Snort 3 Upgrade Manual

REVISION HISTORY			
NUMBER	DATE	DESCRIPTION	NAME
3.1.82.0	2024-03-12 12:51:53 EDT		TST

Contents

1 Overview

Snort 3.0 is an updated version of the Snort Intrusion Prevention System (IPS) which features a new design that provides a superset of Snort 2.X functionality with better efficacy, performance, scalability, usability, and extensibility. Some of the key features of Snort 3.0 are:

- Support multiple packet processing threads
- Use a shared configuration and attribute table
- Autodetect services for portless configuration
- Modular design
- Plugin framework with over 200 plugins
- More scalable memory profile
- LuaJIT configuration, loggers, and rule options
- Hyperscan support
- Rewritten TCP handling
- New rule parser and syntax
- Service rules like alert http
- Rule "sticky" buffers
- Way better SO rules
- New HTTP inspector
- New performance monitor
- New time and space profiling
- New latency monitoring and enforcement
- Inspection Events
- Autogenerate reference documentation

1.1 Efficacy

- Detects and blocks all but 10 HTTP Evader tests (see https://noxxi.de/research/http-evader.html).
- Autodetection of services reduces misses due to incorrect or out of date port configurations and improves detection on unexpected command and control channels.

1.2 Performance

- Vastly improved throughput over Snort 2 for deep flow inspection.
- Many more fast pattern buffers means fewer non-fast pattern rules.
- Hyperscan is used for faster fast patterns, content literals and (optionally) compatible PCRE during signature evaluation, and various searches done by inspectors.
- sd_pattern rules have normal fast patterns, don't require extra searches.
- The snort3_demo repo has a performance suite that can be used to compare Snort 2 and Snort 3.
- The DAQ 3 interface facilitates leveraging your data plane for maximum throughput including checksum offload and acquisition of a vector of packets.
- Snort 3 is not run-to-completion, making it possible to detain packets for lookaside acceleration and other new features.

1.3 Scalability

- Much easier to leverage multiple cores.
- All packet threads share configuration and rule engine which frees up much more memory for packet processing.

1.4 Usability

- Sticky buffers make it easier to write correct rules.
- Autodetection eliminates much port configuration maintenance.
- Service-based detection doesn't preclude port-based detection.
- Builtin defaults and policy tweaks make effective tuning much simpler.
- Command line help and generated reference documentation make it easy to get the correct configuration details.
- A trace mechanism is provided to make it easy to see how packets are processed.
- Build a scaled down image and configure for smaller systems such as IoT devices.
- The snort2lua utility (described later) makes it easy to convert your local rules.
- Extensive peg counts capture the important events and actions to provide more detailed insight into your deployment.

1.5 Extensibility

- Several plugin types are defined and over 220 plugins are available.
- Inspection events make collaboration among plugins possible without framework updates.
- All plugins, including SO rules, can be mixed in the same library.
- A command line shell is available for reloads and other operations.
- The snort3_extra repo has examples to each plugin type to help you get started.
- Easy to add SO rules for 0-day attacks.

2 Snort 3 vs Snort 2

2.1 Features New to Snort 3

Some things Snort++ can do today that Snort can not do:

- regex fast patterns, not just literals
- JSON perf monitor logs
- LuaJIT scriptable rule options and loggers
- pub/sub inspection events (currently used by sip and http_inspect to appid)
- JIT buffer stuffers (notably with new http_inspect)
- C-style comments in rules
- #begin ... #end comment blocks in rules
- rule remarks (comment is part of rule, not just in it)

- process raw files (eg read a PDF and do file processing)
- process raw payload (eg bridge 2 sockets and do inspection)
- fast pattern offload to separate thread (experimental)
- add or override any config item on command line
- set CPU affinity
- pause and resume commands

2.2 Features Improved over Snort 2

Some things Snort++ can do today that Snort can not do as well:

Feature	Snort 2	Snort 3
Packet Threads	1 per process	N per process
Config Memory Use	N processes * M GB	M GB total, more for packets
Config Reload	N processes, slower	1 thread, can be pinned to separate core
Startup	Single threaded, slower	Multithreaded, faster
Plugins	Limited to preprocs and outputs	Full plugin system with over 230 plugins
Plugin Examples	DPX preproc	snort3_extra repo, all types, LuaJIT
DAQ	2.X, run to completion	3.X, vector input, multiple outstanding packets
DAQ Modules	Legacy modules	Stacked modules, IOCTLs, file, socket, text modules
Pcap Readback Speed	X Mbits/sec for Max-Detect	2X with ac, 4X with hyperscan
IP Layers	2 max	Arbitrary / configurable limits
IP Reputation	Complex, shared memory	Simplified process memory
Stream TCP	Complex implementation	New and improved implementation
Service detection	appid only, port configs required	Autodetection, most port configs optional
HTTP inspector	Partly stateful	Fully stateful
HTTP Evader	65 misses	10 misses
Port Scan Detection	High, medium, low thresholds only	Fully configurable detection thresholds
Config parsing	Report one error and quit	Report all errors
Config syntax	Irregular, static, limited variables	Lua, fully scriptable, arbitrary variables
Command Line	Some overlap with config file	Set or override any config from command line
Default Config	Complex, needs tuning	Simplified, effective default config
Policy Examples	None	Tweaks for all standard Talos policies
Policy Bindings	One level	Nested
Rule syntax	Inconsistent, requires line escapes	Uniform syntax, arbitrary whitespace
Rule parsing	Buggy, limited warnings	Robust, many optional warnings
Rule comments	# comments only	#, #begin/#end, C-style, rem option
Rule input	Config includes only	Includes, external file or path, stdin
Hyperscan	External patch available	Native support, regex fast patterns
Sensitive Data	Requires a slow, extra search	Normal fast pattern rule
alert file rules	No	Yes
alert service rules	No	Yes
fast_pattern_only	Hand optimization	Automatic optimization
Fast Pattern Buffers	6 available	14 available
Elided rule headers	No	Yes (omit nets and/or ports)

