

API Documentation

API Documentation

February 28, 2018

Contents

Contents	1
1 Package ethanol	2
1.1 Modules	2
1.2 Variables	5
2 Module ethanol.client_test	6
2.1 Functions	6
3 Package ethanol.ethanol	7
3.1 Modules	7
3.2 Variables	7
4 Module ethanol.ethanol.ap	8
4.1 Functions	8
4.2 Variables	9
4.3 Class AP	9
4.3.1 Methods	10
4.3.2 Properties	12
5 Module ethanol.ethanol.device	13
5.1 Variables	13
5.2 Class Device	13
5.2.1 Methods	13
5.2.2 Properties	17
6 Module ethanol.ethanol.network	18
6.1 Functions	18
6.2 Class Network	19
6.2.1 Methods	19
6.2.2 Properties	20
7 Module ethanol.ethanol.radio	21
7.1 Class Radio	21
7.1.1 Methods	21
7.1.2 Properties	24
8 Module ethanol.ethanol.station	25

8.1	Functions	25
8.2	Variables	25
8.3	Class Station	26
8.3.1	Methods	26
9	Module ethanol.ethanol.switch	28
9.1	Functions	28
9.2	Variables	28
9.3	Class LearningSwitch	28
9.3.1	Methods	28
9.3.2	Properties	28
9.4	Class l2_learning	29
9.4.1	Methods	29
9.4.2	Properties	29
10	Module ethanol.ethanol.vap	30
10.1	Class VAP	30
10.1.1	Methods	30
11	Package ethanol.events	35
11.1	Modules	35
11.2	Class Events	35
11.2.1	Methods	35
11.3	Class EventsException	36
11.3.1	Methods	36
11.3.2	Properties	36
12	Module ethanol.events.events	37
12.1	Variables	37
12.2	Class EventsException	37
12.2.1	Methods	37
12.2.2	Properties	38
12.3	Class Events	38
12.3.1	Methods	38
13	Package ethanol.events.tests	40
13.1	Modules	40
13.2	Variables	40
14	Module ethanol.events.tests.tests	41
14.1	Variables	41
14.2	Class TestBase	41
14.2.1	Methods	41
14.2.2	Properties	42
14.2.3	Class Variables	42
14.3	Class TestEvents	42
14.3.1	Methods	42
14.3.2	Properties	43
14.3.3	Class Variables	43
14.4	Class TestEventSlot	44
14.4.1	Methods	44
14.4.2	Properties	45

14.4.3 Class Variables	45
14.5 Class TestInstanceEvents	45
14.5.1 Methods	45
14.5.2 Properties	46
14.5.3 Class Variables	46
15 Package ethanol.graph_coloring	47
15.1 Modules	47
15.2 Variables	47
16 Module ethanol.graph_coloring.exact_color	48
16.1 Functions	48
17 Package ethanol.ovsdb	49
17.1 Modules	49
17.2 Variables	49
18 Module ethanol.ovsdb.ovsdb	50
18.1 Variables	50
18.2 Class Ovsdb	50
18.2.1 Methods	50
18.2.2 Properties	51
19 Module ethanol.server	52
19.1 Functions	52
19.2 Class ethanol_ap_server	52
19.2.1 Methods	52
19.2.2 Properties	53
20 Package ethanol.ssl_message	54
20.1 Modules	54
20.2 Variables	56
21 Module ethanol.ssl_message.enum	57
21.1 Functions	57
21.2 Variables	57
21.3 Class Enum	57
21.3.1 Methods	57
22 Module ethanol.ssl_message.msg_acs	58
22.1 Functions	58
22.2 Variables	58
23 Module ethanol.ssl_message.msg_ap_broadcastssid	60
23.1 Functions	60
23.2 Variables	61
24 Module ethanol.ssl_message.msg_ap_ctsprotection_enabled	62
24.1 Functions	62
24.2 Variables	63
25 Module ethanol.ssl_message.msg_ap_dtiminterval	64
25.1 Functions	64

25.2 Variables	65
26 Module ethanol.ssl_message.msg_ap_frameburstenabled	66
26.1 Functions	66
26.2 Variables	67
27 Module ethanol.ssl_message.msg_ap_guardinterval	68
27.1 Functions	68
27.2 Variables	69
28 Module ethanol.ssl_message.msg_ap_in_range	70
28.1 Functions	70
28.2 Variables	71
29 Module ethanol.ssl_message.msg_ap_interferencemap	72
29.1 Functions	72
29.2 Variables	72
30 Module ethanol.ssl_message.msg_ap_modes	73
30.1 Functions	73
30.2 Variables	73
31 Module ethanol.ssl_message.msg_ap_rtsthreshold	74
31.1 Functions	74
31.2 Variables	75
32 Module ethanol.ssl_message.msg_ap_ssid	76
32.1 Functions	76
32.2 Variables	76
33 Module ethanol.ssl_message.msg_association	78
33.1 Functions	78
33.2 Variables	78
34 Module ethanol.ssl_message.msg_beacon_interval	80
34.1 Functions	80
34.2 Variables	81
35 Module ethanol.ssl_message.msg_bitrates	82
35.1 Functions	82
35.2 Variables	83
36 Module ethanol.ssl_message.msg_bye	84
36.1 Functions	84
36.2 Variables	84
37 Module ethanol.ssl_message.msg_changed_ap	86
37.1 Functions	86
37.2 Variables	87
38 Module ethanol.ssl_message.msg_channelinfo	88
38.1 Functions	88
38.2 Variables	88

39	Module ethanol.ssl_message.msg_channels	90
39.1	Functions	90
39.2	Variables	91
40	Module ethanol.ssl_message.msg_common	93
40.1	Functions	93
40.2	Variables	94
41	Module ethanol.ssl_message.msg_core	96
41.1	Functions	96
41.2	Variables	97
42	Module ethanol.ssl_message.msg_enabled	98
42.1	Functions	98
42.2	Variables	99
43	Module ethanol.ssl_message.msg_error	100
43.1	Functions	100
43.2	Variables	100
44	Module ethanol.ssl_message.msg_frequency	101
44.1	Functions	101
44.2	Variables	102
45	Module ethanol.ssl_message.msg_handle_snr	103
45.1	Functions	103
45.2	Variables	104
46	Module ethanol.ssl_message.msg_hello	106
46.1	Functions	106
46.2	Variables	106
47	Module ethanol.ssl_message.msg_hostapd_conf	108
47.1	Functions	108
47.2	Variables	109
48	Module ethanol.ssl_message.msg_interfaces	110
48.1	Functions	110
48.2	Variables	111
49	Module ethanol.ssl_message.msg_log	112
49.1	Variables	112
50	Module ethanol.ssl_message.msg_mean_sta_stats	113
50.1	Functions	113
50.2	Variables	115
51	Module ethanol.ssl_message.msg_memcpu	117
51.1	Functions	117
51.2	Variables	118
52	Module ethanol.ssl_message.msg_metric	119
52.1	Functions	119
52.2	Variables	119

53	Module ethanol.ssl_message.msg_mlme	121
53.1	Functions	121
53.2	Variables	122
54	Module ethanol.ssl_message.msg_mtu_qlen	124
54.1	Functions	124
54.2	Variables	124
55	Module ethanol.ssl_message.msg_ping	126
55.1	Functions	126
55.2	Variables	127
56	Module ethanol.ssl_message.msg_powersave	128
56.1	Functions	128
56.2	Variables	129
57	Module ethanol.ssl_message.msg_preamble	130
57.1	Functions	130
57.2	Variables	131
58	Module ethanol.ssl_message.msg_radio_wlans	132
58.1	Functions	132
58.2	Variables	133
59	Module ethanol.ssl_message.msg_sent_received	134
59.1	Functions	135
59.2	Variables	138
60	Module ethanol.ssl_message.msg_server	139
60.1	Functions	139
60.2	Variables	139
61	Module ethanol.ssl_message.msg_snr_power	141
61.1	Functions	141
61.2	Variables	143
62	Module ethanol.ssl_message.msg_ssid	144
62.1	Functions	144
62.2	Variables	144
63	Module ethanol.ssl_message.msg_sta_link_information	146
63.1	Functions	146
63.2	Variables	147
64	Module ethanol.ssl_message.msg_sta_statistics	148
64.1	Functions	148
64.2	Variables	148
65	Module ethanol.ssl_message.msg_station_trigger_transition	150
65.1	Functions	150
65.2	Variables	150
66	Module ethanol.ssl_message.msg_statistics	151
66.1	Functions	151

66.2 Variables	152
67 Module ethanol.ssl_message.msg_tos	153
67.1 Functions	153
67.2 Variables	154
68 Module ethanol.ssl_message.msg_uptime	155
68.1 Functions	155
68.2 Variables	155
69 Module ethanol.ssl_message.msg_wlan_info	156
69.1 Functions	156
69.2 Variables	156
70 Package ethanol.tos	158
70.1 Modules	158
70.2 Variables	158
71 Module ethanol.tos.usecase__tos	159
71.1 Functions	159
71.2 Variables	159
72 Script script-produce__doc__sh	161

1 Package ethanol

This package contains some components to implement Ethanol API.
ethanol should run as a pox module

sample command call:

```
python ./pox.py forwarding.l2_learning ethanol.server
```

ethanol.server is the ~/ethanol/python/server.py file

you must create a symbolic link inside pox subtree, like:

```
cd ~/ethanol/pox/pox
```

```
ln ~/ethanol/python ethanol
```

1.1 Modules

- **client_test**: For TESTING purpose only.
(Section 2, p. 6)
- **ethanol**: This package contains the main classes to implement Ethanol API.
(Section 3, p. 7)
 - **ap**: Defines the AP class.
(Section 4, p. 8)
 - **device**: This module provides: class device.Device
(Section 5, p. 13)
 - **network**: defines the Network class that represents the SSIDs controlled by the Ethanol Controller
(Section 6, p. 18)
 - **radio**: This module provides: class radio.Radio
(Section 7, p. 21)
 - **station** (Section 8, p. 25)
 - **switch**: An L2 learning switch based on L2 learning example from POX
(Section 9, p. 28)
 - **vap**: This module provides: class VAP
(Section 10, p. 30)
- **events** (Section 11, p. 35)
 - **events**: Events ~~~~~
(Section 12, p. 37)
 - **tests** (Section 13, p. 40)
 - * **tests** (Section 14, p. 41)
- **graph_coloring**: This package contains some extra components.
(Section 15, p. 47)
 - **exact_color**: Graph coloring
(Section 16, p. 48)
- **ovsdb**: This package contains a python ovsdb client
(Section 17, p. 49)
 - **ovsdb**: OVSDb calls.
(Section 18, p. 50)
- **server**: This is a pox module.
(Section 19, p. 52)
- **ssl_message**: This package contains some components to implement Ethanol API.

(Section 20, p. 54)

- **enum** (Section 21, p. 57)
- **msg_acs**: implements the following messages:
(Section 22, p. 58)
- **msg_ap_broadcastssid**: implements the following messages:
(Section 23, p. 60)
- **msg_ap_ctsprotection_enabled**: implements the following messages:
(Section 24, p. 62)
- **msg_ap_dtiminterval**: implements the following messages:
(Section 25, p. 64)
- **msg_ap_frameburstenabled**: implements the following messages:
(Section 26, p. 66)
- **msg_ap_guardinterval**: implements the following messages:
(Section 27, p. 68)
- **msg_ap_in_range**: implements the following messages:
(Section 28, p. 70)
- **msg_ap_interferencemap**: implements the following messages:
(Section 29, p. 72)
- **msg_ap_modes**: implements the following messages:
(Section 30, p. 73)
- **msg_ap_rtsthreshold**: implements the following messages:
(Section 31, p. 74)
- **msg_ap_ssid**: implements: * get_ap_ssids
(Section 32, p. 76)
- **msg_association**: implements:
(Section 33, p. 78)
- **msg_beacon_interval**: handles the beacon interval information: gets or sets it.
(Section 34, p. 80)
- **msg_bitrates**: implements the following messages:
(Section 35, p. 82)
- **msg_bye**: implements the BYE message
(Section 36, p. 84)
- **msg_changed_ap**: implements the following messages:
(Section 37, p. 86)
- **msg_channelinfo**: implements the following messages:
(Section 38, p. 88)
- **msg_channels**: implements the following messages:
(Section 39, p. 90)
- **msg_common**: this module contains important constants use through out our implementation
(Section 40, p. 93)
- **msg_core**: All ssl_modules use python construct (<https://pypi.python.org/pypi/construct>).
(Section 41, p. 96)
- **msg_enabled**: implements the following messages:
(Section 42, p. 98)
- **msg_error**: error messages
(Section 43, p. 100)
- **msg_frequency**: implements the following messages:
(Section 44, p. 101)
- **msg_handle_snr**: implements:
(Section 45, p. 103)
- **msg_hello**: basic hello message.

- (Section 46, p. 106)
 - **msg_hostapd_conf**: configure hostapd.
(Section 47, p. 108)
 - **msg_interfaces**: implements the following messages:
(Section 48, p. 110)
 - **msg_log**: defines if our modules will use pox.log facility or python log facility
(Section 49, p. 112)
 - **msg_mean_sta_stats**: implements the following messages:
(Section 50, p. 113)
 - **msg_memcpu**: implements the following messages:
(Section 51, p. 117)
 - **msg_metric**: implements:
(Section 52, p. 119)
 - **msg_mlme**: implements the following MLME messages:
(Section 53, p. 121)
 - **msg_mtu_qlen**: implements: * set_txqueuelen * set_mtu
(Section 54, p. 124)
 - **msg_ping**: implements:
(Section 55, p. 126)
 - **msg_powersave**: implements the following messages:
(Section 56, p. 128)
 - **msg_preamble**: implements: * get_preamble * set_preamble
(Section 57, p. 130)
 - **msg_radio_wlans**: implements the following messages:
(Section 58, p. 132)
 - **msg_sent_received**: implements the following messages:
(Section 59, p. 134)
 - **msg_server**: this is creates the server, that deals with clients (aps and stations) messages the messages implemented are mapped in map_msg_to_procedure main entry to this module is: call run(server)
(Section 60, p. 139)
 - **msg_snr_power**: implements the following messages:
(Section 61, p. 141)
 - **msg_ssid**: implements the following messages:
(Section 62, p. 144)
 - **msg_sta_link_information**: implements the following messages:
(Section 63, p. 146)
 - **msg_sta_statistics**: implements the following messages:
(Section 64, p. 148)
 - **msg_station_trigger_transition**: implements the following messages:
(Section 65, p. 150)
 - **msg_statistics**: implements the following messages:
(Section 66, p. 151)
 - **msg_tos**: implements the following messages:
(Section 67, p. 153)
 - **msg_uptime**: implements the following messages:
(Section 68, p. 155)
 - **msg_wlan_info**: implements: * req_wlan_info(): MSG_WLAN_INFO
(Section 69, p. 156)
- **tos**: This package contains some components to implement Ethanol API.
(Section 70, p. 158)

- **usecase_tos**: This is a module that runs inside POX.
(Section 71, p. 159)

1.2 Variables

Name	Description
__package__	Value: None

2 Module `ethanol.client_test`

For TESTING purpose only. Don't use it as a template to your code. This module uses construct (<https://pypi.python.org/pypi/construct>) See more info at `msg_core.py` on how to install it.

