File Konfigurasi Suricata

Lokasi file `/etc/suricata/suricata-debian.yaml`

```
%YAML 1.1
---

# Suricata configuration file. In addition to the comments describing all
# options in this file, full documentation can be found at:
#
https://redmine.openinfosecfoundation.org/projects/suricata/wiki/Suricatayam
l



# Number of packets allowed to be processed simultaneously.  Default is a
# conservative 1024. A higher number will make sure CPU's/CPU cores will be
# more easily kept busy, but may negatively impact caching.
#
# If you are using the CUDA pattern matcher (b2g_cuda below), different
rules
# apply. In that case try something like 4000 or more. This is because the
CUDA
# pattern matcher scans many packets in parallel.
#max-pending-packets: 1024

# Runmode the engine should use. Please check --list-runmodes to get the
available
# runmodes for each packet acquisition method. Defaults to "autofp" (auto
flow pinned
# load balancing).
#runmode: autofp

# Specifies the kind of flow load balancer used by the flow pinned autofp
mode.
#
# Supported schedulers are:
#
# round-robin       - Flows assigned to threads in a round robin fashion.
# active-packets    - Flows assigned to threads that have the lowest number
of
#                     unprocessed packets (default).
# hash              - Flow alloted usihng the address hash. More of a random
#                     technique. Was the default in Suricata 1.2.1 and
older.
#
autofp-scheduler: active-packets

# Run suricata as user and group.
#run-as:
```

```
#  user: suri
#  group: suri

# Default pid file.
# Will use this file if no --pidfile in command options.
pid-file: /var/run/suricata.pid

# Daemon working directory
# Suricata will change directory to this one if provided
# Default: "/"
#daemon-directory: "/"

# Preallocated size for packet. Default is 1514 which is the classical
# size for pcap on ethernet. You should adjust this value to the highest
# packet size (MTU + hardware header) on your system.
default-packet-size: 1514

# The default logging directory.  Any log or output file will be
# placed here if its not specified with a full path name.  This can be
# overridden with the -l command line parameter.
default-log-dir: /var/log/suricata/

# Unix command socket can be used to pass commands to suricata.
# An external tool can then connect to get information from suricata
# or trigger some modifications of the engine. Set enabled to yes
# to activate the feature. You can use the filename variable to set
# the file name of the socket.
unix-command:
  enabled: no
  filename: custom.socket

# Configure the type of alert (and other) logging you would like.
outputs:

  # a line based alerts log similar to Snort's fast.log
  - fast:
      enabled: yes
      filename: fast.log
      append: yes
      #filetype: regular # 'regular', 'unix_stream' or 'unix_dgram'

  # alert output for use with Barnyard2
  - unified2-alert:
      enabled: yes
      filename: unified2.alert

      # File size limit.  Can be specified in kb, mb, gb.  Just a number
      # is parsed as bytes.
      #limit: 32mb
```

```
      # Sensor ID field of unified2 alerts.
      #sensor-id: 0

  # a line based log of HTTP requests (no alerts)
  - http-log:
      enabled: yes
      filename: http.log
      append: yes
      #extended: yes     # enable this for extended logging information
      #custom: yes       # enabled the custom logging format (defined by
customformat)
      #customformat: "%{%D-%H:%M:%S}t.%z %{X-Forwarded-For}i %H %m %h %u %s
%B %a:%p -> %A:%P"
      #filetype: regular # 'regular', 'unix_stream' or 'unix_dgram'

  # a line based log of TLS handshake parameters (no alerts)
  - tls-log:
      enabled: no  # Log TLS connections.
      filename: tls.log # File to store TLS logs.
      #extended: yes # Log extended information like fingerprint
      certs-log-dir: certs # directory to store the certificates files

  # a line based log to used with pcap file study.
  # this module is dedicated to offline pcap parsing (empty output
  # if used with another kind of input). It can interoperate with
  # pcap parser like wireshark via the suriwire plugin.
  - pcap-info:
      enabled: yes

  # Packet log... log packets in pcap format. 2 modes of operation: "normal"
  # and "sguil".
  #
  # In normal mode a pcap file "filename" is created in the default-log-dir,
  # or are as specified by "dir". In Sguil mode "dir" indicates the base
directory.
  # In this base dir the pcaps are created in th directory structure Sguil
expects:
  #
  # $sguil-base-dir/YYYY-MM-DD/$filename.<timestamp>
  #
  # By default all packets are logged except:
  # - TCP streams beyond stream.reassembly.depth
  # - encrypted streams after the key exchange
  #
  - pcap-log:
      enabled:  yes
      filename: log.pcap

      # File size limit.  Can be specified in kb, mb, gb.  Just a number
      # is parsed as bytes.
      limit: 10mb
```

```
      # If set to a value will enable ring buffer mode. Will keep Maximum of
"max-files" of size "limit"
      max-files: 2000

      mode: normal # normal or sguil.