Rule sticky buffers	Some	All buffers are sticky
SO rule features	Restricted functionality	True superset of text rules
Simple SO rules	No (based on tedious C-structs)	Yes (based on text rule)
Dump builtin stubs	No (SO stubs only)	Yes
Runtime tracing	No (debug traces and misc logs only)	Yes
Event Logging	Legacy unified2 recommended	JSON recommended, integrates with ELK stack, etc.
Statistics	Limited counts, fixed perf monitor	2X counts, configurable perf monitor
Control interface	Limited binary interface	Extensible text shell, socket access
Documentation	LaTeX-based PDF, READMEs	Asciidoc text, html, and PDF (user, ref, dev, upgrade)
Command line help	No	Extensive, supports data driven management
Build system	Automake, Linux, *BSD, Windows	Cmake, Linux, *BSD, Mac
Startup / Shutdown	Noisy, limited	Clean, extensible, non-zero counts only
Sig Quit	Same as Sig Term and Sig Int	Quick exit
Source Code	470K lines C, avg 400 lines/file	389K lines C++, avg 200 lines/file
Tests	A few unit tests	Many unit tests, snort3_demo suite
Distribution	snort.org tarballs, ~6 month updates	github repo, ~2 week updates

2.3 Build Options

- configure --with-lib{pcap,pcre}-* → --with-{pcap,pcre}-*
- control socket, cs_dir, and users were deleted
- POLICY_BY_ID_ONLY code was deleted
- hardened --enable-inline-init-failopen / INLINE_FAILOPEN

2.4 Command Line

- --pause loads config and waits for resume before processing packets
- --require-rule-sid is hardened
- --shell enables interactive Lua shell
- -T is assumed if no input given
- added --help-config prefix to dump all matching settings
- added --script-path
- added -L none|dump|pcap
- added -z <#> and --max-packet-threads <#>
- delete --enable-mpls-multicast, --enable-mpls-overlapping-ip, --max-mpls-labelchain-len, --mpls-payload-type
- deleted --pid-path and --no-interface-pidfile
- deleting command line options which will be available with --lua or some such including: -I, -h, -F, -p, --disable-inline-init-failopen
- hardened -n < 0
- removed --search-method

- replaced "unknown args are bpf" with --bpf
- replaced --dynamic-*-lib[-dir] with --plugin-path (with : separators)
- removed -b, -N, -Z and, --perfmon-file options

2.5 Conf File

- Snort 3 has a default unicode.map
- Snort 3 will not enforce an upper bound on memcaps and the like within 64 bits
- Snort 3 will supply a default *_global config if not specified (Snort 2 would fatal; e.g. http_inspect_server w/o http_inspect_global)
- address list syntax changes: [[and]] must be [[and]] to avoid Lua string parsing errors (unless in quoted string)
- because the Lua conf is live code, we lose file:line locations in app error messages (syntax errors from Lua have file:line)
- changed search-method names for consistency
- delete config include_vlan_in_alerts (not used in code)
- delete config so_rule_memcap (not used in code)
- deleted --disable-attribute-table-reload-thread
- deleted config decode_*_{alerts,drops} (use rules only)
- deleted config dump-dynamic-rules-path
- deleted config ipv6_frag (not actually used)
- deleted config threshold and ips rule threshold (→ event_filter)
- eliminated ac-split; must use ac-full-q split-any-any
- frag3 → defrag, arpspoof → arp_spoof, sfportscan → port_scan, perfmonitor → perf_monitor, bo → back_orifice
- limits like "1234K" are now "limit = 1234, units = *K*"
- lua field names are (lower) case sensitive; snort.conf largely wasn't
- module filenames are not configurable: always <log-dir>/<module-name><suffix> (suffix is determined by module)
- no positional parameters; all name = value
- perf_monitor configuration was simplified
- portscan.detect_ack_scans deleted (exact same as include_midstream)
- removed various run modes - now just one
- frag3 default policy is Linux not bsd
- lowmem* search methods are now in snort_examples
- deleted unused http_inspect stateful mode
- deleted stateless inspection from ftp and telnet
- deleted http and ftp alert options (now strictly rule based)
- preprocessor disabled settings deleted since no longer relevant
- sessions are always created; snort config stateful checks eliminated
- stream5_tcp: prune_log_max deleted; to be replaced with histogram
- stream5_tcp: max_active_responses, min_response_seconds moved to active.max_responses, min_interval
- ips policies support snort variables configured based on type, ips = { variables = { nets = { var1 = expr1, ... }, paths = { var2 = expr2, ... }, ports = { var3 = expr3, ... } } }

2.6 Rules

- all rules must have a sid
- sid == 0 not allowed
- deleted activate / dynamic rules
- deleted metadata engine shared
- deleted metadata: rule-flushing (with PDU flushing rule flushing can cause missed attacks, the opposite of its intent)
- changed metadata:service one[, service two]; to service:one[, two];
- soid is now a non-metadata option
- metadata is now truly metadata with no impact on detection (Snort doesn't care about metadata internal structure / syntax)
- deleted fast_pattern:only; use fast_pattern, nocase (option is not added to detection tree if not required)
- changed fast_pattern:<offset>,<length> to fast_pattern,fast_pattern_offset <offset>,fast_pattern_length <length>
- fast pattern sensitive data with sd_pattern using hyperscan
- hyperscan regex fast patterns with regex:"<regex>", fast_pattern;
- no ; separated content suboptions
- offset, depth, distance, and within must use a space separator not colon (e.g. offset:5; becomes offset 5;)
- content suboptions http_* are now full options
- added sticky buffers: buffer selector options must precede contents and remain in effect until changed
- the following pcre options have been deleted: use sticky buffers instead B, U, P, H, M, C, I, D, K, S, Y
- deleted uricontent option; use sticky buffer uricontent:"foo" -→ http_uri; content:"foo"
- deleted urilen raw and norm; must use http_raw_uri and http_uri instead
- deleted unused http_encode option
- urilen replaced with generic bufferlen which applies to current sticky buffer
- added optional selector to http_header, e.g. http_header:User-Agent;
- the all new http_inspect has new buffers and rule options
- added alert file and alert service rules (service in body not required if there is only one and it is in header; alert service / file rules disable fast pattern searching of raw packets)
- rule option sequence: <stub> soid <hidden>
- arbitrary whitespace and multiline rules w/o \n
- #begin ... #end comments to easily comment out multiple lines
- add rule remarks option with rem:"arbitrary comment"
- nets and/or ports may be omitted from rule headers (matches any)
- parse all rules and output all errors before quitting
- read rules from conf, separate rules file, or stdin
- The symbol =< in a byte test is recognized as a syntax error. The correct symbol is <=.