We use this module to test ethanol messages. It does not use Ethanol architecture, only its messages. We must import the correct message module, and place its call in `launch()`

This is a pox module. It should be called using `pox.py`.

Command sample:

```
cd pox ./pox.py ethanol.client_test -server_address='thunder' -server_port=22223
```

2.1 Functions

msg_acs (<i>connect</i> , <i>intf_name</i> ='wlan0', <i>num_acs_tests</i> =1)

this is a test function. it runs <code>num_acs_tests</code> times on interface <code>wlan0</code>

launch (<i>server_address</i> ='0.0.0.0', <i>server_port</i> ='22223', <i>num_acs_tests</i> =1, <i>intf_name</i> ='wlan0', <i>mac_sta</i> ='0c:84:dc:d4:7a:73')
--

<code>launch</code> is a default method used by pox to load and run this module

3 Package ethanol.ethanol

This package contains the main classes to implement Ethanol API.

See Also: file Entidades-vxxxx.pdf contains the class diagram for this API

Change Log:

- Entidades-v1.pdf
- Entidades-v2.pdf
- Entidades-v3.pdf

3.1 Modules

- **ap:** Defines the AP class.
(Section 4, p. 8)
- **device:** This module provides: class device.Device
(Section 5, p. 13)
- **network:** defines the Network class that represents the SSIDs controlled by the Ethanol Controller
(Section 6, p. 18)
- **radio:** This module provides: class radio.Radio
(Section 7, p. 21)
- **station** (Section 8, p. 25)
- **switch:** An L2 learning switch based on L2 learning example from POX
(Section 9, p. 28)
- **vap:** This module provides: class VAP
(Section 10, p. 30)

3.2 Variables

Name	Description
__package__	Value: None

4 Module *ethanol.ethanol.ap*

Defines the AP class. It represents the physical access point.

Author: Henrique Duarte Moura

Organization: WINET/DCC/UFMG

Copyright: h3dema (c) 2017

Contact: henriquemoura@hotmail.com

Since: July 2015

Status: in development

4.1 Functions

connected_aps()
use this function to get the dictionary that contains all aps currently connected to Ethanol controller
Return Value list of ap's objects

is_ap_with_ip_connected(<i>ip</i>)
Return Value TRUE if an AP with the ip provided as a parameter is connected
Note: this is the ip of the AP's interface that sends packets to the controller, i.e., normally it is an ethernet interface

get_ap_by_ip(<i>ip</i>)
get the AP object with an IP address (of the connection to the controller)
Parameters ip: a string with the ip address in dotted format
Return Value the AP object that has the provided ip address, or None if it doesn't exist

get_vap_by_mac_address(<i>mac_address</i>)
get a VAP object by its MAC address (BSSID)
Parameters mac_address: MAC address in dotted format of the Virtual AP (SSID)
Return Value a VAP object that matches the mac_address or None if doesn't match

add_ap_openflow(*ip*)

called at ethanol.server when connectionUp occurs. inserts an entry in map_openflow_vs_ethanol_ip with the ip detected in pox.openflow.connection. when a Hello message arrives, AP.__init__() searches this mapping and assigns self to this entry

Parameters

ip: a string with the ip address in dotted format
(*type=*str)

add_ap(*client_address*)

Create (and return) an AP object for the the device represented by the tuple client_address. This function updates a list of these objects.

used by the Hello message's process

Parameters

client_address: tuple with (ip, port) used to make a socket connection to the AP
(*type=tuple or list*)

remove_ap_byIP(*ip*)

removes the ap from the list called by AP.__destroy__() or when the server receives a "bye message" from such AP

Parameters

ip: a string with the ip address in dotted format
(*type=*str)

4.2 Variables

Name	Description
map_openflow_vs_ethanol_ip	provides a mapping from the ap's ip address to the ap object Value: {}

4.3 Class AP

object  ethanol.ethanol.ap.AP

defines the AP class that represents the physical wifi device

4.3.1 Methods

<code>__init__(self, ip, port=SERVER_PORT)</code> <hr/> constructor Parameters ip: socket IP address to connect to the physical AP port: socket port to connect to the physical AP Overrides: object. <code>__init__</code>
<code>id(self)</code> <hr/> AP's unique identifier Return Value AP's <code>uuid.uuid4()</code> value
<code>__del__(self)</code> <hr/> Called when the instance is about to be destroyed. Removes this ap from the mapping
<code>__str__(self)</code> <hr/> string Return Value the ip and port of this device Overrides: object. <code>__str__</code>
<code>radios(self)</code> <hr/> get list of AP's radios Return Value a list of radio objects associated with the AP
<code>msg_id(self)</code> <hr/> helper function: returns the next message id to be sent, and increments the message ID by 1 Return Value id for the new message
<code>vaps(self)</code> <hr/> returns a list of the vaps configured in this AP Return Value list of VAP objects

createvirtualap_and_insert_listvap(*self, ssid, radio, mac_address*)

create the VAP based on ssid, radio, and mac_address inserts the vap in self.__listVAP list

Parameters

ssid: BSSID
(type=str)

radio: object RADIO attached to this AP

mac_address: MAC address in dotted format
(type=str)

Return Value

the vap created

destroyvirtualap(*self, vap*)

remove a VAP: deactivate it (remove SSID)

Parameters

vap: a vap object (SSID connected to this AP)
(type=vap.VAP object)

getsupportedinterfacemodes(*self, intf_name*)

indicates the modes supported

Return Value

a list with the supported modes: AP, Station, Mesh, IBSS

getinterferencemap(*self, intf_name*)

NOT IMPLEMENTED YET returns the interference map as defined in 802.11/2012

listwlan_interfaces(*self*)

wireless interfaces in this AP

Return Value

a list with the names of wireless interfaces in this AP

get_interface_stats(*self*)

get statistics for all interfaces

enable_interface_stats(*self*)

disable_interface_stats(*self*)

statistics_time(*self, new_time*)

Parameters

new_time: set the time of collection in milliseconds. send -1 to disable data collection

statistics_alpha (<i>self</i> , <i>alpha</i>)
--

defines alpha value for EWMA

read_hostapd_conf_param (<i>self</i> , <i>param</i>)

reads the hostapd.conf, finds the param requested, and returns its value
--

write_hostapd_conf_param (<i>self</i> , <i>param</i> , <i>value</i>)

reads the hostapd.conf, finds the param requested, and (over)write value to its contents
--

Inherited from object

__delattr__(), __format__(), __getattr__(), __hash__(), __new__(),
 __reduce__(), __reduce_ex__(), __repr__(), __setattr__(), __sizeof__(),
 __subclasshook__()

4.3.2 Properties

Name	Description
<i>Inherited from object</i>	
__class__	

5 Module ethanol.ethanol.device

This module provides: class device.Device

It is a superclass for Station and VAP

Author: Henrique Duarte Moura

Organization: WINET/DCC/UFGM

Copyright: h3dema (c) 2017

Contact: henriquemoura@hotmail.com

Since: July 2015

Status: in development

5.1 Variables

Name	Description
METRIC_TO_SUBSCRIBE	Value: ['bytesReceived', 'bytesSent', 'bytesLost', 'packetsRecei...

5.2 Class Device



this superclass provides the attributes and methods shared by Station and VAP

5.2.1 Methods

__init__ (<i>self, socket, intf_name</i>)
creates a device object (used by VAP and STATION)
Parameters
socket: tuple (ip, port_num)
intf_name: name of the wireless interface that this device uses
Overrides: object.__init__

id(*self*)

unique identifier (UUID) for this device

get_connection(*self*)

returns a tuple representing the socket to connection to the physical station

msg_id(*self*)

helper function: returns the next message id to be sent. increments the message ID by 1

intf_name(*self*)

wireless interface of this device (set during __init__)

mac_address(*self*)

wireless interface's MAC address

ipv4_address(*self*, *ip_conf*)

NOT IMPLEMENTED YET – function in C is ok

set IP v4 parameters: ip, netmask, gateway

ipv6_address(*self*, *ip_conf*)

NOT IMPLEMENTED YET – function in C is ok

set the device's IP address (version 6)

fastBSSTransition_compatible(*self*)

connect to ap requesting if it is "Fast BSS Transition" compatible

bytesReceived(*self*)

number of bytes received on this interface (cumulative value)

bytesSent(*self*)

number of bytes sent on this interface (cumulative value)

bytesLost(*self*)

number of bytes lost on this interface (cumulative value)

packetsReceived(*self*)

number of packets received on this interface (cumulative value)

packetsSent(*self*)

number of packets sent on this interface (cumulative value)

packetsLost(*self*)

number of packets lost on this interface (cumulative value)

jitter(*self*)

NOT IMPLEMENTED YET

Return Value

mean jitter measured at the wireless interface

delay(*self*)

NOT IMPLEMENTED YET

Return Value

mean delay measured at the wireless interface

retries(*self*)

NOT IMPLEMENTED YET

Return Value

number of retries at the wireless interface

failed(*self*)

NOT IMPLEMENTED YET

Return Value

total number of failures at the wireless interface

statistics(*self*)

collect some cumulative statistics – rx_packets, rx_bytes, rx_dropped, tx_packets, tx_bytes. these values are accumulate since the interface went up.

signalStrength(*self*)

NOT IMPLEMENTED YET

SNR(*self*)

retrieve current SNR

txpower(*self*, *new_value*)

set current tx power

tx_bitrate(*self*, *sta_mac*=None)**Return Value**the last seen tx_bitrate for a given station (in Mbps) or a list for each station connected (if *sta_mac* is None)**uptime**(*self*)

system uptime and idle time in seconds

cpu(*self*)

physical device's CPU usage

cpu_usage(*self*)same as `cpu()`. to keep model compatibility**memory**(*self*)

physical device's memory usage

memory_usage(*self*)same as `memory()`. to keep model compatibility**getAPsInRange**(*self*)

get aps that are in range.

Note: this method is not precise, because it relies on the spare time the device has to scan all the channels**clear_mange**(*self*)**add_tos**(*self*, *rules*)**replace_tos**(*self*, *rules*)

subscribe__metric (<i>self</i> , <i>metrics</i> , <i>period</i> =100, <i>activate</i> =True)
--

check if all metrics are valid ones

evMetric (<i>self</i> , <i>metric</i> , <i>value</i>)
--

this method should be overwritten to deal with received metrics

Inherited from object

__delattr__(), __format__(), __getattr__(), __hash__(), __new__(),
 __reduce__(), __reduce_ex__(), __repr__(), __setattr__(), __sizeof__(),
 __str__(), __subclasshook__()

5.2.2 Properties

Name	Description
<i>Inherited from object</i>	
__class__	

6 Module `ethanol.ethanol.network`

defines the Network class that represents the SSIDs controlled by the Ethanol Controller

This module provides:

- 1) `add_network(net)`
- 2) `del_network(net)`
- 3) `get_or_create_network_by_ssid(ssid)`
- 4) class `Network`

Author: Henrique Duarte Moura

Organization: WINET/DCC/UFMG

Copyright: h3dema (c) 2017

Contact: henriquemoura@hotmail.com

Since: July 2015

Status: in development

6.1 Functions

<code>list_of_networks()</code>
--

<code>add_network(ssid, net)</code>
--

returns True if successfully added the network to the set. False if the SSID of the network provided already exists. net is also not added to the set @return boolean

<code>del_network(net)</code>

delete this network: disconfigures all vaps associated to this network @param net the network to be deleted

<code>get_or_create_network_by_ssid(ssid)</code>

@return a Network object representing the ssid. if none exists, a new one is created
--

6.2 Class Network

object —
ethanol.ethanol.network.Network

handle a network - a network is a set of VAPs that share the same SSID

6.2.1 Methods

__init__ (<i>self</i> , <i>ssid</i>)
create a network with ESSID = ssid Overrides: object.__init__
__del__ (<i>self</i>)
class destructor Called when the instance is about to be destroyed.
releaseResources (<i>self</i>)
deconfigure vap's SSID
id (<i>self</i>)
returns the network's internal class ID
vaps (<i>self</i>)
returns VAPs associated to this network
SSID (<i>self</i> , <i>newSSID</i> , <i>keepenabled=False</i>)
change the SSID of the network
associateVirtualAP (<i>self</i> , <i>vap</i>)
join the vap to the network. called by ssid.setter in VAP class
deassociateVirtualAP (<i>self</i> , <i>vap</i>)
releases the vap from the network called by ssid.setter in VAP class

handoffUser(*station*, *new_vap*)

handles handoff. This method relies on 802.11 mobility domain feature. So the station and the AP should be configure to use mobility domain. This method disassociates the station from a vap in the network and moves it to a *new_vap* in this network. It also sends a message to the station, using `station.triggerTransition()`, instructing it to roam to a new ap.

See Also: documentacao-para-handover.pdf for instruction on how to set up the station and the AP for handover. **** not implemented yet ****

Inherited from object

`__delattr__()`, `__format__()`, `__getattr__()`, `__hash__()`, `__new__()`,
`__reduce__()`, `__reduce_ex__()`, `__repr__()`, `__setattr__()`, `__sizeof__()`,
`__str__()`, `__subclasshook__()`

6.2.2 Properties

Name	Description
<i>Inherited from object</i>	
<code>__class__</code>	

7 Module **ethanol.ethanol.radio**

This module provides: class `radio.Radio`

Author: Henrique Duarte Moura

Organization: WINET/DCC/UFMG

Copyright: h3dema (c) 2017

Contact: henriquemoura@hotmail.com

Since: July 2015

Status: in development

7.1 Class **Radio**

object —
ethanol.ethanol.radio.Radio

Radio represents the physical radios attached to an AP

abstracts the physical radio

7.1.1 Methods

__init__ (<i>self</i> , <i>ap</i> , <i>wiphy_name</i> , <i>ip</i> , <i>port</i>) <hr/> creates an object associated with the "ap" must provide the wiphy_name (intf_name) Overrides: object.__init__
id (<i>self</i>) <hr/> Radio UUID
__str__ (<i>self</i>) <hr/> returns the ip and port of this device Overrides: object.__str__

msg_id(*self*)

handles the radio message id's

Return Value

an id to be used in the message and increments the current id

wiphy(*self*)**Return Value**

the wireless interface name

validChannels(*self*)

informs a list of valid channel numbers, supported by the device in its wireless interface

Return Valuethe list of the channels that can be assigned to this interface.
Returns [] if an error occurs**currentChannel**(*self*, *new_channel*)

tries to set the ap channel.

