NAME

esets_http − ESETS HTTP filter module.

SYNOPSIS

esets_http [**OPTION** ...]

DESCRIPTION

The HTTP filter module is HTTP 1.1 compliant special proxy server used to scan the communication between an http client and server for viruses. It receives HTTP message from the HTTP client (a web browser application or other proxy cache) and forwards it to the HTTP server (a web server application) and vice versa. In case any of the messages contains body it is scanned by the **esets_http** module for viruses. Depending on the scanning result the message body is cleaned and/or the message is forwarded to the client or the message is blocked and the alternative predefined response template is sent to particular connection end-point. Particular handling to the messages is applied in case the transfer of the message is long in time. Please, refer to section **LONG TRANSFER HANDLING** to get detailed information on this topic. The **esets_http** is able to act as both the transparent and non-transparent proxy server depending on the integration of the module into the environment.

SIGNALS AND DAEMON PROCESS STRUCTURE

In order to start **esets_http**, the agent manager process must be enabled using parameter **agent_enabled** in section [http] of the main ESETS configuration file. The manager process is responsible for the agent initialization and also for the maintenance of **num_proc** processes each running pool of **num_thrd** client/server sessions servicing threads. This version of ESETS implements 1 vs N threads per sessions model with asynchronous I/O operations, i.e. each thread is able to maintain multiple sessions (resp. multiple opened client/server connections) at a time. Note that each thread performs sessions until termination, resp. until hard termination of the agent processes.

TERM This signal causes termination of the agent processes, i.e. after **esets_http** manager process obtained this signal it terminates all daughter threads and processes as soon as possible and terminates.

USR1 This signal causes so called hard termination of the agent processes, i.e. after **esets_http** manager process obtained this signal it terminates immediately all daughter threads and processes and terminates.

LONG TRANSFER HANDLING

As concerned in the **DESCRIPTION** section of this manual page, **esets_http** handles every object transferred via HTTP in a way that the object is first transferred from the HTTP server (resp. client) to **esets_http**, second, it is scanned for infiltration and finally, it is transferred to the HTTP client (resp. server). Concerning long message transfers this becomes not very suitable scenario as the user agent's timeout or user's impatience can cause interrupts or even canceling of the transfer. Therefore other methods to process the long message transfer must be implemented.

The **esets_http** implements standard so-called 'deferred scan' method of transfer handling. This means if transfer is too long the **esets_http** starts to send the object transparently to an awaiting HTTP end-point (i.e. client or server). After the last part of the object has arrived to **esets_http**, the object is scanned for infiltrations. If the object has been found as infected the last part of the object (current version of ESET Gateway Security defines last part as last 4KB of object's data) is not sent to the awaiting end-point and the connection with the end-point is dropped. In parallel, the e-mail notification is sent to the Gateway administrator with the relevant information about the dangerous file transfer. The URL of the source object is stored in this case in the **esets_http** cache to block the source transfer if requested again. Note that the URL of the source object is stored into the cache only in case of server to client data transfer.

In this place we would like to point out that the 'deferred scan' technique described above presents potential risk for the computer whose user agent requested the infected file for the first time. The risk persists in that even data transfer of an infected object has been deferred some parts of already transferred data can contain executable danger code. That is why the ESET developed modification of the 'deferred scan' technique called 'intermediate scan'.

The 'intermediate scan' technique has been developed to safeguard 'deferred scan' method. Operation principle of the 'intermediate scan' technique is based on the idea that scanning time of a messages is negligible as compared to its overall process time. Note that this condition is fulfilled in case of long HTTP transfer as significantly higher time is needed to transfer the object than to scan it for infiltrations. This assumption allows us to perform more than only one scan during the transfer.

Once parameter **lt_intermediate_scan_enabled** is enabled in [http] section of main ESETS configuration file the message is scanned for infiltrations during its transfer in some predefined intervals and data scanned are sent to awaiting end-point (i.e. to client or to server). Using this method there is no way to pass any infiltration to the computer whose user agent has requested the infected object as each portion of the data sent is already ensured to be secure.

It has been proved that in the common circumstances (by means the speed of the Gateway local network connection is orderly higher than the speed of the Gateway connection to the Internet) the process time of the long transfer with the 'intermediate scan' technique used is approximately the same as when the standard 'deferred scan' method used.

CONFIGURATION FILE OPTIONS

The main ESETS configuration mechanism is assumed to be that using ESETS main configuration file. Note that most principles of this mechanism are described in the esets.cfg(5) manual page while this section contains only additional information related with the list of configuration file options valid for this particular module. Therefore we recommend user to become familiar with the above mentioned documentation prior reading this section.