2.7 Output

- alert_fast includes packet data by default
- all text mode outputs default to stdout
- changed default logging mode to -L none
- deleted layer2resets and flexresp2_*
- deleted log_ascii
- general output guideline: don't print zero counts
- Snort 3 queues decoder and inspector events to the main event queue before ips policy is selected; since some events may not be enabled, the queue needs to be sized larger than with Snort 2 which used an intermediate queue for decoder events.
- deleted the intermediate http and ftp_telnet event queues
- alert_unified2 and log_unified2 have been deleted

2.8 Sensitive Data

The Snort 2.X SDF Preprocessor is gone, replaced by ips option `sd_pattern`. The sd_pattern rule option is synonymous with the sd_pattern option used for gid:138 rules, but has a different syntax. A major difference in syntax is the use of Hyperscan pattern matching library which provides a regex language similar to PCRE.

To facilitate continued performance, sd_pattern rule option is implemented with Hyperscan pattern matching library. The rule option is now also utilized as a "fast pattern" in the Snort engine which provides a significant performance improvement over the separate detection step of earlier implementations.

The preprocessor alert SDF_COMBO_ALERT (139:1) has been removed and has no replacement in Snort 3.X. This is because the rule offered no additional value over gid:138 rules and was difficult to interpret the result of.

For more information, See Features > Sensitive Data Filtering for details.

2.9 Features Not Yet Supported by Snort 3

- Support in http_inspect for Original Client IP is limited to the X-Forwarded-For and True-Client-IP headers in that order. It is not possible to configure additional custom headers to search for Original Client IP.

3 Snort2Lua

One of the major differences between Snort 2 and Snort 3 is the configuration. Snort 2 configuration files are written in Snort-specific syntax while Snort 3 configuration files are written in Lua. Snort2Lua is a program specifically designed to convert valid Snort 2 configuration files into Lua files that Snort 3 can understand.

Snort2Lua reads your legacy Snort conf file(s) and generates Snort 3 Lua and rules files. When running this program, the only mandatory option is to provide Snort2Lua with a Snort 2 configuration file. The default output file file is snort.lua, the default error file will be snort.rej, and the default rule file is the output file (default is snort.lua). When Snort2Lua finishes running, the resulting configuration file can be successfully run as the Snort3.0 configuration file. The sole exception to this rule is when Snort2Lua cannot find an included file. If that occurs, the file will still be included in the output file and you will need to manually adjust or comment the file name. Additionally, if the exit code is not zero, some of the information may not be successfully converted. Check the error file for all of the conversion problems.

Those errors can occur for a multitude of reasons and are not necessarily bad. Snort2Lua expects a valid Snort 2 configuration. Therefore, if the configuration is invalid or has questionable syntax, Snort2Lua may fail to parse the configuration file or create an invalid Snort 3 configuration file.

There are a also few peculiarities of Snort2Lua that may be confusing to a first time user:

- Aside from an initial configuration file (which is specified from the command line or as the file in 'config binding'), every file that is included into Snort 3 must be either a Lua file or a rule file; the file cannot contain both rules and Lua syntax. Therefore, when parsing a file specified with the 'include' command, Snort2Lua will output both a Lua file and a rule file.
- Any line that is a comment in a configuration file will be added in to a comments section at the bottom of the main configuration file.
- Rules that contain unsupported options will be converted to the best of Snort2Lua's capability and then printed as a comment in the rule file.
- Files with a *.rules* suffix are assumed to be Talos 2.X rules files and converted line-by-line. In this case, lines starting with *alert* are converted as usual but lines starting with *# alert* are assumed to be commented out rules which are converted to 3.0 format and remain comments in the output file. All other comments are passed through directly. There is no support for other commented rule actions since these do not appear in Talos rules files.

3.1 Snort2Lua Command Line

By default, Snort2Lua will attempt to parse every 'include' file and every 'binding' file. There is an option to change this functionality.

When specifying a rule file with one of the command line options, Snort2Lua will output all of the converted rules to that specified rule file. This is especially useful when you are only interesting in converting rules since there is no Lua syntax in rule files. There is also an option that tells Snort2Lua to output every rule for a given configuration into a single rule file. Similarly, there is an option pull all of the Lua syntax from every 'include' file into the output file.

There are currently three output modes: default, quiet, and differences. As expected, quiet mode produces a Snort configuration. All errors (aside from Fatal Snort2Lua errors), differences, and comments will omitted from the final output file. Default mode will print everything. That mean you will be able to see exactly what changes have occurred between Snort 2 and Snort 3 in addition to the new syntax, the original file's comments, and all errors that have occurred. Finally, differences mode will not actually output a valid Snort 3 configuration. Instead, you can see the exact options from the input configuration that have changed.