Note: to confirm that the channel was changed, issue currentChannel()**frequency**(*self*, *new_frequency*)

not implemented yet

same as currentChannel() but uses the frequency instead

tx_bitrates(*self*, *tx_bitrates*)

not implemented yet

powerSaveMode(*self*, *new_mode*)

sets the power mode of the ap to (on or off)

fragmentationThreshold(*self*, *new_threshold*)

not implemented yet

channelBandwidth(*self*, *new_chbw*)

not implemented yet

channelInfo(*self*)

uses MSG_GET_CHANNELINFO to get information for each channel available for the wireless interface

Return Value

a list with channel info – active_time, busy_time, channel_type, extension_channel_busy_time, frequency, in_use, noise, receive_time, transmit_time

wireless_interfaces(*self*)

get a list of all wireless interfaces

Return Value

list of interfaces

fastBSSTransition(*self*)

connect to ap requesting if it is "Fast BSS Transition" compatible

beaconInterval(*self*, *value*=100)

connect to AP to set beacon interval value returns nothing

getWirelessInterfaceInfo(*self*)

call ap to get information about this interface

getLinkStatitics(*self*)

not implemented yet

getACS(*self*, *num_tests*=1)

request that the AP computes the ACS factor for each frequency in the intf_name interface

define_msg_to_capture(*self*, *rules*, *func*)

this register in the AP rules to send all matched wireless messages to the Ethanol controller

Parameters

func: handler function for this messages the function has one parameter: func(received_frame)

rules: a list of rules - each rule identifies a type of wireless frame that should be sent to the controller

send_frame (<i>self</i> , <i>packet</i>)

calls the AP so it sends the frame

Parameters

packet : fully formatted (binary) packet to be sent by the AP
--

Inherited from object

```

__delattr__(), __format__(), __getattr__(), __hash__(), __new__(),
__reduce__(), __reduce_ex__(), __repr__(), __setattr__(), __sizeof__(),
__subclasshook__()

```

7.1.2 Properties

Name	Description
<i>Inherited from object</i>	
<code>__class__</code>	

8 Module *ethanol.ethanol.station*

Author: Henrique Duarte Moura

Organization: WINET/DCC/UFGM

Copyright: h3dema (c) 2017

Contact: henriquemoura@hotmail.com

Since: July 2015

Status: in development

8.1 Functions

add_station(*client_address*)

Create (and return) possibly several objects, one for each wireless connections identified by (client_address, interface name). This function updates a list of these objects.

client_address = (ip, port) used by the Hello message's process

get_station_by_mac_address(*mac_address*)

returns a connected station (object), provided its mac address

get_station_by_ip(*ip*)

returns a dictionary containing the connected station, provided its ip address
note: that the object are indexed by the intf_name, in case the station has multiple wireless interfaces e.g. list_of_stations[ip]['wlan0']

is_sta_with_ip_connected(*ip*)

Return Value

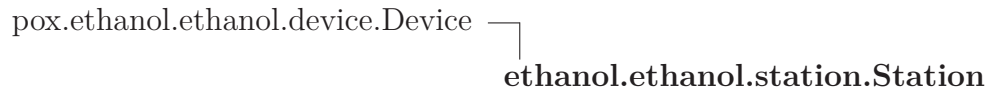
TRUE if an STA with the ip provided as a parameter is connected

Note: this is the ip of the STA's interface that sends packets to the controller, i.e., normally it is an ethernet interface

8.2 Variables

Name	Description
list_of_stations	Value: {}

8.3 Class Station



This module contains the Station class. Its objects represent each user connected to the VAP Each station is identified by its ip address and wireless interface name

8.3.1 Methods

__init__ (<i>self</i> , <i>socket</i> , <i>intf_name</i> ='wlan0')
constructor: creates an object that represents the user connection receives an ip/port pair from the hello message uses this info to connect to the station and retrieve the radio it is connected to

__del__ (<i>self</i>)
destructor

vap (<i>self</i>)
the VAP the station is connected to

radio (<i>self</i>)
this station is connected to radio, if radio is None the AP is not ethanol enabled

wireless_interfaces (<i>self</i>)
returns all wireless enabled interfaces of the device

getInterferenceMap (<i>self</i>)
not implemented yet

getChannelInfo (<i>self</i>)
not implemented yet

getBeaconInfo (<i>self</i>)
not implemented yet

getNoiseInfo(*self*)

not implemented yet

getLinkMeasurement(*self*)

not implemented yet

getStatistics(*self*)

not implemented yet

getLocation(*self*)

not implemented yet

triggerTransition(*self*, *new_vap*)

uses message MSG_TRIGGER_TRANSITION to send to the station a command to change to a new ap

Parameters

new_ap: MAC address of the new AP

__str__(*self*)

string representation of this station

9 Module `ethanol.ethanol.switch`

An L2 learning switch based on L2 learning example from POX

9.1 Functions

<code>launch(transparent=False, hold_down=_flood_delay)</code>
--

Starts an L2 learning switch.

9.2 Variables

Name	Description
log	Value: <code>core.getLogger()</code>

9.3 Class `LearningSwitch`

object └─ `ethanol.ethanol.switch.LearningSwitch`

9.3.1 Methods

<code>__init__(self, connection, transparent, idle_timeout=10, hard_timeout=30)</code>
--

`x.__init__(...)` initializes x; see `help(type(x))` for signature

Overrides: `object.__init__` `exitit`(inherited documentation)

Inherited from object

`__delattr__()`, `__format__()`, `__getattr__()`, `__hash__()`, `__new__()`,
`__reduce__()`, `__reduce_ex__()`, `__repr__()`, `__setattr__()`, `__sizeof__()`,
`__str__()`, `__subclasshook__()`

9.3.2 Properties

Name	Description
<i>Inherited from object</i>	

continued on next page

Name	Description
<code>__class__</code>	

9.4 Class `l2_learning`

object —
 `ethanol.ethanol.switch.l2_learning`

Waits for OpenFlow switches to connect and makes them learning switches.

9.4.1 Methods

<code>__init__</code> (<i>self</i> , <i>transparent</i>)
x. <code>__init__</code> (...) initializes x; see <code>help(type(x))</code> for signature
Overrides: object. <code>__init__</code> extit(inherited documentation)

Inherited from object

`__delattr__`(), `__format__`(), `__getattr__`(), `__hash__`(), `__new__`(),
`__reduce__`(), `__reduce_ex__`(), `__repr__`(), `__setattr__`(), `__sizeof__`(),
`__str__`(), `__subclasshook__`()

9.4.2 Properties

Name	Description
<i>Inherited from object</i>	
<code>__class__</code>	

10 Module `ethanol.ethanol.vap`

This module provides: class VAP

Author: Henrique Duarte Moura

Organization: WINET/DCC/UFMG

Copyright: h3dema (c) 2017

Contact: henriquemoura@hotmail.com

Since: July 2015

Status: in development

10.1 Class VAP

`pox.ethanol.ethanol.device.Device` —
`ethanol.ethanol.vap.VAP`

represents the logical AP (defined by the SSID it contains) inherits DEVICE class

10.1.1 Methods

<code>__init__(self, server, ssid, radio, mac_address)</code>

constructor:

<code>__del__(self)</code>

destructor: not implemented yet

<code>__str__(self)</code>

vap string representation

<code>register_station(self, station=None)</code>

register a station in the list called by station. <code>__init__</code>

<code>unregister_station(self, station)</code>
--

register a station in the list called by station. <code>__del__</code>
--

stations(*self*)

return the stations (objects) currently connected to the VAP and to the controller (ethanol enabled stations)

radio(*self*)

the radio to which the radio is connected

enabled(*self*, *value*)**ssid**(*self*, *value*)

change the vap's SSID

broadcastSSID(*self*, *value*)

not implemented yet

fastBSSTransitionEnabled(*self*)

not implemented yet

security(*self*)

not implemented yet

contention(*self*)

not implemented yet

cac(*self*)

not implemented yet

frameBurstEnabled(*self*)

:return if AP has frame burst feature enabled

guardInterval(*self*)

:return Guard Interval

dtimInterval(*self*)

:return DTIM interval

ctsProtection_enabled (<i>self</i>)
not implemented yet
rtsThreshold (<i>self</i>)
get RTS threshold, if 0 RTS/CTS is not used
getStationInRange (<i>self</i>)
not implemented yet
evUserConnecting (<i>self</i> , <i>mac_station</i>)
evUserAssociating (<i>self</i> , <i>mac_station</i>)
evUserAuthenticating (<i>self</i> , <i>mac_station</i>)
evUserDisassociating (<i>self</i> , <i>mac_station</i>)
evUserReassociating (<i>self</i> , <i>mac_station</i>)
evUserDisconnecting (<i>self</i> , <i>mac_station</i>)
disassociateUser (<i>self</i> , <i>station</i>)
not implemented yet
deauthenticateUser (<i>self</i>)
not implemented yet
evFastTransition (<i>self</i>)
not implemented yet
evFastReassociation (<i>self</i>)
not implemented yet
program_ProbeRequest_Interval (<i>self</i> , <i>Interval=</i> None)
not implemented yet

evProbeRequestReceived(*self*)

not implemented yet

evMgmtFrameReceived(*self*, *msg_type*, *msg*)

not implemented yet

:param *msg_type* indicates the type of the management frame. definition are in ieee80211

```
#define IEEE80211_STYPE_ASSOC_REQ    0x0000
#define IEEE80211_STYPE_ASSOC_RESP  0x0010
#define IEEE80211_STYPE_REASSOC_REQ  0x0020
#define IEEE80211_STYPE_REASSOC_RESP 0x0030
#define IEEE80211_STYPE_PROBE_REQ    0x0040
#define IEEE80211_STYPE_PROBE_RESP   0x0050
#define IEEE80211_STYPE_BEACON        0x0080
#define IEEE80211_STYPE_ATIM          0x0090
#define IEEE80211_STYPE_DISASSOC      0x00A0
#define IEEE80211_STYPE_AUTH          0x00B0
#define IEEE80211_STYPE_DEAUTH        0x00C0
#define IEEE80211_STYPE_ACTION        0x00D0
```

:param *msg* message received

registerMgmtFrame(*self*, *msg_type*, *listener*)

unregisterMgmtFrame(*self*, *msg_type*)

not implemented yet inform the AP that it does not need to send information back to the controller about this type of message

connectNewUser(*self*, *station*, *old_ap*)

not implemented yet transfer information about a station from *old_ap* to this ap

connected_stations(*self*)

:return: list of stations MAC address

mlme_qos_map_request(*self*, *mac_station*, *mappings*)

This primitive is used by an AP to transmit an unsolicited QoS Map Configure frame to a specified STA MAC entity.

mlme_scan_request(*self*, *mac_station*, *configs*)

Requests a survey of potential BSSs that the STA can later elect to try to join.
blocking: waits for response or timeout

mlme_channel_switch(*self*, *mac_station*, *configs*)

requests a switch to a new operating channel. blocking: waits for response or timeout

mlme_neighbor_report(*self*, *mac_station*)

requests a Neighbor Report. blocking: waits for response or timeout

mlme_link_measurement(*self*, *mac_station*, *configs*)

measurement of link path loss and the estimation of link margin between peer entities. blocking: waits for response or timeout

mlme_bss_transition(*self*, *mac_station*, *new_ap*)

measurement of link path loss and the estimation of link margin between peer entities. non-blocking

11 Package *ethanol.events*

Version: 0.3

11.1 Modules

- **events**: Events ~~~~~
(Section 12, p. 37)
- **tests** (Section 13, p. 40)
 - **tests** (Section 14, p. 41)

11.2 Class Events

Encapsulates the core to event subscription and event firing, and feels like a "natural" part of the language.