ESETS MODULE PRIVATE OPTIONS

agent_enabled = *yes/no*

type: bool

default: no

Enables/disables operation of **esets_http** daemon manager process.

num_proc = *value*

type: integer

default: *value* = 1

Defines number of **esets_http** daemon processes.

num_thrd = *value*

type: integer

default: *value* = 2

Defines number of **esets_http** daemon threads per process.

listen_addr = *addr*

type: string

default: no default

Listen on *addr* for client connections. *LA* can be a host name or IP address. When IP address is set to 0.0.0.0 then **esets_http** listens on all available network interfaces.

listen_port = *value*

type: integer

default: no default

Listen on the specified TCP port number *value*.

parent_addr = *addr*

type: string

default: no default

Use address *addr* of the parent proxy to access internet. This parameter, if defined, causes forwarding of every request received from client to server specified by address *addr* rather than to server specified in request.

parent_port = *value*

type: integer

default: no default

Connect to TCP port number *value* of the parent proxy to access internet. If option **parent_addr** specified, option **parent_port** must be specified.

timeout_client = *value*

type: integer

default: *value* = 30

Timeout for the communication of **esets_http** agent with http client measured in seconds. Zero (0) means no timeout. The time limit starts to be measured since last message sent to the client. After no further message is sent during this period from the client the **esets_http** closes connection with both the client and server.

timeout_server = *value*

type: integer

default: *value* = 30

Timeout for the communication of **esets_http** agent with http server measured in seconds. Zero (0) means no timeout. The time limit starts to be measured since last message sent to the server. After no further message is sent during this period from the server the **esets_http** closes connection with this server.

transfer_delay = *value*

type: integer

default: *value* = 10

Transfer delay (in seconds) introduced to store files to be scanned. Every transfer above the limit defined by *value* is handled as long transfer. To get detailed information on this topic refer to section **LONG TRANSFER HANDLING** of this manual page.

lt_intermediate_scan_enabled = *yes/no*

type: bool

default: yes

Specifies whether the **intermediate scan** technique developed by the ESET will be used to handle long transfer. If not, the **deferred scan** is used.

lt_infected_alert_enabled = *yes/no*

type: bool

default: *lt_infected_alert_enabled* = yes

Specifies whether to enable or disable alert notification sent via e-mail to gateway administrator in case the message transferred during long transfer has been found as infected. Note that the alert message is not sent in case the **intermediate scan** technique is used.

lt_infected_alert_script_file = *path*

type: string

default: *path* = "*/etc/opt/eset/esets/scripts/esets_http_lt_infected_alert_script*" (base Linux installation), "*/usr/local/etc/esets/scripts/esets_http_lt_infected_alert_script*" (BSD installation), "*/etc/opt/esets/scripts/esets_http_lt_infected_alert_script*" (Solaris installation).

Specifies path to the alert notification script used to alert gateway administrator in case the message transferred during long transfer has been found as infected.

cache_url_blocked_size = *value*

type: integer

default: *value* = 10000

The size (measured in entries) of the cache used to store URLs blocked by **esets_http**. An URL is being inserted into the cache in case the the message transferred during long transfer has been found as infected.

cache_url_blocked_eltime = *value*

type: integer

default: *value* = 180

This parameter defines the time interval (measured in minutes) during which the URL blocked is being kept in the **esets_http** cache. The default value corresponds to time period of three hours.

scan_obj_max_size = *size*

type: integer

default: *scan_obj_max_size* = 0

Maximum size in bytes of objects scanned by proxy. Zero (0) means no limit.

scan_obj_max_size_accepted = *yes/no*

type: bool

default: *scan_obj_max_size_accepted* = yes

Specifies whether accept or reject objects with size higher than *scan_obj_max_size*).

partial_transfer_enabled = *yes/no*

type: bool

default: *partial_transfer_enabled* = no

Specifies whether to allow or deny the partial data transfer. If disabled the HTTP header fields Range, If-Range, Accept-Ranges are removed from the message header.

http_template_infected = *template*

type: string

default: *template* = "PREDEFINED"

A template used to be sent to the client in case the AV-scanned object is 'infected'. Please refer to ESETS main configuration file (esets.cfg) to get default of this parameter. The template may contain the following directives:

%log%
This directive is replaced by Anti-Virus logging output concerned with this message.

http_template_notscanned = *template*

type: string

default: *template* = "PREDEFINED"

A template used to be sent to the client in case the AV-scanned object is 'notscanned'. Please refer to ESETS main configuration file (esets.cfg) to get default of this parameter. The template may contain the following directives:

%log%
This directive is replaced by Anti-Virus logging output concerned with this message.

tunnel_ports = *portlist*

type: string

default: *tunnel_ports* = "443"

Specifies list of port numbers allowed when CONNECT method used. The string argument *portlist* is assumed to be of the format "port1:port2:port3:...", where port1, port2, port3 etc. are specified ports.