3.1.1 Usage: snort2lua [OPTIONS]... -c <snort_conf> ...

Converts the Snort configuration file specified by the -c or --conf-file options into a Snort++ configuration file

Options:

- **-?** show usage
- **-h** this overview of snort2lua
- **-a** default option. print all data
- **-c <snort_conf>** The Snort <snort_conf> file to convert
- **-d** print the differences, and only the differences, between the Snort and Snort++ configurations to the <out_file>
- **-e <error_file>** output all errors to <error_file>
- **-i** if <snort_conf> file contains any <include_file> or <policy_file> (i.e. *include path/to/conf/other_conf*), do NOT parse those files
- **-m** add a remark to the end of every converted rule
- **-o <out_file>** output the new Snort++ lua configuration to <out_file>
- **-q** quiet mode. Only output valid configuration information to the <out_file>
- **-r <rule_file>** output any converted rule to <rule_file>

- **-s** when parsing <include_file>, write <include_file>'s rules to <rule_file>. Meaningless if *-i* provided
- **-t** when parsing <include_file>, write <include_file>'s information, excluding rules, to <out_file>. Meaningless if *-i* provided
- **-V** Print the current Snort2Lua version
- **--bind-wizard** Add default wizard to bindings
- **--bind-port** Convert port bindings
- **--conf-file** Same as *-c*. A Snort <snort_conf> file which will be converted
- **--dont-parse-includes** Same as *-p*. if <snort_conf> file contains any <include_file> or <policy_file> (i.e. *include path/to/conf/other_conf*), do NOT parse those files
- **--dont-convert-max-sessions** do not convert max_tcp, max_udp, max_icmp, max_ip to max_session
- **--error-file=<error_file>** Same as *-e*. output all errors to <error_file>
- **--help** Same as *-h*. this overview of snort2lua
- **--ips-policy-pattern** Convert config bindings matching this path to ips policy bindings
- **--markup** print help in asciidoc compatible format
- **--output-file=<out_file>** Same as *-o*. output the new Snort++ lua configuration to <out_file>
- **--print-all** Same as *-a*. default option. print all data
- **--print-differences** Same as *-d*. output the differences, and only the differences, between the Snort and Snort++ configurations to the <out_file>
- **--quiet** Same as *-q*. quiet mode. Only output valid configuration information to the <out_file>
- **--remark** same as *-m*. add a remark to the end of every converted rule
- **--rule-file=<rule_file>** Same as *-r*. output any converted rule to <rule_file>
- **--single-conf-file** Same as *-t*. when parsing <include_file>, write <include_file>'s information, excluding rules, to <out_file>
- **--single-rule-file** Same as *-s*. when parsing <include_file>, write <include_file>'s rules to <rule_file>.
- **--version** Same as *-V*. Print the current Snort2Lua version

Required option:

- A Snort configuration file to convert. Set with either *-c* or *--conf-file*

Default values:

- <out_file> = snort.lua
- <rule_file> = <out_file> = snort.lua. Rules are written to the *local_rules* variable in the <out_file>
- <error_file> = snort.rej. This file will not be created in quiet mode.

3.2 Known Problems

- Any Snort 2 'string' which is dependent on a variable will no longer have that variable in the Lua string.
- Snort2Lua currently does not handle variables well. First, that means variables will not always be parsed correctly. Second, sometimes a variables value will be output in the lua file rather than a variable For instance, if Snort2Lua attempted to convert the line *include $RULE_PATH/example.rule*, the output may output *include /etc/rules/example.rule* instead.
- When Snort2Lua parses a 'binding' configuration file, the rules and configuration will automatically be combined into the same file. Also, the new files name will automatically become the old file's name with a .lua extension. There is currently no way to specify or change that files name.
- If a rule's action is a custom ruletype, that rule action will be silently converted to the rule's *type*. No warnings or errors are currently emitted. Additionally, the custom ruletypes outputs will be silently discarded.
- If the original configuration contains a binding that points to another file and the binding file contains an error, Snort2Lua will output the number of rejects for the binding file in addition to the number of rejects in the main file. The two numbers will eventually be combined into one output.
- If the original configuration contains a replace rule with alert action, Snort2Lua won't translate the rule from alert to rewrite action. It will keep the action as alert, which does not actually replace the content in Snort 3. To replace content, the rule action needs to be rewrite, which can be added manually or by tooling.

3.3 Usage

Snort2Lua is included in the Snort 3 distribution. The Snort2Lua source code is located in the tools/snort2lua directory. The program is automatically built and installed.

Translating your configuration

To run Snort2Lua, the only requirement is a file containing Snort 2 syntax. Assuming your configuration file is named snort.conf, run the command

```
snort2lua -c snort.conf
```

Snort2Lua will output a file named snort.lua. Assuming your snort.conf file is a valid Snort 2 configuration file, than the resulting snort.lua file will always be a valid Snort 3 configuration file; any errors that occur are because Snort 3 currently does not support all of the Snort 2 options.

Every keyword from the Snort configuration can be found in the output file. If the option or keyword has changed, then a comment containing both the option or keyword's old name and new name will be present in the output file.

Translating a rule file

Snort2Lua can also accommodate translating individual rule files. Assuming the Snort 2 rule file is named snort.rules and you want the new rule file to be name updated.rules, run the command

```
snort2lua -c snort.rules -r updated.rules
```

Snort2Lua will output a file named updated.rules. That file, updated.rules, will always be a valid Snort 3 rule file. Any rule that contains unsupported options will be a comment in the output file.

Understanding the Output

Although Snort2Lua outputs very little to the console, there are several things that occur when Snort2Lua runs. This is a list of Snort2Lua outputs.

The console. Every line that Snort2Lua is unable to translate from the Snort 2.X format to the Snort 3 format is considered an error. Upon exiting, Snort2Lua will print the number of errors that occurred. Snort2Lua will also print the name of the error file.

The output file. As previously mentioned, Snort2Lua will create a Lua file with valid Snort 3 syntax. The default Lua file is named snort.lua. This file is the equivalent of your main Snort 2 configuration file.