The class `Events` is there mainly for 3 reasons:

- Events (Slots) are added automatically, so there is no need to declare/create them separately. This is great for prototyping. (Note that `'__events__'` is optional and should primarily help detect misspelled event names.)
- To provide (and encapsulate) some level of introspection.
- To "steel the name" and hereby remove unneeded redundancy in a call like:

```
xxx.OnChange = event('OnChange')
```

11.2.1 Methods

```
__init__(self, events=None)
```

```
__getattr__(self, name)
```

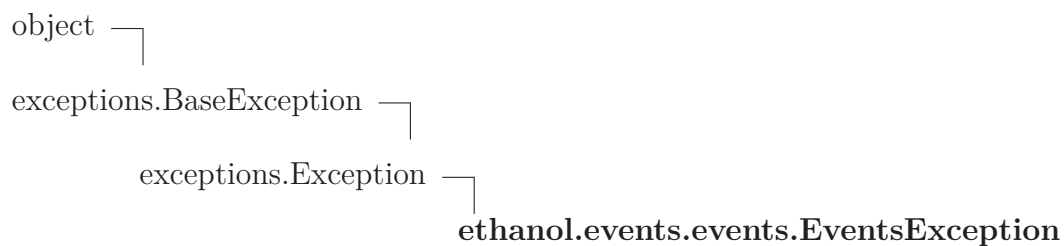
```
__repr__(self)
```

```
__str__(self)
```

```
__len__(self)
```

<code>__iter__(self)</code>

11.3 Class EventsException



11.3.1 Methods

Inherited from exceptions.Exception

`__init__()`, `__new__()`

Inherited from exceptions.BaseException

`__delattr__()`, `__getattribute__()`, `__getitem__()`, `__getslice__()`, `__reduce__()`, `__repr__()`, `__setattr__()`, `__setstate__()`, `__str__()`, `__unicode__()`

Inherited from object

`__format__()`, `__hash__()`, `__reduce_ex__()`, `__sizeof__()`, `__subclasshook__()`

11.3.2 Properties

Name	Description
<i>Inherited from exceptions.BaseException</i>	
	args, message
<i>Inherited from object</i>	
<code>__class__</code>	

12 Module `ethanol.events.events`

`Events`

~~~~~

Implements C#-Style Events.

Derived from the original work by Zoran Isailovski:

<http://code.activestate.com/recipes/410686/> - Copyright (c) 2005

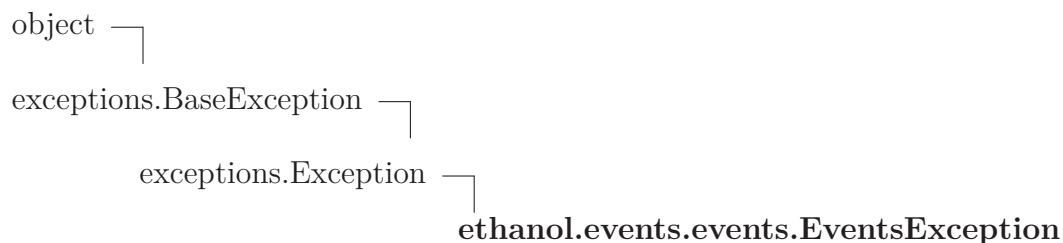
:copyright: (c) 2014-2017 by Nicola Iarocci.

:license: BSD, see LICENSE for more details.

### 12.1 Variables

| Name                     | Description        |
|--------------------------|--------------------|
| <code>__package__</code> | <b>Value:</b> None |

### 12.2 Class `EventsException`



#### 12.2.1 Methods

*Inherited from `exceptions.Exception`*

`__init__()`, `__new__()`

*Inherited from `exceptions.BaseException`*

`__delattr__()`, `__getattr__()`, `__getitem__()`, `__getslice__()`, `__reduce__()`, `__repr__()`, `__setattr__()`, `__setstate__()`, `__str__()`, `__unicode__()`

*Inherited from `object`*

`__format__()`, `__hash__()`, `__reduce_ex__()`, `__sizeof__()`, `__subclasshook__()`

### 12.2.2 Properties

| Name                                                        | Description |
|-------------------------------------------------------------|-------------|
| <i>Inherited from <code>exceptions.BaseException</code></i> |             |
| <code>args</code> , <code>message</code>                    |             |
| <i>Inherited from object</i>                                |             |
| <code>__class__</code>                                      |             |

## 12.3 Class Events

Encapsulates the core to event subscription and event firing, and feels like a "natural" part of the language.

The class `Events` is there mainly for 3 reasons:

- Events (Slots) are added automatically, so there is no need to declare/create them separately. This is great for prototyping. (Note that `'__events__'` is optional and should primarily help detect misspelled event names.)
- To provide (and encapsulate) some level of introspection.
- To "steel the name" and hereby remove unneeded redundancy in a call like:

```
xxx.OnChange = event('OnChange')
```

### 12.3.1 Methods

|                                          |
|------------------------------------------|
| <code>__init__(self, events=None)</code> |
| <code>__getattr__(self, name)</code>     |
| <code>__repr__(self)</code>              |
| <code>__str__(self)</code>               |
| <code>__len__(self)</code>               |

|                             |
|-----------------------------|
| <code>__iter__(self)</code> |
|-----------------------------|

## 13 Package ethanol.events.tests

### 13.1 Modules

- **tests** (*Section 14, p. 41*)

### 13.2 Variables

| Name                     | Description        |
|--------------------------|--------------------|
| <code>__package__</code> | <b>Value:</b> None |

## 14 Module *ethanol.events.tests.tests*

### 14.1 Variables

| Name                     | Description                                       |
|--------------------------|---------------------------------------------------|
| <code>__package__</code> | <b>Value:</b> <code>'ethanol.events.tests'</code> |

### 14.2 Class *TestBase*



**Known Subclasses:** *ethanol.events.tests.tests.TestEventSlot*, *ethanol.events.tests.tests.TestEvents*, *ethanol.events.tests.tests.TestInstanceEvents*

#### 14.2.1 Methods

**setUp(*self*)**

Hook method for setting up the test fixture before exercising it.

Overrides: *unittest.case.TestCase.setUp* *exitit*(inherited documentation)

**callback1(*self*)**

**callback2(*self*)**

**callback3(*self*)**

**Inherited from *unittest.case.TestCase***

`__call__()`, `__eq__()`, `__hash__()`, `__init__()`, `__ne__()`, `__repr__()`, `__str__()`, `addCleanup()`, `addTypeEqualityFunc()`, `assertAlmostEqual()`, `assertAlmostEquals()`, `assertDictContainsSubset()`, `assertDictEqual()`, `assertEqual()`, `assertEquals()`, `assertFalse()`, `assertGreater()`, `assertGreaterEqual()`, `assertIn()`, `assertIs()`, `assertIsInstance()`, `assertIsNone()`, `assertIsNot()`, `assertIsNotNone()`, `assertItemsEqual()`, `assertLess()`, `assertLessEqual()`, `assertListEqual()`, `assertMultiLineEqual()`, `assertNotAlmostEqual()`, `assertNotAlmostEquals()`, `assertNotEqual()`, `assertNotEquals()`, `assertNotIn()`, `assertNotIsInstance()`, `assertNotRegexpMatches()`,

assertRaises(), assertRaisesRegexp(), assertRegexpMatches(), assertSequenceEqual(), assertSetEqual(), assertTrue(), assertTupleEqual(), assert\_(), countTestCases(), debug(), defaultTestResult(), doCleanups(), fail(), failIf(), failIfAlmostEqual(), failIfEqual(), failUnless(), failUnlessAlmostEqual(), failUnlessEqual(), failUnlessRaises(), id(), run(), setUpClass(), shortDescription(), skipTest(), tearDown(), tearDownClass()

### *Inherited from object*

\_\_delattr\_\_(), \_\_format\_\_(), \_\_getattr\_\_(), \_\_new\_\_(), \_\_reduce\_\_(), \_\_reduce\_ex\_\_(), \_\_setattr\_\_(), \_\_sizeof\_\_(), \_\_subclasshook\_\_()

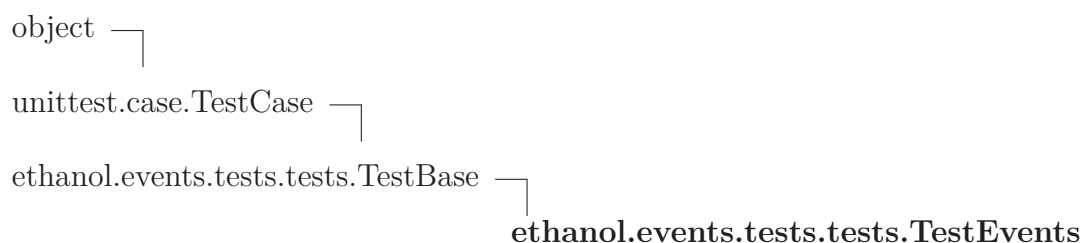
#### 14.2.2 Properties

| Name                                      | Description |
|-------------------------------------------|-------------|
| <i>Inherited from object</i><br>__class__ |             |

#### 14.2.3 Class Variables

| Name                                                                 | Description |
|----------------------------------------------------------------------|-------------|
| <i>Inherited from unittest.case.TestCase</i><br>longMessage, maxDiff |             |

### 14.3 Class TestEvents



#### 14.3.1 Methods

|                              |
|------------------------------|
| <b>test__getattr__(self)</b> |
| <b>test__len__(self)</b>     |



|                              |
|------------------------------|
| <code>test_iter(self)</code> |
|------------------------------|

*Inherited from ethanol.events.tests.tests.TestBase(Section 14.2)*

callback1(), callback2(), callback3(), setUp()

*Inherited from unittest.case.TestCase*

\_\_call\_\_(), \_\_eq\_\_(), \_\_hash\_\_(), \_\_init\_\_(), \_\_ne\_\_(), \_\_repr\_\_(),  
 \_\_str\_\_(), addCleanup(), addTypeEqualityFunc(), assertAlmostEqual(), asser-  
 tAlmostEquals(), assertDictContainsSubset(), assertDictEqual(), assertEquals(), as-  
 sertEquals(), assertFalse(), assertGreater(), assertGreaterEqual(), assertIn(), as-  
 sertIs(), assertIsInstance(), assertIsNone(), assertIsNot(), assertIsNotNone(), as-  
 sertItemsEqual(), assertLess(), assertLessEqual(), assertListEqual(), assertMulti-  
 LineEqual(), assertNotAlmostEqual(), assertNotAlmostEquals(), assertNotEqual(),  
 assertNotEquals(), assertNotIn(), assertNotIsInstance(), assertNotRegexMatches(),  
 assertRaises(), assertRaisesRegex(), assertRegexMatches(), assertSequenceEqual(),  
 assertSetEqual(), assertTrue(), assertTupleEqual(), assert\_(), countTestCases(),  
 debug(), defaultTestResult(), doCleanups(), fail(), failIf(), failIfAlmostEqual(),  
 failIfEqual(), failUnless(), failUnlessAlmostEqual(), failUnlessEqual(), failUnless-  
 Raises(), id(), run(), setUpClass(), shortDescription(), skipTest(), tearDown(),  
 tearDownClass()

*Inherited from object*

\_\_delattr\_\_(), \_\_format\_\_(), \_\_getattr\_\_(), \_\_new\_\_(), \_\_reduce\_\_(),  
 \_\_reduce\_ex\_\_(), \_\_setattr\_\_(), \_\_sizeof\_\_(), \_\_subclasshook\_\_()

### 14.3.2 Properties

| Name                         | Description |
|------------------------------|-------------|
| <i>Inherited from object</i> |             |
| __class__                    |             |

### 14.3.3 Class Variables

| Name                                         | Description |
|----------------------------------------------|-------------|
| <i>Inherited from unittest.case.TestCase</i> |             |
| longMessage, maxDiff                         |             |

## 14.4 Class *TestEventSlot*



### 14.4.1 Methods

**setUp**(*self*)

Hook method for setting up the test fixture before exercising it.

Overrides: `unittest.case.TestCase.setUp` `exitit`(inherited documentation)

**test\_type**(*self*)

**test\_len**(*self*)

**test\_repr**(*self*)

**test\_iter**(*self*)

**test\_getitem**(*self*)

**test\_isub**(*self*)

*Inherited from `ethanol.events.tests.tests.TestBase` (Section 14.2)*

`callback1()`, `callback2()`, `callback3()`

*Inherited from `unittest.case.TestCase`*

`__call__()`, `__eq__()`, `__hash__()`, `__init__()`, `__ne__()`, `__repr__()`, `__str__()`, `addCleanup()`, `addTypeEqualityFunc()`, `assertAlmostEqual()`, `assertAlmostEquals()`, `assertDictContainsSubset()`, `assertDictEqual()`, `assertEqual()`, `assertEquals()`, `assertFalse()`, `assertGreater()`, `assertGreaterEqual()`, `assertIn()`, `assertIs()`, `assertIsInstance()`, `assertIsNone()`, `assertIsNot()`, `assertIsNotNone()`, `assertItemsEqual()`, `assertLess()`, `assertLessEqual()`, `assertListEqual()`, `assertMultiLineEqual()`, `assertNotAlmostEqual()`, `assertNotAlmostEquals()`, `assertNotEqual()`, `assertNotEquals()`, `assertNotIn()`, `assertNotIsInstance()`, `assertNotRegexpMatches()`,

assertRaises(), assertRaisesRegexp(), assertRegexpMatches(), assertSequenceEqual(), assertSetEqual(), assertTrue(), assertTupleEqual(), assert\_(), countTestCases(), debug(), defaultTestResult(), doCleanups(), fail(), failIf(), failIfAlmostEqual(), failIfEqual(), failUnless(), failUnlessAlmostEqual(), failUnlessEqual(), failUnlessRaises(), id(), run(), setUpClass(), shortDescription(), skipTest(), tearDown(), tearDownClass()

### *Inherited from object*

\_\_delattr\_\_(), \_\_format\_\_(), \_\_getattr\_\_(), \_\_new\_\_(), \_\_reduce\_\_(), \_\_reduce\_ex\_\_(), \_\_setattr\_\_(), \_\_sizeof\_\_(), \_\_subclasshook\_\_()

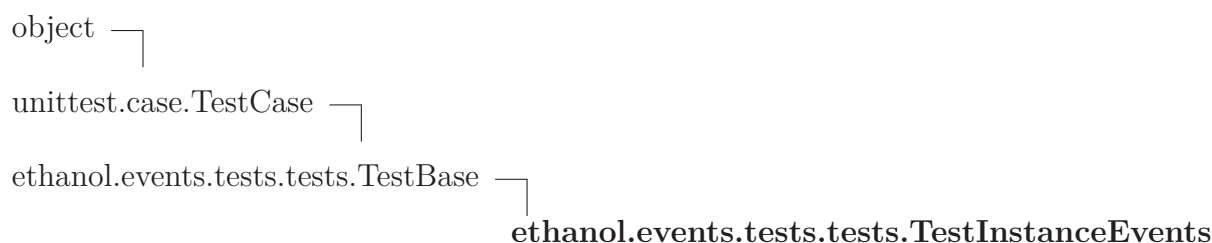
#### 14.4.2 Properties

| Name                                      | Description |
|-------------------------------------------|-------------|
| <i>Inherited from object</i><br>__class__ |             |

#### 14.4.3 Class Variables

| Name                                                                 | Description |
|----------------------------------------------------------------------|-------------|
| <i>Inherited from unittest.case.TestCase</i><br>longMessage, maxDiff |             |

### 14.5 Class *TestInstanceEvents*



#### 14.5.1 Methods

|                                                |
|------------------------------------------------|
| <code>test__getattr__(self)</code>             |
| <code>test__instance__restriction(self)</code> |

*Inherited from ethanol.events.tests.tests.TestBase(Section 14.2)*

callback1(), callback2(), callback3(), setUp()

### ***Inherited from unittest.case.TestCase***

\_\_call\_\_(), \_\_eq\_\_(), \_\_hash\_\_(), \_\_init\_\_(), \_\_ne\_\_(), \_\_repr\_\_(), \_\_str\_\_(), addCleanup(), addTypeEqualityFunc(), assertAlmostEqual(), assertAlmostEquals(), assertDictContainsSubset(), assertDictEqual(), assertEquals(), assertEquals(), assertFalse(), assertGreater(), assertGreaterEqual(), assertIn(), assertIs(), assertIsInstance(), assertIsNone(), assertIsNot(), assertIsNotNone(), assertItemsEqual(), assertLess(), assertLessEqual(), assertListEqual(), assertMultiLineEqual(), assertNotAlmostEqual(), assertNotAlmostEquals(), assertNotEqual(), assertNotEquals(), assertNotIn(), assertNotIsInstance(), assertNotRegexpMatches(), assertRaises(), assertRaisesRegexp(), assertRegexpMatches(), assertSequenceEqual(), assertSetEqual(), assertTrue(), assertTupleEqual(), assert\_(), countTestCases(), debug(), defaultTestResult(), doCleanups(), fail(), failIf(), failIfAlmostEqual(), failIfEqual(), failUnless(), failUnlessAlmostEqual(), failUnlessEqual(), failUnlessRaises(), id(), run(), setUpClass(), shortDescription(), skipTest(), tearDown(), tearDownClass()

### ***Inherited from object***

\_\_delattr\_\_(), \_\_format\_\_(), \_\_getattr\_\_(), \_\_new\_\_(), \_\_reduce\_\_(), \_\_reduce\_ex\_\_(), \_\_setattr\_\_(), \_\_sizeof\_\_(), \_\_subclasshook\_\_()

#### **14.5.2 Properties**

| Name                         | Description |
|------------------------------|-------------|
| <i>Inherited from object</i> |             |
| __class__                    |             |

#### **14.5.3 Class Variables**

| Name                                         | Description |
|----------------------------------------------|-------------|
| <i>Inherited from unittest.case.TestCase</i> |             |
| longMessage, maxDiff                         |             |

## 15 Package ethanol.graph\_coloring

This package contains some extra components.

exact\_color: contains an exact graph coloring algorithm

### 15.1 Modules

- **exact\_color**: Graph coloring  
(Section 16, p. 48)

### 15.2 Variables

| Name        | Description        |
|-------------|--------------------|
| __package__ | <b>Value:</b> None |

## 16 Module `ethanol.graph_coloring.exact_color`

Graph coloring

**Author:** Henrique Moura

**Change Log:** April 04, 2017

**Requires:** `networkx`

### 16.1 Functions

|                                                                         |
|-------------------------------------------------------------------------|
| <code>assign_colors(<i>index_k</i>, <i>graph</i>, <i>colors</i>)</code> |
|-------------------------------------------------------------------------|

|                                                                    |
|--------------------------------------------------------------------|
| <code>coloring(<i>index_k</i>, <i>graph</i>, <i>colors</i>)</code> |
|--------------------------------------------------------------------|

|                                                |
|------------------------------------------------|
| algoritmo de coloracao exata ref.: puntambekar |
|------------------------------------------------|

|                                        |
|----------------------------------------|
| <code>color_graph(<i>graph</i>)</code> |
|----------------------------------------|

|                                          |
|------------------------------------------|
| <code>read_graph(<i>clq_file</i>)</code> |
|------------------------------------------|

## 17 Package ethanol.ovsdb

This package contains a python ovsdb client

### 17.1 Modules

- **ovsdb**: OVSDB calls.  
(Section 18, p. 50)

### 17.2 Variables

| Name        | Description        |
|-------------|--------------------|
| __package__ | <b>Value:</b> None |

## 18 Module ethanol.ovsdb.ovsdb

OVSDb calls. see more information in <https://tools.ietf.org/html/rfc7047>

**Author:** Henrique Duarte Moura

**Organization:** WINET/DCC/UFGM

**Copyright:** h3dema (c) 2017

**Contact:** henriquemoura@hotmail.com

**Since:** July 2015

**Status:** in development

### 18.1 Variables

| Name        | Description                   |
|-------------|-------------------------------|
| __package__ | <b>Value:</b> 'ethanol.ovsdb' |

### 18.2 Class Ovsdb

#### 18.2.1 Methods

|                                                                                                           |
|-----------------------------------------------------------------------------------------------------------|
| <b>__init__</b> ( <i>self</i> , <i>server_ip</i> , <i>server_port</i> =6632, <i>buffer_size</i> =4096)    |
| <b>connect</b> ( <i>self</i> )<br>connect to the openvswitch ovsdb port                                   |
| <b>gather_data</b> ( <i>self</i> )<br>receive all json data                                               |
| <b>echo</b> ( <i>self</i> )<br>perform an echo (ping) to the ovsdb                                        |
| <b>list_dbs</b> ( <i>self</i> )<br>list all databases                                                     |
| <b>monitor</b> ( <i>self</i> , <i>db</i> , <i>columns</i> , <i>monitor_id</i> =None, <i>msg_id</i> =None) |



|                                                                                                                  |
|------------------------------------------------------------------------------------------------------------------|
| <b>monitor_cancel</b> ( <i>self</i> , <i>db</i> , <i>columns</i> , <i>monitor_id</i> =None, <i>msg_id</i> =None) |
|------------------------------------------------------------------------------------------------------------------|

|                                                      |
|------------------------------------------------------|
| <b>list_bridges</b> ( <i>self</i> , <i>db</i> =None) |
|------------------------------------------------------|

|                                                       |
|-------------------------------------------------------|
| :param db: the database name :return: list of bridges |
|-------------------------------------------------------|

|                                               |
|-----------------------------------------------|
| <b>get_schema</b> ( <i>self</i> , <i>db</i> ) |
|-----------------------------------------------|

|                         |
|-------------------------|
| get the database schema |
|-------------------------|

|                              |
|------------------------------|
| :param db: the database name |
|------------------------------|

|                                                    |
|----------------------------------------------------|
| :return: A JSON object with the following members: |
|----------------------------------------------------|

|                                       |          |
|---------------------------------------|----------|
| "name": <id>                          | required |
| "version": <version>                  | required |
| "cksum": <string>                     | optional |
| "tables": {<id>: <table-schema>, ...} | required |

|                                                |
|------------------------------------------------|
| <b>list_tables</b> ( <i>self</i> , <i>db</i> ) |
|------------------------------------------------|

|                                |
|--------------------------------|
| get the tables from a database |
|--------------------------------|

|            |
|------------|
| :param db: |
|------------|

|                                                    |
|----------------------------------------------------|
| :return: A JSON object with the following members: |
|----------------------------------------------------|

|                                         |          |
|-----------------------------------------|----------|
| "columns": {<id>: <column-schema>, ...} | required |
| "maxRows": <integer>                    | optional |
| "isRoot": <boolean>                     | optional |
| "indexes": [<column-set>*]              | optional |

|                                                     |
|-----------------------------------------------------|
| <b>list_table_names</b> ( <i>self</i> , <i>db</i> ) |
|-----------------------------------------------------|

|                                  |
|----------------------------------|
| return a list of all table names |
|----------------------------------|

### 18.2.2 Properties

| Name | Description                |
|------|----------------------------|
| id   | message id (autoincrement) |

## 19 Module `ethanol.server`

This is a pox module. It should be called using `pox.py`.

Command sample:

