.subscribe**: subscribe (sum)
- **sip.total_requests**: total requests (sum)
- **sip.total_responses**: total responses (sum)
- **sip.update**: update (sum)
- **smtp.b64_attachments**: total base64 attachments decoded (sum)
- **smtp.b64_decoded_bytes**: total base64 decoded bytes (sum)
- **smtp.concurrent_sessions**: total concurrent smtp sessions (now)
- **smtp.js_pdf_scripts**: total number of PDF files processed (sum)
- **smtp.max_concurrent_sessions**: maximum concurrent smtp sessions (max)
- **smtp.non_encoded_attachments**: total non-encoded attachments extracted (sum)

- **smtp.non_encoded_bytes**: total non-encoded extracted bytes (sum)
- **smtp.packets**: total packets processed (sum)
- **smtp.qp_attachments**: total quoted-printable attachments decoded (sum)
- **smtp.qp_decoded_bytes**: total quoted-printable decoded bytes (sum)
- **smtp.sessions**: total smtp sessions (sum)
- **smtp.ssl_search_abandoned**: total SSL search abandoned (sum)
- **smtp.ssl_srch_abandoned_early**: total SSL search abandoned too soon (sum)
- **smtp.start_tls**: total STARTTLS events generated (sum)
- **smtp.total_bytes**: total number of bytes processed (sum)
- **smtp.uu_attachments**: total uu attachments decoded (sum)
- **smtp.uu_decoded_bytes**: total uu decoded bytes (sum)
- **snort.attribute_table_hosts**: number of hosts added to the attribute table (sum)
- **snort.attribute_table_overflow**: number of host additions that failed due to attribute table full (sum)
- **snort.attribute_table_reloads**: number of times hosts attribute table was reloaded (sum)
- **snort.conf_reloads**: number of times configuration was reloaded (sum)
- **snort.daq_reloads**: number of times daq configuration was reloaded (sum)
- **snort.inspector_deletions**: number of times inspectors were deleted (sum)
- **snort.local_commands**: total local commands processed (sum)
- **snort.policy_reloads**: number of times policies were reloaded (sum)
- **snort.remote_commands**: total remote commands processed (sum)
- **snort.signals**: total signals processed (sum)
- **ssh.concurrent_sessions**: total concurrent ssh sessions (now)
- **ssh.max_concurrent_sessions**: maximum concurrent ssh sessions (max)
- **ssh.packets**: total packets (sum)
- **ssh.total_bytes**: total number of bytes processed (sum)
- **ssl.alert**: total ssl alert records (sum)
- **ssl.bad_handshakes**: total bad handshakes (sum)
- **ssl.certificate**: total ssl certificates (sum)
- **ssl.change_cipher**: total change cipher records (sum)
- **ssl.client_application**: total client application records (sum)
- **ssl.client_hello**: total client hellos (sum)
- **ssl.client_key_exchange**: total client key exchanges (sum)
- **ssl.concurrent_sessions**: total concurrent ssl sessions (now)
- **ssl.decoded**: ssl packets decoded (sum)
- **ssl.detection_disabled**: total detection disabled (sum)

- **ssl.finished**: total handshakes finished (sum)
- **ssl.handshakes_completed**: total completed ssl handshakes (sum)
- **ssl.max_concurrent_sessions**: maximum concurrent ssl sessions (max)
- **ssl.packets**: total packets processed (sum)
- **ssl.server_application**: total server application records (sum)
- **ssl.server_done**: total server done (sum)
- **ssl.server_hello**: total server hellos (sum)
- **ssl.server_key_exchange**: total server key exchanges (sum)
- **ssl.sessions_ignored**: total sessions ignore (sum)
- **ssl.unrecognized_records**: total unrecognized records (sum)
- **stream.current_flows**: current number of flows in cache (now)
- **stream.excess_prunes**: sessions pruned due to excess (sum)
- **stream.expected_flows**: total expected flows created within snort (sum)
- **stream.expected_overflows**: number of expected cache overflows (sum)
- **stream.expected_pruned**: number of expected flows pruned (sum)
- **stream.expected_realized**: number of expected flows realized (sum)
- **stream.file_memcap_prunes**: number of FILE flows pruned due to memcap (sum)
- **stream.file_timeout_prunes**: number of FILE flows pruned due to timeout (sum)
- **stream.flows**: total sessions (sum)
- **stream.ha_prunes**: sessions pruned by high availability sync (sum)
- **stream_icmp.created**: icmp session trackers created (sum)
- **stream_icmp.max**: max icmp sessions (max)
- **stream.icmp_memcap_prunes**: number of ICMP flows pruned due to memcap (sum)
- **stream_icmp.prunes**: icmp session prunes (sum)
- **stream_icmp.released**: icmp session trackers released (sum)
- **stream_icmp.sessions**: total icmp sessions (sum)
- **stream.icmp_timeout_prunes**: number of ICMP flows pruned due to timeout (sum)
- **stream_icmp.timeouts**: icmp session timeouts (sum)
- **stream.idle_prunes_max_flows**: sessions pruned due to pruning timeout since max flows is reached (sum)
- **stream.idle_prunes_proto_timeout**: sessions pruned due to protocol timeout (sum)
- **stream_ip.alerts**: alerts generated (sum)
- **stream_ip.anomalies**: anomalies detected (sum)
- **stream_ip.created**: ip session trackers created (sum)
- **stream_ip.current_frags**: current fragments (now)
- **stream_ip.discards**: fragments discarded (sum)

- **stream_ip.drops**: fragments dropped (sum)
- **stream_ip.fragmented_bytes**: total fragmented bytes (sum)
- **stream_ip.frag_timeouts**: datagrams abandoned (sum)
- **stream_ip.max_frags**: max fragments (sum)
- **stream_ip.max**: max ip sessions (max)
- **stream.ip_memcap_prunes**: number of IP flows pruned due to memcap (sum)
- **stream_ip.nodes_deleted**: fragments deleted from tracker (sum)
- **stream_ip.nodes_inserted**: fragments added to tracker (sum)
- **stream_ip.overlaps**: overlapping fragments (sum)
- **stream_ip.prunes**: ip session prunes (sum)
- **stream_ip.reassembled_bytes**: total reassembled bytes (sum)
- **stream_ip.reassembled**: reassembled datagrams (sum)
- **stream_ip.released**: ip session trackers released (sum)
- **stream_ip.sessions**: total ip sessions (sum)
- **stream.ip_timeout_prunes**: number of IP flows pruned due to timeout (sum)
- **stream_ip.timeouts**: ip session timeouts (sum)
- **stream_ip.total_bytes**: total number of bytes processed (sum)
- **stream_ip.total_frags**: total fragments (sum)
- **stream_ip.trackers_added**: datagram trackers created (sum)
- **stream_ip.trackers_cleared**: datagram trackers cleared (sum)
- **stream_ip.trackers_completed**: datagram trackers completed (sum)
- **stream_ip.trackers_freed**: datagram trackers released (sum)
- **stream.memcap_prunes**: sessions pruned due to memcap (sum)
- **stream.pdu_memcap_prunes**: number of PDU flows pruned due to memcap (sum)
- **stream.pdu_timeout_prunes**: number of PDU flows pruned due to timeout (sum)
- **stream.reload_allowed_deletes**: number of allowed flows deleted by config reloads (sum)
- **stream.reload_blocked_deletes**: number of blocked flows deleted by config reloads (sum)
- **stream.reload_freelist_deletes**: number of flows deleted from the free list by config reloads (sum)
- **stream.reload_offloaded_deletes**: number of offloaded flows deleted by config reloads (sum)
- **stream.reload_total_adds**: number of flows added by config reloads (sum)
- **stream.reload_total_deletes**: number of flows deleted by config reloads (sum)
- **stream.reload_tuning_idle**: number of times stream resource tuner called while idle (sum)
- **stream.reload_tuning_packets**: number of times stream resource tuner called while processing packets (sum)
- **stream.stale_prunes**: sessions pruned due to stale connection (sum)
- **stream_tcp.client_cleanups**: number of times data from server was flushed when session released (sum)

- **stream_tcp.closing**: number of sessions currently closing (now)
- **stream_tcp.created**: tcp session trackers created (sum)
- **stream_tcp.cur_packets_held**: number of packets currently held (now)
- **stream_tcp.data_trackers**: tcp session tracking started on data (sum)
- **stream_tcp.discards_skipped**: tcp packet discards skipped due to normalization disabled (sum)
- **stream_tcp.discards**: tcp packets discarded (sum)
- **stream_tcp.established**: number of sessions currently established (now)
- **stream_tcp.events**: events generated (sum)
- **stream_tcp.exceeded_max_bytes**: number of times the maximum queued byte limit was reached (sum)
- **stream_tcp.exceeded_max_segs**: number of times the maximum queued segment limit was reached (sum)
- **stream_tcp.fins**: number of fin packets (sum)
- **stream_tcp.gaps**: missing data between PDUs (sum)
- **stream_tcp.held_packet_purges**: number of held packets that were purged without flushing (sum)
- **stream_tcp.held_packet_retries**: number of held packets that were added to the retry queue (sum)
- **stream_tcp.held_packet_rexmits**: number of retransmits of held packets (sum)
- **stream_tcp.held_packets_dropped**: number of held packets dropped (sum)
- **stream_tcp.held_packets_passed**: number of held packets passed (sum)
- **stream_tcp.held_packet_timeouts**: number of held packets that timed out (sum)
- **stream_tcp.ignored**: tcp packets ignored (sum)
- **stream_tcp.initializing**: number of sessions currently initializing (now)
- **stream_tcp.inspector_fallbacks**: count of fallbacks from assigned service inspector (sum)
- **stream_tcp.instantiated**: new sessions instantiated (sum)
- **stream_tcp.internal_events**: 135:X events generated (sum)
- **stream_tcp.invalid_ack**: tcp packets received with an invalid ack number (sum)
- **stream_tcp.invalid_seq_num**: tcp packets received with an invalid sequence number (sum)
- **stream_tcp.max_bytes**: maximum number of bytes queued in any flow (max)
- **stream_tcp.max**: max tcp sessions (max)
- **stream_tcp.max_packets_held**: maximum number of packets held simultaneously (max)
- **stream_tcp.max_segs**: maximum number of segments queued in any flow (max)
- **stream.tcp_memcap_prunes**: number of TCP flows pruned due to memcap (sum)
- **stream_tcp.memory**: current memory in use (now)
- **stream_tcp.meta_acks**: number of meta acks processed (sum)
- **stream_tcp.no_flags_set**: tcp packets received with no TCP flags set (sum)
- **stream_tcp.overlaps**: overlapping segments queued (sum)
- **stream_tcp.packets_held**: number of packets held (sum)

- **stream_tcp.partial_fallbacks**: count of fallbacks from assigned service stream splitter (sum)
- **stream_tcp.partial_flush_bytes**: partial flush total bytes (sum)
- **stream_tcp.partial_flushes**: number of partial flushes initiated (sum)
- **stream_tcp.payload_fully_trimmed**: segments with no data after trimming (sum)
- **stream_tcp.prunes**: tcp session prunes (sum)
- **stream_tcp.rebuilt_buffers**: rebuilt PDU sections (sum)
- **stream_tcp.rebuilt_bytes**: total rebuilt bytes (sum)
- **stream_tcp.rebuilt_packets**: total reassembled PDUs (sum)
- **stream_tcp.released**: tcp session trackers released (sum)
- **stream_tcp.resets**: number of reset packets (sum)
- **stream_tcp.restarts**: sessions restarted (sum)
- **stream_tcp.resyns**: SYN received on established session (sum)
- **stream_tcp.segs_queued**: total segments queued (sum)
- **stream_tcp.segs_released**: total segments released (sum)
- **stream_tcp.segs_split**: tcp segments split when reassembling PDUs (sum)
- **stream_tcp.segs_used**: queued tcp segments applied to reassembled PDUs (sum)
- **stream_tcp.server_cleanups**: number of times data from client was flushed when session released (sum)
- **stream_tcp.sessions**: total tcp sessions (sum)
- **stream_tcp.setups**: session initializations (sum)
- **stream_tcp.syn_acks**: number of syn-ack packets (sum)
- **stream_tcp.syn_ack_trackers**: tcp session tracking started on syn-ack (sum)
- **stream_tcp.syns**: number of syn packets (sum)
- **stream_tcp.syn_trackers**: tcp session tracking started on syn (sum)
- **stream_tcp.three_way_trackers**: tcp session tracking started on ack (sum)
- **stream.tcp_timeout_prunes**: number of TCP flows pruned due to timeout (sum)
- **stream_tcp.timeouts**: tcp session timeouts (sum)
- **stream_tcp.untracked**: tcp packets not tracked (sum)
- **stream_tcp.zero_len_tcp_opt**: number of zero length tcp options (sum)
- **stream_tcp.zero_win_probes**: number of tcp zero window probes (sum)
- **stream.total_prunes**: total sessions pruned (sum)
- **stream_udp.created**: udp session trackers created (sum)
- **stream_udp.ignored**: udp packets ignored (sum)
- **stream_udp.max**: max udp sessions (max)
- **stream.udp_memcap_prunes**: number of UDP flows pruned due to memcap (sum)
- **stream_udp.prunes**: udp session prunes (sum)

- **stream_udp.released**: udp session trackers released (sum)
- **stream_udp.sessions**: total udp sessions (sum)
- **stream.udp_timeout_prunes**: number of UDP flows pruned due to timeout (sum)
- **stream_udp.timeouts**: udp session timeouts (sum)
- **stream_udp.total_bytes**: total number of bytes processed (sum)
- **stream.uni_flows**: number of uni flows in cache (now)
- **stream.uni_ip_flows**: number of uni ip flows in cache (now)
- **stream.uni_prunes**: uni sessions pruned (sum)
- **stream.user_memcap_prunes**: number of USER flows pruned due to memcap (sum)
- **stream.user_timeout_prunes**: number of USER flows pruned due to timeout (sum)
- **tcp.bad_tcp4_checksum**: nonzero tcp over ip checksums (sum)
- **tcp.bad_tcp6_checksum**: nonzero tcp over ipv6 checksums (sum)
- **tcp.checksum_bypassed**: checksum calculations bypassed (sum)
- **tcp_connector.messages**: total messages (sum)
- **telnet.concurrent_sessions**: total concurrent Telnet sessions (now)
- **telnet.max_concurrent_sessions**: maximum concurrent Telnet sessions (max)
- **telnet.total_packets**: total packets (sum)
- **tenant_selector.no_match**: selection evaluations that had no matches (sum)
- **tenant_selector.packets**: packets evaluated (sum)
- **udp.bad_udp4_checksum**: nonzero udp over ipv4 checksums (sum)
- **udp.bad_udp6_checksum**: nonzero udp over ipv6 checksums (sum)
- **udp.checksum_bypassed**: checksum calculations bypassed (sum)
- **wizard.tcp_hits**: tcp identifications (sum)
- **wizard.tcp_misses**: tcp searches abandoned (sum)
- **wizard.tcp_scans**: tcp payload scans (sum)
- **wizard.udp_hits**: udp identifications (sum)
- **wizard.udp_misses**: udp searches abandoned (sum)
- **wizard.udp_scans**: udp payload scans (sum)
- **wizard.user_hits**: user identifications (sum)
- **wizard.user_misses**: user searches abandoned (sum)
- **wizard.user_scans**: user payload scans (sum)

11.6 Generators

- **2**: output
- **105**: back_orifice
- **106**: rpc_decode
- **112**: arp_spoof
- **116**: arp
- **116**: auth
- **116**: ciscometadata
- **116**: decode
- **116**: eapol
- **116**: erspan2
- **116**: erspan3
- **116**: esp
- **116**: eth
- **116**: fabricpath
- **116**: geneve
- **116**: gre
- **116**: gtp
- **116**: icmp4
- **116**: icmp6
- **116**: igmp
- **116**: ipv4
- **116**: ipv6
- **116**: llc
- **116**: mpls
- **116**: pbb
- **116**: pgm
- **116**: pppoe
- **116**: tcp
- **116**: token_ring
- **116**: udp
- **116**: vlan
- **116**: wlan
- **119**: http_inspect
- **121**: http2_inspect

- **122**: port_scan
- **123**: stream_ip
- **124**: smtp
- **125**: ftp_server
- **126**: telnet
- **128**: ssh
- **129**: stream_tcp
- **131**: dns
- **133**: dce_http_proxy
- **133**: dce_http_server
- **133**: dce_smb
- **133**: dce_tcp
- **133**: dce_udp
- **134**: latency
- **135**: stream
- **136**: reputation
- **137**: ssl
- **140**: sip
- **141**: imap
- **142**: pop
- **143**: gtp_inspect
- **144**: modbus
- **145**: dnp3
- **148**: cip
- **149**: s7commplus
- **150**: file_id
- **151**: iec104
- **152**: mms
- **154**: js_norm
- **175**: domain_filter
- **256**: dpx

11.7 Builtin Rules

2:1 (output) tagged packet

A tagged packet was logged.

105:1 (back_orifice) Back Orifice traffic detected, unknown direction

Back Orifice traffic detected, unknown direction

105:2 (back_orifice) Back Orifice client traffic detected

Back Orifice client traffic detected

105:3 (back_orifice) Back Orifice server traffic detected

Back Orifice server traffic detected

105:4 (back_orifice) Back Orifice length field >= 1024 bytes

Back Orifice length field >= 1024 bytes

106:1 (rpc_decode) fragmented RPC records

Detected fragmented RPC records.

106:2 (rpc_decode) multiple RPC records

Detected multiple RPC records in the packet.

106:3 (rpc_decode) large RPC record fragment

Large RPC record fragment. RPC fragment length is greater than packet data size.

106:4 (rpc_decode) incomplete RPC segment

Incomplete RPC segment. Packet data size is less than required RPC fragment length.

106:5 (rpc_decode) zero-length RPC fragment

Zero-length RPC fragment.

112:1 (arp_spoof) unicast ARP request

ARP request is unicast, not broadcast.

112:2 (arp_spoof) ethernet/ARP mismatch for source hardware address

Mismatch between ethernet source hardware address and ARP source hardware address.

112:3 (arp_spoof) ethernet/ARP mismatch for destination hardware address in reply

Mismatch between ethernet destination hardware address and ARP destination hardware address in an ARP reply.