The rule file. By default, all rules will be printed to the Lua file. However, if a rule file is specified on the command line, any rules found in the Snort 2 configuration will be written to the rule file instead

The error file. By default, the error file is snort.rej. It will only be created if errors exist. Every error referenced on the command line can be found in this file. There are two reasons an error can occur.

- The Snort 2 configuration file has invalid syntax. If Snort 2 cannot parse the configuration file, neither can Snort2Lua. In the example below, Snort2Lua could not convert the line *config bad_option*. Since that is not valid Snort 2 syntax, this is a syntax error.
- The Snort 2 configuration file contains preprocessors and rule options that are not supported in Snort 3. If Snort 2 can parse a line that Snort2Lua cannot parse, than Snort 3 does not support something in the line. As Snort 3 begins supporting these preprocessors and rule options, Snort2Lua will also begin translating these lines. One example of such an error is dcerpc2.

Additional .lua and .rules files. Every time Snort2Lua parses the include or binding keyword, the program will attempt to parse the file referenced by the keyword. Snort2Lua will then create one or two new files. The new files will have a .lua or .rules extension appended to the original filename.

4 Configuration Changes

```
change -> attribute_table: 'STREAM_POLICY' ==> 'hosts: tcp_policy'
change -> attribute_table: 'filename <file_name>' ==> 'hosts[]'
change -> config 'addressspace_agnostic'  ==> 'packets.address_space_agnostic'
change -> config 'checksum_mode'  ==> 'network.checksum_eval'
change -> config 'daq_dir'  ==> 'daq.module_dirs'
change -> config 'detection_filter'  ==> 'alerts.detection_filter_memcap'
change -> config 'enable_deep_teredo_inspection'  ==> 'udp.deep_teredo_inspection'
change -> config 'enable_mpls_overlapping_ip'  ==> 'packets.mpls_agnostic'
change -> config 'event_filter'  ==> 'alerts.event_filter_memcap'
change -> config 'max_attribute_hosts'  ==> 'attribute_table.max_hosts'
change -> config 'max_attribute_services_per_host'  ==> 'attribute_table. ↩
    max_services_per_host'
change -> config 'nopcre'  ==> 'detection.pcre_enable'
change -> config 'pkt_count'  ==> 'packets.limit'
change -> config 'rate_filter'  ==> 'alerts.rate_filter_memcap'
change -> config 'react'  ==> 'react.page'
change -> config 'threshold'  ==> 'alerts.event_filter_memcap'
change -> converter: 'gen_id' ==> 'gid'
change -> converter: 'sid_id' ==> 'sid'
change -> csv: 'csv' ==> 'fields'
change -> csv: 'dgmlen' ==> 'pkt_len'
change -> csv: 'dst' ==> 'dst_addr'
change -> csv: 'dstport' ==> 'dst_port'
change -> csv: 'ethdst' ==> 'eth_dst'
change -> csv: 'ethlen' ==> 'eth_len'
change -> csv: 'ethsrc' ==> 'eth_src'
change -> csv: 'ethtype' ==> 'eth_type'
change -> csv: 'icmpcode' ==> 'icmp_code'
change -> csv: 'icmpid' ==> 'icmp_id'
change -> csv: 'icmpseq' ==> 'icmp_seq'
change -> csv: 'icmptype' ==> 'icmp_type'
change -> csv: 'id' ==> 'ip_id'
change -> csv: 'iplen' ==> 'ip_len'
change -> csv: 'sig_generator' ==> 'gid'
change -> csv: 'sig_id' ==> 'sid'
change -> csv: 'sig_rev' ==> 'rev'
change -> csv: 'src' ==> 'src_addr'
change -> csv: 'srcport' ==> 'src_port'
change -> csv: 'tcpack' ==> 'tcp_ack'
```

```
change -> csv: 'tcpflags' ==> 'tcp_flags'
change -> csv: 'tcplen' ==> 'tcp_len'
change -> csv: 'tcpseq' ==> 'tcp_seq'
change -> csv: 'tcpwindow' ==> 'tcp_win'
change -> csv: 'udplength' ==> 'udp_len'
change -> daq: 'config daq:' ==> 'name'
change -> daq_mode: 'config daq_mode:' ==> 'mode'
change -> daq_var: 'config daq_var:' ==> 'variables'
change -> detection: 'ac' ==> 'ac_full'
change -> detection: 'ac-banded' ==> 'ac_full'
change -> detection: 'ac-bnfa' ==> 'ac_bnfa'
change -> detection: 'ac-bnfa-nq' ==> 'ac_bnfa'
change -> detection: 'ac-bnfa-q' ==> 'ac_bnfa'
change -> detection: 'ac-nq' ==> 'ac_full'
change -> detection: 'ac-q' ==> 'ac_full'
change -> detection: 'ac-sparsebands' ==> 'ac_full'
change -> detection: 'ac-split' ==> 'ac_full'
change -> detection: 'ac-split' ==> 'split_any_any'
change -> detection: 'ac-std' ==> 'ac_full'
change -> detection: 'acs' ==> 'ac_full'
change -> detection: 'bleedover-port-limit' ==> 'bleedover_port_limit'
change -> detection: 'debug-print-fast-pattern' ==> 'show_fast_patterns'
change -> detection: 'intel-cpm' ==> 'hyperscan'
change -> detection: 'lowmem-nq' ==> 'lowmem'
change -> detection: 'lowmem-q' ==> 'lowmem'
change -> detection: 'max-pattern-len' ==> 'max_pattern_len'
change -> detection: 'no_stream_inserts' ==> 'detect_raw_tcp'