      #sguil-base-dir: /nsm_data/
      #ts-format: usec # sec or usec second format (default) is filename.sec
usec is filename.sec.usec
      use-stream-depth: no #If set to "yes" packets seen after reaching
stream inspection depth are ignored. "no" logs all packets

  # a full alerts log containing much information for signature writers
  # or for investigating suspected false positives.
  - alert-debug:
      enabled: no
      filename: alert-debug.log
      append: yes
      filetype: regular # 'regular', 'unix_stream' or 'unix_dgram'

  # alert output to prelude (http://www.prelude-technologies.com/) only
  # available if Suricata has been compiled with --enable-prelude
  - alert-prelude:
      enabled: no
      profile: suricata
      log-packet-content: no
      log-packet-header: yes

  # Stats.log contains data from various counters of the suricata engine.
  # The interval field (in seconds) tells after how long output will be
written
  # on the log file.
  - stats:
      enabled: yes
      filename: stats.log
      interval: 8

  # a line based alerts log similar to fast.log into syslog
  - syslog:
      enabled: yes
      # reported identity to syslog. If ommited the program name (usually
      # suricata) will be used.
      identity: "suricata"
      facility: local5
      level: Warning ## possible levels: Emergency, Alert, Critical,
                     ## Error, Warning, Notice, Info, Debug

  # a line based information for dropped packets in IPS mode
  - drop:
      enabled: no
```

```
      filename: drop.log
      append: yes
      #filetype: regular # 'regular', 'unix_stream' or 'unix_dgram'

  # output module to store extracted files to disk
  #
  # The files are stored to the log-dir in a format "file.<id>" where <id>
is
  # an incrementing number starting at 1. For each file "file.<id>" a meta
  # file "file.<id>.meta" is created.
  #
  # File extraction depends on a lot of things to be fully done:
  # - stream reassembly depth. For optimal results, set this to 0
(unlimited)
  # - http request / response body sizes. Again set to 0 for optimal
results.
  # - rules that contain the "filestore" keyword.
  - file-store:
      enabled: no       # set to yes to enable
      log-dir: files    # directory to store the files
      force-magic: no   # force logging magic on all stored files
      force-md5: no     # force logging of md5 checksums
      #waldo: file.waldo # waldo file to store the file_id across runs

  # output module to log files tracked in a easily parsable json format
  - file-log:
      enabled: no
      filename: files-json.log
      append: yes
      #filetype: regular # 'regular', 'unix_stream' or 'unix_dgram'

      force-magic: no   # force logging magic on all logged files
      force-md5: no     # force logging of md5 checksums

# Magic file. The extension .mgc is added to the value here.
#magic-file: /usr/share/file/magic
magic-file: /usr/share/file/magic

# When running in NFQ inline mode, it is possible to use a simulated
# non-terminal NFQUEUE verdict.
# This permit to do send all needed packet to suricata via this a rule:
#       iptables -I FORWARD -m mark ! --mark $MARK/$MASK -j NFQUEUE
# And below, you can have your standard filtering ruleset. To activate
# this mode, you need to set mode to 'repeat'
# If you want packet to be sent to another queue after an ACCEPT decision
# set mode to 'route' and set next-queue value.
# On linux >= 3.6, you can set the fail-open option to yes to have the
kernel
# accept the packet if suricata is not able to keep pace.
nfq:
#  mode: accept
```

```
#  repeat-mark: 1
#  repeat-mask: 1
#  route-queue: 2
#  fail-open: yes

# af-packet support
# Set threads to > 1 to use PACKET_FANOUT support
af-packet:
  - interface: venet0:0
    # Number of receive threads (>1 will enable experimental flow pinned
    # runmode)
    threads: 1
    # Default clusterid.  AF_PACKET will load balance packets based on flow.
    # All threads/processes that will participate need to have the same
    # clusterid.
    cluster-id: 99
    # Default AF_PACKET cluster type. AF_PACKET can load balance per flow or
per hash.
    # This is only supported for Linux kernel > 3.1
    # possible value are:
    #  * cluster_round_robin: round robin load balancing
    #  * cluster_flow: all packets of a given flow are send to the same
socket
    #  * cluster_cpu: all packets treated in kernel by a CPU are send to the
same socket
    cluster-type: cluster_flow
    # In some fragmentation case, the hash can not be computed. If "defrag"
is set
    # to yes, the kernel will do the needed defragmentation before sending
the packets.
    defrag: yes
    # To use the ring feature of AF_PACKET, set 'use-mmap' to yes
    use-mmap: yes
    # Ring size will be computed with respect to max_pending_packets and
number
    # of threads. You can set manually the ring size in number of packets by
setting
    # the following value. If you are using flow cluster-type and have
really network
    # intensive single-flow you could want to set the ring-size
independantly of the number
    # of threads:
    #ring-size: 2048
    # On busy system, this could help to set it to yes to recover from a
packet drop
    # phase. This will result in some packets (at max a ring flush) being
non treated.