112:4 (arp_spoof) attempted ARP cache overwrite attack

Attempted ARP cache overwrite attack. The ethernet source hardware address or ARP source hardware address doesn't match the one provided for this IP address in the configured host table.

116:1 (ipv4) not IPv4 datagram

The packet is not an IPv4 datagram (based on the ip header's version field).

116:2 (ipv4) IPv4 header length < minimum

The IPv4 header length (based on the header's length field) is less than the ip version 4's minimum header length (20 bytes).

116:3 (ipv4) IPv4 datagram length < header field

The total IPv4 datagram length is less than the length calculated using the ipv4 header length field.

116:4 (ipv4) IPv4 options found with bad lengths

The IPv4 options field has a bad/incorrect length.

116:5 (ipv4) truncated IPv4 options

The IPv4 options field is truncated.

116:6 (ipv4) IPv4 datagram length > captured length

The IPv4 datagram length is greater than the captured packet's length.

116:45 (tcp) TCP packet length is smaller than 20 bytes

The TCP packet length is smaller than the minimum tcp header length (20 bytes).

116:46 (tcp) TCP data offset is less than 5

The TCP data offset is less than five 32 bit words (20 bytes) and is invalid.

116:47 (tcp) TCP header length exceeds packet length

The TCP header length exceeds the packet's length.

116:54 (tcp) TCP options found with bad lengths

The TCP options are invalid and/or have bad lengths.

116:55 (tcp) truncated TCP options

The TCP options field is truncated.

116:56 (tcp) T/TCP detected

A tcp packet was detected with the CC Echo field set.

116:57 (tcp) obsolete TCP options found

A tcp packet was detected that contained obsolete TCP options.

116:58 (tcp) experimental TCP options found

A tcp packet was detected that contained experimental TCP options.

116:59 (tcp) TCP window scale option found with length > 14

The TCP window scale option found with a length greater than 14.

116:95 (udp) truncated UDP header

A truncated UDP header has been detected.

116:96 (udp) invalid UDP header, length field < 8

An invalid UDP header detected. The header's length is less than 8 bytes.

116:97 (udp) short UDP packet, length field > payload length

The UDP length field is greater than the payload length.

116:98 (udp) long UDP packet, length field < payload length

The UDP length field is less than the payload length.

116:105 (icmp4) ICMP header truncated

An ICMP packet was detected with the header truncated.

116:106 (icmp4) ICMP timestamp header truncated

The ICMP packet's timestamp header is truncated.

116:107 (icmp4) ICMP address header truncated

The ICMP packet's address header is truncated.

116:109 (arp) truncated ARP

The packet length is less than ethernet arp's minimum length of 28 bytes.

116:110 (eapol) truncated EAP header

(eapol) truncated EAP header

116:111 (eapol) EAP key truncated

(eapol) EAP key truncated

116:112 (eapol) EAP header truncated

(eapol) EAP header truncated

116:120 (pppoe) bad PPPOE frame detected

A bad PPPOE frame has been detected. The frames length is less than the PPPOE frame minimum (6 bytes).

116:130 (vlan) bad VLAN frame

A bad VLAN frame was detected due to either the packet being smaller than the minimum VLAN header size or the VLAN ID being invalid (0 or 4095).

116:131 (llc) bad LLC header

An invalid LLC header has been detected (less than 3 bytes).

116:132 (llc) bad extra LLC info

(llc) bad extra LLC info

116:133 (wlan) bad 802.11 LLC header

(wlan) bad 802.11 LLC header

116:134 (wlan) bad 802.11 extra LLC info

(wlan) bad 802.11 extra LLC info

116:140 (token_ring) bad Token Ring header

(token_ring) bad Token Ring header

116:141 (token_ring) bad Token Ring ETHLLC header

(token_ring) bad Token Ring ETHLLC header

116:142 (token_ring) bad Token Ring MRLEN header

(token_ring) bad Token Ring MRLEN header

116:143 (token_ring) bad Token Ring MR header

(token_ring) bad Token Ring MR header

116:150 (decode) loopback IP

A loopback IP was detected within a packet.

116:151 (decode) same src/dst IP

The same source and destination IP was detected.

116:160 (gre) GRE header length > payload length

The payload length is greater than the packet length.

116:161 (gre) multiple encapsulations in packet

There are multiple encapsulations within the GRE packet.

116:162 (gre) invalid GRE version

The detected GRE version field value is invalid (should be 0 or 1).

116:163 (gre) invalid GRE header

Invalid flag set in GRE header.

116:164 (gre) invalid GRE v.1 PPTP header

Invalid GRE v.1 PPTP header detected.

116:165 (gre) GRE trans header length > payload length

The GRE trans header length is greater than the payload length.

116:170 (mpls) bad MPLS frame

The MPLS frame is invalid. The MPLS header length is less than the MPLS minimum frame size (4 bytes).

116:171 (mpls) MPLS label 0 appears in bottom header when not decoding as ip4

The MPLS label 0 appears in bottom header when not decoding as an ip4 packet.

116:172 (mpls) MPLS label 1 appears in bottom header

The MPLS label 1 appears in bottom header.

116:173 (mpls) MPLS label 2 appears in bottom header when not decoding as ip6

The MPLS label 2 appears in bottom header when not decoding as an ip6 packet.

116:174 (mpls) MPLS label 3 appears in header

A MPLS label 3 (Implicit NULL Label) appears in header.

116:175 (mpls) MPLS label 4, 5,.. or 15 appears in header

A reserved MPLS label (4, 5 or 15) appears in header.

116:176 (mpls) too many MPLS headers

There were too many MPLS headers detected. (Use the mpls.max_stack_depth setting to set the max value).

116:180 (geneve) insufficient room for geneve header

The packet length is less than the expected GENEVE header length.

116:181 (geneve) invalid version

The version number in the GENEVE header is not valid (not equal to zero).

116:182 (geneve) invalid header

The packet length is less than the minimum GENEVE header length.

116:183 (geneve) invalid flags

There are several scenarios for this event. 1) The C flag is clear but critical options are present. 2) The C flag is set but critical options are absent. 3) If the critical header present bit is set the option's length cannot be 0.

116:184 (geneve) invalid options

The options length field extends past the end of the GENEVE header.

116:250 (icmp4) ICMP original IP header truncated

The ICMP error message's original IP header is truncated.

116:251 (icmp4) ICMP version and original IP header versions differ

The ICMP error message's original IP packet's version and original IP header versions differ.

116:252 (icmp4) ICMP original datagram length < original IP header length

The ICMP error message's original datagram's length is less than the original IP's header length.

116:253 (icmp4) ICMP original IP payload < 64 bits

The ICMP error message's original IP packet's payload is less than 64 bits.

116:254 (icmp4) ICMP original IP payload > 576 bytes

The ICMP error message's original IP packet's payload is greater than the expected max of 576 bytes.

116:255 (icmp4) ICMP original IP fragmented and offset not 0

An ICMP original IP is fragmented and the offset is not 0.

116:270 (ipv6) IPv6 packet below TTL limit

The IPv6 packet has a TTL value that is below the TTL limit.

116:271 (ipv6) IPv6 header claims to not be IPv6

The IPv6 header claims to not be an IPv6 packet.

116:272 (ipv6) IPv6 truncated extension header

The IPv6 packet has a truncated extension header.

116:273 (ipv6) IPv6 truncated header

The IPv6 packet has a truncated header.

116:274 (ipv6) IPv6 datagram length < header field

The IPv6 datagram length field is less than the header field.

116:275 (ipv6) IPv6 datagram length > captured length

The IPv6 datagram's length is greater than the captured packet's length.

116:276 (ipv6) IPv6 packet with destination address ::0

An IPv6 packet was detected with a destination address of ::0

116:277 (ipv6) IPv6 packet with multicast source address

An IPv6 packet with a multicast source address has been detected.

116:278 (ipv6) IPv6 packet with reserved multicast destination address

An IPv6 packet with a reserved multicast destination address has been detected.

116:279 (ipv6) IPv6 header includes an undefined option type

The IPv6 header includes an undefined option type.

116:280 (ipv6) IPv6 address includes an unassigned multicast scope value

The IPv6 address includes an unassigned multicast scope value.

116:281 (ipv6) IPv6 header includes an invalid value for the *next header* field

The IPv6 header includes an invalid value for the *next header* field.

116:282 (ipv6) IPv6 header includes a routing extension header followed by a hop-by-hop header

The IPv6 header includes a routing extension header followed by a hop-by-hop header.

116:283 (ipv6) IPv6 header includes two routing extension headers

The IPv6 header includes two routing extension headers.

116:285 (icmp6) ICMPv6 packet of type 2 (message too big) with MTU field < 1280

An ICMPv6 packet of type 2 (message too big) that contains an MTU field of less than 1280 bytes has been detected.

116:286 (icmp6) ICMPv6 packet of type 1 (destination unreachable) with non-RFC 2463 code

An ICMPv6 packet of type 1 (destination unreachable) that contains a non-RFC 2463 code has been detected.

116:287 (icmp6) ICMPv6 router solicitation packet with a code not equal to 0

An ICMPv6 router solicitation packet with a code not equal to 0 has been detected.

116:288 (icmp6) ICMPv6 router advertisement packet with a code not equal to 0

An ICMPv6 router advertisement packet with a code not equal to 0 has been detected.

116:289 (icmp6) ICMPv6 router solicitation packet with the reserved field not equal to 0

An ICMPv6 router solicitation packet with the reserved field not equal to 0 has been detected.

116:290 (icmp6) ICMPv6 router advertisement packet with the reachable time field set > 1 hour

An ICMPv6 router advertisement packet with the reachable time field set to greater than 1 hour was detected.

116:291 (ipv6) IPV6 tunneled over IPv4, IPv6 header truncated, possible Linux kernel attack

An IPV6 tunnel over IPv4 packet was received. The IPv6 header truncated which could possibly be a Linux kernel attack.

116:292 (ipv6) IPv6 header has destination options followed by a routing header

The IPv6 header has destination options followed by a routing header.

116:293 (decode) two or more IP (v4 and/or v6) encapsulation layers present

There are two or more IP (v4 and/or v6) encapsulation layers present.

116:294 (esp) truncated encapsulated security payload header

The encapsulated security payload header was too short (less than 22 bytes).

116:295 (ipv6) IPv6 header includes an option which is too big for the containing header

The IPv6 header includes an option which is too big for the containing header.

116:296 (ipv6) IPv6 packet includes out-of-order extension headers

The IPv6 packet includes out-of-order extension headers.

116:297 (gtp) two or more GTP encapsulation layers present

There are multiple GTP encapsulation layers present.

116:298 (gtp) GTP header length is invalid

The packet data is smaller than the GTP header length making the packet invalid.

116:400 (tcp) XMAS attack detected

A XMAS attack detected.

116:401 (tcp) Nmap XMAS attack detected

A NMAP XMAS attack detected.

116:402 (tcp) DOS NAPTHA vulnerability detected

(tcp) DOS NAPTHA vulnerability detected.

116:403 (tcp) SYN to multicast address

A SYN packet was sent to a multicast address.

116:404 (ipv4) IPv4 packet with zero TTL

IPv4 packet was detected with a zero TTL value.

116:405 (ipv4) IPv4 packet with bad frag bits (both MF and DF set)

The IPv4 packet contains an invalid frag bits combination (both MF and DF are set).

116:406 (udp) invalid IPv6 UDP packet, checksum zero

An invalid IPv6 UDP packet was detected. The checksum value is zero.

116:407 (ipv4) IPv4 packet frag offset + length exceed maximum

The IPv4 packet's frag offset + the datagram length field exceeds the maximum packet size (65535)

116:408 (ipv4) IPv4 packet from *current net* source address

The IPv4 packet's source address is from the *current net* (value of zero)

116:409 (ipv4) IPv4 packet to *current net* dest address

The IPv4 packet's destination address is to the *current net* (value of zero)

116:410 (ipv4) IPv4 packet from multicast source address

The IPv4 packet has a multicast source address.

116:411 (ipv4) IPv4 packet from reserved source address

The IPv4 packet has a reserved source address.

116:412 (ipv4) IPv4 packet to reserved dest address

The IPv4 packet has a reserved destination address.

116:413 (ipv4) IPv4 packet from broadcast source address

The IPv4 packet has a broadcast source address.

116:414 (ipv4) IPv4 packet to broadcast dest address

The IPv4 packet has a broadcast destination address

116:415 (icmp4) ICMP4 packet to multicast dest address

ICMP4 packet to multicast destination address

116:416 (icmp4) ICMP4 packet to broadcast dest address

ICMP4 packet to broadcast destination address

116:418 (icmp4) ICMP4 type other

The ICMP4 packet *type* is not known.

116:419 (tcp) TCP urgent pointer exceeds payload length or no payload

The TCP urgent pointer exceeds payload length or has no payload.

116:420 (tcp) TCP SYN with FIN

An invalid tcp flag combination was detected (SYN and FIN).

116:421 (tcp) TCP SYN with RST

An invalid tcp flag combination was detected (SYN with RST)

116:422 (tcp) TCP PDU missing ack for established session

The TCP packet is missing the acknowledgment flag for an established session.

116:423 (tcp) TCP has no SYN, ACK, or RST

The TCP packet is invalid because it doesn't have a SYN, ACK, or RST flag set.

116:424 (pbb) truncated ethernet header

The packet length is less than the minimum ethernet header size (14 bytes)

116:424 (pbb) truncated ethernet header

A truncated ethernet header was detected.

116:425 (ipv4) truncated IPv4 header

The IPv4 header is truncated.

116:426 (icmp4) truncated ICMP4 header

The ICMP4 header is truncated.

116:427 (icmp6) truncated ICMPv6 header

The ICMPv6 header is truncated.

116:428 (ipv4) IPv4 packet below TTL limit

An IPv4 packet was received after the TTL limit.

116:429 (ipv6) IPv6 packet has zero hop limit

An IPv6 packet has a zero hop limit count.

116:430 (ipv4) IPv4 packet both DF and offset set

An invalid IPv4 packet was detected. The DF bit and an offset value are set.

116:431 (icmp6) ICMPv6 type not decoded

The ICMPv6 type is unknown and not decoded.

116:432 (icmp6) ICMPv6 packet to multicast address

An ICMPv6 packet to a multicast address was detected.

116:433 (tcp) DDOS shaft SYN flood

A tcp DDOS shaft SYN flood was detected.

116:434 (icmp4) ICMP ping Nmap

An ICMP ping from NMAP was detected.

116:435 (icmp4) ICMP icmpenum v1.1.1

An ICMP icmpenum v1.1.1 packet was received (the payload length is zero and icmp seq number equals 666).

116:436 (icmp4) ICMP redirect host

An ICMP host redirect packet was received.

116:437 (icmp4) ICMP redirect net

An ICMP network redirect packet was received.

116:438 (icmp4) ICMP traceroute ipopts

An ICMP packet with trace route ipopts was detected.

116:439 (icmp4) ICMP source quench

An ICMP packet with the source quench field set was detected.

116:440 (icmp4) broadscan smurf scanner

Broadscan smurf scanner traffic was detected.

116:441 (icmp4) ICMP destination unreachable communication administratively prohibited

ICMP destination unreachable traffic was detected (communication administratively prohibited).

116:442 (icmp4) ICMP destination unreachable communication with destination host is administratively prohibited

ICMP destination unreachable traffic detected (communication with destination host is administratively prohibited).

116:443 (icmp4) ICMP destination unreachable communication with destination network is administratively prohibited

ICMP destination unreachable traffic detected (communication with destination network is administratively prohibited).

116:444 (ipv4) IPv4 option set

(ipv4) IPv4 option set

116:445 (udp) large UDP packet (> 4000 bytes)

A large UDP packet was received (greater than 4000 bytes).

116:446 (tcp) TCP port 0 traffic

TCP port 0 traffic was detected.

116:447 (udp) UDP port 0 traffic

UDP port 0 traffic was detected.

116:448 (ipv4) IPv4 reserved bit set

An IPv4 packet was detected that has the reserved bit set.

116:449 (decode) unassigned/reserved IP protocol

An IP packet has an unassigned/reserved IP protocol number.

116:450 (decode) bad IP protocol

An invalid/bad IP protocol number has been detected.

116:451 (icmp4) ICMP path MTU denial of service attempt

An ICMP path MTU denial of service attempt has been detected.

116:452 (icmp4) Linux ICMP header DOS attempt

A Linux ICMP header DOS attempt has been detected.

116:453 (ipv6) ISATAP-addressed IPv6 traffic spoofing attempt

(ipv6) ISATAP-addressed IPv6 traffic spoofing attempt

116:454 (pgm) PGM nak list overflow attempt

(pgm) PGM nak list overflow attempt

116:455 (igmp) DOS IGMP IP options validation attempt

An IGMP IP options validation DOS attempt was detected.

116:456 (ipv6) too many IPv6 extension headers

The decoder detected more than the configured amount of IPv6 extension headers.

116:457 (icmp6) ICMPv6 packet of type 1 (destination unreachable) with non-RFC 4443 code

An ICMPv6 packet of type 1 (destination unreachable) was received with non-RFC 4443 code.

116:458 (ipv6) bogus fragmentation packet, possible BSD attack

An invalid fragmentation packet was detected. Could be a possible BSD attack.

116:459 (decode) fragment with zero length

An ip fragment was received with a zero length payload.

116:460 (icmp6) ICMPv6 node info query/response packet with a code greater than 2

The ICMPv6 node info query/response packet has a code value greater than 2.

116:461 (ipv6) IPv6 routing type 0 extension header

An IPv6 packet was received with a routing type 0 extension header.

116:462 (erspan2) ERSpan header version mismatch

The ERSpan2 version is not equal to 1 (the value of 1 signals that it's ERSpan2).

116:463 (erspan2) captured length < ERSpan type2 header length

The packet's length is less than the ERSpan2 headers minimum length (8 bytes).

116:464 (erspan3) captured < ERSpan type3 header length

The packet's length is less than the ERSpan3 header's minimum length (20 bytes).

116:465 (auth) truncated authentication header

The length of the packet received is less than the expected minimum of 16 bytes.

116:466 (auth) bad authentication header length

The authentication header length is greater than the packet data length.

116:467 (fabricpath) truncated FabricPath header

The packet header length is less than the minimum FabricPath header size of 16 bytes.