change -> detection: 'search-method' ==> 'search_method'
change -> detection: 'split-any-any' ==> 'split_any_any = true by default'
change -> detection: 'split-any-any' ==> 'split_any_any'
change -> dnp3: 'ports' ==> 'bindings'
change -> dns: 'ports' ==> 'bindings'
change -> dynamicdetection ==> 'snort.--plugin_path=<path>'
change -> dynamicengine ==> 'snort.--plugin_path=<path>'
change -> dynamicpreprocessor ==> 'snort.--plugin_path=<path>'
change -> dynamicsidechannel ==> 'snort.--plugin_path=<path>'
change -> event_filter: 'gen_id' ==> 'gid'
change -> event_filter: 'sig_id' ==> 'sid'
change -> event_filter: 'threshold' ==> 'event_filter'
change -> file: 'config file: file_block_timeout' ==> 'block_timeout'
change -> file: 'config file: file_capture_block_size' ==> 'capture_block_size'
change -> file: 'config file: file_capture_max' ==> 'capture_max_size'
change -> file: 'config file: file_capture_memcap' ==> 'capture_memcap'
change -> file: 'config file: file_capture_min' ==> 'capture_min_size'
change -> file: 'config file: file_type_depth' ==> 'type_depth'
change -> file: 'config file: signature' ==> 'enable_signature'
change -> file: 'config file: type_id' ==> 'enable_type'
change -> file: 'ver' ==> 'version'
change -> frag3_engine: 'min_fragment_length' ==> 'min_frag_length'
change -> frag3_engine: 'overlap_limit' ==> 'max_overlaps'
change -> frag3_engine: 'policy bsd-right' ==> 'policy = bsd_right'
change -> frag3_engine: 'timeout' ==> 'session_timeout'
change -> ftp_telnet_protocol: 'alt_max_param_len' ==> 'cmd_validity'
change -> ftp_telnet_protocol: 'data_chan' ==> 'ignore_data_chan'
change -> ftp_telnet_protocol: 'ports' ==> 'bindings'
change -> gtp: 'ports' ==> 'bindings'
change -> http_inspect_server: 'bare_byte' ==> 'utf8_bare_byte'
change -> http_inspect_server: 'client_flow_depth' ==> 'request_depth'
change -> http_inspect_server: 'double_decode' ==> 'iis_double_decode'
change -> http_inspect_server: 'http_inspect_server' ==> 'http_inspect'
change -> http_inspect_server: 'iis_backslash' ==> 'backslash_to_slash'
change -> http_inspect_server: 'inspect_gzip' ==> 'unzip'
```

```
change -> http_inspect_server: 'non_rfc_char' ==> 'bad_characters'
change -> http_inspect_server: 'ports' ==> 'bindings'
change -> http_inspect_server: 'u_encode' ==> 'percent_u'
change -> http_inspect_server: 'utf_8' ==> 'utf8'
change -> imap: 'ports' ==> 'bindings'
change -> modbus: 'ports' ==> 'bindings'
change -> na_policy_mode: 'na_policy_mode' ==> 'mode'
change -> paf_max: 'paf_max [0:63780]' ==> 'max_pdu [1460:32768]'
change -> perfmonitor: 'console' ==> 'format = 'text''
change -> perfmonitor: 'console' ==> 'output = 'console''
change -> perfmonitor: 'file' ==> 'format = 'csv''
change -> perfmonitor: 'file' ==> 'output = 'file''
change -> perfmonitor: 'flow-file' ==> 'format = 'csv''
change -> perfmonitor: 'flow-file' ==> 'output = 'file''
change -> perfmonitor: 'flow-ip' ==> 'flow_ip'
change -> perfmonitor: 'flow-ip-file' ==> 'format = 'csv''
change -> perfmonitor: 'flow-ip-file' ==> 'output = 'file''
change -> perfmonitor: 'flow-ip-memcap' ==> 'flow_ip_memcap'
change -> perfmonitor: 'flow-ports' ==> 'flow_ports'
change -> perfmonitor: 'pktcnt' ==> 'packets'
change -> perfmonitor: 'snortfile' ==> 'format = 'csv''
change -> perfmonitor: 'snortfile' ==> 'output = 'file''
change -> perfmonitor: 'time' ==> 'seconds'
change -> policy_mode: 'inline_test' ==> 'inline-test'
change -> pop: 'ports' ==> 'bindings'
change -> ppm: 'fastpath-expensive-packets' ==> 'packet.fastpath'
change -> ppm: 'max-pkt-time' ==> 'packet.max_time'
change -> ppm: 'max-rule-time' ==> 'rule.max_time'
change -> ppm: 'ppm' ==> 'latency'
change -> ppm: 'suspend-expensive-rules' ==> 'rule.suspend'
change -> ppm: 'suspend-timeout' ==> 'max_suspend_time'
change -> ppm: 'threshold' ==> 'rule.suspend_threshold'
change -> preprocessor 'normalize_icmp4' ==> 'normalize.icmp4'
change -> preprocessor 'normalize_icmp6' ==> 'normalize.icmp6'
change -> preprocessor 'normalize_ip6' ==> 'normalize.ip6'
change -> profile: 'print' ==> 'count'
change -> profile: 'sort avg_ticks' ==> 'sort = avg_check'
change -> profile: 'sort total_ticks' ==> 'sort = total_time'
change -> rate_filter: 'gen_id' ==> 'gid'
change -> rate_filter: 'sig_id' ==> 'sid'
change -> reputation: 'shared_mem' ==> 'list_dir'
change -> sfportscan: 'proto' ==> 'protos'
change -> sfportscan: 'scan_type' ==> 'scan_types'
change -> sip: 'max_requestName_len' ==> 'max_request_name_len'
change -> sip: 'ports' ==> 'bindings'
change -> smtp: 'ports' ==> 'bindings'
change -> ssh: 'server_ports' ==> 'bindings'
change -> ssl: 'ports' ==> 'bindings'