```
./pox.py ethanol.server
```

**Requires:** `construct` (<https://pypi.python.org/pypi/construct>)

**See Also:** more info at `msg_core.py`

### 19.1 Functions

|                                                                            |
|----------------------------------------------------------------------------|
| <code>run_server(server_address='0.0.0.0', server_port=SERVER_PORT)</code> |
|----------------------------------------------------------------------------|

|                                                                        |
|------------------------------------------------------------------------|
| creates an Ethanol server at <code>SERVER_PORT</code> and activates it |
|------------------------------------------------------------------------|

|                       |
|-----------------------|
| <code>launch()</code> |
|-----------------------|

|                                                 |
|-------------------------------------------------|
| registra a classe que trata as conexões dos Aps |
|-------------------------------------------------|

### 19.2 Class `ethanol_ap_server`

```

object └─ ethanol.server.ethanol_ap_server

```

Waits for OpenFlow switches to connect and saves their information to match with Ethanol AP.

#### 19.2.1 Methods

|                             |
|-----------------------------|
| <code>__init__(self)</code> |
|-----------------------------|

|                                                                                                        |
|--------------------------------------------------------------------------------------------------------|
| <code>x.__init__(...)</code> initializes <code>x</code> ; see <code>help(type(x))</code> for signature |
|--------------------------------------------------------------------------------------------------------|

|                                                                                       |
|---------------------------------------------------------------------------------------|
| Overrides: <code>object.__init__</code> <code>exitit</code> (inherited documentation) |
|---------------------------------------------------------------------------------------|

#### *Inherited from `object`*

```

__delattr__(), __format__(), __getattr__(), __hash__(), __new__(),
__reduce__(), __reduce_ex__(), __repr__(), __setattr__(), __sizeof__(),
__str__(), __subclasshook__()

```

### 19.2.2 Properties

| Name                                                   | Description |
|--------------------------------------------------------|-------------|
| <i>Inherited from object</i><br><code>__class__</code> |             |

## 20 Package `ethanol.ssl_message`

This package contains some components to implement Ethanol API. This module provides messaging capabilities to Ethanol using SSL sockets. This module is used by the ethanol classes.

See `msg_common.py` for the message types supported

### 20.1 Modules

- **enum** (*Section 21, p. 57*)
- **msg\_acs**: implements the following messages:  
(*Section 22, p. 58*)
- **msg\_ap\_broadcastssid**: implements the following messages:  
(*Section 23, p. 60*)
- **msg\_ap\_ctsprotection\_enabled**: implements the following messages:  
(*Section 24, p. 62*)
- **msg\_ap\_dtiminterval**: implements the following messages:  
(*Section 25, p. 64*)
- **msg\_ap\_frameburstenabled**: implements the following messages:  
(*Section 26, p. 66*)
- **msg\_ap\_guardinterval**: implements the following messages:  
(*Section 27, p. 68*)
- **msg\_ap\_in\_range**: implements the following messages:  
(*Section 28, p. 70*)
- **msg\_ap\_interferencemap**: implements the following messages:  
(*Section 29, p. 72*)
- **msg\_ap\_modes**: implements the following messages:  
(*Section 30, p. 73*)
- **msg\_ap\_rtsthreshold**: implements the following messages:  
(*Section 31, p. 74*)
- **msg\_ap\_ssid**: implements: \* `get_ap_ssids`  
(*Section 32, p. 76*)
- **msg\_association**: implements:  
(*Section 33, p. 78*)
- **msg\_beacon\_interval**: handles the beacon interval information: gets or sets it.  
(*Section 34, p. 80*)
- **msg\_bitrates**: implements the following messages:  
(*Section 35, p. 82*)
- **msg\_bye**: implements the BYE message  
(*Section 36, p. 84*)
- **msg\_changed\_ap**: implements the following messages:  
(*Section 37, p. 86*)
- **msg\_channelinfo**: implements the following messages:

(Section 38, p. 88)

- **msg\_channels**: implements the following messages:  
(Section 39, p. 90)
- **msg\_common**: this module contains important constants use throughout our implementation  
(Section 40, p. 93)
- **msg\_core**: All *ssl\_message* modules use python construct (<https://pypi.python.org/pypi/construct>).  
(Section 41, p. 96)
- **msg\_enabled**: implements the following messages:  
(Section 42, p. 98)
- **msg\_error**: error messages  
(Section 43, p. 100)
- **msg\_frequency**: implements the following messages:  
(Section 44, p. 101)
- **msg\_handle\_snr**: implements:  
(Section 45, p. 103)
- **msg\_hello**: basic hello message.  
(Section 46, p. 106)
- **msg\_hostapd\_conf**: configure hostapd.  
(Section 47, p. 108)
- **msg\_interfaces**: implements the following messages:  
(Section 48, p. 110)
- **msg\_log**: defines if our modules will use *pox.log* facility or python log facility  
(Section 49, p. 112)
- **msg\_mean\_sta\_stats**: implements the following messages:  
(Section 50, p. 113)
- **msg\_memcpu**: implements the following messages:  
(Section 51, p. 117)
- **msg\_metric**: implements:  
(Section 52, p. 119)
- **msg\_mlme**: implements the following MLME messages:  
(Section 53, p. 121)
- **msg\_mtu\_qlen**: implements: \* *set\_txqueuelen* \* *set\_mtu*  
(Section 54, p. 124)
- **msg\_ping**: implements:  
(Section 55, p. 126)
- **msg\_powersave**: implements the following messages:  
(Section 56, p. 128)
- **msg\_preamble**: implements: \* *get\_preamble* \* *set\_preamble*  
(Section 57, p. 130)
- **msg\_radio\_wlans**: implements the following messages:  
(Section 58, p. 132)
- **msg\_sent\_received**: implements the following messages:  
(Section 59, p. 134)

- **msg\_server**: this is creates the server, that deals with clients (aps and stations) messages the messages implemented are mapped in `map_msg_to_procedure` main entry to this module is: `call run(server)`  
(Section 60, p. 139)
- **msg\_snr\_power**: implements the following messages:  
(Section 61, p. 141)
- **msg\_ssid**: implements the following messages:  
(Section 62, p. 144)
- **msg\_sta\_link\_information**: implements the following messages:  
(Section 63, p. 146)
- **msg\_sta\_statistics**: implements the following messages:  
(Section 64, p. 148)
- **msg\_station\_trigger\_transition**: implements the following messages:  
(Section 65, p. 150)
- **msg\_statistics**: implements the following messages:  
(Section 66, p. 151)
- **msg\_tos**: implements the following messages:  
(Section 67, p. 153)
- **msg\_uptime**: implements the following messages:  
(Section 68, p. 155)
- **msg\_wlan\_info**: implements: \* `req_wlan_info()`: MSG\_WLAN\_INFO  
(Section 69, p. 156)

## 20.2 Variables

| Name                     | Description        |
|--------------------------|--------------------|
| <code>__package__</code> | <b>Value:</b> None |

## 21 Module ethanol.ssl\_message.enum

### 21.1 Functions

|                                                       |
|-------------------------------------------------------|
| <b>Enums</b> (* <i>sequential</i> , ** <i>named</i> ) |
| helper function - creates an enumeration              |

### 21.2 Variables

| Name        | Description        |
|-------------|--------------------|
| __package__ | <b>Value:</b> None |

### 21.3 Class Enum

helper function - creates an enumeration