116:468 (ciscometadata) truncated Cisco Metadata header

The packet length is less than the Cisco Metadata header length.

116:469 (ciscometadata) invalid Cisco Metadata option length

The Cisco Metadata option length value is greater than zero.

116:470 (ciscometadata) invalid Cisco Metadata option type

The Cisco metadata option type is not set to 1.

116:471 (ciscometadata) invalid Cisco Metadata security group tag

The Cisco Metadata security group tag value is invalid (0xFFFF).

116:472 (decode) too many protocols present

The decoder detected that there were too many protocols present.

116:473 (decode) ether type out of range

An ether type value is below the minimum of 0x0600 (1536) and therefore out of range.

116:474 (icmp6) ICMPv6 not encapsulated in IPv6

An ICMPv6 packet was received that was not encapsulated in IPv6.

116:475 (ipv6) IPv6 mobility header includes an invalid value for the *payload protocol* field

The IPv6 mobility header includes an invalid value for the payload protocol field.

116:476 (ipv6) IPv6 packet from reserved source address

The IPv6 packet has a reserved source address.

116:477 (ipv6) IPv6 packet to reserved dest address

The IPv6 packet has a reserved destination address.

119:1 (http_inspect) URI has percent-encoding of an unreserved character

URI has percent encoding of an unreserved character. The ignore_unreserved option designates specific unreserved characters that are exempted from triggering this alert.

119:2 (http_inspect) URI contains double-encoded hexadecimal characters

URI contains double-encoded hexadecimal characters. This alert can only be generated if the iis_double_decode option is configured.

119:3 (http_inspect) URI has non-standard %u-style Unicode encoding

URI has non-standard %u-style Unicode encoding. This alert can only be generated if the percent_u option is configured.

119:4 (http_inspect) URI has Unicode encodings containing bytes that were not percent-encoded

URI has Unicode encodings containing bytes that were not percent-encoded as required by the HTTP RFC. This is sometimes called "bare byte" encoding. This alert can only be generated if the utf8_bare_byte option is configured.

119:6 (http_inspect) URI has two-byte or three-byte UTF-8 encoding

URI has two-byte or three-byte UTF-8 encoding. This alert can only be generated if the utf8 option is configured.

119:7 (http_inspect) URI has unicode map code point encoding

URI includes a two-byte or three-byte unicode character that normalized through the unicode map to some byte other than 0xFF. This alert can only be generated if the iis_unicode option is configured.

119:8 (http_inspect) URI path contains consecutive slash characters

URI path contains consecutive slash characters which are redundant. This alert can only be generated if the simplify_path option is configured.

119:9 (http_inspect) backslash character appears in the path portion of a URI

The backslash character appears in the path portion of a URI. This alert can only be generated if the backslash_to_slash option is configured

119:10 (http_inspect) URI path contains /./ pattern repeating the current directory

URI path contains "/./" pattern repeating the current directory. Alternatively the path may end with "/." repeating the current directory. This alert can only be generated if the simplify_path option is configured.

119:11 (http_inspect) URI path contains /../ pattern moving up a directory

URI path contains "/../" pattern moving upward a directory. Alternatively the path may end with "/.." with the same effect. This alert can only be generated if the simplify_path option is configured.

119:12 (http_inspect) Tab character in HTTP start line

The HTTP start line has a tab character among the blank space separators.

119:13 (http_inspect) HTTP start line or header line terminated by LF without a CR

HTTP start line or header line terminated by LF without a CR.

119:14 (http_inspect) Normalized URI includes character from bad_characters list

Normalized URI (after percent decoding) contains a forbidden character specified by the bad_characters option.

119:15 (http_inspect) URI path contains a segment that is longer than the oversize_dir_length parameter

URI path contains a segment (directory or file name) that is longer than the oversize_dir_length parameter.

119:16 (http_inspect) chunk length exceeds configured maximum_chunk_length

Chunk length as given in the chunk header exceeds maximum_chunk_length parameter.

119:18 (http_inspect) URI path includes /../ that goes above the root directory

The URI path has used /../ segments to go above the root of the directory tree. For example /foo/../../bar which specifies an object not under the root directory /. This alert can only be generated if the simplify_path option is configured.

119:19 (http_inspect) HTTP header line exceeds maximum_header_length option bytes

HTTP header line exceeds maximum_header_length option bytes. This does not apply to the start line. Header line length includes both header field name and value.

119:20 (http_inspect) HTTP message has more than maximum_headers option header fields

HTTP message has more than maximum_headers option header fields.

119:21 (http_inspect) HTTP message has more than one Content-Length header value

HTTP message has more than one Content-Length header value. This may be multiple header lines or comma-separated values on one line.

119:24 (http_inspect) Host header field appears more than once or has multiple values

Host header field appears more than once or has multiple values.

119:25 (http_inspect) length of HTTP Host header field value exceeds maximum_host_length option

Length of HTTP Host header field value exceeds maximum_host_length option.

119:28 (http_inspect) HTTP POST or PUT request without content-length or chunks

HTTP request uses POST or PUT method without delimiting the message body using either the Content-Length header or Transfer-Encoding chunked.

119:31 (http_inspect) HTTP request method is not known to Snort

HTTP request method is not known to Snort. Snort is familiar with all RFC methods and dozens of other methods.

119:32 (http_inspect) HTTP request uses primitive HTTP format known as HTTP/0.9

HTTP request uses primitive HTTP format known as HTTP/0.9.

119:33 (http_inspect) HTTP request URI has space character that is not percent-encoded

HTTP request URI has space character that is not percent-encoded.

119:34 (http_inspect) HTTP connection has more than maximum_pipelined_requests simultaneous pipelined requests that have not been answered

HTTP connection has more than maximum_pipelined_requests simultaneous pipelined requests that have not been answered.

119:102 (http_inspect) invalid status code in HTTP response

Invalid status code in HTTP response. Either it is outside the range 100-599 or it is not a number.

119:104 (http_inspect) HTTP response has UTF character set that failed to normalize

HTTP response has Content-Type charset=utf-16le, utf-16be, utf-32le, or utf-32be, but UTF decoding of the message body failed.

119:105 (http_inspect) HTTP response has UTF-7 character set

HTTP response has Content-Type charset=utf-7.

119:109 (http_inspect) more than one level of JavaScript obfuscation

More than one level of JavaScript obfuscation. This alert can only be generated when normalize_javascript configuration option is true or enhanced JavaScript normalizer is enabled.

119:110 (http_inspect) consecutive JavaScript whitespaces exceed maximum allowed

Consecutive whitespaces within a JavaScript exceed max_javascript_whitespaces configuration option. This alert can only be generated when normalize_javascript configuration option is true.

119:111 (http_inspect) multiple encodings within JavaScript obfuscated data

More than one encoding within JavaScript obfuscated data. This alert can only be generated when normalize_javascript configuration option is true or enhanced JavaScript normalizer is enabled.

119:112 (http_inspect) SWF file zlib decompression failure

The HTTP message body contains compressed SWF file data with errors that cannot be decompressed.

119:113 (http_inspect) SWF file LZMA decompression failure

The HTTP message body contains compressed LZMA file data with errors that cannot be decompressed.

119:114 (http_inspect) PDF file deflate decompression failure

The HTTP message body contains compressed PDF file data with errors that cannot be decompressed.

119:115 (http_inspect) PDF file unsupported compression type

The HTTP message body contains a compressed PDF file that uses a compression type other than deflate ("FlateDecode" and "Fl").

119:116 (http_inspect) PDF file with more than one compression applied

The HTTP message body contains a PDF file with more than one compression applied.

119:117 (http_inspect) PDF file parse failure

The HTTP message body contains PDF file data with an error that made the start of the PDF compressed stream unable to be located.

119:201 (http_inspect) not HTTP traffic or unrecoverable HTTP protocol error

HTTP inspector is unable to parse this flow. Either the connection is not actually using HTTP or some sort of unrecoverable HTTP protocol error has occurred. This conclusion applies only to one direction of the flow. The opposite direction may be OK.

119:202 (http_inspect) chunk length has excessive leading zeros

Chunk length has five or more leading zeros.

119:203 (http_inspect) white space before or between HTTP messages

White space characters before the first HTTP message or inserted between HTTP messages.

119:204 (http_inspect) request message without URI

HTTP request message does not include a URI. There is nothing between the method and the version except whitespace. Alternatively the 0.9 equivalent which is GET followed by nothing except whitespace.

119:205 (http_inspect) control character in HTTP response reason phrase

The reason phrase in an HTTP response message contains a control character.

119:206 (http_inspect) illegal extra whitespace in start line

There is more than one space (or other whitespace) character between two elements of an HTTP request or status line.

119:207 (http_inspect) corrupted HTTP version

The HTTP version in the start line begins with "HTTP/" but the remainder is not in the expected <digit>.<digit> format.

119:209 (http_inspect) format error in HTTP header

An HTTP header line contains a format error. A well-formed header consists of a field name followed by a colon followed by the field value.

119:210 (http_inspect) chunk header options present

A chunked transfer-encoded HTTP message body contains chunk extensions. A chunk extension is an optional parameter following the chunk length in the chunk header.

119:211 (http_inspect) URI badly formatted

The HTTP request URI is not well-formatted as one of the four types defined for the HTTP protocol.

119:212 (http_inspect) unrecognized type of percent encoding in URI

The HTTP URI contains an unrecognized type of percent encoding.

119:213 (http_inspect) HTTP chunk misformatted

A chunked transfer-encoded HTTP message body contains a misformatted chunk. The following conditions make a chunk misformatted: there are at least five leading whitespaces before the chunk length in the chunk header, there is an illegal character in the chunk length (expressed as the hex number in ASCII), the chunk length is longer than 32 bits, the chunk header is terminated by lone CR (\r) without an LF (\n), the chunk header does not contain the length, or the chunk data is terminated by a character other than CR or LF

119:214 (http_inspect) white space adjacent to chunk length

A chunked transfer-encoded HTTP message body contains a chunk header with white space adjacent to the chunk length. This covers leading and trailing whitespace.

119:215 (http_inspect) white space within header name

An HTTP header name contains whitespace.

119:216 (http_inspect) excessive gzip compression

A gzip-encoded HTTP message body was found to have an excessive compression ratio during decompression.

119:217 (http_inspect) gzip decompression failed

An error was encountered during decompression of a gzip-encoded HTTP message body.

119:218 (http_inspect) HTTP 0.9 requested followed by another request

An HTTP connection contains an HTTP 0.9 request followed by another request. There can only be one 0.9 response per connection because it ends the server-to-client connection.

119:219 (http_inspect) HTTP 0.9 request following a normal request

An HTTP connection contains an HTTP 0.9 request following a normal request.

119:220 (http_inspect) message has both Content-Length and Transfer-Encoding

An HTTP message has both Content-Length and Transfer-Encoding headers. These headers conflict since the size of the message body will be determined by either the Content-Length value or by the chunked transfer-encoding formatting.

119:221 (http_inspect) status code implying no body combined with Transfer-Encoding or nonzero Content-Length

An HTTP server sent a response with a status code implying there will be no body but also sent a Transfer-Encoding or nonzero Content-Length header. The status codes that imply no message body are the informational (1XX) codes, 204 No Content and 304 Not Modified. Transfer-Encoding and nonzero Content-Length headers indicate that there will be a message body.

119:222 (http_inspect) Transfer-Encoding not ending with chunked

The HTTP Transfer-Encoding header value does not end with "chunked". The HTTP protocol specifies that when a transfer coding is applied to a message, "chunked" must the last transfer coding applied to the message body so that the length of the message body can be determined by the client.

119:223 (http_inspect) Transfer-Encoding with encodings before chunked

An HTTP message includes a Transfer-Encoding header value that specifies other encodings before "chunked."

119:225 (http_inspect) unsupported Content-Encoding used

The HTTP Content-Encoding header contains a coding other than gzip and deflate decompression.

119:226 (http_inspect) unknown Content-Encoding used

The HTTP Content-Encoding header contains an unknown coding.

119:227 (http_inspect) multiple Content-Encodings applied

The HTTP Content-Encoding header has multiple values, meaning multiple content encodings have been applied.

119:228 (http_inspect) server response before client request

An HTTP server response was seen before a corresponding client request.

119:229 (http_inspect) PDF/SWF/ZIP decompression of server response too big

The decompressed size of the PDF/SWF/ZIP file contained in the HTTP message body exceeded the configured limit. The decompression limit can be configured with file_id.decompress_buffer_size.

119:230 (http_inspect) nonprinting character in HTTP message header name

An HTTP message header field name contains a nonprinting character.

119:231 (http_inspect) bad Content-Length value in HTTP header

The HTTP Content-Length header value is not a valid decimal length.

119:232 (http_inspect) HTTP header line wrapped

The HTTP header contains a wrapped header line. This means that the header field value has been folded onto multiple lines, indicated by beginning the continuation line with a space or horizontal tab.

119:233 (http_inspect) HTTP header line terminated by CR without a LF

An HTTP header line is terminated by CR (\r) without LF (\n). The HTTP protocol specifies that header lines should be terminated by CRLF (\r\n).

119:234 (http_inspect) chunk terminated by nonstandard separator

A chunked transfer-encoded HTTP message body contains a chunk terminated by a nonstandard separator. The separator defined by the protocol that should terminate each chunk is CRLF (\r\n).

119:235 (http_inspect) chunk length terminated by LF without CR

A chunked transfer-encoded HTTP message body contains a chunk length that is terminated by LF (\n) without CR (\r). The protocol specifies that chunk lengths should be terminated by CRLF (\r\n) as the line separator.

119:236 (http_inspect) more than one response with 100 status code

An HTTP server sent more than one response with 100 Continue status code.

119:237 (http_inspect) 100 status code not in response to Expect header

An HTTP server sent a response with a status code other than 100 Continue in response to a request with an Expect header. The Expect header informs the server that the client will send a (presumably large) message body, and requests that the server send an interim 100 Continue response if it can handle the request.

119:238 (http_inspect) 1XX status code other than 100 or 101

An HTTP server sent an informational (1XX) response with a status code other than 100 Continue or 101 Switching Protocols.

119:239 (http_inspect) Expect header sent without a message body

An HTTP client sent an Expect header without sending a request message body. The Expect header informs the server that the client will send a (presumably large) message body, and requests that the server send an interim 100 Continue response if it can handle the request.

119:240 (http_inspect) HTTP 1.0 message with Transfer-Encoding header

An HTTP 1.0 message contains a Transfer-Encoding header, which is disallowed for that version.

119:241 (http_inspect) Content-Transfer-Encoding used as HTTP header

The Content-Transfer-Encoding field is used as an HTTP header. Content-Transfer-Encoding is a MIME header and is not registered as an HTTP header.

119:242 (http_inspect) illegal field in chunked message trailers

The HTTP trailer contains a header field that is disallowed in chunked message trailers.

119:243 (http_inspect) header field inappropriately appears twice or has two values

The HTTP Age header field appears twice or has two values.

119:244 (http_inspect) invalid value chunked in Content-Encoding header

An HTTP Content-Encoding header has a value of "chunked", which is not a registered content encoding.

119:245 (http_inspect) 206 response sent to a request without a Range header

A partial content (status code 206) response was sent to a request without a Range header, meaning the client did not request the message body be fragmented.

119:246 (http_inspect) *HTTP* in version field not all upper case

An HTTP start line contains a version field where the letters in *HTTP* are not all upper case.

119:247 (http_inspect) white space embedded in critical header value

There is whitespace embedded in the Content-Length header value other than leading and trailing whitespace.

119:248 (http_inspect) gzip compressed data followed by unexpected non-gzip data

While decompressing a gzip-encoded message body, the zipped data stream ended before the end of the message body, so there is unexpected non-gzip data following the compressed data.

119:249 (http_inspect) excessive HTTP parameter key repeats

There is an HTTP parameter key that is repeated at least 100 times within a request query.

119:250 (http_inspect) HTTP/2 Transfer-Encoding header other than identity

There is an HTTP/2 Transfer-Encoding header value other than identity. The HTTP/2 protocol specifies that the chunked transfer encoding is not allowed.

119:251 (http_inspect) HTTP/2 message body overruns Content-Length header value

An HTTP/2 message header contained a Content-Length header value, but the actual message body transferred is larger than that value. The Content-Length header is not used to determine the length of the message body for HTTP/2 traffic.

119:252 (http_inspect) HTTP/2 message body smaller than Content-Length header value

An HTTP/2 message header contained a Content-Length header value, but the actual message body transferred is smaller than that value. The Content-Length header is not used to determine the length of the message body for HTTP/2 traffic.

119:253 (http_inspect) HTTP CONNECT request with a message body

An HTTP client sent a CONNECT request with a request message body.

119:254 (http_inspect) HTTP client-to-server traffic after CONNECT request but before CONNECT response

There was traffic from an HTTP client after the client sent a CONNECT request but before the CONNECT response from the server was received.

119:255 (http_inspect) HTTP CONNECT 2XX response with Content-Length header

An HTTP server sent a successful (2XX) CONNECT response with a Content-Length header.

119:256 (http_inspect) HTTP CONNECT 2XX response with Transfer-Encoding header

An HTTP server sent a successful (2XX) CONNECT response with a Transfer-Encoding header.

119:257 (http_inspect) HTTP CONNECT response with 1XX status code

An HTTP server sent a CONNECT response with an informational (1XX) status code.

119:258 (http_inspect) HTTP CONNECT response before request message completed

An HTTP CONNECT response was received before the request message from the client was completed.

119:259 (http_inspect) malformed HTTP Content-Disposition filename parameter

A Content-Disposition HTTP header field contains a malformed filename parameter.

119:260 (http_inspect) HTTP Content-Length message body was truncated

The TCP connection was closed before the full HTTP message body was transferred. The length of the full message body was determined by the Content-Length HTTP header field.

119:261 (http_inspect) HTTP chunked message body was truncated

The TCP connection was closed before the full HTTP message body was transferred. The message uses the chunked transfer-encoding, so this means there was no well-formed chunk of length zero to terminate the message.

119:262 (http_inspect) HTTP URI scheme longer than 10 characters

The scheme portion of an HTTP URI is longer than 10 characters.