change -> stream5_global: 'max_active_responses' ==> 'max_responses'
change -> stream5_global: 'min_response_seconds' ==> 'min_interval'
change -> stream5_global: 'tcp_cache_nominal_timeout' ==> 'idle_timeout'
change -> stream5_global: 'udp_cache_nominal_timeout' ==> 'idle_timeout'
change -> stream5_ha: 'min_session_lifetime' ==> 'min_age'
change -> stream5_ha: 'min_sync_interval' ==> 'min_sync'
change -> stream5_ha: 'stream5_ha' ==> 'high_availability'
change -> stream5_ha: 'use_daq' ==> 'daq_channel'
change -> stream5_ip: 'timeout' ==> 'session_timeout'
change -> stream5_tcp: 'bind_to' ==> 'bindings'
change -> stream5_tcp: 'dont_reassemble_async' ==> 'reassemble_async'
change -> stream5_tcp: 'max_queued_bytes' ==> 'queue_limit.max_bytes'
change -> stream5_tcp: 'max_queued_segs' ==> 'queue_limit.max_segments'
change -> stream5_tcp: 'policy hpux' ==> 'stream_tcp.policy = hpux11'
```

```
change -> stream5_tcp: 'timeout' ==> 'session_timeout'
change -> stream5_udp: 'timeout' ==> 'session_timeout'
change -> suppress: 'gen_id' ==> 'gid'
change -> suppress: 'sig_id' ==> 'sid'
change -> syslog: 'log_alert' ==> 'level = alert'
change -> syslog: 'log_auth' ==> 'facility = auth'
change -> syslog: 'log_authpriv' ==> 'facility = authpriv'
change -> syslog: 'log_cons' ==> 'options = cons'
change -> syslog: 'log_crit' ==> 'level = crit'
change -> syslog: 'log_daemon' ==> 'facility = daemon'
change -> syslog: 'log_debug' ==> 'level = debug'
change -> syslog: 'log_emerg' ==> 'level = emerg'
change -> syslog: 'log_err' ==> 'level = err'
change -> syslog: 'log_info' ==> 'level = info'
change -> syslog: 'log_local0' ==> 'facility = local0'
change -> syslog: 'log_local1' ==> 'facility = local1'
change -> syslog: 'log_local2' ==> 'facility = local2'
change -> syslog: 'log_local3' ==> 'facility = local3'
change -> syslog: 'log_local4' ==> 'facility = local4'
change -> syslog: 'log_local5' ==> 'facility = local5'
change -> syslog: 'log_local6' ==> 'facility = local6'
change -> syslog: 'log_local7' ==> 'facility = local7'
change -> syslog: 'log_ndelay' ==> 'options = ndelay'
change -> syslog: 'log_notice' ==> 'level = notice'
change -> syslog: 'log_perror' ==> 'options = perror'
change -> syslog: 'log_pid' ==> 'options = pid'
change -> syslog: 'log_user' ==> 'facility = user'
change -> syslog: 'log_warning' ==> 'level = warning'
change -> threshold: 'ips_option: threshold' ==> 'event_filter'
change -> unified2: 'alert_unified2' ==> 'unified2'
change -> unified2: 'log_unified2' ==> 'unified2'
change -> unified2: 'unified2' ==> 'unified2'
deleted -> arpspoof: 'unicast'
deleted -> attribute_table: '<FRAG_POLICY>hpux</FRAG_POLICY>'
deleted -> attribute_table: '<FRAG_POLICY>irix</FRAG_POLICY>'
deleted -> attribute_table: '<FRAG_POLICY>old-linux</FRAG_POLICY>'
deleted -> attribute_table: '<FRAG_POLICY>unknown</FRAG_POLICY>'
deleted -> attribute_table: '<STREAM_POLICY>noack</STREAM_POLICY>'
deleted -> attribute_table: '<STREAM_POLICY>unknown</STREAM_POLICY>'
deleted -> config 'cs_dir'
deleted -> config 'decode_data_link'
deleted -> config 'disable_attribute_reload_thread'
deleted -> config 'disable_decode_alerts'
deleted -> config 'disable_decode_drops'
deleted -> config 'disable_inline_init_failopen'
deleted -> config 'disable_ipopt_alerts'
deleted -> config 'disable_ipopt_drops'
deleted -> config 'disable_replace'
deleted -> config 'disable_tcpopt_alerts'
deleted -> config 'disable_tcpopt_drops'
deleted -> config 'disable_tcpopt_experimental_alerts'
deleted -> config 'disable_tcpopt_experimental_drops'
deleted -> config 'disable_tcpopt_obsolete_alerts'
deleted -> config 'disable_tcpopt_obsolete_drops'
deleted -> config 'disable_tcpopt_ttcp_alerts'
deleted -> config 'disable_ttcp_alerts'
deleted -> config 'disable_ttcp_drops'
deleted -> config 'dump_dynamic_rules_path'
deleted -> config 'dynamicoutput'
deleted -> config 'enable_decode_drops'
deleted -> config 'enable_decode_oversized_alerts'
deleted -> config 'enable_decode_oversized_drops'
```

```
deleted -> config 'enable_gtp'
deleted -> config 'enable_ipopt_drops'
deleted -> config 'enable_mpls_multicast'
deleted -> config 'enable_tcpopt_drops'
deleted -> config 'enable_tcpopt_experimental_drops'
deleted -> config 'enable_tcpopt_obsolete_drops'
deleted -> config 'enable_tcpopt_ttcp_drops'
deleted -> config 'enable_ttcp_drops'
deleted -> config 'flexresp2_attempts'
deleted -> config 'flexresp2_interface'
deleted -> config 'flexresp2_memcap'
deleted -> config 'flexresp2_rows'
deleted -> config 'flowbits_size'
deleted -> config 'include_vlan_in_alerts'
deleted -> config 'interface'
deleted -> config 'layer2resets'