    #use-emergency-flush: yes
    # recv buffer size, increase value could improve performance
    # buffer-size: 32768
```

```
    # Set to yes to disable promiscuous mode
    # disable-promisc: no
    # Choose checksum verification mode for the interface. At the moment
    # of the capture, some packets may be with an invalid checksum due to
    # offloading to the network card of the checksum computation.
    # Possible values are:
    #  - kernel: use indication sent by kernel for each packet (default)
    #  - yes: checksum validation is forced
    #  - no: checksum validation is disabled
    #  - auto: suricata uses a statistical approach to detect when
    #  checksum off-loading is used.
    # Warning: 'checksum-validation' must be set to yes to have any
validation
    #checksum-checks: kernel
    # BPF filter to apply to this interface. The pcap filter syntax apply
here.
    bpf-filter: port 80 or udp
    # You can use the following variables to activate AF_PACKET tap od IPS
mode.
    # If copy-mode is set to ips or tap, the traffic coming to the current
    # interface will be copied to the copy-iface interface. If 'tap' is set,
the
    # copy is complete. If 'ips' is set, the packet matching a 'drop' action
    # will not be copied.
    #copy-mode: ips
    #copy-iface: eth1
  - interface: venet0:0
    threads: 1
    cluster-id: 98
    cluster-type: cluster_flow
    defrag: yes
    # buffer-size: 32768
    # disable-promisc: no
  # Put default values here
  - interface: default
    threads: 2
    use-mmap: yes

# You can specify a threshold config file by setting "threshold-file"
# to the path of the threshold config file:
# threshold-file: /etc/suricata/threshold.config

# The detection engine builds internal groups of signatures. The engine
# allow us to specify the profile to use for them, to manage memory on an
# efficient way keeping a good performance. For the profile keyword you
# can use the words "low", "medium", "high" or "custom". If you use custom
# make sure to define the values at "- custom-values" as your convenience.
# Usually you would prefer medium/high/low.
#
# "sgh mpm-context", indicates how the staging should allot mpm contexts for
# the signature groups.  "single" indicates the use of a single context for
```

```
# all the signature group heads.  "full" indicates a mpm-context for each
# group head.  "auto" lets the engine decide the distribution of contexts
# based on the information the engine gathers on the patterns from each
# group head.
#
# The option inspection-recursion-limit is used to limit the recursive calls
# in the content inspection code.  For certain payload-sig combinations, we
# might end up taking too much time in the content inspection code.
# If the argument specified is 0, the engine uses an internally defined
# default limit.  On not specifying a value, we use no limits on the
recursion.
detect-engine:
  - profile: medium
  - custom-values:
      toclient-src-groups: 2
      toclient-dst-groups: 2
      toclient-sp-groups: 2
      toclient-dp-groups: 3
      toserver-src-groups: 2
      toserver-dst-groups: 4
      toserver-sp-groups: 2
      toserver-dp-groups: 25
  - sgh-mpm-context: auto
  - inspection-recursion-limit: 3000
  # When rule-reload is enabled, sending a USR2 signal to the Suricata
process
  # will trigger a live rule reload. Experimental feature, use with care.
  #- rule-reload: true
  # If set to yes, the loading of signatures will be made after the capture
  # is started. This will limit the downtime in IPS mode.
  #- delayed-detect: yes

# Suricata is multi-threaded. Here the threading can be influenced.
threading:
  # On some cpu's/architectures it is beneficial to tie individual threads
  # to specific CPU's/CPU cores. In this case all threads are tied to CPU0,
  # and each extra CPU/core has one "detect" thread.
  #
  # On Intel Core2 and Nehalem CPU's enabling this will degrade performance.
  #
  set-cpu-affinity: no
  # Tune cpu affinity of suricata threads. Each family of threads can be
bound
  # on specific CPUs.
  cpu-affinity:
    - management-cpu-set:
        cpu: [ 0 ]  # include only these cpus in affinity settings
    - receive-cpu-set:
        cpu: [ 0 ]  # include only these cpus in affinity settings
    - decode-cpu-set:
```

```
        cpu: [ 0, 1 ]
        mode: "balanced"
    - stream-cpu-set:
        cpu: [ "0-1" ]
    - detect-cpu-set:
        cpu: [ "all" ]
        mode: "exclusive" # run detect threads in these cpus
        # Use explicitely 3 threads and don't compute number by using
        # detect-thread-ratio variable:
        # threads: 3
        prio:
          low: [ 0 ]
          medium: [ "1-2" ]
          high: [ 3 ]
          default: "medium"
    - verdict-cpu-set:
        cpu: [ 0 ]
        prio:
          default: "high"
    - reject-cpu-set:
        cpu: [ 0 ]
        prio:
          default: "low"
    - output-cpu-set:
        cpu: [ "all" ]
        prio:
           default: "medium"
  #
  # By default Suricata creates one "detect" thread per available CPU/CPU
core.