```
> Number = Enum('a', 'b', 'c') > print Number.a 0
```

#### 21.3.1 Methods

|                                                 |
|-------------------------------------------------|
| <b>__init__</b> ( <i>self</i> , * <i>keys</i> ) |
|-------------------------------------------------|

## 22 Module `ethanol.ssl_message.msg_acs`

implements the following messages:

\* `get_acs`

no process is implemented: the controller is not supposed to respond to these message

**Author:** Henrique Duarte Moura

**Organization:** WINET/DCC/UFGM

**Copyright:** h3dema (c) 2017

**Contact:** henriquemoura@hotmail.com

**Since:** July 2015

**Status:** in development

**Requires:** construct 2.5.2

### 22.1 Functions

```
get_acs(server, id=0, intf_name=None, sta_ip=None, sta_port=0,
num_tests=1)
```

request the ap to provide ACS information

#### Parameters

|                         |                                                                                                                                 |
|-------------------------|---------------------------------------------------------------------------------------------------------------------------------|
| <code>server:</code>    | tuple (ip, port_num)                                                                                                            |
| <code>id:</code>        | message id                                                                                                                      |
| <code>intf_name:</code> | name of the wireless interface<br>( <i>type=</i> <code>str</code> )                                                             |
| <code>sta_ip:</code>    | ip address of a station to which this message should be relayed. If None don't relay message, server should process the request |
| <code>sta_port:</code>  | socket port of the station                                                                                                      |
| <code>num_tests:</code> | number of tests (greater than or equal to 1) that should be executed                                                            |
| <code>num_tests:</code> | int                                                                                                                             |

### 22.2 Variables



| Name             | Description                                                                   |
|------------------|-------------------------------------------------------------------------------|
| msg_acs          | <b>Value:</b> Struct('msg_ap_in_range',<br>Embed(msg_default), Embed(field... |
| ACS_SCALE_FACTOR | <b>Value:</b> 1000000000000000000.0                                           |

## 23 Module `ethanol.ssl_message.msg_ap_broadcastssid`

implements the following messages:

\* `get_acs`

no process is implemented: the controller is not supposed to respond to these message

**Author:** Henrique Duarte Moura

**Organization:** WINET/DCC/UFMG

**Copyright:** h3dema (c) 2017

**Contact:** henriquemoura@hotmail.com

**Since:** July 2015

**Status:** in development

**Requires:** construct 2.5.2

### 23.1 Functions

|                                                                         |
|-------------------------------------------------------------------------|
| <code>get_broadcastssid(server, id=0, intf_name=None, ssid=None)</code> |
| verify is the interface is broadcasting the SSID                        |
| <b>Parameters</b>                                                       |
| <b>server:</b> tuple (ip, port_num)                                     |
| <b>id:</b> message id                                                   |
| <b>intf_name:</b> name of the wireless interface                        |
| ( <i>type=str</i> )                                                     |

```
set__broadcastssid(server, id=0, intf_name=None, enable=False,
ssid=None)
```

enable or disable the broadcasting of the SSID

omitted fieldlist **Parameters**

**id:** message id

**intf\_name:** name of the wireless interface  
(*type=*str)

**enable:** set if the SSID should be broadcasted or if it is a hidden SSID

**enable:** bool

## 23.2 Variables

| Name                 | Description                                                                |
|----------------------|----------------------------------------------------------------------------|
| msg_ap_broadcastssid | <b>Value:</b> Struct('msg_ap_broadcastssid', Embed(msg_default), Embed(... |

## 24 Module *ethanol.ssl\_message.msg\_ap\_ctsprotection\_enabled*

implements the following messages:

\* *get\_ctsprotection\_enabled*

\* *set\_ctsprotection\_enabled*

no process is implemented: the controller is not supposed to respond to these message

**Author:** Henrique Duarte Moura

**Organization:** WINET/DCC/UFGM

**Copyright:** h3dema (c) 2017

**Contact:** henriquemoura@hotmail.com

**Since:** July 2015

**Status:** in development

**Requires:** construct 2.5.2

### 24.1 Functions

|                                                                                                 |
|-------------------------------------------------------------------------------------------------|
| <b><i>get_ctsprotection_enabled</i></b> ( <i>server</i> , <i>id</i> =0, <i>intf_name</i> =None) |
| Verify if RTS/CTS mechanism is activated                                                        |
| <b>Parameters</b>                                                                               |
| <b>server:</b> tuple (ip, port_num)                                                             |
| <b>id:</b> message id                                                                           |
| <b>intf_name:</b> name of the wireless interface.                                               |
| ( <i>type=</i> str)                                                                             |

```
set_ctsprotection_enabled(server, id=0, intf_name=None,
enable=False)
```

enable or disable RTS/CTS mechanism

#### Parameters

**server:** tuple (ip, port\_num)  
**id:** message id  
**intf\_name:** name of the wireless interface.  
*(type=str)*  
**enable:** true activates RTS/CTS mechanism  
**enable:** bool

## 24.2 Variables

| Name                      | Description                                                                |
|---------------------------|----------------------------------------------------------------------------|
| msg_ctsprotection_enabled | <b>Value:</b> Struct('ctsprotection_enabled', Embed(msg_default), Embed... |

## 25 Module `ethanol.ssl_message.msg_ap_dtiminterval`

implements the following messages:

\* `set_ap_dtiminterval`

\* `get_ap_dtiminterval`

no process is implemented: the controller is not supposed to respond to these message

**Author:** Henrique Duarte Moura

**Organization:** WINET/DCC/UFGM

**Copyright:** h3dema (c) 2017

**Contact:** henriquemoura@hotmail.com

**Since:** July 2015

**Status:** in development

**Requires:** construct 2.5.2

### 25.1 Functions

|                                                                         |
|-------------------------------------------------------------------------|
| <code>get_ap_dtiminterval(server, id=0, intf_name=None)</code>          |
| get the DTIM interval set in the interface <code>intf_name</code>       |
| <b>Parameters</b>                                                       |
| <b>server:</b> tuple (ip, port_num)                                     |
| <b>id:</b> message id                                                   |
| <b>intf_name:</b> name of the wireless interface<br>( <i>type=</i> str) |

```
set_ap_dtiminterval(server, id=0, intf_name=None, dtim_interval=100)
```

set the DTIM interval of the interface *intf\_name*

#### Parameters

**server:** tuple (ip, port\_num)  
**id:** message id  
**intf\_name:** name of the wireless interface  
*(type=str)*  
**dtim\_interval:** DTIM interval  
*(type=int)*

**Note:** <https://routerguide.net/dtim-interval-period-best-setting/>

## 25.2 Variables

| Name                | Description                                                                   |
|---------------------|-------------------------------------------------------------------------------|
| msg_ap_dtiminterval | <b>Value:</b> Struct('msg_ap_dtiminterval',<br>Embed(msg_default), Embed(f... |

## 26 Module `ethanol.ssl_message.msg_ap_frameburstenabled`

implements the following messages:

\* `get_ap_frameburstenabled`

\* `set_ap_frameburstenabled`

no process is implemented: the controller is not supposed to respond to these message

**Author:** Henrique Duarte Moura

**Organization:** WINET/DCC/UFGM

**Copyright:** h3dema (c) 2017

**Contact:** henriquemoura@hotmail.com

**Since:** July 2015

**Status:** in development

**Requires:** construct 2.5.2

### 26.1 Functions

|                                                                     |
|---------------------------------------------------------------------|
| <code>get_ap_frameburstenabled(server, id=0, intf_name=None)</code> |
| if frame burst is enabled                                           |
| <b>Parameters</b>                                                   |
| <b>server:</b> tuple (ip, port_num)                                 |
| <b>id:</b> message id                                               |
| <b>intf_name:</b> name of the wireless interface                    |
| ( <i>type=str</i> )                                                 |



```
set_ap_frameburstenabled(server, id=0, intf_name=None,
enabled=False)
```

#### Parameters

**server:** tuple (ip, port\_num)  
**id:** message id  
**intf\_name:** name of the wireless interface  
*(type=str)*  
**enabled:** enables or disables frame burst  
*(type=bool)*

## 26.2 Variables

| Name                     | Description                                                                      |
|--------------------------|----------------------------------------------------------------------------------|
| msg_ap_frameburstenabled | <b>Value:</b><br>Struct('msg_ap_frameburstenabled',<br>Embed(msg_default), Em... |

## 27 Module *ethanol.ssl\_message.msg\_ap\_guardinterval*

implements the following messages:

\* *get\_ap\_guardinterval*

\* *set\_ap\_guardinterval*

no process is implemented: the controller is not supposed to respond to these message

**Author:** Henrique Duarte Moura

**Organization:** WINET/DCC/UFGM

**Copyright:** h3dema (c) 2017

**Contact:** henriquemoura@hotmail.com

**Since:** July 2015

**Status:** in development

**Requires:** construct 2.5.2

### 27.1 Functions

|                                                                                            |
|--------------------------------------------------------------------------------------------|
| <b><i>get_ap_guardinterval</i></b> ( <i>server</i> , <i>id</i> =0, <i>intf_name</i> =None) |
| get the guard interval set in the interface <i>intf_name</i>                               |
| <b>Parameters</b>                                                                          |
| <b><i>server</i>:</b> tuple (ip, port_num)                                                 |
| <b><i>id</i>:</b> message id                                                               |
| <b><i>intf_name</i>:</b> name of the wireless interface                                    |
| ( <i>type</i> =str)                                                                        |

```
set_ap_guardinterval(server, id=0, intf_name=None,
guard_interval=100)
```

set the guard interval of the interface `intf_name`

#### Parameters

**server:** tuple (ip, port\_num)  
**id:** message id  
**intf\_name:** name of the wireless interface  
*(type=str)*  
**guard\_interval:** time used as guard interval between transmissions  
*(type=int)*

## 27.2 Variables

| Name                              | Description                                                                                 |
|-----------------------------------|---------------------------------------------------------------------------------------------|
| <code>msg_ap_guardinterval</code> | <b>Value:</b> <code>Struct('msg_ap_guardinterval',<br/>Embed(msg_default), Embed(...</code> |

## 28 Module `ethanol.ssl_message.msg_ap_in_range`

implements the following messages:

\* `get_ap_in_range`

no process is implemented: the controller is not supposed to respond to these message

**Author:** Henrique Duarte Moura

**Organization:** WINET/DCC/UFMG

**Copyright:** h3dema (c) 2017

**Contact:** henriquemoura@hotmail.com

**Since:** July 2015

**Status:** in development

**Requires:** construct 2.5.2

### 28.1 Functions

```
get_ap_in_range(server, id=0, intf_name=None, sta_ip=None,
sta_port=0)
```

request the ap or the client to try to detect the aps in range, using 802.11 scanning capability

#### Parameters

**server:** tuple (ip, port\_num)  
**id:** message id  
**intf\_name:** name of the wireless interface  
(*type=str*)  
**sta\_ip:** ip address of the station that this message should be relayed to, if sta\_ip is different from None  
(*type=str*)  
**sta\_port:** socket port number of the station  
(*type=int*)

#### Return Value

msg, num\_aps, aps the received message (a Container), the number of aps in range, a list of aps (ap\_in\_range struct)

## 28.2 Variables

| Name            | Description                                                                   |
|-----------------|-------------------------------------------------------------------------------|
| ap_in_range     | <b>Value:</b> Struct('ap_in_range',<br>Embed(field_intf_name), Embed(field... |
| msg_ap_in_range | <b>Value:</b> Struct('msg_ap_in_range',<br>Embed(msg_default), Embed(field... |

## 29 Module *ethanol.ssl\_message.msg\_ap\_interferencemap*

implements the following messages:

\* `get_ap_interferenceMap`

no process is implemented: the controller is not supposed to respond to these message

**Author:** Henrique Duarte Moura

**Organization:** WINET/DCC/UFGM

**Copyright:** h3dema (c) 2017

**Contact:** henriquemoura@hotmail.com

**Since:** July 2015

**Status:** in development

**Requires:** construct 2.5.2

### 29.1 Functions

|                                                                                          |
|------------------------------------------------------------------------------------------|
| <code>get_ap_interferenceMap(<i>server</i>, <i>m_id</i>=0, <i>intf_name</i>=None)</code> |
|------------------------------------------------------------------------------------------|

### 29.2 Variables

| Name                     | Description        |
|--------------------------|--------------------|
| <code>__package__</code> | <b>Value:</b> None |

### 30 Module *ethanol.ssl\_message.msg\_ap\_modes*

implements the following messages:

\* `get_ap_supported_intf_modes`

no process is implemented: the controller is not supposed to respond to these message

**Author:** Henrique Duarte Moura

**Organization:** WINET/DCC/UFMG

**Copyright:** h3dema (c) 2017

**Contact:** henriquemoura@hotmail.com

**Since:** July 2015

**Status:** in development

**Requires:** construct 2.5.2

#### 30.1 Functions

|                                                                                               |
|-----------------------------------------------------------------------------------------------|
| <code>get_ap_supported_intf_modes(<i>server</i>, <i>m_id</i>=0, <i>intf_name</i>=None)</code> |
|-----------------------------------------------------------------------------------------------|

#### 30.2 Variables

| Name                     | Description        |
|--------------------------|--------------------|
| <code>__package__</code> | <b>Value:</b> None |

## 31 Module `ethanol.ssl_message.msg_ap_rtsthreshold`

implements the following messages:

\* `get_ap_rtsthreshold`

\* `set_ap_rtsthreshold`

no process is implemented: the controller is not supposed to respond to these message

**Author:** Henrique Duarte Moura

**Organization:** WINET/DCC/UFGM

**Copyright:** h3dema (c) 2017

**Contact:** henriquemoura@hotmail.com

**Since:** July 2015

**Status:** in development

**Requires:** construct 2.5.2

### 31.1 Functions

|                                                                                                 |
|-------------------------------------------------------------------------------------------------|
| <b><code>get_ap_rtsthreshold</code></b> ( <i>server</i> , <i>id</i> =0, <i>intf_name</i> =None) |
| verify is the interface is broadcasting the SSID                                                |
| <b>Parameters</b>                                                                               |
| <b><code>server</code>:</b> tuple (ip, port_num)                                                |
| <b><code>id</code>:</b> message id                                                              |
| <b><code>intf_name</code>:</b> name of the wireless interface                                   |
| ( <i>type=</i> <i>str</i> )                                                                     |
| <b>Return Value</b>                                                                             |
| msg, value                                                                                      |