119:263 (http_inspect) HTTP/1 client requested HTTP/2 upgrade

A client sent a request to upgrade an HTTP/1 connection to HTTP/2.

119:264 (http_inspect) HTTP/1 server granted HTTP/2 upgrade

A server granted a request to upgrade a connection from HTTP/1 to HTTP/2.

119:268 (http_inspect) JavaScript code under the external script tags

When HTML <script> tag contains a reference to an external script, it must not contain any executable JavaScript code. This alert is raised if executable (i.e. not comment) code is found inside a script tag that has an external reference. This alert is raised by the enhanced JavaScript normalizer.

119:269 (http_inspect) script opening tag in a short form

In HTML, a script tag must not be self-closing (written as <script /> without a following end-tag). If a self-closing "short-form" script tag is encountered, this alert is raised. This alert is raised by the enhanced JavaScript normalizer.

119:272 (http_inspect) Consecutive commas in HTTP Accept-Encoding header

There are consecutive commas, possibly separated by whitespace, in an HTTP Accept-Encoding header. This pattern constitutes a Microsoft Windows HTTP protocol stack remote code execution attempt. Reference: CVE-2021-31166.

119:275 (http_inspect) HTTP/1 version other than 1.0 or 1.1

The HTTP version in the start line has a valid 1.<subversion> format, but the subversion is not 0 or 1.

119:276 (http_inspect) HTTP version in start line is 0

The HTTP version in the start line has a valid format but the version is 0.

119:277 (http_inspect) HTTP version in start line is higher than 1

The HTTP version in the start line has a valid format but the version is higher than 1. This alert does not apply to HTTP/2 or HTTP/3 traffic.

119:278 (http_inspect) HTTP gzip body with the FEXTRA flag set

The HTTP message body is gzip encoded and the FEXTRA flag is set in the gzip header.

119:279 (http_inspect) invalid status line

HTTP Status-Line failed validation checks. Checks include minimum length, format, characters used, etc.

119:280 (http_inspect) HTTP message headers longer than 63780 bytes

HTTP message headers longer than 63780 bytes

119:281 (http_inspect) invalid request line

HTTP Request-Line failed validation checks. Checks include minimum length, format, characters used, etc.

119:282 (http_inspect) too many white space characters when start line is expected

Packet with more than 20 white space characters when an HTTP Start-Line is required.

119:283 (http_inspect) HTTP message status line longer than 63780 bytes

HTTP message Status-Line longer than 63780 bytes

119:284 (http_inspect) partial start line

Connection closed in the middle of a Request-Line or Status-Line.

119:285 (http_inspect) HTTP message request line longer than 63780 bytes

HTTP message Request-Line longer than 63780 bytes

119:286 (http_inspect) HTTP/2 preface received instead of an HTTP/1 method

HTTP/2 preface received instead of an HTTP/1 method

119:287 (http_inspect) HTTP request method is not on allowed methods list or is on disallowed methods list

HTTP request method is not on allowed methods list or is on disallowed methods list.

119:288 (http_inspect) HTTP gzip body with reserved flag set

HTTP reserved GZIP flags are set

121:1 (http2_inspect) invalid flag set on HTTP/2 frame

Invalid flag set on HTTP/2 frame header

121:2 (http2_inspect) HPACK integer value has leading zeros

HPACK integer value has leading zeros

121:3 (http2_inspect) HTTP/2 stream initiated with invalid stream id

HTTP/2 stream initiated with invalid stream ID. Either server initiated push promise with odd promised stream ID or new stream with stream ID that is not greater than the last one seen on this side.

121:4 (http2_inspect) missing HTTP/2 continuation frame

HTTP/2 Headers, Continuation or Push promise frame without the END_HEADERS flag set was not followed by a Continuation frame.

121:5 (http2_inspect) unexpected HTTP/2 continuation frame

HTTP/2 Continuation frame not preceded by Headers, Continuation or Push promise frame without the END_HEADERS flag.

121:6 (http2_inspect) HTTP/2 headers HPACK decoding error

HTTP/2 headers HPACK decoding error

121:7 (http2_inspect) HTTP/2 connection preface does not match

HTTP/2 connection preface does not match

121:8 (http2_inspect) HTTP/2 request missing required header field

HTTP/2 request missing required header field. CONNECT request without authority, non-CONNECT request without a scheme, or http/https scheme without a path.

121:9 (http2_inspect) HTTP/2 response has no status code

HTTP/2 response has no status code

121:10 (http2_inspect) HTTP/2 CONNECT request with scheme or path

HTTP/2 CONNECT request with scheme or path

121:11 (http2_inspect) error in HTTP/2 settings frame

HTTP/2 settings frame error: stream ID isn't 0, length isn't multiple of 6, or ACK flag is set and length isn't 0.

121:12 (http2_inspect) unknown parameter in HTTP/2 settings frame

Unknown parameter in HTTP/2 settings frame. Parameter identifier is not one of the six RFC-defined values.

121:13 (http2_inspect) invalid HTTP/2 frame sequence

Invalid HTTP/2 frame sequence. Frame type is not valid for current stream state.

121:14 (http2_inspect) HTTP/2 dynamic table has more than 512 entries

HTTP/2 dynamic table has more than 512 entries

121:15 (http2_inspect) HTTP/2 push promise frame with promised stream ID already in use

HTTP/2 push promise frame with promised stream ID already in use

121:16 (http2_inspect) HTTP/2 padding length is bigger than frame data size

HTTP/2 padding length is bigger than frame data size

121:17 (http2_inspect) HTTP/2 pseudo-header after regular header

HTTP/2 pseudo-header after regular header

121:18 (http2_inspect) HTTP/2 pseudo-header in trailers

HTTP/2 pseudo-header in trailers

121:19 (http2_inspect) invalid HTTP/2 pseudo-header

Invalid HTTP/2 pseudo header. For response only :status is valid. For request only :authority, :method, :path and :scheme are valid. Any other pseudo-header or seeing one of these more than once will trigger the alert.

121:20 (http2_inspect) HTTP/2 trailers without END_STREAM bit

HTTP/2 trailers without END_STREAM bit

121:21 (http2_inspect) HTTP/2 push promise frame sent when prohibited by receiver

HTTP/2 push promise frame sent when prohibited by receiver. Receiver prohibited push promise by sending settings frame with SETTINGS_ENABLE_PUSH 0.

121:22 (http2_inspect) padding flag set on HTTP/2 frame with zero length

Padding flag set on HTTP/2 frame with zero length

121:23 (http2_inspect) HTTP/2 push promise frame in client-to-server direction

HTTP/2 push promise frame in client-to-server direction

121:24 (http2_inspect) invalid HTTP/2 push promise frame

Invalid HTTP/2 push promise frame, length is less than promised stream ID length.

121:25 (http2_inspect) HTTP/2 push promise frame sent at invalid time

HTTP/2 push promise frame sent at invalid time. Client didn't send headers yet for this stream, END_STREAM already seen on server side or server side in error state.

121:26 (http2_inspect) invalid parameter value sent in HTTP/2 settings frame

Invalid SETTINGS_ENABLE_PUSH value sent in HTTP/2 settings frame

121:27 (http2_inspect) excessive concurrent HTTP/2 streams

HTTP/2 flow exceed concurrent streams limit, as configured by concurrent_streams_limit.

121:28 (http2_inspect) invalid HTTP/2 rst stream frame

Invalid HTTP/2 RST_STREAM frame. Stream ID is not 0 or length is not 4.

121:29 (http2_inspect) HTTP/2 rst stream frame sent at invalid time

HTTP/2 RST_STREAM frame sent at invalid time. Stream is not in idle state, already started with a push promise or headers frame.

121:30 (http2_inspect) uppercase HTTP/2 header field name

Uppercase HTTP/2 header field name

121:31 (http2_inspect) invalid HTTP/2 window update frame

HTTP/2 window update frame length is not 4

121:32 (http2_inspect) HTTP/2 window update frame with zero increment

HTTP/2 window update frame with zero increment

121:33 (http2_inspect) HTTP/2 request without a method

HTTP/2 request without a method

121:34 (http2_inspect) HTTP/2 HPACK table size update not at the start of a header block

HTTP/2 HPACK table size update not at the start of a header block

121:35 (http2_inspect) More than two HTTP/2 HPACK table size updates in a single header block

More than two HTTP/2 HPACK table size updates in a single header block

121:36 (http2_inspect) HTTP/2 HPACK table size update exceeds max value set by decoder in SETTINGS frame

HTTP/2 HPACK table size update exceeds max value set by decoder in SETTINGS frame

121:37 (http2_inspect) Nonempty HTTP/2 Data frame where message body not expected

Nonempty HTTP/2 Data frame where a message body was not expected.

121:38 (http2_inspect) HTTP/2 non-Data frame longer than 63780 bytes

HTTP/2 non-Data frame longer than 63780 bytes. For HEADERS and PUSH_PROMISE frames this includes the size of any following continuation frames.

121:39 (http2_inspect) not HTTP/2 traffic or unrecoverable HTTP/2 protocol error

HTTP/2 inspector is unable to parse this flow. Either the connection is not actually using HTTP/2 or some sort of unrecoverable HTTP/2 protocol error has occurred. This conclusion applies only to one direction of the flow. The opposite direction may be OK.

121:40 (http2_inspect) invalid HTTP/2 PRIORITY frame

Invalid HTTP/2 PRIORITY frame. Stream ID is 0 or length is not 5.

121:41 (http2_inspect) invalid HTTP/2 GOAWAY frame

Invalid HTTP/2 GOAWAY frame. R bit is set or stream ID is not 0 or length is less than 8.

121:42 (http2_inspect) too many unacknowledged settings

More than 6 unacknowledged settings frames.

121:43 (http2_inspect) setting acknowledgment without actual settings

Unexpected settings ACK.

122:1 (port_scan) TCP portscan

Basic one host to one host TCP portscan where multiple TCP ports are scanned on the destination host from a single host

122:2 (port_scan) TCP decoy portscan

Decoy TCP portscan where the real scanner's host address was mixed with multiple decoy hosts to connect to a single port multiple times

122:3 (port_scan) TCP portsweep

One host to many hosts TCP portsweep where multiple TCP ports are scanned on each destination host

122:4 (port_scan) TCP distributed portscan

Many hosts to one host TCP distributed portscan where many hosts connect to a single destination host and multiple ports are scanned on the destination host

122:5 (port_scan) TCP filtered portscan

Filtered one host to one host TCP portscan where multiple firewall filtered TCP ports are scanned on the destination host from a single host

122:6 (port_scan) TCP filtered decoy portscan

Filtered decoy TCP portscan where the real scanner's host address was mixed with multiple decoy hosts to connect to a single firewall filtered port multiple times

122:7 (port_scan) TCP filtered portsweep

Filtered one host to many hosts TCP portsweep where multiple firewall filtered TCP ports are scanned on each destination host

122:8 (port_scan) TCP filtered distributed portscan

Filtered many hosts to one host TCP distributed portscan where many hosts connect to a single destination host and multiple firewall filtered ports are scanned on the destination host

122:9 (port_scan) IP protocol scan

One host to one host IP protocol scan where multiple IP protocols are scanned on the destination host from a single host

122:10 (port_scan) IP decoy protocol scan

Decoy IP protocol scan where the real scanner's host address was mixed with multiple decoy hosts to scan IP protocols on a single host multiple times

122:11 (port_scan) IP protocol sweep

One host to many hosts IP protocol sweep where multiple IP protocols are scanned on each host

122:12 (port_scan) IP distributed protocol scan

Many hosts to one host distributed IP protocol scan where many hosts attempt to scan multiple IP protocols on a single destination host

122:13 (port_scan) IP filtered protocol scan

Filtered one host to one host IP protocol scan where multiple firewall filtered IP protocols are scanned on the destination host from a single host

122:14 (port_scan) IP filtered decoy protocol scan

Filtered decoy IP protocol scan where the real scanner's host address was mixed with multiple decoy hosts to scan firewall filtered IP protocols on a single host multiple times

122:15 (port_scan) IP filtered protocol sweep

Filtered one host to many hosts IP protocol sweep where multiple firewall filtered IP protocols are scanned on each host

122:16 (port_scan) IP filtered distributed protocol scan

Filtered many hosts to one host distributed IP protocol scan where many hosts attempt to scan multiple firewall filtered IP protocols on a single destination host

122:17 (port_scan) UDP portscan

Basic one host to one host UDP portscan where multiple UDP ports are scanned on the destination host from a single host

122:18 (port_scan) UDP decoy portscan

Decoy UDP portscan where the real scanner's host address was mixed with multiple decoy hosts to scan a single UDP port on the single destination host multiple times

122:19 (port_scan) UDP portsweep

One host to many hosts UDP portsweep where multiple UDP ports are scanned on each destination host from a single host

122:20 (port_scan) UDP distributed portscan

Many hosts to one host distributed UDP portscan where many hosts scan multiple UDP ports on a single destination host

122:21 (port_scan) UDP filtered portscan

Filtered one host to one host UDP portscan where multiple firewall filtered UDP ports are scanned on the destination host from a single host

122:22 (port_scan) UDP filtered decoy portscan

Filtered decoy UDP portscan where the real scanner's host address was mixed with multiple decoy hosts to scan a single firewall filtered UDP port on the single destination host multiple times

122:23 (port_scan) UDP filtered portsweep

Filtered one host to many hosts UDP portsweep where multiple firewall filtered UDP ports are scanned on each destination host from a single host

122:24 (port_scan) UDP filtered distributed portscan

Filtered many hosts to one host distributed UDP portscan where many hosts scan multiple firewall filtered UDP ports on a single destination host

122:25 (port_scan) ICMP sweep

One host to many hosts ICMP sweep scan where multiple ICMP scan occurred on each destination host from a single host

122:26 (port_scan) ICMP filtered sweep

Filtered one host to many hosts ICMP sweep scan where multiple ICMP scan occurred on each firewall filtered destination host from a single host

122:27 (port_scan) open port

open port

123:1 (stream_ip) inconsistent IP options on fragmented packets

Received inconsistent IP options on fragmented packets.

123:2 (stream_ip) teardrop attack

Received indicators of a teardrop attack on fragmented packets.

123:3 (stream_ip) short fragment, possible DOS attempt

Received short fragment, possible DOS attempt (possible boink/bolt/jolt attack). The minimum length required to throw this alert is specified by stream_ip.min_frag_length.

123:4 (stream_ip) fragment packet ends after defragmented packet

Overlap anomaly: fragment packet ends after defragmented packet.

123:5 (stream_ip) zero-byte fragment packet

Received a zero-byte fragment.

123:6 (stream_ip) bad fragment size, packet size is negative

Bad fragment size encountered, packet size is negative.

123:7 (stream_ip) bad fragment size, packet size is greater than 65536

Bad fragment size encountered, packet size is greater than 65536.

123:8 (stream_ip) fragmentation overlap

Fragmentation results in overlap between segments.

123:11 (stream_ip) TTL value less than configured minimum, not using for reassembly

TTL value is less than configured minimum, not using for reassembly. Minimum TTL can be configured with stream_ip.min_ttl.

123:12 (stream_ip) excessive fragment overlap

Fragment overlap limit exceeded, event will be raised for all successive fragments. The max fragment overlaps that can occur before alerting is configurable by changing stream_ip.max_overlaps.

123:13 (stream_ip) tiny fragment

Received a tiny fragment (less than minimum fragment length).

124:1 (smtp) attempted command buffer overflow

SMTP command exceeds the configured max_command_line_len.

124:2 (smtp) attempted data header buffer overflow

SMTP data header exceeds the configured max_header_line_len.

124:3 (smtp) attempted response buffer overflow

SMTP response exceeds the configured max_response_line_len.

124:4 (smtp) attempted specific command buffer overflow

SMTP command that is specified in the alt_max_command_line_len array is detected, and its length exceeds the maximum length that is configured in the array.

124:5 (smtp) unknown command

Command did not match valid_cmds list.

124:6 (smtp) illegal command

Invalid command(invalid_cmds) is detected.

124:7 (smtp) attempted header name buffer overflow

SMTP header name exceeds 64 characters.

124:8 (smtp) attempted X-Link2State command buffer overflow

Microsoft Exchange X-Link2State command exceeds maximum length of 520 characters.

124:10 (smtp) base64 decoding failed

Base64 decoding failed.

124:11 (smtp) quoted-printable decoding failed

Quoted-printable data decoding failed.

124:13 (smtp) Unix-to-Unix decoding failed

Uudecoding failed.

124:14 (smtp) Cyrus SASL authentication attack

Cyrus SASL authentication attack is detected.

124:15 (smtp) attempted authentication command buffer overflow

AUTH command exceeds the configured max_auth_command_line_len.

124:16 (smtp) file decompression failed

File decompression failed.

124:17 (smtp) STARTTLS command injection attempt

SMTP STARTTLS command injection attempt.

124:18 (smtp) mix of LF and CRLF as end of line

SMTP traffic has a mix of LF and CRLF as end of line

125:1 (ftp_server) TELNET cmd on FTP command channel

TELNET command is detected on FTP control channel.

125:2 (ftp_server) invalid FTP command

Invalid FTP command is detected.

125:3 (ftp_server) FTP command parameters were too long

The length of a FTP command parameter is longer than the configured maximum parameter length.

125:4 (ftp_server) FTP command parameters were malformed

One or more FTP command parameters are malformed.

125:5 (ftp_server) FTP command parameters contained potential string format

FTP command parameter had invalid string format. Two or more than % signs are detected in FTP command parameter.

125:6 (ftp_server) FTP response message was too long

FTP response message is longer than the maximum configured response length.

125:7 (ftp_server) FTP traffic encrypted

FTP traffic is encrypted

125:8 (ftp_server) FTP bounce attempt

FTP servers can allow an attacker to connect to arbitrary ports on machines other than the FTP client. This is called as FTP bounce attempt and bounce attempt has been detected.

125:9 (ftp_server) evasive (incomplete) TELNET cmd on FTP command channel

Evasive (incomplete) TELNET command is detected on FTP control channel.

126:1 (telnet) consecutive Telnet AYT commands beyond threshold

Consecutive Telnet AYT(Are you There) commands are detected beyond the configured AYT threshold limit.

126:2 (telnet) Telnet traffic encrypted

Telnet traffic is encrypted.