deleted -> config 'log_ipv6_extra_data'
deleted -> config 'no_promisc'
deleted -> config 'nolog'
deleted -> config 'protected_content'
deleted -> config 'sfalert_unified2'
deleted -> config 'sflog_unified2'
deleted -> config 'sidechannel'
deleted -> config 'so_rule_memcap'
deleted -> config 'stateful'
deleted -> csv: '<filename> can no longer be specific'
deleted -> csv: 'default'
deleted -> csv: 'trheader'
deleted -> detection: 'mwm'
deleted -> detection: 'search-optimize is always true'
deleted -> dnp3: 'disabled'
deleted -> dnp3: 'memcap'
deleted -> dns: 'enable_experimental_types'
deleted -> dns: 'enable_obsolete_types'
deleted -> dns: 'enable_rdata_overflow'
deleted -> event_trace: 'file'
deleted -> fast: '<filename> can no longer be specific'
deleted -> frag3_engine: 'detect_anomalies'
deleted -> frag3_global: 'disabled'
deleted -> ftp_telnet_protocol: 'detect_anomalies'
deleted -> full: '<filename> can no longer be specific'
deleted -> http_inspect: 'detect_anomalous_servers'
deleted -> http_inspect: 'disabled'
deleted -> http_inspect: 'fast_blocking'
deleted -> http_inspect: 'normalize_random_nulls_in_text'
deleted -> http_inspect: 'proxy_alert'
deleted -> http_inspect_server: 'allow_proxy_use'
deleted -> http_inspect_server: 'enable_cookie'
deleted -> http_inspect_server: 'enable_xff'
deleted -> http_inspect_server: 'extended_ascii_uri'
deleted -> http_inspect_server: 'extended_response_inspection'
deleted -> http_inspect_server: 'iis_unicode_map not allowed in sever'
deleted -> http_inspect_server: 'inspect_uri_only'
deleted -> http_inspect_server: 'log_hostname'
deleted -> http_inspect_server: 'log_uri'
deleted -> http_inspect_server: 'no_alerts'
deleted -> http_inspect_server: 'no_pipeline_req'
deleted -> http_inspect_server: 'non_strict'
deleted -> http_inspect_server: 'normalize_cookies'
deleted -> http_inspect_server: 'normalize_headers'
deleted -> http_inspect_server: 'small_chunk_length'
deleted -> http_inspect_server: 'tab_uri_delimiter'
```

```
deleted -> http_inspect_server: 'unlimited_decompress'
deleted -> imap: 'disabled'
deleted -> imap: 'max_mime_mem'
deleted -> imap: 'memcap'
deleted -> perfmonitor: 'accumulate'
deleted -> perfmonitor: 'atexitonly'
deleted -> perfmonitor: 'atexitonly: base-stats'
deleted -> perfmonitor: 'atexitonly: events-stats'
deleted -> perfmonitor: 'atexitonly: flow-ip-stats'
deleted -> perfmonitor: 'atexitonly: flow-stats'
deleted -> perfmonitor: 'atexitonly: reset'
deleted -> perfmonitor: 'events'
deleted -> perfmonitor: 'max'
deleted -> pop: 'disabled'
deleted -> pop: 'max_mime_mem'
deleted -> pop: 'memcap'
deleted -> ppm: 'debug-pkts'
deleted -> reputation: 'shared_max_instances'
deleted -> reputation: 'shared_refresh'
deleted -> rpc_decode: 'alert_fragments'
deleted -> rpc_decode: 'no_alert_incomplete'
deleted -> rpc_decode: 'no_alert_large_fragments'
deleted -> rpc_decode: 'no_alert_multiple_requests'
deleted -> rule_state: 'action'
deleted -> rule_state: 'enable'
deleted -> sfportscan: 'detect_ack_scans'
deleted -> sfportscan: 'disabled'
deleted -> sfportscan: 'logfile'
deleted -> sfportscan: 'sense_level'
deleted -> sip: 'disabled'
deleted -> sip: 'max_sessions'
deleted -> smtp: 'alert_unknown_cmds'
deleted -> smtp: 'disabled'
deleted -> smtp: 'enable_mime_decoding'
deleted -> smtp: 'inspection_type'
deleted -> smtp: 'max_mime_depth'
deleted -> smtp: 'max_mime_mem'
deleted -> smtp: 'memcap'
deleted -> smtp: 'no_alerts'
deleted -> smtp: 'print_cmds'
deleted -> ssh: 'autodetect'
deleted -> ssh: 'enable_badmsgdir'
deleted -> ssh: 'enable_paysize'
deleted -> ssh: 'enable_protomismatch'
deleted -> ssh: 'enable_recognition'
deleted -> ssh: 'enable_respoverflow'
deleted -> ssh: 'enable_srvoverflow'
deleted -> ssh: 'enable_ssh1crc32'
deleted -> ssl: 'noinspect_encrypted'
deleted -> stream5_global: 'disabled'
deleted -> stream5_global: 'flush_on_alert'
deleted -> stream5_global: 'memcap'
deleted -> stream5_global: 'no_midstream_drop_alerts'
deleted -> stream5_tcp: 'check_session_hijacking'
deleted -> stream5_tcp: 'detect_anomalies'
deleted -> stream5_tcp: 'dont_store_large_packets'
deleted -> stream5_tcp: 'ignore_any_rules'
deleted -> stream5_tcp: 'log_asymmetric_traffic'
deleted -> stream5_tcp: 'policy noack'
deleted -> stream5_tcp: 'policy unknown'
deleted -> stream5_tcp: 'use_static_footprint_sizes'
deleted -> stream5_udp: 'ignore_any_rules'
```

```
deleted -> tcpdump: '<filename> can no longer be specific'
deleted -> test: 'file'
deleted -> test: 'stdout'
deleted -> unified2: 'filename'
deleted -> unified2: 'mpls_event_types'
deleted -> unified2: 'vlan_event_types'
```