```
set_ap_rtsthreshold(server, id=0, intf_name=None, rts_threshold=0)
```

enable or disable the broadcasting of the SSID

#### Parameters

**server:** tuple (ip, port\_num)

**id:** message id

**intf\_name:** name of the wireless interface

(*type=**str*)

## 31.2 Variables

| Name                | Description                                                                   |
|---------------------|-------------------------------------------------------------------------------|
| msg_ap_rtsthreshold | <b>Value:</b> Struct('msg_ap_rtsthreshold',<br>Embed(msg_default), Embed(f... |

## 32 Module `ethanol.ssl_message.msg_ap_ssid`

implements: \* `get_ap_ssids`

**Author:** Henrique Duarte Moura

**Organization:** WINET/DCC/UFGM

**Copyright:** h3dema (c) 2017

**Contact:** henriquemoura@hotmail.com

**Since:** July 2015

**Status:** in development

**Requires:** `construct 2.5.2`

### 32.1 Functions

```
get_ap_ssids(server, id=0, sta_ip=None, sta_port=0, intf_names=[])
```

returns the channel and frequency of the ssid for each `intf_names`

#### Parameters

|                    |                                                                                                                                                        |
|--------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>server:</b>     | tuple (ip, port_num)                                                                                                                                   |
| <b>id:</b>         | message id                                                                                                                                             |
| <b>intf_names:</b> | names of the wireless interface<br>( <i>type=list of str</i> )                                                                                         |
| <b>sta_ip:</b>     | ip address of the station that this message should be<br>relayed to, if <code>sta_ip</code> is different from <code>None</code><br>( <i>type=str</i> ) |
| <b>sta_port:</b>   | socket port number of the station<br>( <i>type=int</i> )                                                                                               |

### 32.2 Variables

| Name                   | Description                                                                                                                                                                         |
|------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>ssid_info</code> | information about the configured SSID: wiphy,<br>ESSID, channel, frequency, mode<br><b>Value:</b> <code>Struct('ssid_info',<br/>Embed(field_intf_name),<br/>Embed(field_s...</code> |

*continued on next page*

| Name        | Description                                                                                        |
|-------------|----------------------------------------------------------------------------------------------------|
| msg_ap_ssid | message structure<br><b>Value:</b> Struct('msg_ap_ssid',<br>Embed(msg_default), Embed(field_sta... |

### 33 Module *ethanol.ssl\_message.msg\_association*

implements:

- \* the default process function used by the controller
- \* `process_association()`
- \* `get_association()`
- \* `register_functions()` used in VAP
- \* `set_event_association()`

omitted fieldlist **Author:** Henrique Duarte Moura

**Organization:** WINET/DCC/UFMG

**Copyright:** h3dema (c) 2017

**Contact:** henriquemoura@hotmail.com

**Since:** July 2015

**Status:** in development

**Requires:** construct 2.5.2

#### 33.1 Functions

|                                                                                              |
|----------------------------------------------------------------------------------------------|
| <code>get_association(server, id=0, association_type=None, mac_sta=None, mac_ap=None)</code> |
|----------------------------------------------------------------------------------------------|

|                                                  |
|--------------------------------------------------|
| only for tests. the controller don't use this!!! |
|--------------------------------------------------|

|                                           |
|-------------------------------------------|
| <code>register_functions(mac, vap)</code> |
|-------------------------------------------|

|                                                                                                                                                             |
|-------------------------------------------------------------------------------------------------------------------------------------------------------------|
| use this function to register the VAP object <code>process_association</code> will call the object's methods to deal with each one of the association steps |
|-------------------------------------------------------------------------------------------------------------------------------------------------------------|

|                                                          |
|----------------------------------------------------------|
| <code>process_association(received_msg, fromaddr)</code> |
|----------------------------------------------------------|

|                                                                                        |
|----------------------------------------------------------------------------------------|
| <code>set_event_association(server, id=0, mac_sta=None, events=[], action=True)</code> |
|----------------------------------------------------------------------------------------|

#### 33.2 Variables

| Name                         | Description                                                                                                                                                                  |
|------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| field_mac_ap                 | handles the ap's mac address used in msg_association<br><b>Value:</b> Struct('mac_ap', SLInt32('mac_ap_size'), If(lambda ctx: c...                                           |
| field_mac_sta                | handles the station's mac address used in msg_association<br><b>Value:</b> Struct('mac_sta', SLInt32('mac_sta_size'), If(lambda ctx:...                                      |
| msg_association              | all association message types are the same, and use msg_association struct to send information<br><b>Value:</b> Struct('msg_association', Embed(msg_default), Embed(field... |
| registered_functions         | <b>Value:</b> {}                                                                                                                                                             |
| EVENT_MSG_ASSOCIATION        | <b>Value:</b> 1 << 0                                                                                                                                                         |
| EVENT_MSG_DISASSOCIATION     | <b>Value:</b> 1 << 1                                                                                                                                                         |
| EVENT_MSG_REASSOCIATION      | <b>Value:</b> 1 << 2                                                                                                                                                         |
| EVENT_MSG_AUTHORIZATION      | <b>Value:</b> 1 << 3                                                                                                                                                         |
| EVENT_MSG_USER_DISCONNECTING | <b>Value:</b> 1 << 4                                                                                                                                                         |
| EVENT_MSG_USER_CONNECTING    | <b>Value:</b> 1 << 5                                                                                                                                                         |
| msg_event_association        | <b>Value:</b> Struct('msg_event_association', Embed(msg_default), Embed...                                                                                                   |

## 34 Module `ethanol.ssl_message.msg_beacon_interval`

handles the beacon interval information: gets or sets it. Implements:

\* `get_beacon_interval()`

\* `set_beacon_interval()`

no process is implemented: the controller is not supposed to respond to these message

**Author:** Henrique Duarte Moura

**Organization:** WINET/DCC/UFGM

**Copyright:** h3dema (c) 2017

**Contact:** henriquemoura@hotmail.com

**Since:** July 2015

**Status:** in development

**Requires:** construct 2.5.2

### 34.1 Functions

|                                                                                                 |
|-------------------------------------------------------------------------------------------------|
| <b><code>get_beacon_interval</code></b> ( <i>server</i> , <i>id</i> =0, <i>intf_name</i> =None) |
| get beacon interval in miliseconds for the interface <i>intf_name</i>                           |
| <b>Parameters</b>                                                                               |
| <b><i>server</i>:</b> tuple (ip, port_num)                                                      |
| <b><i>id</i>:</b> message id                                                                    |
| <b><i>intf_name</i>:</b> name of the wireless interface                                         |
| ( <i>type</i> =str)                                                                             |
| <b>Return Value</b>                                                                             |
| -1 if an error occurs                                                                           |

```
set_beacon_interval(server, id=0, intf_name=None,
                    beacon_interval=100)
```

set the beacon interval (in ms) default = 100ms different brands and models offer different allowable beacon interval ranges

#### Parameters

**server:** tuple (ip, port\_num)  
**id:** message id  
**intf\_name:** name of the wireless interface  
*(type=str)*  
**beacon\_interval:** *(type=int)*

## 34.2 Variables

| Name                | Description                                                                   |
|---------------------|-------------------------------------------------------------------------------|
| msg_beacon_interval | <b>Value:</b> Struct('msg_beacon_interval',<br>Embed(msg_default), Embed(f... |
| ERROR               | <b>Value:</b> -1                                                              |

## 35 Module *ethanol.ssl\_message.msg\_bitrates*

implements the following messages:

- \* MSG\_GET\_TX\_BITRATES: *get\_tx\_bitrates*
- \* MSG\_GET\_TX\_BITRATE : *get\_tx\_bitrate*
- \* MSG\_SET\_TX\_BITRATES: TODO

no process is implemented: the controller is not supposed to respond to these message

**Author:** Henrique Duarte Moura

**Organization:** WINET/DCC/UFGM

**Copyright:** h3dema (c) 2017

**Contact:** henriquemoura@hotmail.com

**Since:** July 2015

**Status:** in development

**Requires:** construct 2.5.2

### 35.1 Functions

***get\_tx\_bitrates***(*server*, *id*=0, *intf\_name*=None, *sta\_ip*=None, *sta\_port*=0)

get the channels the interface *intf\_name* supports, this function applies to access points

**Parameters**

- server:** tuple (ip, port\_num)
- id:** message id
- intf\_name:** name of the wireless interface  
(*type=str*)
- sta\_ip:** ip address of the station that this message should be relayed to, if *sta\_ip* is different from None  
(*type=str*)
- sta\_port:** socket port number of the station  
(*type=int*)

**Return Value**

a dictionary, the index is the band



```
get_tx_bitrate(server, id=0, intf_name=None, sta_ip=None, sta_port=0,
sta_mac=None)
```

get the channels the interface `intf_name` supports, applies to access points

#### Parameters

**server:** tuple (ip, port\_num)  
**id:** message id  
**intf\_name:** name of the wireless interface  
*(type=str)*  
**sta\_ip:** ip address of the station that this message should be relayed to, if `sta_ip` is different from `None`  
*(type=str)*  
**sta\_port:** socket port number of the station  
*(type=int)*  
**sta\_mac:** if `None`, scan for all stations. If specified (str with MAC address dotted format), returns only the station, if connected

## 35.2 Variables

| Name                         | Description                                                                                                                |
|------------------------------|----------------------------------------------------------------------------------------------------------------------------|
| <code>iw_bitrates</code>     | <b>Value:</b> Struct('iw_bitrates', LFloat32("bitrate"), UInt8('is_sho...                                                  |
| <code>iw_bands</code>        | <b>Value:</b> Struct('iw_bands', Embed(field_intf_name), UInt32('band'...                                                  |
| <code>msg_tx_bitrates</code> | <b>Value:</b> Struct('msg_tx_bitrates', Embed(msg_default), Embed(field...                                                 |
| <code>msg_tx_bitrate</code>  | *****<br>MSG_TYPE.MSG_GET_TX_BITRATE<br>*****<br><b>Value:</b> Struct('msg_tx_bitrate', Embed(msg_default), Embed(field... |

## 36 Module *ethanol.ssl\_message.msg\_bye*

implements the BYE message

**Author:** Henrique Duarte Moura

**Organization:** WINET/DCC/UFGM

**Copyright:** h3dema (c) 2017

**Contact:** henriquemoura@hotmail.com

**Since:** July 2015

**Status:** in development

**Requires:** construct 2.5.2

### 36.1 Functions

|                                                                            |
|----------------------------------------------------------------------------|
| <b>send_msg_bye</b> ( <i>server</i> , <i>id</i> =0, <i>tcp_port</i> =None) |
|----------------------------------------------------------------------------|

disconnects the ethanol device from the controller

**Parameters**

**server:** tuple (ip, port\_num)

**id:** message id

**tcp\_port:** socket port number of the device  
(*type=int*)

|                                                              |
|--------------------------------------------------------------|
| <b>process_bye</b> ( <i>received_msg</i> , <i>fromaddr</i> ) |
|--------------------------------------------------------------|

returns the message to the ssl server process. nothing to be done, only send back the same message

**Parameters**

**func\_bye:** event

|                                                |
|------------------------------------------------|
| <b>bogus_bye_on_change</b> (** <i>kwargs</i> ) |
|------------------------------------------------|

### 36.2 Variables

| Name       | Description                                                                                                                                                                   |
|------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| events_bye | to handle a receiving bye messages, just add your function to events_bye your function must use 'def my_func(**kwargs)' signature for compatibility<br><b>Value:</b> Events() |
| msg_bye    | <b>Value:</b> Struct('msg_bye',<br>Embed(msg_default),<br>SLInt32('tcp_port'),)                                                                                               |

## 37 Module `ethanol.ssl_message.msg_changed_ap`

implements the following messages:

\* `changed_ap`

\* `process_changed_ap`

no process is implemented: the controller is not supposed to respond to these message

**Author:** Henrique Duarte Moura

**Organization:** WINET/DCC/UFGM

**Copyright:** h3dema (c) 2017

**Contact:** henriquemoura@hotmail.com

**Since:** July 2015

**Status:** in development

**Requires:** construct 2.5.2

### 37.1 Functions

|                                                                                                                      |
|----------------------------------------------------------------------------------------------------------------------|
| <b>changed_ap</b> ( <i>server</i> , <i>id</i> =0, <i>status</i> =0, <i>current_ap</i> =None, <i>intf_name</i> =None) |
| verify is the interface is broadcasting the SSID                                                                     |
| <b>Parameters</b>                                                                                                    |
| <b>server:</b> tuple (ip, port_num)                                                                                  |
| <b>id:</b> message id                                                                                                |
| <b>intf_name:</b> names of the wireless interface<br>( <i>type=list of str</i> )                                     |
| <b>status:</b> inform the status of the operation (result from change<br>ap operation)<br>( <i>type=int</i> )        |
| <b>current_ap:</b> MAC address of the ap<br>( <i>type=str</i> )                                                      |

```
process_hello(received_msg, fromaddr)
```

for now, only logs the information

#### Parameters

**received\_msg**: stream of bytes to be decoded

**fromaddr**: IP address from the device that sent this message

## 37.2 Variables

| Name             | Description                                                                |
|------------------|----------------------------------------------------------------------------|
| field_current_ap | <b>Value:</b> Struct('current_ap', SLInt32('current_ap_size'), If(lambd... |
| msg_changed_ap   | <b>Value:</b> Struct('msg_changed_ap', Embed(msg_default), Embed(field_... |

## 38 Module *ethanol.ssl\_message.msg\_channelinfo*

implements the following messages:

\* MSG\_GET\_CHANNELINFO: *get\_channelinfo*

no process is implemented: the controller is not supposed to respond to these message

**Author:** Henrique Duarte Moura

**Organization:** WINET/DCC/UFMG

**Copyright:** h3dema (c) 2017

**Contact:** henriquemoura@hotmail.com

**Since:** July 2015

**Status:** in development

**Requires:** construct 2.5.2

### 38.1 Functions