  # This setting allows controlling this behaviour. A ratio setting of 2
will
  # create 2 detect threads for each CPU/CPU core. So for a dual core CPU
this
  # will result in 4 detect threads. If values below 1 are used, less
threads
  # are created. So on a dual core CPU a setting of 0.5 results in 1 detect
  # thread being created. Regardless of the setting at a minimum 1 detect
  # thread will always be created.
  #
  detect-thread-ratio: 1.5

# Cuda configuration.
cuda:
  # The "mpm" profile.  On not specifying any of these parameters, the
engine's
  # internal default values are used, which are same as the ones specified
here.
  - mpm:
      # Threshold limit for no of packets buffered to the GPU.  Once we hit
this
```

```
      # limit, we pass the buffer to the gpu.
      packet-buffer-limit: 2400
      # The maximum length for a packet that we would buffer to the gpu.
      # Anything over this is MPM'ed on the CPU.  All entries > 0 are valid.
      # Can be specified in kb, mb, gb.  Just a number indicates it's in
bytes.
      packet-size-limit: 1500
      # No of packet buffers we initialize.  All entries > 0 are valid.
      packet-buffers: 10
      # The timeout limit for batching of packets in secs.  If we don't fill
the
      # buffer within this timeout limit, we pass the currently filled
buffer to the gpu.
      # All entries > 0 are valid.
      batching-timeout: 1
      # Specifies whether to use page-locked memory whereever possible.
Accepted values
      # are "enabled" and "disabled".
      page-locked: enabled
      # The device to use for the mpm.  Currently we don't support load
balancing
      # on multiple gpus.  In case you have multiple devices on your system,
you
      # can specify the device to use, using this conf.  By default we hold
0, to
      # specify the first device cuda sees.  To find out device-id
associated with
      # the card(s) on the system run "suricata --list-cuda-cards".
      device-id: 0
      # No of Cuda streams used for asynchronous processing. All values > 0
are valid.
      # For this option you need a device with Compute Capability > 1.0 and
      # page-locked enabled to have any effect.
      cuda-streams: 2

# Select the multi pattern algorithm you want to run for scan/search the
# in the engine. The supported algorithms are b2g, b2gc, b2gm, b3g,
wumanber,
# ac and ac-gfbs.
#
# The mpm you choose also decides the distribution of mpm contexts for
# signature groups, specified by the conf - "detect-engine.sgh-mpm-context".
# Selecting "ac" as the mpm would require "detect-engine.sgh-mpm-context"
# to be set to "single", because of ac's memory requirements, unless the
# ruleset is small enough to fit in one's memory, in which case one can
# use "full" with "ac".  Rest of the mpms can be run in "full" mode.
#
# There is also a CUDA pattern matcher (only available if Suricata was
# compiled with --enable-cuda: b2g_cuda. Make sure to update your
# max-pending-packets setting above as well if you use b2g_cuda.
```

```
mpm-algo: ac

