E2Guardian V5 – Storyboarding

NB – This document and the development of v5 is work in progress and so not all the functionally described in this document is implemented at this time.

Version 5 has a revised model for logic flow and using lists.

Storyboarding is a simple scripting language which defines functions that control list checking, map actions to list matches and defines logic flow. Each filter group can use a different storyboard and so can have different logic if required. The lists used and logic can now also be changed without stopping and re-starting e2guardian.

The Storyboard File

The storyboard file defines functions. Certain 'entry point' functions must be defined as e2guardian will use these as entry points into the function engine. These functions are 'checkrequest' and 'checkresponse' for storyboards included in e2guardianf1.conf and 'pre-authcheck' for storyboard included in e2guardian.conf.

A storyboard file can include other storyboard files, to allow a structured approach to function definitions with common functions being defined in a common included file. Functions can be redefined with the last read version overwriting the previous one.

Blank lines and lines starting with '#' are ignored.

Function Definition

The start of a function is defined with:-

```
function(function_name)
```

function_name is a label made up of alphanumeric, '_' and '-' (no spaces). Do not use labels starting with 'true', 'false', 'set' or 'return' as these may conflict with built-in actions.

The end of a function is defined with:-

```
end()
```

or by the start of a new function

or by an include

or by the end of the file.

A function must be completely defined in a single file and consists of one or more command lines which are executed in order.

Command Line Format

The format of a command line is:-

Command(*Condition*)[**return** || **returnif**]*Action*

where:-

Command is **if** – if *Condition* is true do *Action*

or is **ifnot** - if *Condition* is false do *Action*

Condition format is:-

state[,[*list*][,*message_no*[,*logmessage_no*]]]

where:-

state is as listed in Table 1.

list is list name (mandatory if *state* ends in 'in')

message_no is a message number (overrides messageno in list definition or used when no list) default 0

logmessage_no – (overrides *logmessageno* in list definition or used when no list) default *message_no*

Action is a built-in action (see Table 2) or a function name

if *Action* is prefixed with **return** then return from the current function with the return value of *Action*

if Action is prefixed with **returnif** then return true from the current function if *Action* returns **true**

Table 1a – States which require lists

State	Description	List types checked in this order
urlin	Is url in named list(s)?	siteiplist, sitelist, urllist, regexpboollist
sitein	Is site in named list(s)?	siteiplist, sitelist, regexpboollist
searchin	Is search term in named list?	searchlist
embeddedin	Are any embedded URL in named list(s)?	siteiplist, sitelist, urllist, regexpboollist
refererin	Is referer in named list(s)?	siteiplist, sitelist, urllist
headerin	Is a header in the named list?	regexpboollist, regexpreplacelist
fullurlin	Is full url in named list?	regexpreplacelist
clientin	Is client host in named list?	iplist, sitelist
extensionin	Is file extension in named list?	fileextlist
mimein	Is mime type in named list?	mimelist
Note: lists are only checked if present and when required:- i.e. siteiplist is only checked if site is an IP & urllist is not checked if URL is site-only.		

Table 1b – States without lists

State	Description
connect	Is it a CONNECT request?
blockset	Is block flag set?
doneset	Is done flag set?
exceptionset	Is exception flag set?
get	Is it a GET request?
greyset	Is grey flag set?
mitmset	Are we in a MITM session?

State	Description
post	Is it a POST request?
returnset	Was return from last action true?
siteisip	Is site an IP?
true	Always true

Table 2 – Built-in Actions

Name	Action
false	Return false
setaddheader	Set addheader flag to true and add header – return true if successful
setblock	Set block flag to true – return true
setdone	Set done flag to true – return true
setexception	Set exception flag to true – return true
setgomitm	Set gomitm flag to true – return true
setgrey	Set grey flag to true – return true
setlogcategory	Log category without blocking – return true
setmodurl	Set modurl flag to true and modify URL – return true
setredirect	Set redirect flag to true – return true if successful
true	Return true