ESETS MODULE COMMON OPTIONS

To get detailed description of ESETS module common options, please refer to the section **ESETS MODULE COMMON OPTIONS** of the **esets.cfg(5)** manual page.

ESETS SCANNER COMMON OPTIONS

To get detailed description of ESETS scanner common options, please refer to the section **ESETS SCANNER COMMON OPTIONS** of the **esets.cfg(5)** manual page.

USER SPECIFIC CONFIGURATION

The ESETS system implements possibility to define so called user specific configuration, i.e. relevant configuration parameters specific for client's and/or server's IP address can be defined.

As described in section **USER SPECIFIC CONFIGURATION** of esets.cfg(5) manual page the user specific configuration is created when an appropriate special configuration section created within a special configuration file *path* referenced from this agent section (see main ESETS configuration file) by option **user_config =** *path*.

The header name of user specific section must be in general of the following format,

[s_addr/s_mask|c_addr/c_mask]

where 's_addr' is server's network IP address, 's_mask' is server's network mask (represented by plain number, specifying the number of 1's at the left side of the network mask), 'c_addr' is client's network IP address, 'c_mask' is client's network mask (represented by plain number, specifying the number of 1's at the left side of the network mask). Thus the part 's_addr/s_mask' represents server's network address range

for which we would like to define special configuration and part 'c_addr/c_mask' represents client's network address range for which we would like to define special configuration.

Note that it is not mandatory to define both client's and server's parts of the header name. In this case the appropriate part not present within header name will be assumed to be not restricted. The following example shows definition of section header name compound only from the client's address range for which we would like to define special configuration.

```
[|192.168.1.0/24]
```

Please, note that thanks to '|' character present at the beginning of section header name, the main ESETS daemon knows that an appropriate IP address range represents the client's part of the section header name. In case you omit the character '|', the appropriate content of the section header name will be assumed to be its server's part as shown in an example below.

```
[192.168.1.0/24]
```

If user specific configuration defined, it will be used automatically, as this agent automatically provides main ESETS scanning daemon **esets_daemon** with the client and server IP addresses received from the header of transferred packet during the FTP communication.

Once client's and/or server's IP address passed to the daemon, it is compared with section header names found in the special configuration file. The comparison is performed with all section header names consecutively in order as they are written within the file. The configuration appropriate to the first matched section is chosen. If no section header name matches the client's/server's IP address passed to the daemon, the configuration appropriate to the agent section from main ESETS configuration file is chosen. The section header name matching algorithm is as follows:

1. If no client's IP address passed to the daemon or no client's part of the section header name present, the algorithm returns match for this part of section header name. If client's IP address passed to the daemon, it is checked whether it matches network address range represented by client's part of the section header name.
2. Similarly if no server's IP address passed to the daemon or no server's part of the section header name present, the algorithm returns match for this part of section header name. If server's IP address passed to the daemon, it is checked whether it matches network address range represented by server's part of the section header name.

If both comparison steps described above return match the configuration appropriate to the section header name is chosen. On the other hand if at least one of the steps returns no match, an appropriate section is skipped.

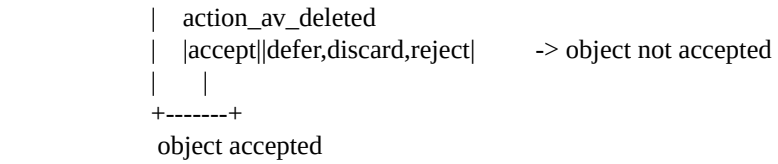
URL WHITELIST

An URL's whitelist is a list of URL addresses whose communication transferred from is not scanned for infiltration. The syntax of URL's whitelist is simply a list of URL addresses (one per line) as shown in the *object* specification of **esets_http** logging output. Note, that the wildcard characters are allowed as well. **esets_http** reads the list from within the file *whitelist_url* in *http* subdirectory of ESETS configuration directory during the initialization stage. To add or remove some URLs one has to restart the daemon.