# The memory settings for hash size of these algorithms can vary from lowest
# (2048) - low (4096) - medium (8192) - high (16384) - higher (32768) - max
# (65536). The bloomfilter sizes of these algorithms can vary from low (512)
-
# medium (1024) - high (2048).
#
# For B2g/B3g algorithms, there is a support for two different scan/search
# algorithms. For B2g the scan algorithms are B2gScan & B2gScanBNDMq, and
# search algorithms are B2gSearch & B2gSearchBNDMq. For B3g scan algorithms
# are B3gScan & B3gScanBNDMq, and search algorithms are B3gSearch &
# B3gSearchBNDMq.
#
# For B2g the different scan/search algorithms and, hash and bloom
# filter size settings. For B3g the different scan/search algorithms and,
hash
# and bloom filter size settings. For wumanber the hash and bloom filter
size
# settings.

pattern-matcher:
  - b2gc:
      search-algo: B2gSearchBNDMq
      hash-size: low
      bf-size: medium
  - b2gm:
      search-algo: B2gSearchBNDMq
      hash-size: low
      bf-size: medium
  - b2g:
      search-algo: B2gSearchBNDMq
      hash-size: low
      bf-size: medium
  - b3g:
      search-algo: B3gSearchBNDMq
      hash-size: low
      bf-size: medium
  - wumanber:
      hash-size: low
      bf-size: medium

# Defrag settings:

defrag:
  memcap: 32mb
  hash-size: 65536
  trackers: 65535 # number of defragmented flows to follow
  max-frags: 65535 # number of fragments to keep (higher than trackers)
  prealloc: yes
  timeout: 60
```

```
# Flow settings:
# By default, the reserved memory (memcap) for flows is 32MB. This is the
limit
# for flow allocation inside the engine. You can change this value to allow
# more memory usage for flows.
# The hash-size determine the size of the hash used to identify flows inside
# the engine, and by default the value is 65536.
# At the startup, the engine can preallocate a number of flows, to get a
better
# performance. The number of flows preallocated is 10000 by default.
# emergency-recovery is the percentage of flows that the engine need to
# prune before unsetting the emergency state. The emergency state is
activated
# when the memcap limit is reached, allowing to create new flows, but
# prunning them with the emergency timeouts (they are defined below).
# If the memcap is reached, the engine will try to prune flows
# with the default timeouts. If it doens't find a flow to prune, it will set
# the emergency bit and it will try again with more agressive timeouts.
# If that doesn't work, then it will try to kill the last time seen flows
# not in use.
# The memcap can be specified in kb, mb, gb.  Just a number indicates it's
# in bytes.

flow:
  memcap: 32mb
  hash-size: 65536
  prealloc: 10000
  emergency-recovery: 30

# Specific timeouts for flows. Here you can specify the timeouts that the
# active flows will wait to transit from the current state to another, on
each
# protocol. The value of "new" determine the seconds to wait after a
hanshake or
# stream startup before the engine free the data of that flow it doesn't
# change the state to established (usually if we don't receive more packets
# of that flow). The value of "established" is the amount of
# seconds that the engine will wait to free the flow if it spend that amount
# without receiving new packets or closing the connection. "closed" is the
# amount of time to wait after a flow is closed (usually zero).
#
# There's an emergency mode that will become active under attack
circumstances,
# making the engine to check flow status faster. This configuration
variables
# use the prefix "emergency-" and work similar as the normal ones.
# Some timeouts doesn't apply to all the protocols, like "closed", for udp
and
# icmp.
```

```
flow-timeouts:

  default:
    new: 30
    established: 300
    closed: 0
    emergency-new: 10
    emergency-established: 100
    emergency-closed: 0
  tcp:
    new: 60
    established: 3600
    closed: 120
    emergency-new: 10
    emergency-established: 300
    emergency-closed: 20
  udp:
    new: 30
    established: 300
    emergency-new: 10
    emergency-established: 100
  icmp:
    new: 30
    established: 300
    emergency-new: 10
    emergency-established: 100

# Stream engine settings. Here the TCP stream tracking and reassembly
# engine is configured.
#
# stream:
#   memcap: 32mb                  # Can be specified in kb, mb, gb.  Just a
#                                 # number indicates it's in bytes.
#   checksum-validation: yes      # To validate the checksum of received
#                                 # packet. If csum validation is specified as
#                                 # "yes", then packet with invalid csum will
not
#                                 # be processed by the engine stream/app
layer.
#                                 # Warning: locally generated trafic can be
#                                 # generated without checksum due to hardware
offload
#                                 # of checksum. You can control the handling
of checksum
#             # on a per-interface basis via the 'checksum-checks'
#             # option
#   max-sessions: 262144          # 256k concurrent sessions
#   prealloc-sessions: 32768      # 32k sessions prealloc'd
#   midstream: false              # don't allow midstream session pickups
#   async-oneside: false          # don't enable async stream handling
#   inline: no                    # stream inline mode
```

```
#
#   reassembly:
#     memcap: 64mb              # Can be specified in kb, mb, gb.  Just a
number
#                               # indicates it's in bytes.
#     depth: 1mb                # Can be specified in kb, mb, gb.  Just a
number
#                               # indicates it's in bytes.
#     toserver-chunk-size: 2560 # inspect raw stream in chunks of at least
#                               # this size.  Can be specified in kb, mb,
#                               # gb.  Just a number indicates it's in
bytes.
#     toclient-chunk-size: 2560 # inspect raw stream in chunks of at least
#                               # this size.  Can be specified in kb, mb,
#                               # gb.  Just a number indicates it's in
bytes.