126:3 (telnet) Telnet subnegotiation begin command without subnegotiation end

Telnet subnegotiation begin command is detected without subnegotiation end.

128:1 (ssh) challenge-response overflow exploit

SSH challenge-response overflow exploit. Amount of data transferred from client is more than configured maximum.

128:2 (ssh) SSH1 CRC32 exploit

SSH1 CRC32 exploit. Amount of data transferred from client is more than configured maximum.

128:3 (ssh) server version string overflow

SSH version string is greater than the configured maximum.

128:5 (ssh) bad message direction

SSH bad message direction.

128:6 (ssh) payload size incorrect for the given payload

SSH payload size incorrect for the given payload.

128:7 (ssh) failed to detect SSH version string

Failed to detect SSH version string.

129:1 (stream_tcp) SYN on established session

Received a TCP SYN on an already established TCP session.

129:2 (stream_tcp) data on SYN packet

Data present on SYN packet.

129:3 (stream_tcp) data sent on stream not accepting data

Data was sent on a stream not accepting data. The stream is in the TIME-WAIT, FIN-WAIT, CLOSED, or CLOSE-WAIT state.

129:4 (stream_tcp) TCP timestamp is outside of PAWS window

The TCP timestamp is outside of PAWS (protection against wrapped sequences) window.

129:5 (stream_tcp) bad segment, adjusted size $\Leftarrow$ 0 (deprecated)

Bad segment, adjusted size $\Leftarrow$ 0 (deprecated)

129:6 (stream_tcp) window size (after scaling) larger than policy allows

Window size (after scaling) is larger than policy allows. stream_tcp.max_window can be increased to allow for larger window sizes if desired.

129:7 (stream_tcp) limit on number of overlapping TCP packets reached

Limit on number of overlapping TCP packets per session was reached. stream_tcp.overlap_limit can be increased to allow for more overlaps per session, if desired.

129:8 (stream_tcp) data sent on stream after TCP reset sent

Data was sent on stream after a TCP reset was sent, and the stream is in CLOSED state.

129:9 (stream_tcp) TCP client possibly hijacked, different ethernet address

TCP client is possibly hijacked, MAC addresses on received packets differ from what was originally seen on this flow.

129:10 (stream_tcp) TCP server possibly hijacked, different ethernet address

TCP server is possibly hijacked, MAC addresses on received packets differ from what was originally seen on this flow.

129:11 (stream_tcp) TCP data with no TCP flags set

Received TCP data with no TCP flags set.

129:12 (stream_tcp) consecutive TCP small segments exceeding threshold

Consecutive (in the order of received packets, not the order of sequence numbers) TCP small segments exceed the configured threshold. The size required to be a small segment can be configured via stream_tcp.small_segments.maximum_size, and the maximum number of these small segments can be configured with int stream_tcp.small_segments.count.

129:13 (stream_tcp) 4-way handshake detected

stream_tcp detected a 4-way handshake, which includes a TCP SYN (without ACK) in response to the initiating client SYN. stream_tcp.require_3whs = 0 should be set to ensure this can be detected in all cases.

129:14 (stream_tcp) TCP timestamp is missing

TCP timestamp is missing, which could cause a failure in PAWS checking, or RTT calculation.

129:15 (stream_tcp) reset outside window

TCP reset was requested outside window (bad RST).

129:16 (stream_tcp) FIN number is greater than prior FIN

TCP Anomaly: FIN number is greater than prior FIN while the connection is in TIME-WAIT.

129:17 (stream_tcp) ACK number is greater than prior FIN

TCP Anomaly: ACK number is greater than prior FIN while the connection is in FIN-WAIT-2.

129:18 (stream_tcp) data sent on stream after TCP reset received

Data was sent on stream after TCP reset received.

129:19 (stream_tcp) TCP window closed before receiving data

TCP window was closed before receiving data.

129:20 (stream_tcp) TCP session without 3-way handshake

The TCP 3-way handshake was not seen for this TCP session.

131:1 (dns) obsolete DNS RR types

DNS Response Resource Record Type is Obsolete.

131:2 (dns) experimental DNS RR types

DNS Response Resource Record Type is Experimental.

131:3 (dns) DNS client rdata txt overflow

DNS Response Resource Record Type is Client rdata Overflow.

133:2 (dce_smb) SMB - bad NetBIOS session service session type

Invalid NetBIOS session service type specified in the header. Valid types are keep alive, request from client, positive response, negative response, and retarget response from the server.

133:3 (dce_smb) SMB - bad SMB message type

Invalid SMB message type specified in the header. Either a request was made by server or a response was given by client.

133:4 (dce_smb) SMB - bad SMB Id (not xffSMB for SMB1 or not xfeSMB for SMB2)

SMB id not equal to \xffSMB for SMB1 or not \xfeSMB for SMB2.

133:5 (dce_smb) SMB - bad word count or structure size

Invalid word count for the command or structure size. SMB commands have specific word counts and if a command with word count not matching with the required word count, this alert is raised.

133:6 (dce_smb) SMB - bad byte count

Bad byte count for the command. Either word count is zero and byte count isn't or byte count is not in the range of minimum and maximum required byte count for the SMB command.

133:7 (dce_smb) SMB - bad format type

Bad format type for the SMB command.

133:8 (dce_smb) SMB - bad offset

Bad Offset. Offset points to beginning of SMB header. Offset is bad, if it points to the data already looked at or after the end of payload.

133:9 (dce_smb) SMB - zero total data count

SMB command has a field containing total amount of data to be transmitted. If this field is zero, an alert is raised.

133:10 (dce_smb) SMB - NetBIOS data length less than SMB header length

NetBIOS data length value is less than size of the SMB header.

133:11 (dce_smb) SMB - remaining NetBIOS data length less than command length

Remaining NetBIOS data length is less than SMB command length.

133:12 (dce_smb) SMB - remaining NetBIOS data length less than command byte count

Remaining NetBIOS data length is less than the SMB command byte count.

133:13 (dce_smb) SMB - remaining NetBIOS data length less than command data size

Remaining NetBIOS data length is less than SMB command data size.

133:14 (dce_smb) SMB - remaining total data count less than this command data size

Total data count is less than SMB command data size. Total data count must always be greater than or equal to current data size.

133:15 (dce_smb) SMB - total data sent (STDu64) greater than command total data expected

Total data sent in transaction is greater than SMB command total data expected.

133:16 (dce_smb) SMB - byte count less than command data size (STDu64)

Byte count in the SMB command header is less than the command data size.

133:17 (dce_smb) SMB - invalid command data size for byte count

Byte count minus predetermined value for the SMB command is not equal to data size.

133:18 (dce_smb) SMB - excessive tree connect requests with pending tree connect responses

Excessive SMB tree connect requests with pending tree connect responses. Tree connect requests queue up and wait for server response. This alert raised for excessing pending tree connect requests.

133:19 (dce_smb) SMB - excessive read requests with pending read responses

Excessive SMB read requests with pending read responses. After client is done writing data, read request is queued and gets dequeued upon receiving response. This alert raised for excessive pending read requests

133:20 (dce_smb) SMB - excessive command chaining

Excessive command chaining. Number of SMB chained commands in a single request is greater than or equal to the configured value.

133:21 (dce_smb) SMB - multiple chained tree connect requests

It is possible to chain multiple Session Setup AndX commands within the same request. There is, however, only one place in the SMB header to return a login handle (or Uid). Windows does not allow this behavior, however Samba does. This is an anomalous behavior.

133:22 (dce_smb) SMB - multiple chained tree connect requests

It is possible to chain multiple Tree Connect AndX commands within the same request. There is, however, only one place in the SMB header to return a tree handle (or Tid). Windows does not allow this behavior, however Samba does. This is anomalous behavior.

133:23 (dce_smb) SMB - chained/compounded login followed by logoff

When a Session Setup AndX request is sent to the server, the server responds with a user id or login handle. This is used by the client in subsequent requests to indicate that it has authenticated. A Logoff AndX request is sent by the client to indicate it wants to end the session and invalidate the login handle. With SMB commands that are chained after a Session Setup AndX request, the login handle returned by the server is used for the subsequent chained commands. The combination of a Session Setup AndX command with a chained Logoff AndX command, essentially logins in and logs off in the same request and is anomalous behavior.

133:24 (dce_smb) SMB - chained/compounded tree connect followed by tree disconnect

A SMB Tree Connect AndX command is used to connect to a share. The Tree Disconnect command is used to disconnect from that share. The combination of a Tree Connect AndX command with a chained Tree Disconnect command, essentially connects to a share and disconnects from the same share in the same request and is anomalous behavior.

133:25 (dce_smb) SMB - chained/compounded open pipe followed by close pipe

An SMB Open AndX or Nt Create AndX command is used to open/create a file handle. The Close command is used to close that file handle. The combination of a Open AndX or Nt Create AndX command with a chained Close command, essentially opens and closes the file handle in the same request and is anomalous behavior.

133:26 (dce_smb) SMB - invalid share access

Invalid SMB shares configured. It looks for a Tree Connect or Tree Connect AndX to the share.

133:27 (dce_tcp) connection oriented DCE/RPC - invalid major version

Major version contained in the connection oriented DCE/RPC header is not equal to 5.

133:28 (dce_tcp) connection oriented DCE/RPC - invalid minor version

Minor version contained in the connection oriented DCE/RPC header is not equal to 0.

133:29 (dce_tcp) connection-oriented DCE/RPC - invalid PDU type

Connection oriented DCE/RPC PDU type contained in the header is not a valid PDU type.

133:30 (dce_tcp) connection-oriented DCE/RPC - fragment length less than header size

Fragment length less than connection oriented DCE/RPC header size.

133:31 (dce_tcp) connection-oriented DCE/RPC - remaining fragment length less than size needed

Connection oriented DCE/RPC remaining fragment length less than size needed.

133:32 (dce_tcp) connection-oriented DCE/RPC - no context items specified

In connection oriented DCE/RPC Client's Bind or Alter Context request, there are no context items specified.

133:33 (dce_tcp) connection-oriented DCE/RPC -no transfer syntaxes specified

In connection oriented DCE/RPC Client's Bind or Alter context request, there are no transfer syntaxes to go with the requested interface.

133:34 (dce_tcp) connection-oriented DCE/RPC - fragment length on non-last fragment less than maximum negotiated fragment transmit size for client

Connection oriented DCE/RPC non-last fragment is less than the size of the negotiated maximum fragment length. Most evasion techniques try to fragment the data as much as possible and usually each fragment comes well below the negotiated transmit size.

133:35 (dce_tcp) connection-oriented DCE/RPC - fragment length greater than maximum negotiated fragment transmit size

Connection oriented DCE/RPC fragment length greater than maximum negotiated fragment length.

133:36 (dce_tcp) connection-oriented DCE/RPC - alter context byte order different from bind

Alter context byte order different from bind. The byte order of the request data is determined by the Bind in connection-oriented DCE/RPC for Windows. It is anomalous behavior to attempt to change the byte order.

133:37 (dce_tcp) connection-oriented DCE/RPC - call id of non first/last fragment different from call id established for fragmented request

Call id of non first/last fragment different from call id established for fragmented request in connection oriented DCE/RPC. The call id for a set of fragments in a fragmented request should stay the same.

133:38 (dce_tcp) connection-oriented DCE/RPC - opnum of non first/last fragment different from opnum established for fragmented request

Connection-oriented DCE/RPC opnum of non first/last fragment different from opnum established for fragmented request. The operation number specifies which function the request is calling on the bound interface. If a request is fragmented, this number should stay the same for all fragments.

133:39 (dce_tcp) connection-oriented DCE/RPC - context id of non first/last fragment different from context id established for fragmented request

Connection-oriented DCE/RPC context id of non first/last fragment different from context id established for fragmented request. The context id is a handle to a interface that was bound to. If a request if fragmented, this number should stay same for all fragments.

133:40 (dce_udp) connection-less DCE/RPC - invalid major version

Connection-less DCE/RPC invalid major version. Major version is not equal to 4.

133:41 (dce_udp) connection-less DCE/RPC - invalid PDU type

Connection-less DCE/RPC PDU type is not a valid PDU type.

133:42 (dce_udp) connection-less DCE/RPC - data length less than header size

Connection-less DCE/RPC packet data length is less than size of the header.

133:43 (dce_udp) connection-less DCE/RPC - bad sequence number

Connection-less DCE/RPC bad sequence number. The sequence number used in a request is the same or less than a previously used sequence number on the session.

133:44 (dce_smb) SMB - invalid SMB version 1 seen

Invalid SMB version 1 seen.

133:45 (dce_smb) SMB - invalid SMB version 2 seen

Invalid SMB version 2 seen.

133:46 (dce_smb) SMB - invalid user, tree connect, file binding

SMB invalid user, tree connect, file binding seen.

133:47 (dce_smb) SMB - excessive command compounding

SMB excessive command compounding seen.

133:48 (dce_smb) SMB - zero data count

SMB Data count is zero.

133:50 (dce_smb) SMB - maximum number of outstanding requests exceeded

Maximum number of outstanding SMB requests exceeded.

133:51 (dce_smb) SMB - outstanding requests with same MID

Multiple outstanding SMB requests with same MID. When a client sends a request it uses a value called the MID (multiplex id) to match a response, which the server is supposed to echo, to a request.

133:52 (dce_smb) SMB - deprecated dialect negotiated

Deprecated dialect negotiated. In the Negotiate request a client gives a list of SMB dialects it supports, normally in order from least desirable to most desirable and the server responds with the index of the dialect to be used on the SMB session. If the client doesn't offer it as a supported dialect or the server chooses a lesser dialect, it is deprecated dialect negotiated.

133:53 (dce_smb) SMB - deprecated command used

Deprecated SMB command used. There are a number of commands that are considered deprecated and/or obsolete by Microsoft (see MS-CIFS and MS-SMB). Detected use of a deprecated/obsolete command.

133:54 (dce_smb) SMB - unusual command used

Unusual SMB command used. There are some commands considered unusual in the context they are used. Some of the commands such as : TRANS_READ_NMPIPE/TRANS_WRITE_NMPIPE/TRANS2_OPEN2/NT_TRANSACT_CREATE/NT_TRANSAC

133:55 (dce_smb) SMB - invalid setup count for command

Transaction SMB commands have a setup count field that indicates word count in the transaction setup, Alert raised if setup count is invalid for transaction command.

133:56 (dce_smb) SMB - client attempted multiple dialect negotiations on session

Client attempted multiple SMB dialect negotiations on session. There can be only one Negotiate transaction per session and it is the first thing a client and server do to determine the SMB dialect each supports.

133:57 (dce_smb) SMB - client attempted to create or set a file's attributes to readonly/hidden/system

SMB client attempted to create or set a file's attributes to readonly/hidden/system. Malware will often set a files attributes to ReadOnly/Hidden/System if it is successful in installing itself as a Windows service or is able to write an autorun.inf file since it doesn't want the user to see the file and the default folder options in Windows is not to display Hidden files.

133:58 (dce_smb) SMB - file offset provided is greater than file size specified

SMB file offset provided is greater than file size specified.

133:59 (dce_smb) SMB - next command specified in SMB2 header is beyond payload boundary

SMB protocol allows multiple smb commands to be grouped in a single packet. Next command specified in SMB2 header is greater than the payload boundary.

134:1 (latency) rule tree suspended due to latency

(latency) rule tree suspended due to latency

134:2 (latency) rule tree re-enabled after suspend timeout

(latency) rule tree re-enabled after suspend timeout

134:3 (latency) packet fastpathed due to latency

(latency) packet fastpathed due to latency

135:1 (stream) TCP SYN received

A TCP SYN was received.

135:2 (stream) TCP session established

A TCP session was established.

135:3 (stream) TCP session cleared

A TCP session was cleared.

136:1 (reputation) packets blocked based on source

The flow was blocked based on the source IP address, since it appears on the IP reputation block list. Configure either the discovery filter, or the reputation IP lists to change this behavior.

136:2 (reputation) packets trusted based on source

The flow was trusted based on the source IP address, since it appears on the IP reputation trust list. Configure either the discovery filter, or the reputation IP lists to change this behavior.

136:3 (reputation) packets monitored based on source

The flow was monitored based on the source IP address, since it appears on the IP reputation monitor list. Configure either the discovery filter, or the reputation IP lists to change this behavior.

136:4 (reputation) packets blocked based on destination

The flow was blocked based on the destination IP address, since it appears on the IP reputation block list. If the flow contained proxy traffic, the IP address could also be the address of the (inner-layer) proxied connection. Configure either the discovery filter, or the reputation IP lists to change this behavior.

136:5 (reputation) packets trusted based on destination

The flow was trusted based on the destination IP address, since it appears on the IP reputation trust list. If the flow contained proxy traffic, the IP address could also be the address of the (inner-layer) proxied connection. Configure either the discovery filter, or the reputation IP lists to change this behavior.

136:6 (reputation) packets monitored based on destination

The flow was monitored (passed to further inspection) based on the destination IP address, since it appears on the IP reputation monitor list. If the flow contained proxy traffic, the IP address could also be the address of the (inner-layer) proxied connection. Configure either the discovery filter, or the reputation IP lists to change this behavior.

137:1 (ssl) invalid client HELLO after server HELLO detected

An invalid SSL client HELLO was received after an SSL server HELLO has been detected.

137:2 (ssl) invalid server HELLO without client HELLO detected

An invalid SSL server HELLO was received without an SSL client HELLO having been detected.

137:3 (ssl) heartbeat read overrun attempt detected

An SSL heartbeat read overrun attempt has been detected.

137:4 (ssl) large heartbeat response detected

A large SSL heartbeat response was detected.

140:2 (sip) empty request URI

SIP Request_URI header field is empty.

140:3 (sip) URI is too long

SIP Request_URI header field is larger than the defined length in configuration.

140:4 (sip) empty call-Id

SIP Call-ID header field is empty.

140:5 (sip) Call-Id is too long

SIP Call-ID header field is larger than the defined length in configuration.

140:6 (sip) CSeq number is too large or negative

SIP header field CSeq number is too large or negative. The CSeq number value must be expressible as a 32-bit unsigned integer and must be less than 2^31.

140:7 (sip) request name in CSeq is too long

The request name in the CSeq is larger than the defined length in configuration.

140:8 (sip) empty From header

SIP From header field is empty.

140:9 (sip) From header is too long

SIP From field in header is larger than the defined length in configuration.