HANDLE OBJECT POLICY

The Handle Object Policy (see figure below) is a mechanism that provides handling of the scanned objects depending on their scanning status. The mechanism implemented in this module is based on so called action configuration options **action_av**, **action_av_infected**, **action_av_notscanned** and **action_av_deleted**. To get description of these configuration options, see esets.cfg(5) manual page.

```
 action_av
 |accept||scan||defer,discard,reject|    -> object not accepted
   |       |
   |   action_av_infected
   |   action_av_notscanned
```



Every object processed by this module is first handled with respect to the setting of the configuration option **action_av**. Once the option is set to 'accept' (resp. 'defer', 'discard', 'reject') the object is accepted (resp. deferred, discarded, rejected). If the option is set to 'scan' the object is scanned (resp. also cleaned if requested by configuration option **av_clean_mode**) for virus infiltrations and set of action configuration options **action_av_infected**, **action_av_notscanned** and **action_av_deleted** is taken into account to evaluate further handling of the object. If action 'accept' has been taken as a result of the above action options the object is accepted, i.e. the access to the object is allowed. On the other hand if any of action configuration options caused other than 'accept' value, the object is blocked, i.e. access to the object is denied.

You have probably noticed that each of the action configuration options discussed above accepts a variety of the values whose list can be found in esets.cfg(5) manual page. As also stated there the values listed are handled individually by every ESETS agent module. Thus to be consistent in the following we review the meaning of the values for this ESETS agent module.

accept Accept object on this level of Handle Object Policy, i.e. access to the object is allowed by the particular action configuration option.

scan Scan object for virus infiltrations and clean infected objects if requested by configuration option **av_clean_mode**.

defer, discard, reject
Block access to the object, i.e. the access to the object is denied by this particular action configuration option.

LOGGING

Logging functionality of the ESETS agent modules has been developed to fine tune or to troubleshoot the agent module performance. Thus all the ESETS agent modules support only logging using syslogd daemon which logs system messages on *nix systems. To get more information on this topic please, refer to manual pages syslog(2), syslog.conf(5) and syslogd(8).

Regarding ESETS agent modules, this functionality can be invoked by setting ESETS module common configuration option *syslog_facility* to value other than **none**. To get description of the introduced ESETS module common configuration options please, refer to esets.cfg(5) manual page.

Once the syslog logging enabled, the ESETS agent module messages are logged with one of the following syslog priorities:

LOG_ERR
Error messages concerned with the ESETS agent module performance are logged with this priority. Message logged with this priority usually means that error occurred during the ESETS agent module running and thus the module could not accomplish its operation or even the module process exited.

LOG_WARNING
Warning messages concerned with the ESETS agent module performance are logged or 'summary' messages concerned with action other than 'accept' taken as a consequence of object scanning status are logged with this priority.

LOG_NOTICE
The 'summary' messages concerned with action taken as a consequence of object scanning status are logged with this priority.

LOG_INFO
The common tasks are logged with this priority.

LOG_DEBUG

Debug information concerned with the ESETS agent module performance is logged with this priority.

COMMAND LINE OPTIONS

This section contains list of command line options valid for this module.

Note that present version of ESETS implements in general three types of command line options parameters:

Integer parameters accept integer values, e.g.:

--integer-parameter 26

For integer parameter it is also possible to append unit K (1K = 1024), M (1M=1024*1024), G (1=1024*1024*1024), e.g.:

--integer-parameter 4K

Boolean (logical) parameters do not accept any arguments. Their appearance in the command line automatically signal enabling of an appropriate functionality related with the parameter, e.g.:

--bool-parameter

Thus in order to provide negation of an appropriate logical parameter prescribe this parameter with --no, e.g.:

--no-bool-parameter

String parameters accept string values. They can be put into quotation marks, e.g.:

--string-parameter "string value"

Besides options on the command line, there are also commands. All ESETS programs accept at least these commands for displaying the version number and a simple help screen.

-v, --version

Print version information to stdout and exit.

-h, --help

Print help screen to stdout and exit.

REPORTING BUGS

In order to report bugs, please visit http://www.eset.com/support <URL:http://www.eset.com/support> or use directly the support form at http://www.eset.eu/support/form <URL:http://www.eset.eu/support/form>.

COPYRIGHT

SEE ALSO

esets_cgp(1) esets_cli(1) esets_dac(1) esets_ftp(1) esets_gwia(1) esets_http(1) esets_icap(1) esets_imap(1) esets_mda(1) esets_mird(1) libesets_pac.so(1) esets_pipe(1) esets_pop3(1) esets_smfi(1) esets_smtp(1) esets_ssfi.so(1) esets_wwwi(1) esets_zmfi(1) esets(5) esets.cfg(5) esets_daemon(8) esets_inst(8) esets_lic(8) esets_quar(8) esets_scan(8) esets_set(8) esets_update(8) eset_efs_userguide.pdf eset_egs_userguide.pdf eset_ems_userguide.pdf