stream:
  memcap: 32mb
  checksum-validation: yes      # reject wrong csums
  inline: auto                  # auto will use inline mode in IPS mode, yes
or no set it statically
  reassembly:
    memcap: 64mb
    depth: 1mb                  # reassemble 1mb into a stream
    toserver-chunk-size: 2560
    toclient-chunk-size: 2560

# Host table:
#
# Host table is used by tagging and per host thresholding subsystems.
#
host:
  hash-size: 4096
  prealloc: 1000
  memcap: 16777216

# Logging configuration.  This is not about logging IDS alerts, but
# IDS output about what its doing, errors, etc.
logging:

  # The default log level, can be overridden in an output section.
  # Note that debug level logging will only be emitted if Suricata was
  # compiled with the --enable-debug configure option.
  #
  # This value is overriden by the SC_LOG_LEVEL env var.
  default-log-level: info

  # The default output format.  Optional parameter, should default to
  # something reasonable if not provided.  Can be overriden in an
```

```
  # output section.  You can leave this out to get the default.
  #
  # This value is overriden by the SC_LOG_FORMAT env var.
  #default-log-format: "[%i] %t - (%f:%l) <%d> (%n) -- "

  # A regex to filter output.  Can be overridden in an output section.
  # Defaults to empty (no filter).
  #
  # This value is overriden by the SC_LOG_OP_FILTER env var.
  default-output-filter:

  # Define your logging outputs.  If none are defined, or they are all
  # disabled you will get the default - console output.
  outputs:
  - console:
      enabled: yes
  - file:
      enabled: no
      filename: /var/log/suricata.log
  - syslog:
      enabled: no
      facility: local5
      format: "[%i] <%d> -- "

# PF_RING configuration. for use with native PF_RING support
# for more info see http://www.ntop.org/PF_RING.html
pfring:
  - interface: venet0:0
    # Number of receive threads (>1 will enable experimental flow pinned
    # runmode)
    threads: 1

    # Default clusterid.  PF_RING will load balance packets based on flow.
    # All threads/processes that will participate need to have the same
    # clusterid.
    cluster-id: 99

    # Default PF_RING cluster type. PF_RING can load balance per flow or per
hash.
    # This is only supported in versions of PF_RING > 4.1.1.
    cluster-type: cluster_flow
    # bpf filter for this interface
    #bpf-filter: tcp
    # Choose checksum verification mode for the interface. At the moment
    # of the capture, some packets may be with an invalid checksum due to
    # offloading to the network card of the checksum computation.
    # Possible values are:
    #  - rxonly: only compute checksum for packets received by network card.
    #  - yes: checksum validation is forced
    #  - no: checksum validation is disabled
    #  - auto: suricata uses a statistical approach to detect when
```

```
    #  checksum off-loading is used. (default)
    # Warning: 'checksum-validation' must be set to yes to have any
validation
    #checksum-checks: auto
  # Second interface
  - interface: venet0:1
    threads: 3
    cluster-id: 93
    cluster-type: cluster_flow
  # Put default values here
  - interface: default
    #threads: 2

pcap:
  - interface: venet0:0
    # On Linux, pcap will try to use mmaped capture and will use buffer-size
    # as total of memory used by the ring. So set this to something bigger
    # than 1% of your bandwidth.
    #buffer-size: 16777216
    #bpf-filter: "tcp and port 25"
    # Choose checksum verification mode for the interface. At the moment
    # of the capture, some packets may be with an invalid checksum due to
    # offloading to the network card of the checksum computation.
    # Possible values are:
    #  - yes: checksum validation is forced
    #  - no: checksum validation is disabled
    #  - auto: suricata uses a statistical approach to detect when
    #  checksum off-loading is used. (default)
    # Warning: 'checksum-validation' must be set to yes to have any
validation
    #checksum-checks: auto
    # With some accelerator cards using a modified libpcap (like myricom),
you
    # may want to have the same number of capture threads as the number of
capture
    # rings. In this case, set up the threads variable to N to start N
threads
    # listening on the same interface.
    threads: 16
    # set to no to disable promiscuous mode:
    promisc: no
    # set snaplen, if not set it defaults to MTU if MTU can be known
    # via ioctl call and to full capture if not.
    #snaplen: 1518
  # Put default values here
  - interface: default
    #checksum-checks: auto

# For FreeBSD ipfw(8) divert(4) support.
# Please make sure you have ipfw_load="YES" and ipdivert_load="YES"
```

```
# in /etc/loader.conf or kldload'ing the appropriate kernel modules.
# Additionally, you need to have an ipfw rule for the engine to see
# the packets from ipfw.  For Example:
#
#   ipfw add 100 divert 8000 ip from any to any
#
# The 8000 above should be the same number you passed on the command
# line, i.e. -d 8000
#
ipfw:

  # Reinject packets at the specified ipfw rule number.  This config
  # option is the ipfw rule number AT WHICH rule processing continues
  # in the ipfw processing system after the engine has finished
  # inspecting the packet for acceptance.  If no rule number is specified,
  # accepted packets are reinjected at the divert rule which they entered
  # and IPFW rule processing continues.  No check is done to verify
  # this will rule makes sense so care must be taken to avoid loops in ipfw.
  #
  ## The following example tells the engine to reinject packets
  # back into the ipfw firewall AT rule number 5500:
  #
  # ipfw-reinjection-rule-number: 5500

# Set the default rule path here to search for the files.
# if not set, it will look at the current working dir
default-rule-path: /etc/snort/rules
rule-files:
 - botcc.rules
 - ciarmy.rules
 - compromised.rules
 - drop.rules
 - dshield.rules
 - emerging-activex.rules
 - emerging-attack_response.rules
 - emerging-chat.rules
 - emerging-current_events.rules
 - emerging-dns.rules
 - emerging-dos.rules
 - emerging-exploit.rules
 - emerging-ftp.rules
 - emerging-games.rules
 - emerging-icmp_info.rules
 - emerging-icmp.rules
 - emerging-imap.rules
 - emerging-inappropriate.rules
 - emerging-malware.rules
 - emerging-misc.rules
 - emerging-mobile_malware.rules
 - emerging-netbios.rules
 - emerging-p2p.rules
```

```
  - emerging-policy.rules
  - emerging-pop3.rules
  - emerging-rpc.rules
  - emerging-scada.rules
  - emerging-scan.rules
  - emerging-shellcode.rules
  - emerging-smtp.rules
  - emerging-snmp.rules
  - emerging-sql.rules
  - emerging-telnet.rules
  - emerging-tftp.rules
  - emerging-trojan.rules
  - emerging-user_agents.rules
  - emerging-virus.rules
  - emerging-voip.rules
  - emerging-web_client.rules
  - emerging-web_server.rules
  - emerging-web_specific_apps.rules
  - emerging-worm.rules
  - rbn-malvertisers.rules
  - rbn.rules
  - tor.rules
  - decoder-events.rules # available in suricata sources under rules dir
  - stream-events.rules  # available in suricata sources under rules dir
  - http-events.rules    # available in suricata sources under rules dir
  - smtp-events.rules    # available in suricata sources under rules dir

classification-file: /etc/suricata/classification.config
reference-config-file: /etc/suricata/reference.config

# Holds variables that would be used by the engine.
vars:

  # Holds the address group vars that would be passed in a Signature.
  # These would be retrieved during the Signature address parsing stage.
  address-groups:

    HOME_NET: "[192.168.0.0/16,10.0.0.0/8,172.16.0.0/12]"

    EXTERNAL_NET: "!$HOME_NET"

    HTTP_SERVERS: "$HOME_NET"

    SMTP_SERVERS: "$HOME_NET"

    SQL_SERVERS: "$HOME_NET"

    DNS_SERVERS: "$HOME_NET"

    TELNET_SERVERS: "$HOME_NET"
```

```
    AIM_SERVERS: "$EXTERNAL_NET"

    DNP3_SERVER: "$HOME_NET"

    DNP3_CLIENT: "$HOME_NET"

    MODBUS_CLIENT: "$HOME_NET"

    MODBUS_SERVER: "$HOME_NET"

    ENIP_CLIENT: "$HOME_NET"

    ENIP_SERVER: "$HOME_NET"

  # Holds the port group vars that would be passed in a Signature.
  # These would be retrieved during the Signature port parsing stage.
  port-groups:

    HTTP_PORTS: "80"

    SHELLCODE_PORTS: "!80"

    ORACLE_PORTS: 1521

    SSH_PORTS: 22

    DNP3_PORTS: 20000

# Set the order of alerts bassed on actions
# The default order is pass, drop, reject, alert
action-order:
  - pass
  - drop
  - reject
  - alert

# IP Reputation
#reputation-categories-file: /etc/suricata/iprep/categories.txt
#default-reputation-path: /etc/suricata/iprep
#reputation-files:
# - reputation.list

# Host specific policies for defragmentation and TCP stream
# reassembly.  The host OS lookup is done using a radix tree, just
# like a routing table so the most specific entry matches.
host-os-policy:
  # Make the default policy windows.
  windows: [0.0.0.0/0]
  bsd: []
  bsd-right: []
  old-linux: []
```

```
  linux: [10.0.0.0/8, 192.168.1.100,
"8762:2352:6241:7245:E000:0000:0000:0000"]
  old-solaris: []
  solaris: ["::1"]
  hpux10: []
  hpux11: []
  irix: []
  macos: []
  vista: []
  windows2k3: []


# Limit for the maximum number of asn1 frames to decode (default 256)
asn1-max-frames: 256

# When run with the option --engine-analysis, the engine will read each of
# the parameters below, and print reports for each of the enabled sections
# and exit.  The reports are printed to a file in the default log dir
# given by the parameter "default-log-dir", with engine reporting
# subsection below printing reports in its own report file.
engine-analysis:
  # enables printing reports for fast-pattern for every rule.
  rules-fast-pattern: yes
  # enables printing reports for each rule
  rules: yes

#recursion and match limits for PCRE where supported
pcre:
  match-limit: 3500
  match-limit-recursion: 1500