140:10 (sip) empty To header

SIP To field in header is empty.

140:11 (sip) To header is too long

SIP To field in header is larger than the defined length in configuration.

140:12 (sip) empty Via header

SIP Via field in header is empty.

140:13 (sip) Via header is too long

SIP Via field in header is larger than the defined length in configuration.

140:14 (sip) empty Contact

SIP contact field in header is empty.

140:15 (sip) contact is too long

SIP contact field in header is larger than the defined length in configuration.

140:16 (sip) content length is too large or negative

SIP content length is too large or negative.

140:17 (sip) multiple SIP messages in a packet

SIP packet has multiple requests in a single packet.

140:18 (sip) content length mismatch

Inconsistencies present between the Content-Length in SIP header and actual body data.

140:19 (sip) request name is invalid

SIP request name field is invalid in response.

140:20 (sip) Invite replay attack

SIP received authenticated invite message, but no challenge from server is received. This is the case of Invite replay attack.

140:21 (sip) illegal session information modification

SIP received authenticated invite message, but session information has been changed. This is different from re-INVITE, where the dialog has been established and authenticated.

140:22 (sip) response status code is not a 3 digit number

SIP response status code is not a 3 digit number.

140:23 (sip) empty Content-type header

SIP Content-type header field is empty.

140:24 (sip) SIP version is invalid

SIP version is invalid. SIP version other than 1.0, 1.1, and 2.0 is invalid.

140:25 (sip) mismatch in METHOD of request and the CSEQ header

Mismatch in method of request and the CSEQ header detected.

140:26 (sip) method is unknown

SIP method is unknown.

140:27 (sip) maximum dialogs within a session reached

SIP dialog numbers in the stream session exceeds the maximal value.

141:1 (imap) unknown IMAP3 command

Unknown IMAP3 command is detected.

141:2 (imap) unknown IMAP3 response

Unknown IMAP3 response is detected.

141:4 (imap) base64 decoding failed

Base64 decoding failed.

141:5 (imap) quoted-printable decoding failed

Quoted-printable decoding failed.

141:7 (imap) Unix-to-Unix decoding failed

Uudecoding failed.

141:8 (imap) file decompression failed

File decompression failed.

142:1 (pop) unknown POP3 command

Unknown POP3 command is detected.

142:2 (pop) unknown POP3 response

Unknown POP3 response is detected.

142:4 (pop) base64 decoding failed

Base64 decoding failed.

142:5 (pop) quoted-printable decoding failed

Quoted-printable decoding failed.

142:7 (pop) Unix-to-Unix decoding failed

Uudecoding failed.

142:8 (pop) file decompression failed

File decompression failed.

143:1 (gtp_inspect) message length is invalid

gtp_inspect detected invalid message length

143:2 (gtp_inspect) information element length is invalid

gtp_inspect detected invalid information element length

143:3 (gtp_inspect) information elements are out of order

gtp_inspect detected information elements are out of order

143:4 (gtp_inspect) TEID is missing

gtp_inspect detected tunnel endpoint identifier having zero

144:1 (modbus) length in Modbus MBAP header does not match the length needed for the given function

Length in Modbus MBAP header does not match the length needed for the given function or length mismatch discovered while parsing the PDU

144:2 (modbus) Modbus protocol ID is non-zero

Modbus protocol ID is non-zero

144:3 (modbus) reserved Modbus function code in use

Modbus using reserved function code

145:1 (dnp3) DNP3 link-layer frame contains bad CRC

DNP3 link-layer frame contains bad CRC

145:2 (dnp3) DNP3 link-layer frame is truncated or frame length is invalid

DNP3 link-layer frame is truncated or frame length is invalid

145:3 (dnp3) DNP3 transport-layer segment sequence number is incorrect

DNP3 transport-layer segment sequence number is incorrect

145:4 (dnp3) DNP3 transport-layer segment flag violation is detected

DNP3 transport-layer segment flag violation is detected, FIR flag was set in middle fragment

145:5 (dnp3) DNP3 link-layer frame uses a reserved address

DNP3 link-layer frame uses a reserved address (0xFFF0 to 0xFFFB)

145:6 (dnp3) DNP3 application-layer fragment uses a reserved function code

DNP3 application-layer fragment uses an undefined function code, defined function codes: Requests (0 to 33) and Responses (129 to 131)

148:1 (cip) CIP data is malformed

(cip) CIP data is malformed

148:2 (cip) CIP data is non-conforming to ODVA standard

(cip) CIP data is non-conforming to ODVA standard

148:3 (cip) CIP connection limit exceeded. Least recently used connection removed

(cip) CIP connection limit exceeded. Least recently used connection removed

148:4 (cip) CIP unconnected request limit exceeded. Oldest request removed

(cip) CIP unconnected request limit exceeded. Oldest request removed

149:1 (s7commplus) length in S7commplus MBAP header does not match the length needed for the given S7commplus function

(s7commplus) length in S7commplus MBAP header does not match the length needed for the given S7commplus function

149:2 (s7commplus) S7commplus protocol ID is non-zero

(s7commplus) S7commplus protocol ID is non-zero

149:3 (s7commplus) reserved S7commplus function code in use

(s7commplus) reserved S7commplus function code in use

150:1 (file_id) file not processed due to per flow limit

(file_id) file not processed due to per flow limit

151:1 (iec104) Length in IEC104 APCI header does not match the length needed for the given IEC104 ASDU type id

(iec104) Length in IEC104 APCI header does not match the length needed for the given IEC104 ASDU type id

151:2 (iec104) IEC104 Start byte does not match 0x68

(iec104) IEC104 Start byte does not match 0x68

151:3 (iec104) Reserved IEC104 ASDU type id in use

(iec104) Reserved IEC104 ASDU type id in use

151:4 (iec104) IEC104 APCI U Reserved field contains a non-default value

(iec104) IEC104 APCI U Reserved field contains a non-default value

151:5 (iec104) IEC104 APCI U message type was set to an invalid value

(iec104) IEC104 APCI U message type was set to an invalid value

151:6 (iec104) IEC104 APCI S Reserved field contains a non-default value

(iec104) IEC104 APCI S Reserved field contains a non-default value

151:7 (iec104) IEC104 APCI I number of elements set to zero

(iec104) IEC104 APCI I number of elements set to zero

151:8 (iec104) IEC104 APCI I SQ bit set on an ASDU that does not support the feature

(iec104) IEC104 APCI I SQ bit set on an ASDU that does not support the feature

151:9 (iec104) IEC104 APCI I number of elements set to greater than one on an ASDU that does not support the feature

(iec104) IEC104 APCI I number of elements set to greater than one on an ASDU that does not support the feature

151:10 (iec104) IEC104 APCI I Cause of Initialization set to a reserved value

(iec104) IEC104 APCI I Cause of Initialization set to a reserved value

151:11 (iec104) IEC104 APCI I Qualifier of Interrogation Command set to a reserved value

(iec104) IEC104 APCI I Qualifier of Interrogation Command set to a reserved value

151:12 (iec104) IEC104 APCI I Qualifier of Counter Interrogation Command request parameter set to a reserved value

(iec104) IEC104 APCI I Qualifier of Counter Interrogation Command request parameter set to a reserved value

151:13 (iec104) IEC104 APCI I Qualifier of Parameter of Measured Values kind of parameter set to a reserved value

(iec104) IEC104 APCI I Qualifier of Parameter of Measured Values kind of parameter set to a reserved value

151:14 (iec104) IEC104 APCI I Qualifier of Parameter of Measured Values local parameter change set to a technically valid but unused value

(iec104) IEC104 APCI I Qualifier of Parameter of Measured Values local parameter change set to a technically valid but unused value

151:15 (iec104) IEC104 APCI I Qualifier of Parameter of Measured Values parameter option set to a technically valid but unused value

(iec104) IEC104 APCI I Qualifier of Parameter of Measured Values parameter option set to a technically valid but unused value

151:16 (iec104) IEC104 APCI I Qualifier of Parameter Activation set to a reserved value

(iec104) IEC104 APCI I Qualifier of Parameter Activation set to a reserved value

151:17 (iec104) IEC104 APCI I Qualifier of Command set to a reserved value

(iec104) IEC104 APCI I Qualifier of Command set to a reserved value

151:18 (iec104) IEC104 APCI I Qualifier of Reset Process set to a reserved value

(iec104) IEC104 APCI I Qualifier of Reset Process set to a reserved value

151:19 (iec104) IEC104 APCI I File Ready Qualifier set to a reserved value

(iec104) IEC104 APCI I File Ready Qualifier set to a reserved value

151:20 (iec104) IEC104 APCI I Section Ready Qualifier set to a reserved value

(iec104) IEC104 APCI I Section Ready Qualifier set to a reserved value

151:21 (iec104) IEC104 APCI I Select and Call Qualifier set to a reserved value

(iec104) IEC104 APCI I Select and Call Qualifier set to a reserved value

151:22 (iec104) IEC104 APCI I Last Section or Segment Qualifier set to a reserved value

(iec104) IEC104 APCI I Last Section or Segment Qualifier set to a reserved value

151:23 (iec104) IEC104 APCI I Acknowledge File or Section Qualifier set to a reserved value

(iec104) IEC104 APCI I Acknowledge File or Section Qualifier set to a reserved value

151:24 (iec104) IEC104 APCI I Structure Qualifier set on a message where it should have no effect

(iec104) IEC104 APCI I Structure Qualifier set on a message where it should have no effect

151:25 (iec104) IEC104 APCI I Single Point Information Reserved field contains a non-default value

(iec104) IEC104 APCI I Single Point Information Reserved field contains a non-default value

151:26 (iec104) IEC104 APCI I Double Point Information Reserved field contains a non-default value

(iec104) IEC104 APCI I Double Point Information Reserved field contains a non-default value

151:27 (iec104) IEC104 APCI I Cause of Transmission set to a reserved value

(iec104) IEC104 APCI I Cause of Transmission set to a reserved value

151:28 (iec104) IEC104 APCI I Cause of Transmission set to a value not allowed for the ASDU

(iec104) IEC104 APCI I Cause of Transmission set to a value not allowed for the ASDU

151:29 (iec104) IEC104 APCI I invalid two octet common address value detected

(iec104) IEC104 APCI I invalid two octet common address value detected

151:30 (iec104) IEC104 APCI I Quality Descriptor Structure Reserved field contains a non-default value

(iec104) IEC104 APCI I Quality Descriptor Structure Reserved field contains a non-default value

151:31 (iec104) IEC104 APCI I Quality Descriptor for Events of Protection Equipment Structure Reserved field contains a non-default value

(iec104) IEC104 APCI I Quality Descriptor for Events of Protection Equipment Structure Reserved field contains a non-default value

151:32 (iec104) IEC104 APCI I IEEE STD 754 value results in NaN

(iec104) IEC104 APCI I IEEE STD 754 value results in NaN

151:33 (iec104) IEC104 APCI I IEEE STD 754 value results in infinity

(iec104) IEC104 APCI I IEEE STD 754 value results in infinity

151:34 (iec104) IEC104 APCI I Single Event of Protection Equipment Structure Reserved field contains a non-default value

(iec104) IEC104 APCI I Single Event of Protection Equipment Structure Reserved field contains a non-default value

151:35 (iec104) IEC104 APCI I Start Event of Protection Equipment Structure Reserved field contains a non-default value

(iec104) IEC104 APCI I Start Event of Protection Equipment Structure Reserved field contains a non-default value

151:36 (iec104) IEC104 APCI I Output Circuit Information Structure Reserved field contains a non-default value

(iec104) IEC104 APCI I Output Circuit Information Structure Reserved field contains a non-default value

151:37 (iec104) IEC104 APCI I Abnormal Fixed Test Bit Pattern detected

(iec104) IEC104 APCI I Abnormal Fixed Test Bit Pattern detected

151:38 (iec104) IEC104 APCI I Single Command Structure Reserved field contains a non-default value

(iec104) IEC104 APCI I Single Command Structure Reserved field contains a non-default value

151:39 (iec104) IEC104 APCI I Double Command Structure contains an invalid value

(iec104) IEC104 APCI I Double Command Structure contains an invalid value

151:40 (iec104) IEC104 APCI I Regulating Step Command Structure Reserved field contains a non-default value

(iec104) IEC104 APCI I Regulating Step Command Structure Reserved field contains a non-default value

151:41 (iec104) IEC104 APCI I Time2a Millisecond set outside of the allowable range

(iec104) IEC104 APCI I Time2a Millisecond set outside of the allowable range

151:42 (iec104) IEC104 APCI I Time2a Minute set outside of the allowable range

(iec104) IEC104 APCI I Time2a Minute set outside of the allowable range

151:43 (iec104) IEC104 APCI I Time2a Minute Reserved field contains a non-default value

(iec104) IEC104 APCI I Time2a Minute Reserved field contains a non-default value

151:44 (iec104) IEC104 APCI I Time2a Hours set outside of the allowable range

(iec104) IEC104 APCI I Time2a Hours set outside of the allowable range

151:45 (iec104) IEC104 APCI I Time2a Hours Reserved field contains a non-default value

(iec104) IEC104 APCI I Time2a Hours Reserved field contains a non-default value

151:46 (iec104) IEC104 APCI I Time2a Day of Month set outside of the allowable range

(iec104) IEC104 APCI I Time2a Day of Month set outside of the allowable range

151:47 (iec104) IEC104 APCI I Time2a Month set outside of the allowable range

(iec104) IEC104 APCI I Time2a Month set outside of the allowable range

151:48 (iec104) IEC104 APCI I Time2a Month Reserved field contains a non-default value

(iec104) IEC104 APCI I Time2a Month Reserved field contains a non-default value

151:49 (iec104) IEC104 APCI I Time2a Year set outside of the allowable range

(iec104) IEC104 APCI I Time2a Year set outside of the allowable range

151:50 (iec104) IEC104 APCI I Time2a Year Reserved field contains a non-default value

(iec104) IEC104 APCI I Time2a Year Reserved field contains a non-default value

151:51 (iec104) IEC104 APCI I a null Length of Segment value has been detected

(iec104) IEC104 APCI I a null Length of Segment value has been detected

151:52 (iec104) IEC104 APCI I an invalid Length of Segment value has been detected

(iec104) IEC104 APCI I an invalid Length of Segment value has been detected

151:53 (iec104) IEC104 APCI I Status of File set to a reserved value

(iec104) IEC104 APCI I Status of File set to a reserved value

151:54 (iec104) IEC104 APCI I Qualifier of Set Point Command ql field set to a reserved value

(iec104) IEC104 APCI I Qualifier of Set Point Command ql field set to a reserved value

154:1 (js_norm) nested unescape functions

Enhanced JavaScript normalizer has encountered nested unescape functions

154:2 (js_norm) mixed unescape sequence

Enhanced JavaScript normalizer has encountered mixed unescape sequence

154:3 (js_norm) bad token

Enhanced JavaScript normalizer has encountered a symbol that is not expected as a part of a valid JavaScript statement, making further normalization impossible.

154:4 (js_norm) unexpected HTML script opening tag

HTML <script> tag must not have a nested <script> tag inside it. If a nested tag is encountered, this alert is raised. This alert is raised by the enhanced JavaScript normalizer.

154:5 (js_norm) unexpected HTML script closing tag

This alert is raised when </script> end-tag is encountered inside a JavaScript comment or literal, which is a syntax error, as the last comment or literal is not closed before script end. This alert is raised by the enhanced JavaScript normalizer.

154:6 (js_norm) max number of unique identifiers reached

JavaScript normalization includes identifier substitution, which brings arbitrary JavaScript identifiers to a common form. Amount of unique identifiers to normalize is limited, for memory considerations, with http_inspect.js_norm_identifier_depth parameter. When this threshold is reached, a corresponding alert is raised. This alert is not expected for typical network traffic and may be an indication that an attacker is trying to exhaust resources. This alert is raised by the enhanced JavaScript normalizer.

154:7 (js_norm) excessive bracket nesting

In JavaScript, template literals can have substitutions, that in turn can have nested template literals, which requires a stack to track for proper whitespace normalization. Also, the normalization tracks the current bracket scope, which requires a stack as well. When the depth of nesting exceeds limit set in http_inspect.js_norm_max_tmpl_nest or in http_inspect.js_norm_max_bracket_depth, this alert is raised. This alert is not expected for typical network traffic and may be an indication that an attacker is trying to exhaust resources. This alert is raised by the enhanced JavaScript normalizer.

154:8 (js_norm) data gaps during normalization

This alert is raised for the following situation. During JavaScript normalization some data can be lost and not normalized. Usually it happens when rules have file_data and js_data ips options and fast-pattern (FP) search is applying to file_data. Some data doesn't match file_data FP search and JavaScript normalization won't be executed for it. The following normalization for inline/external scripts will be stopped for current request within the flow. This alert is raised by the enhanced JavaScript normalizer.

154:9 (js_norm) excessive scope nesting

To resolve variable names in JavaScript, a current stack of variable scopes has to be tracked. When the depth of nesting exceeds the limit set in http_inspect.js_norm_max_scope_depth, this alert is raised. This alert is not expected for typical network traffic and may be an indication that an attacker is trying to exhaust resources. This alert is raised by the enhanced JavaScript normalizer.

175:1 (domain_filter) configured domain detected

(domain_filter) configured domain detected

256:1 (dpx) too much data sent to port

(dpx) too much data sent to port

11.8 Command Set

- **appid.enable_debug**(proto, src_ip, src_port, dst_ip, dst_port): enable appid debugging
- **appid.disable_debug**(): disable appid debugging
- **appid.reload_third_party**(): reload appid third-party module
- **appid.reload_detectors**(): reload appid detectors
- **host_cache.dump**(file_name): dump host cache

- **host_cache.delete_host**(host_ip): delete host from host cache
- **host_cache.delete_network_proto**(host_ip, proto): delete network protocol from host
- **host_cache.delete_transport_proto**(host_ip, proto): delete transport protocol from host
- **host_cache.delete_service**(host_ip, port, proto): delete service from host
- **host_cache.delete_client**(host_ip, id, service, version): delete client from host
- **host_cache.get_stats**(): get current host cache usage and pegs
- **host_cache.get_segment_stats**(segment): get usage and pegs for cache segment(s)
- **network.set_policy**(id): set the network policy for commands given the user policy id
- **packet_capture.enable**(filter, group): dump raw packets
- **packet_capture.disable**(): stop packet dump
- **packet_tracer.enable**(proto, src_ip, src_port, dst_ip, dst_port): enable packet tracer debugging
- **packet_tracer.disable**(): disable packet tracer
- **perf_monitor.enable_flow_ip_profiling**(seconds, packets): enable statistics on host pairs
- **perf_monitor.disable_flow_ip_profiling**(): disable statistics on host pairs
- **perf_monitor.show_flow_ip_profiling**(): show status of statistics on host pairs
- **profiler.rule_start**(): enable rule profiler
- **profiler.rule_stop**(): disable rule profiler
- **profiler.rule_status**(): print rule profiler status
- **profiler.rule_dump**(output): print rule statistics in table or json format (json format prints dates as Unix epoch)
- **profiler.module_start**(): enable module time profiling
- **profiler.module_stop**(): disable module time profiling
- **profiler.module_dump**(): print module time profiling statistics
- **profiler.module_status**(): show module time profiler status
- **reputation.reload**(): reload reputation data
- **rna.dump_macs**(): dump rna's internal MAC trackers
- **rna.delete_mac_host**(mac): delete a MAC from rna's MAC cache
- **rna.delete_mac_host_proto**(mac, proto): delete a protocol associated with a MAC host
- **rna.purge_data**(): purge all host cache and mac cache data
- **snort.set_watchdog_params**(timer, min_thread_count): set watchdog parameters
- **snort.show_plugins**(): show available plugins
- **snort.delete_inspector**(inspector): delete an inspector from the default policy
- **snort.dump_stats**(): show summary statistics
- **snort.dump_heap_stats**(): show heap statistics
- **snort.reset_stats**(type): clear summary statistics. Type can be: daq|module|appid|file_id|snort|ha|all. reset_stats() without a parameter clears all statistics.

- **snort.rotate_stats**(): roll perfmonitor log files
- **snort.reload_config**(filename): load new configuration
- **snort.reload_policy**(filename): reload part or all of the default policy
- **snort.reload_daq**(): reload daq module
- **snort.reload_hosts**(filename): load a new hosts table
- **snort.log_command**(command, logging): enable or disable command logging
- **snort.show_config_generation**(): show loaded configuration ID
- **snort.pause**(): suspend packet processing
- **snort.resume**(pkt_num): continue packet processing. If number of packets is specified, will resume for n packets and pause
- **snort.detach**(): detach from control shell (without shutting down)
- **snort.quit**(): shutdown and dump-stats
- **snort.help**(): this output
- **trace.set**(modules, constraints, ntuple, timestamp): set modules traces, constraints, ntuple and timestamp options
- **trace.clear**(): clear modules traces and constraints

11.9 Signals

Important
Signal numbers are for the system that generated this documentation and are not applicable elsewhere.

- **term**(15): shutdown normally
- **int**(2): shutdown normally
- **quit**(3): shutdown as if started with --dirty-pig
- **stats**(10): dump stats to stdout
- **rotate**(12): rotate stats files
- **reload**(1): reload config file
- **hosts**(23): reload hosts file

11.10 Module Listing

- **ack** (ips_option): rule option to match on TCP ack numbers
- **active** (basic): configure responses
- **address_space_selector** (policy_selector): configure traffic processing based on address space
- **alert_csv** (logger): output event in csv format
- **alert_ex** (logger): output gid:sid:rev for alerts
- **alert_fast** (logger): output event with brief text format

- **alert_full** (logger): output event with full packet dump
- **alert_json** (logger): output event in json format
- **alert_syslog** (logger): output event to syslog
- **alert_talos** (logger): output event in Talos alert format
- **alert_unixsock** (logger): output event over unix socket
- **alerts** (basic): configure alerts
- **appid** (inspector): application and service identification
- **appid_listener** (inspector): log selected published data to appid_listener.log
- **appids** (ips_option): detection option for application ids
- **arp** (codec): support for address resolution protocol
- **arp_spoof** (inspector): detect ARP attacks and anomalies
- **attribute_table** (basic): configure hosts loading
- **auth** (codec): support for IP authentication header
- **back_orifice** (inspector): back orifice detection
- **base64_decode** (ips_option): rule option to decode base64 data - must be used with base64_data option
- **ber_data** (ips_option): rule option to move to the data for a specified BER element
- **ber_skip** (ips_option): rule option to skip BER element
- **binder** (inspector): configure processing based on CIDRs, ports, services, etc.
- **bufferlen** (ips_option): rule option to check length of current buffer
- **byte_extract** (ips_option): rule option to convert data to an integer variable
- **byte_jump** (ips_option): rule option to move the detection cursor
- **byte_math** (ips_option): rule option to perform mathematical operations on extracted value and a specified value or existing variable
- **byte_test** (ips_option): rule option to convert data to integer and compare
- **cip** (inspector): cip inspection
- **cip_attribute** (ips_option): detection option to match CIP attribute
- **cip_class** (ips_option): detection option to match CIP class
- **cip_conn_path_class** (ips_option): detection option to match CIP Connection Path Class
- **cip_instance** (ips_option): detection option to match CIP instance
- **cip_req** (ips_option): detection option to match CIP request
- **cip_rsp** (ips_option): detection option to match CIP response
- **cip_service** (ips_option): detection option to match CIP service
- **cip_status** (ips_option): detection option to match CIP response status
- **ciscometadata** (codec): support for cisco metadata
- **classifications** (basic): define rule categories with priority

- **classtype** (ips_option): general rule option for rule classification
- **content** (ips_option): payload rule option for basic pattern matching
- **cpeos_test** (inspector): for testing CPE OS RNA event generation
- **cvs** (ips_option): payload rule option for detecting specific attacks
- **daq** (basic): configure packet acquisition interface
- **data_log** (inspector): log selected published data to data.log
- **dce_http_proxy** (inspector): dce over http inspection - client to/from proxy
- **dce_http_server** (inspector): dce over http inspection - proxy to/from server
- **dce_iface** (ips_option): detection option to check dcerpc interface
- **dce_opnum** (ips_option): detection option to check dcerpc operation number
- **dce_smb** (inspector): dce over smb inspection
- **dce_stub_data** (ips_option): sets the cursor to dcerpc stub data
- **dce_tcp** (inspector): dce over tcp inspection
- **dce_udp** (inspector): dce over udp inspection
- **decode** (basic): general decoder rules
- **detection** (basic): configure general IPS rule processing parameters
- **detection_filter** (ips_option): rule option to require multiple hits before a rule generates an event
- **dnp3** (inspector): dnp3 inspection
- **dnp3_data** (ips_option): sets the cursor to dnp3 data
- **dnp3_func** (ips_option): detection option to check DNP3 function code
- **dnp3_ind** (ips_option): detection option to check DNP3 indicator flags
- **dnp3_obj** (ips_option): detection option to check DNP3 object headers
- **dns** (inspector): dns inspection
- **domain_filter** (inspector): alert on configured HTTP domains
- **dpx** (inspector): dynamic inspector example
- **dsize** (ips_option): rule option to test payload size
- **eapol** (codec): support for extensible authentication protocol over LAN
- **enable** (ips_option): stub rule option to enable or disable full rule
- **enip_command** (ips_option): detection option to match CIP Enip Command
- **enip_req** (ips_option): detection option to match ENIP Request
- **enip_rsp** (ips_option): detection option to match ENIP response
- **erspan2** (codec): support for encapsulated remote switched port analyzer - type 2
- **erspan3** (codec): support for encapsulated remote switched port analyzer - type 3
- **esp** (codec): support for encapsulating security payload
- **eth** (codec): support for ethernet protocol (DLT 1) (DLT 51)

- **event_filter** (basic): configure thresholding of events
- **event_queue** (basic): configure event queue parameters
- **fabricpath** (codec): support for fabricpath
- **file_connector** (connector): implement the file based connector
- **file_data** (ips_option): rule option to set detection cursor to file data
- **file_id** (inspector): configure file identification
- **file_log** (inspector): log file event to file.log
- **file_meta** (ips_option): rule option to set file metadata (file type and id)
- **file_policy** (basic): configure file policy
- **file_type** (ips_option): rule option to check file type
- **flags** (ips_option): rule option to test TCP control flags
- **flow** (ips_option): rule option to check session properties
- **flowbits** (ips_option): rule option to set and test arbitrary boolean flags
- **fragbits** (ips_option): rule option to test IP frag flags
- **fragoffset** (ips_option): rule option to test IP frag offset
- **ftp_client** (inspector): FTP client configuration module for use with ftp_server
- **ftp_data** (inspector): FTP data channel handler
- **ftp_server** (inspector): main FTP module; ftp_client should also be configured
- **geneve** (codec): support for Geneve: Generic Network Virtualization Encapsulation
- **gid** (ips_option): rule option specifying rule generator
- **gre** (codec): support for generic routing encapsulation
- **gtp** (codec): support for general-packet-radio-service tunneling protocol
- **gtp_info** (ips_option): rule option to check gtp info element
- **gtp_inspect** (inspector): gtp control channel inspection
- **gtp_type** (ips_option): rule option to check gtp types
- **gtp_version** (ips_option): rule option to check GTP version
- **high_availability** (basic): implement flow tracking high availability
- **host_cache** (basic): global LRU cache of host_tracker data about hosts
- **host_tracker** (basic): configure hosts
- **hosts** (basic): configure hosts
- **http2_inspect** (inspector): HTTP/2 inspector
- **http_client_body** (ips_option): rule option to set the detection cursor to the request body
- **http_cookie** (ips_option): rule option to set the detection cursor to the HTTP cookie
- **http_header** (ips_option): rule option to set the detection cursor to the normalized headers

- **http_header_test** (ips_option): rule option to perform range check on specified header field, check whether it is a number, or check if the field is absent
- **http_inspect** (inspector): HTTP inspector
- **http_max_header_line** (ips_option): rule option to perform range check on longest header line
- **http_max_trailer_line** (ips_option): rule option to perform range check on longest trailer line
- **http_method** (ips_option): rule option to set the detection cursor to the HTTP request method
- **http_num_cookies** (ips_option): rule option to perform range check on number of cookies
- **http_num_headers** (ips_option): rule option to perform range check on number of headers
- **http_num_trailers** (ips_option): rule option to perform range check on number of trailers
- **http_param** (ips_option): rule option to set the detection cursor to the value of the specified HTTP parameter key which may be in the query or body
- **http_raw_body** (ips_option): rule option to set the detection cursor to the unnormalized message body
- **http_raw_cookie** (ips_option): rule option to set the detection cursor to the unnormalized cookie
- **http_raw_header** (ips_option): rule option to set the detection cursor to the unnormalized headers
- **http_raw_request** (ips_option): rule option to set the detection cursor to the unnormalized request line
- **http_raw_status** (ips_option): rule option to set the detection cursor to the unnormalized status line
- **http_raw_trailer** (ips_option): rule option to set the detection cursor to the unnormalized trailers
- **http_raw_uri** (ips_option): rule option to set the detection cursor to the unnormalized URI
- **http_stat_code** (ips_option): rule option to set the detection cursor to the HTTP status code
- **http_stat_msg** (ips_option): rule option to set the detection cursor to the HTTP status message
- **http_trailer** (ips_option): rule option to set the detection cursor to the normalized trailers
- **http_trailer_test** (ips_option): rule option to perform range check on specified trailer field, check whether it is a number, or check if the field is absent
- **http_true_ip** (ips_option): rule option to set the detection cursor to the final client IP address
- **http_uri** (ips_option): rule option to set the detection cursor to the normalized URI buffer
- **http_version** (ips_option): rule option to set the detection cursor to the version buffer
- **http_version_match** (ips_option): rule option to match version to listed values
- **hyperscan** (search_engine): intel hyperscan-based mpse with regex support
- **icmp4** (codec): support for Internet control message protocol v4
- **icmp6** (codec): support for Internet control message protocol v6
- **icmp_id** (ips_option): rule option to check ICMP ID
- **icmp_seq** (ips_option): rule option to check ICMP sequence number
- **icode** (ips_option): rule option to check ICMP code
- **id** (ips_option): rule option to check the IP ID field
- **iec104** (inspector): iec104 inspection
- **iec104_apci_type** (ips_option): rule option to check iec104 apci type

- **iec104_asdu_func** (ips_option): rule option to check iec104 function code
- **igmp** (codec): support for Internet group management protocol
- **imap** (inspector): imap inspection
- **inspection** (basic): configure basic inspection policy parameters
- **ip_proto** (ips_option): rule option to check the IP protocol number
- **ipopts** (ips_option): rule option to check for IP options
- **ips** (basic): configure IPS rule processing
- **ipv4** (codec): support for Internet protocol v4 (DLT 228)
- **ipv6** (codec): support for Internet protocol v6 (DLT 229)
- **isdataat** (ips_option): rule option to check for the presence of payload data
- **itype** (ips_option): rule option to check ICMP type
- **js_data** (ips_option): rule option to set detection cursor to normalized JavaScript data
- **js_norm** (basic): JavaScript normalizer
- **latency** (basic): packet and rule latency monitoring and control
- **llc** (codec): support for logical link control
- **log_codecs** (logger): log protocols in packet by layer
- **log_hext** (logger): output payload suitable for daq hext
- **log_pcap** (logger): log packet in pcap format
- **md5** (ips_option): payload rule option for hash matching
- **mem_test** (inspector): for testing memory management
- **memory** (basic): memory management configuration
- **metadata** (ips_option): rule option for conveying arbitrary comma-separated name, value data within the rule text
- **mms** (inspector): mms inspection
- **mms_data** (ips_option): rule option to set cursor to MMS data
- **mms_func** (ips_option): rule option to check MMS function
- **modbus** (inspector): modbus inspection
- **modbus_data** (ips_option): rule option to set cursor to modbus data
- **modbus_func** (ips_option): rule option to check modbus function code
- **modbus_unit** (ips_option): rule option to check Modbus unit ID
- **mpls** (codec): support for multiprotocol label switching
- **msg** (ips_option): rule option summarizing rule purpose output with events
- **mss** (ips_option): detection for TCP maximum segment size
- **netflow** (inspector): netflow inspection
- **network** (basic): configure basic network parameters
- **normalizer** (inspector): packet scrubbing for inline mode

- **null_trace_logger** (inspector): trace logger with a null printout
- **output** (basic): configure general output parameters
- **packet_capture** (inspector): raw packet dumping facility
- **packet_tracer** (basic): generate debug trace messages for packets
- **packets** (basic): configure basic packet handling
- **payload_injector** (basic): payload injection utility
- **pbb** (codec): support for 802.1ah protocol
- **pcre** (ips_option): rule option for matching payload data with pcre
- **perf_monitor** (inspector): performance monitoring and flow statistics collection
- **pgm** (codec): support for pragmatic general multicast
- **pkt_data** (ips_option): rule option to set the detection cursor to the normalized packet data
- **pkt_num** (ips_option): alert on raw packet number
- **pop** (inspector): pop inspection
- **port_scan** (inspector): detect various ip, icmp, tcp, and udp port or protocol scans
- **pppoe** (codec): support for point-to-point protocol over ethernet
- **priority** (ips_option): rule option for prioritizing events
- **process** (basic): configure basic process setup
- **profiler** (basic): configure profiling of rules and/or modules
- **rate_filter** (basic): configure rate filters (which change rule actions)
- **raw_data** (ips_option): rule option to set the detection cursor to the raw packet data
- **react** (ips_action): send response to client and terminate session
- **reference** (ips_option): rule option to indicate relevant attack identification system
- **references** (basic): define reference systems used in rules
- **regex** (ips_option): rule option for matching payload data with hyperscan regex; uses pcre syntax
- **reject** (ips_action): terminate session with TCP reset or ICMP unreachable
- **rem** (ips_option): rule option to convey an arbitrary comment in the rule body
- **replace** (ips_option): rule option to overwrite payload data; use with "rewrite" action; works for raw packets only
- **reputation** (inspector): reputation inspection
- **rev** (ips_option): rule option to indicate current revision of signature
- **rna** (inspector): Real-time network awareness and OS fingerprinting (experimental)
- **rpc** (ips_option): rule option to check SUNRPC CALL parameters
- **rpc_decode** (inspector): RPC inspector
- **s7commplus** (inspector): s7commplus inspection
- **s7commplus_content** (ips_option): rule option to set cursor to s7commplus content
- **s7commplus_func** (ips_option): rule option to check s7commplus function code

- **s7commplus_opcode** (ips_option): rule option to check s7commplus opcode code
- **sd_pattern** (ips_option): rule option for detecting sensitive data
- **search_engine** (basic): configure fast pattern matcher
- **seq** (ips_option): rule option to check TCP sequence number
- **service** (ips_option): rule option to specify list of services for grouping rules
- **sha256** (ips_option): payload rule option for hash matching
- **sha512** (ips_option): payload rule option for hash matching
- **sid** (ips_option): rule option to indicate signature number
- **side_channel** (basic): implement the side-channel asynchronous messaging subsystem
- **sip** (inspector): sip inspection
- **sip_body** (ips_option): rule option to set the detection cursor to the request body
- **sip_header** (ips_option): rule option to set the detection cursor to the SIP header buffer
- **sip_method** (ips_option): detection option for sip method
- **sip_stat_code** (ips_option): detection option for sip stat code
- **smtp** (inspector): smtp inspection
- **snort** (basic): command line configuration and shell commands
- **so** (ips_option): rule option to call custom eval function
- **so_proxy** (inspector): a proxy inspector to track flow data from SO rules (internal use only)
- **soid** (ips_option): rule option to specify a shared object rule ID
- **ssh** (inspector): ssh inspection
- **ssl** (inspector): ssl inspection
- **ssl_state** (ips_option): detection option for ssl state
- **ssl_version** (ips_option): detection option for ssl version
- **stream** (inspector): common flow tracking
- **stream_file** (inspector): stream inspector for file flow tracking and processing
- **stream_icmp** (inspector): stream inspector for ICMP flow tracking
- **stream_ip** (inspector): stream inspector for IP flow tracking and defragmentation
- **stream_reassemble** (ips_option): detection option for stream reassembly control
- **stream_size** (ips_option): detection option for stream size checking
- **stream_tcp** (inspector): stream inspector for TCP flow tracking and stream normalization and reassembly
- **stream_udp** (inspector): stream inspector for UDP flow tracking
- **stream_user** (inspector): stream inspector for user flow tracking and reassembly
- **suppress** (basic): configure event suppressions
- **tag** (ips_option): rule option to log additional packets
- **target** (ips_option): rule option to indicate target of attack