##########################################################################
# Configure libhtp.
#
#
# default-config:           Used when no server-config matches
#   personality:            List of personalities used by default
#   request-body-limit:     Limit reassembly of request body for inspection
#                           by http_client_body & pcre /P option.
#   response-body-limit:    Limit reassembly of response body for inspection
#                           by file_data, http_server_body & pcre /Q option.
#   double-decode-path:     Double decode path section of the URI
#   double-decode-query:    Double decode query section of the URI
#
# server-config:            List of server configurations to use if address
matches
#   address:                List of ip addresses or networks for this block
#   personalitiy:           List of personalities used by this block
#   request-body-limit:     Limit reassembly of request body for inspection
#                           by http_client_body & pcre /P option.
```

```
#   response-body-limit:    Limit reassembly of response body for inspection
#                           by file_data, http_server_body & pcre /Q option.
#   double-decode-path:     Double decode path section of the URI
#   double-decode-query:    Double decode query section of the URI
#
# Currently Available Personalities:
#   Minimal
#   Generic
#   IDS (default)
#   IIS_4_0
#   IIS_5_0
#   IIS_5_1
#   IIS_6_0
#   IIS_7_0
#   IIS_7_5
#   Apache
#   Apache_2_2
##########################################################################
libhtp:

   default-config:
     personality: IDS

     # Can be specified in kb, mb, gb.  Just a number indicates
     # it's in bytes.
     request-body-limit: 3072
     response-body-limit: 3072

     # inspection limits
     request-body-minimal-inspect-size: 32kb
     request-body-inspect-window: 4kb
     response-body-minimal-inspect-size: 32kb
     response-body-inspect-window: 4kb

     # decoding
     double-decode-path: no
     double-decode-query: no

   server-config:

     - apache:
         address: [192.168.1.0/24, 127.0.0.0/8, "::1"]
         personality: Apache_2_2
         # Can be specified in kb, mb, gb.  Just a number indicates
         # it's in bytes.
         request-body-limit: 4096
         response-body-limit: 4096
         double-decode-path: no
         double-decode-query: no

     - iis7:
```

```
        address:
          - 192.168.0.0/24
          - 192.168.10.0/24
        personality: IIS_7_0
        # Can be specified in kb, mb, gb.  Just a number indicates
        # it's in bytes.
        request-body-limit: 4096
        response-body-limit: 4096
        double-decode-path: no
        double-decode-query: no

# Profiling settings. Only effective if Suricata has been built with the
# the --enable-profiling configure flag.
#
profiling:

  # rule profiling
  rules:

    # Profiling can be disabled here, but it will still have a
    # performance impact if compiled in.
    enabled: yes
    filename: rule_perf.log
    append: yes

    # Sort options: ticks, avgticks, checks, matches, maxticks
    sort: avgticks

    # Limit the number of items printed at exit.
    limit: 100

  # packet profiling
  packets:

    # Profiling can be disabled here, but it will still have a
    # performance impact if compiled in.
    enabled: yes
    filename: packet_stats.log
    append: yes

    # per packet csv output
    csv:

      # Output can be disabled here, but it will still have a
      # performance impact if compiled in.
      enabled: no
      filename: packet_stats.csv

  # profiling of locking. Only available when Suricata was built with
  # --enable-profiling-locks.
```

```
  locks:
    enabled: no
    filename: lock_stats.log
    append: yes

# Suricata core dump configuration. Limits the size of the core dump file to
# approximately max-dump. The actual core dump size will be a multiple of
the
# page size. Core dumps that would be larger than max-dump are truncated. On
# Linux, the actual core dump size may be a few pages larger than max-dump.
# Setting max-dump to 0 disables core dumping.
# Setting max-dump to 'unlimited' will give the full core dump file.
# On 32-bit Linux, a max-dump value >= ULONG_MAX may cause the core dump
size
# to be 'unlimited'.

coredump:
  max-dump: unlimited

napatech:
    # The Host Buffer Allowance for all streams
    # (-1 = OFF, 1 - 100 = percentage of the host buffer that can be held
back)
    hba: -1

    # use_all_streams set to "yes" will query the Napatech service for all
configured
    # streams and listen on all of them. When set to "no" the streams config
array
    # will be used.
    use-all-streams: yes

    # The streams to listen on
    streams: [1, 2, 3]
```