- **tcp** (codec): support for transmission control protocol
- **tcp_connector** (connector): implement the tcp stream connector
- **telnet** (inspector): telnet inspection and normalization
- **tenant_selector** (policy_selector): configure traffic processing based on tenants
- **token_ring** (codec): support for token ring decoding
- **tos** (ips_option): rule option to check type of service field
- **trace** (basic): configure trace log messages
- **ttl** (ips_option): rule option to check time to live field
- **udp** (codec): support for user datagram protocol
- **unified2** (logger): output event and packet in unified2 format file
- **urg** (ips_option): detection for TCP urgent pointer
- **vba_data** (ips_option): rule option to set the detection cursor to the MS Office Visual Basic for Applications macros buffer
- **vlan** (codec): support for local area network
- **window** (ips_option): rule option to check TCP window field
- **wizard** (inspector): inspector that implements port-independent protocol identification
- **wlan** (codec): support for wireless local area network protocol (DLT 105)
- **wscale** (ips_option): detection for TCP window scale

11.11 Plugin Listing

- **codec::arp**: support for address resolution protocol
- **codec::auth**: support for IP authentication header
- **codec::bad_proto**: bad protocol id
- **codec::ciscometadata**: support for cisco metadata
- **codec::eapol**: support for extensible authentication protocol over LAN
- **codec::erspan2**: support for encapsulated remote switched port analyzer - type 2
- **codec::erspan3**: support for encapsulated remote switched port analyzer - type 3
- **codec::esp**: support for encapsulating security payload
- **codec::eth**: support for ethernet protocol (DLT 1) (DLT 51)
- **codec::fabricpath**: support for fabricpath
- **codec::geneve**: support for Geneve: Generic Network Virtualization Encapsulation
- **codec::gre**: support for generic routing encapsulation
- **codec::gtp**: support for general-packet-radio-service tunneling protocol
- **codec::icmp4**: support for Internet control message protocol v4
- **codec::icmp4_ip**: support for IP in ICMPv4
- **codec::icmp6**: support for Internet control message protocol v6

- **codec::icmp6_ip**: support for IP in ICMPv6
- **codec::igmp**: support for Internet group management protocol
- **codec::ipv4**: support for Internet protocol v4 (DLT 228)
- **codec::ipv6**: support for Internet protocol v6 (DLT 229)
- **codec::ipv6_dst_opts**: support for ipv6 destination options
- **codec::ipv6_frag**: support for IPv6 fragment decoding
- **codec::ipv6_hop_opts**: support for IPv6 hop options
- **codec::ipv6_mobility**: support for mobility
- **codec::ipv6_no_next**: sentinel codec
- **codec::ipv6_routing**: support for IPv6 routing extension
- **codec::linux_sll**: support for Linux SLL (DLT 113)
- **codec::llc**: support for logical link control
- **codec::mpls**: support for multiprotocol label switching
- **codec::null**: support for null encapsulation (DLT 0)
- **codec::pbb**: support for 802.1ah protocol
- **codec::pflog**: support for OpenBSD PF log (DLT 117)
- **codec::pgm**: support for pragmatic general multicast
- **codec::ppp**: support for point-to-point encapsulation (DLT 9)
- **codec::ppp_encap**: support for point-to-point encapsulation
- **codec::pppoe_disc**: support for point-to-point discovery
- **codec::pppoe_sess**: support for point-to-point session
- **codec::raw**: support for raw IP (DLT 12)
- **codec::slip**: support for slip protocol (DLT 8)
- **codec::tcp**: support for transmission control protocol
- **codec::teredo**: support for teredo
- **codec::token_ring**: support for token ring decoding
- **codec::trans_bridge**: support for trans-bridging
- **codec::udp**: support for user datagram protocol
- **codec::user**: support for user sessions (DLT 230)
- **codec::vlan**: support for local area network
- **codec::vxlan**: support for Virtual Extensible LAN
- **codec::wlan**: support for wireless local area network protocol (DLT 105)
- **connector::file_connector**: implement the file based connector
- **connector::tcp_connector**: implement the tcp stream connector
- **inspector::appid**: application and service identification

- **inspector::appid_listener**: log selected published data to appid_listener.log
- **inspector::arp_spoof**: detect ARP attacks and anomalies
- **inspector::back_orifice**: back orifice detection
- **inspector::binder**: configure processing based on CIDRs, ports, services, etc.
- **inspector::cip**: cip inspection
- **inspector::cpeos_test**: for testing CPE OS RNA event generation
- **inspector::data_log**: log selected published data to data.log
- **inspector::dce_http_proxy**: dce over http inspection - client to/from proxy
- **inspector::dce_http_server**: dce over http inspection - proxy to/from server
- **inspector::dce_smb**: dce over smb inspection
- **inspector::dce_tcp**: dce over tcp inspection
- **inspector::dce_udp**: dce over udp inspection
- **inspector::dnp3**: dnp3 inspection
- **inspector::dns**: dns inspection
- **inspector::domain_filter**: alert on configured HTTP domains
- **inspector::dpx**: dynamic inspector example
- **inspector::file_id**: configure file identification
- **inspector::file_log**: log file event to file.log
- **inspector::ftp_client**: FTP inspector client module
- **inspector::ftp_data**: FTP data channel handler
- **inspector::ftp_server**: FTP inspector server module
- **inspector::gtp_inspect**: gtp control channel inspection
- **inspector::http2_inspect**: the HTTP/2 inspector
- **inspector::http_inspect**: the new HTTP inspector!
- **inspector::iec104**: iec104 inspection
- **inspector::imap**: imap inspection
- **inspector::mem_test**: for testing memory management
- **inspector::mms**: mms inspection
- **inspector::modbus**: modbus inspection
- **inspector::netflow**: netflow inspection
- **inspector::normalizer**: packet scrubbing for inline mode
- **inspector::null_trace_logger**: trace logger with a null printout
- **inspector::packet_capture**: raw packet dumping facility
- **inspector::perf_monitor**: performance monitoring and flow statistics collection
- **inspector::pop**: pop inspection

- **inspector::port_scan**: detect various ip, icmp, tcp, and udp port or protocol scans
- **inspector::reputation**: reputation inspection
- **inspector::rna**: Real-time network awareness and OS fingerprinting (experimental)
- **inspector::rpc_decode**: RPC inspector
- **inspector::s7commplus**: s7commplus inspection
- **inspector::sip**: sip inspection
- **inspector::smtp**: smtp inspection
- **inspector::so_proxy**: a proxy inspector to track flow data from SO rules (internal use only)
- **inspector::ssh**: ssh inspection
- **inspector::ssl**: ssl inspection
- **inspector::stream**: common flow tracking
- **inspector::stream_file**: stream inspector for file flow tracking and processing
- **inspector::stream_icmp**: stream inspector for ICMP flow tracking
- **inspector::stream_ip**: stream inspector for IP flow tracking and defragmentation
- **inspector::stream_tcp**: stream inspector for TCP flow tracking and stream normalization and reassembly
- **inspector::stream_udp**: stream inspector for UDP flow tracking
- **inspector::stream_user**: stream inspector for user flow tracking and reassembly
- **inspector::telnet**: telnet inspection and normalization
- **inspector::wizard**: inspector that implements port-independent protocol identification
- **ips_action::alert**: generate alert on the current packet
- **ips_action::block**: block current packet and all the subsequent packets in this flow
- **ips_action::drop**: drop the current packet
- **ips_action::file_id**: file_id file type id
- **ips_action::log**: log the current packet
- **ips_action::pass**: mark the current packet as passed
- **ips_action::react**: send response to client and terminate session
- **ips_action::reject**: terminate session with TCP reset or ICMP unreachable
- **ips_action::rewrite**: overwrite packet contents with the "replace" option content
- **ips_option::ack**: rule option to match on TCP ack numbers
- **ips_option::appids**: detection option for application ids
- **ips_option::base64_data**: set detection cursor to decoded Base64 data
- **ips_option::base64_decode**: rule option to decode base64 data - must be used with base64_data option
- **ips_option::ber_data**: rule option to move to the data for a specified BER element
- **ips_option::ber_skip**: rule option to skip BER element
- **ips_option::bufferlen**: rule option to check length of current buffer

- **ips_option::byte_extract**: rule option to convert data to an integer variable
- **ips_option::byte_jump**: rule option to move the detection cursor
- **ips_option::byte_math**: rule option to perform mathematical operations on extracted value and a specified value or existing variable
- **ips_option::byte_test**: rule option to convert data to integer and compare
- **ips_option::cip_attribute**: detection option to match CIP attribute
- **ips_option::cip_class**: detection option to match CIP class
- **ips_option::cip_conn_path_class**: detection option to match CIP Connection Path Class
- **ips_option::cip_instance**: detection option to match CIP instance
- **ips_option::cip_req**: detection option to match CIP request
- **ips_option::cip_rsp**: detection option to match CIP response
- **ips_option::cip_service**: detection option to match CIP service
- **ips_option::cip_status**: detection option to match CIP response status
- **ips_option::classtype**: general rule option for rule classification
- **ips_option::content**: payload rule option for basic pattern matching
- **ips_option::cvs**: payload rule option for detecting specific attacks
- **ips_option::dce_iface**: detection option to check dcerpc interface
- **ips_option::dce_opnum**: detection option to check dcerpc operation number
- **ips_option::dce_stub_data**: sets the cursor to dcerpc stub data
- **ips_option::detection_filter**: rule option to require multiple hits before a rule generates an event
- **ips_option::dnp3_data**: sets the cursor to dnp3 data
- **ips_option::dnp3_func**: detection option to check DNP3 function code
- **ips_option::dnp3_ind**: detection option to check DNP3 indicator flags
- **ips_option::dnp3_obj**: detection option to check DNP3 object headers
- **ips_option::dsize**: rule option to test payload size
- **ips_option::enable**: stub rule option to enable or disable full rule
- **ips_option::enip_command**: detection option to match CIP Enip Command
- **ips_option::enip_req**: detection option to match ENIP Request
- **ips_option::enip_rsp**: detection option to match ENIP response
- **ips_option::file_data**: rule option to set detection cursor to file data
- **ips_option::file_meta**: rule option to set file metadata (file type and id)
- **ips_option::file_type**: rule option to check file type
- **ips_option::flags**: rule option to test TCP control flags
- **ips_option::flow**: rule option to check session properties
- **ips_option::flowbits**: rule option to set and test arbitrary boolean flags

- **ips_option::fragbits**: rule option to test IP frag flags
- **ips_option::fragoffset**: rule option to test IP frag offset
- **ips_option::gid**: rule option specifying rule generator
- **ips_option::gtp_info**: rule option to check gtp info element
- **ips_option::gtp_type**: rule option to check gtp types
- **ips_option::gtp_version**: rule option to check GTP version
- **ips_option::http_client_body**: rule option to set the detection cursor to the request body
- **ips_option::http_cookie**: rule option to set the detection cursor to the HTTP cookie
- **ips_option::http_header**: rule option to set the detection cursor to the normalized headers
- **ips_option::http_header_test**: rule option to perform range check on specified header field, check whether it is a number, or check if the field is absent
- **ips_option::http_max_header_line**: rule option to perform range check on longest header line
- **ips_option::http_max_trailer_line**: rule option to perform range check on longest trailer line
- **ips_option::http_method**: rule option to set the detection cursor to the HTTP request method
- **ips_option::http_num_cookies**: rule option to perform range check on number of cookies
- **ips_option::http_num_headers**: rule option to perform range check on number of headers
- **ips_option::http_num_trailers**: rule option to perform range check on number of trailers
- **ips_option::http_param**: rule option to set the detection cursor to the value of the specified HTTP parameter key which may be in the query or body
- **ips_option::http_raw_body**: rule option to set the detection cursor to the unnormalized message body
- **ips_option::http_raw_cookie**: rule option to set the detection cursor to the unnormalized cookie
- **ips_option::http_raw_header**: rule option to set the detection cursor to the unnormalized headers
- **ips_option::http_raw_request**: rule option to set the detection cursor to the unnormalized request line
- **ips_option::http_raw_status**: rule option to set the detection cursor to the unnormalized status line
- **ips_option::http_raw_trailer**: rule option to set the detection cursor to the unnormalized trailers
- **ips_option::http_raw_uri**: rule option to set the detection cursor to the unnormalized URI
- **ips_option::http_stat_code**: rule option to set the detection cursor to the HTTP status code
- **ips_option::http_stat_msg**: rule option to set the detection cursor to the HTTP status message
- **ips_option::http_trailer**: rule option to set the detection cursor to the normalized trailers
- **ips_option::http_trailer_test**: rule option to perform range check on specified trailer field, check whether it is a number, or check if the field is absent
- **ips_option::http_true_ip**: rule option to set the detection cursor to the final client IP address
- **ips_option::http_uri**: rule option to set the detection cursor to the normalized URI buffer
- **ips_option::http_version**: rule option to set the detection cursor to the version buffer
- **ips_option::http_version_match**: rule option to match version to listed values
- **ips_option::icmp_id**: rule option to check ICMP ID

- **ips_option::icmp_seq**: rule option to check ICMP sequence number
- **ips_option::icode**: rule option to check ICMP code
- **ips_option::id**: rule option to check the IP ID field
- **ips_option::iec104_apci_type**: rule option to check iec104 apci type
- **ips_option::iec104_asdu_func**: rule option to check iec104 function code
- **ips_option::ip_proto**: rule option to check the IP protocol number
- **ips_option::ipopts**: rule option to check for IP options
- **ips_option::isdataat**: rule option to check for the presence of payload data
- **ips_option::itype**: rule option to check ICMP type
- **ips_option::js_data**: rule option to set detection cursor to normalized JavaScript data
- **ips_option::md5**: payload rule option for hash matching
- **ips_option::metadata**: rule option for conveying arbitrary comma-separated name, value data within the rule text
- **ips_option::mms_data**: rule option to set cursor to MMS data
- **ips_option::mms_func**: rule option to check MMS function
- **ips_option::modbus_data**: rule option to set cursor to modbus data
- **ips_option::modbus_func**: rule option to check modbus function code
- **ips_option::modbus_unit**: rule option to check Modbus unit ID
- **ips_option::msg**: rule option summarizing rule purpose output with events
- **ips_option::mss**: detection for TCP maximum segment size
- **ips_option::pcre**: rule option for matching payload data with pcre
- **ips_option::pkt_data**: rule option to set the detection cursor to the normalized packet data
- **ips_option::pkt_num**: alert on raw packet number
- **ips_option::priority**: rule option for prioritizing events
- **ips_option::raw_data**: rule option to set the detection cursor to the raw packet data
- **ips_option::reference**: rule option to indicate relevant attack identification system
- **ips_option::regex**: rule option for matching payload data with hyperscan regex; uses pcre syntax
- **ips_option::rem**: rule option to convey an arbitrary comment in the rule body
- **ips_option::replace**: rule option to overwrite payload data; use with "rewrite" action; works for raw packets only
- **ips_option::rev**: rule option to indicate current revision of signature
- **ips_option::rpc**: rule option to check SUNRPC CALL parameters
- **ips_option::s7commplus_content**: rule option to set cursor to s7commplus content
- **ips_option::s7commplus_func**: rule option to check s7commplus function code
- **ips_option::s7commplus_opcode**: rule option to check s7commplus opcode code
- **ips_option::sd_pattern**: rule option for detecting sensitive data
- **ips_option::seq**: rule option to check TCP sequence number

- **ips_option::service**: rule option to specify list of services for grouping rules
- **ips_option::sha256**: payload rule option for hash matching
- **ips_option::sha512**: payload rule option for hash matching
- **ips_option::sid**: rule option to indicate signature number
- **ips_option::sip_body**: rule option to set the detection cursor to the request body
- **ips_option::sip_header**: rule option to set the detection cursor to the SIP header buffer
- **ips_option::sip_method**: detection option for sip method
- **ips_option::sip_stat_code**: detection option for sip stat code
- **ips_option::so**: rule option to call custom eval function
- **ips_option::soid**: rule option to specify a shared object rule ID
- **ips_option::ssl_state**: detection option for ssl state
- **ips_option::ssl_version**: detection option for ssl version
- **ips_option::stream_reassemble**: detection option for stream reassembly control
- **ips_option::stream_size**: detection option for stream size checking
- **ips_option::tag**: rule option to log additional packets
- **ips_option::target**: rule option to indicate target of attack
- **ips_option::tos**: rule option to check type of service field
- **ips_option::ttl**: rule option to check time to live field
- **ips_option::urg**: detection for TCP urgent pointer
- **ips_option::vba_data**: rule option to set the detection cursor to the MS Office Visual Basic for Applications macros buffer
- **ips_option::window**: rule option to check TCP window field
- **ips_option::wscale**: detection for TCP window scale
- **logger::alert_csv**: output event in csv format
- **logger::alert_ex**: output gid:sid:rev for alerts
- **logger::alert_fast**: output event with brief text format
- **logger::alert_full**: output event with full packet dump
- **logger::alert_json**: output event in json format
- **logger::alert_syslog**: output event to syslog
- **logger::alert_talos**: output event in Talos alert format
- **logger::alert_unixsock**: output event over unix socket
- **logger::log_codecs**: log protocols in packet by layer
- **logger::log_hext**: output payload suitable for daq hext
- **logger::log_null**: disable logging of packets
- **logger::log_pcap**: log packet in pcap format
- **logger::unified2**: output event and packet in unified2 format file

- **policy_selector::address_space_selector**: configure traffic processing based on address space
- **policy_selector::tenant_selector**: configure traffic processing based on tenants
- **search_engine::ac_bnfa**: Aho-Corasick Binary NFA (low memory, high performance) MPSE
- **search_engine::ac_full**: Aho-Corasick Full (high memory, best performance), implements search_all()
- **search_engine::hyperscan**: intel hyperscan-based mpse with regex support
- **search_engine::lowmem**: Keyword Trie (low memory, moderate performance) MPSE
- **so_rule::3|18758**: SO rule example