```
get_channelinfo(server, id=0, intf_name=None, channel=0,  
only_channel_in_use=False)
```

get the channels the interface *intf\_name* supports, this function applies to access points

#### Parameters

|                             |                                                                |
|-----------------------------|----------------------------------------------------------------|
| <b>server:</b>              | tuple (ip, port_num)                                           |
| <b>id:</b>                  | message id                                                     |
| <b>intf_name:</b>           | names of the wireless interface<br>( <i>type=list of str</i> ) |
| <b>channel:</b>             | specify a channel to scan<br>( <i>type=int</i> )               |
| <b>only_channel_in_use:</b> | return only the channel in use<br>( <i>type=bool</i> )         |

#### Return Value

*msg* - received message a list

### 38.2 Variables

| Name            | Description                                                                   |
|-----------------|-------------------------------------------------------------------------------|
| channel_info    | <b>Value:</b> Struct('channel_info',<br>ULInt32('frequency'), SLInt8('in_u... |
| msg_channelinfo | <b>Value:</b> Struct('msg_channelinfo',<br>Embed(msg_default), Embed(field... |

## 39 Module *ethanol.ssl\_message.msg\_channels*

implements the following messages:

- \* `get_channels`
- \* `get_currentchannel`
- \* `set_currentchannel`

no process is implemented: the controller is not supposed to respond to these message

**Author:** Henrique Duarte Moura

**Organization:** WINET/DCC/UFMG

**Copyright:** h3dema (c) 2017

**Contact:** henriquemoura@hotmail.com

**Since:** July 2015

**Status:** in development

**Requires:** construct 2.5.2

### 39.1 Functions

|                                                                                          |
|------------------------------------------------------------------------------------------|
| <b><code>get_channels</code></b> ( <i>server</i> , <i>id</i> =0, <i>intf_name</i> =None) |
| get the channels the interface <i>intf_name</i> supports, applies to access points       |
| <b>Parameters</b>                                                                        |
| <b><i>server</i>:</b> tuple (ip, port_num)                                               |
| <b><i>id</i>:</b> message id                                                             |
| <b><i>intf_name</i>:</b> names of the wireless interface                                 |
| ( <i>type=list of str</i> )                                                              |
| <b>Return Value</b>                                                                      |
| msg - received message                                                                   |



---

```
get_currentchannel(server, id=0, intf_name=None, sta_ip=None,
sta_port=0)
```

---

get the channel the interface is configured to use . You can ask the AP to relay this request to the station if (sta\_ip, sta\_port) is provided

**Parameters**

**server:** tuple (ip, port\_num)  
**id:** message id  
**intf\_name:** names of the wireless interface  
*(type=list of str)*  
**sta\_ip:** ip address of the station that this message should be relayed to, if sta\_ip is different from None  
*(type=str)*  
**sta\_port:** socket port number of the station  
*(type=int)*

**Return Value**

msg - received message

---

```
set_currentchannel(server, id=0, channel=None, intf_name=None,
sta_ip=None, sta_port=0)
```

---

set the current channel to channel

**Parameters**

**server:** tuple (ip, port\_num)  
**id:** message id  
**intf\_name:** names of the wireless interface  
*(type=list of str)*  
**sta\_ip:** ip address of the station that this message should be relayed to, if sta\_ip is different from None  
*(type=str)*  
**sta\_port:** socket port number of the station  
*(type=int)*

**Return Value**

msg - received message

## 39.2 Variables

| Name               | Description                                                                    |
|--------------------|--------------------------------------------------------------------------------|
| valid_channel      | <b>Value:</b> Struct('valid_channel',<br>ULInt32('frequency'), ULInt32('ch...' |
| msg_channels       | <b>Value:</b> Struct('msg_channels',<br>Embed(msg_default), Embed(field_in...' |
| msg_currentchannel | <b>Value:</b> Struct('msg_currentchannel',<br>Embed(msg_default), Embed(fi...' |

## 40 Module `ethanol.ssl_message.msg_common`

this modules contains important constants use throught out our implementation

**Author:** Henrique Duarte Moura

**Organization:** WINET/DCC/UFMG

**Copyright:** h3dema (c) 2017

**Contact:** henriquemoura@hotmail.com

**Since:** July 2015

**Status:** in development

**Requires:** construct 2.5.2

### 40.1 Functions

|                                            |
|--------------------------------------------|
| <b>tri_boolean</b> ( <i>v</i> , <i>d</i> ) |
|--------------------------------------------|

|                                 |
|---------------------------------|
| <b>hexadecimal</b> ( <i>s</i> ) |
|---------------------------------|

|                                                |
|------------------------------------------------|
| converts a string of bytes to a string of hexa |
|------------------------------------------------|

|                                             |
|---------------------------------------------|
| <b>connect_ssl_socket</b> ( <i>server</i> ) |
|---------------------------------------------|

|                                |
|--------------------------------|
| creates a ssl socket to server |
|--------------------------------|

|                   |
|-------------------|
| <b>Parameters</b> |
|-------------------|

|                                      |
|--------------------------------------|
| <b>server:</b> is a tuple (ip, port) |
|--------------------------------------|

|                                             |
|---------------------------------------------|
| <b>is_error_msg</b> ( <i>received_msg</i> ) |
|---------------------------------------------|

|                                              |
|----------------------------------------------|
| <b>get_error_msg</b> ( <i>received_msg</i> ) |
|----------------------------------------------|

```
send_and_receive_msg(server, msg_struct, builder, parser,
only_send=False)
```

generic function to send and receive message

#### Parameters

**server:** (serverIp, serverPort)  
**msg\_struct:** Container with message fields  
**builder:** Struct.build  
**parser:** Struc.parse this Struct class must be able to interpret  
 Cointainer fields

#### Return Value

error : false if something goes wrong msg : a Container with the  
 message

```
len_of_string(v)
```

```
return_from_dict(d, v, error)
```

## 40.2 Variables

| Name        | Description                                                                                                                                                                                                       |
|-------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| VERSION     | ethanol version<br><b>Value:</b> "1.0.3"                                                                                                                                                                          |
| MSG_TYPE    | contains all constants used as message type<br><b>Value:</b> Enum('MSG_HELLO_TYPE',<br>'MSG_BYE_TYPE', 'MSG_ERR_TYPE', 'M...                                                                                      |
| SERVER_ADDR | this is the default address our server is going to<br>bind for tests, connect only to the loopback<br>interface if you want to connect to all available<br>interfaces, use "0.0.0.0"<br><b>Value:</b> "localhost" |
| SERVER_PORT | this is the default port used in the AP the port<br>in the station is SERVER_PORT+1 (by<br>default)<br><b>Value:</b> 22222                                                                                        |
| BUFFER_SIZE | size of the buffer used by the python socket<br><b>Value:</b> 65536                                                                                                                                               |

*continued on next page*

| Name                   | Description                                                                                                                                                                                               |
|------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| MSG_ERROR_TYPE         | constantes usadas para definição de erro de mensagens usadas no campo <code>error_type</code> in <code>msg_error.py</code><br><b>Value:</b> Enum('ERROR_UNKNOWN', 'ERROR_VERSION_MISMATCH', 'ERROR_PR...) |
| DEFAULT_WIFI_INTF-NAME | <b>Value:</b> 'wlan0'                                                                                                                                                                                     |

## 41 Module `ethanol.ssl_message.msg_core`

All `ssl_modules` use `python construct` (<https://pypi.python.org/pypi/construct>). To install this module:

```
wget -c https://pypi.python.org/packages/source/c/construct/construct-2.5.2.tar.gz tar zxvf
construct-2.5.2.tar.gz cd construct-2.5.2 sudo ./setup.py install
```

**See Also:** documentation at <http://construct.readthedocs.io/en/latest/>

**Author:** Henrique Duarte Moura

**Organization:** WINET/DCC/UFGM

**Copyright:** h3dema (c) 2017

**Contact:** [henriquemoura@hotmail.com](mailto:henriquemoura@hotmail.com)

**Since:** July 2015

**Status:** in development

**Requires:** `construct 2.5.2`

### 41.1 Functions

|                                                                                          |
|------------------------------------------------------------------------------------------|
| <b>toHex</b> ( <i>s</i> )                                                                |
| <b>Parameters</b><br><i>s</i> : is a number stored in an string                          |
| <b>Return Value</b><br>a string, each byte of <i>s</i> is coded as a two char hex string |

|                                                                                                                         |
|-------------------------------------------------------------------------------------------------------------------------|
| <b>int32_to_bytes</b> ( <i>i</i> , <i>endian</i> ='1')                                                                  |
| helper function to <code>BooleanFlag()</code> returns boolean value coded as string of 4 bytes default is little endian |

|                                                                                                         |
|---------------------------------------------------------------------------------------------------------|
| <b>BooleanFlag</b> ( <i>name</i> , <i>truth_value</i> =1, <i>false_value</i> =0, <i>default</i> =False) |
| Defines a Construct boolean type. The flag is coded as a 32 bit value                                   |

|                                                                                         |
|-----------------------------------------------------------------------------------------|
| <b>decode__default__fields</b> ( <i>received_msg</i> )                                  |
| handles the default header of all ethanol's messages                                    |
| <b>Parameters</b>                                                                       |
| <i>received_msg</i> : byte stream to be decoded (parsed) using construct message struct |

## 41.2 Variables

| Name                         | Description                                                                                                                              |
|------------------------------|------------------------------------------------------------------------------------------------------------------------------------------|
| <code>msg_default</code>     | default message structure to be embedded in the first part of every message<br><b>Value:</b> <code>Struct('msg_default')</code>          |
| <code>field_intf_name</code> | handles an interface name field (a C char * field)<br><b>Value:</b> <code>Struct('_intf_name')</code>                                    |
| <code>field_mac_addr</code>  | handles a mac address field (a C char * field)<br><b>Value:</b> <code>Struct('mac_addr')</code>                                          |
| <code>field_ssid</code>      | handles a ssid field (a C char * field)<br><b>Value:</b> <code>Struct('ssid')</code>                                                     |
| <code>field_station</code>   | handles a station IP address (a C char * field), and its port (a C int field)<br><b>Value:</b> <code>Struct('station_connection')</code> |
| <code>__package__</code>     | <b>Value:</b> <code>'ethanol.ssl_message'</code>                                                                                         |

## 42 Module *ethanol.ssl\_message.msg\_enabled*

implements the following messages:

\* *is\_802\_11e\_enabled*

\* *is\_fastbsstransition\_compatible*

no process is implemented: the controller is not supposed to respond to these message

**Author:** Henrique Duarte Moura

**Organization:** WINET/DCC/UFGM

**Copyright:** h3dema (c) 2017

**Contact:** henriquemoura@hotmail.com

**Since:** July 2015

**Status:** in development

**Requires:** construct 2.5.2

### 42.1 Functions

```
is_802_11e_enabled(server, id=0, intf_name=DEFAULT_WIFI_INTFNAME,
sta_ip=None, sta_port=0)
```

verifies if 802.11e is supported and is enabled

#### Parameters

**server:** tuple (ip, port\_num)  
**id:** message id  
**intf\_name:** names of the wireless interface  
*(type=list of str)*  
**sta\_ip:** ip address of the station that this message should be  
relayed to, if *sta\_ip* is different from None  
*(type=str)*  
**sta\_port:** socket port number of the station  
*(type=int)*

#### Return Value

msg - received message



```
is_fastbsstransition_compatible(server, id=0,  
intf_name=DEFAULT_WIFI_INTFNAME, sta_ip=None, sta_port=0)
```

checks if the interface supports fast BSS transition feature

#### Parameters

**server:** tuple (ip, port\_num)  
**id:** message id  
**intf\_name:** names of the wireless interface  
*(type=list of str)*  
**sta\_ip:** ip address of the station that this message should be  
 relayed to, if sta\_ip is different from None  
*(type=str)*  
**sta\_port:** socket port number of the station  
*(type=int)*

#### Return Value

msg - received message

## 42.2 Variables

| Name        | Description                                                                   |
|-------------|-------------------------------------------------------------------------------|
| msg_enabled | <b>Value:</b> Struct('msg_enabled',<br>Embed(msg_default), Embed(field_int... |

## 43 Module *ethanol.ssl\_message.msg\_error*

error messagens

**Author:** Henrique Duarte Moura

**Organization:** WINET/DCC/UFMG

**Copyright:** h3dema (c) 2017

**Contact:** henriquemoura@hotmail.com

**Since:** July 2015

**Status:** in development

**Requires:** construct 2.5.2

### 43.1 Functions

```
return_error_msg_struct(m_id,  
error_type=MSG_ERROR_TYPE.ERROR_UNKNOWN)
```

return error message as an array of bytes

**Parameters**

*id*: message id

**Return Value**

*msg* - received message

```
process_msg_not_implemented(received_msg, fromaddr)
```

generates an error message for the case where the process procedure is not implemented in Python returns an error

(not implemented)

### 43.2 Variables

| Name                   | Description                                                                       |
|------------------------|-----------------------------------------------------------------------------------|
| <code>msg_error</code> | <b>Value:</b> Struct('msg_error',<br>Embed(msg_default),<br>SLInt32('error_ty...' |

## 44 Module *ethanol.ssl\_message.msg\_frequency*

implements the following messages:

\* *get\_frequency*

\* *set\_frequency*

no process is implemented: the controller is not supposed to answer these message

**Author:** Henrique Duarte Moura

**Organization:** WINET/DCC/UFGM

**Copyright:** h3dema (c) 2017

**Contact:** henriquemoura@hotmail.com

**Since:** July 2015

**Status:** in development

**Requires:** construct 2.5.2

### 44.1 Functions

```
get_frequency(server, id=0, intf_name=None, sta_ip=None, sta_port=0)
```

the interface is configured to use the frequency returned by this function  
can ask the AP to relay this request to the station if (sta\_ip, sta\_port) is provided

@param server: tuple (ip, port\_num)

@param id: message id

@param intf\_name: name of the wireless interface

@type intf\_name: str

@param sta\_ip: ip address of the station that this message should be relayed to, if provided

@type sta\_ip: str

@param sta\_port: socket port number of the station

@type sta\_port: int

@return: msg - received message

```
set_currentchannel(server, id=0, frequency=None, intf_name=None,
sta_ip=None, sta_port=0)
```

set the current frequency to value provided by the parameter "frequency"

#### Parameters

**frequency:** new channel based on frequency

(*type=int*)

**server:** tuple (ip, port\_num)

**id:** message id

**intf\_name:** name of the wireless interface

(*type=str*)

**sta\_ip:** ip address of the station that this message should be relayed to, if sta\_ip is different from None

(*type=str*)

**sta\_port:** socket port number of the station

(*type=int*)

#### Return Value

msg - received message

## 44.2 Variables

| Name          | Description                                                                   |
|---------------|-------------------------------------------------------------------------------|
| msg_frequency | <b>Value:</b> Struct('msg_frequency',<br>Embed(msg_default), Embed(field_s... |

## 45 Module *ethanol.ssl\_message.msg\_handle\_snr*

implements:

\* *snr\_threshold\_interval\_reached* and *process\_snr\_threshold*

\* *set\_snr\_threshold\_interval*

**Author:** Henrique Duarte Moura

**Organization:** WINET/DCC/UFGM

**Copyright:** h3dema (c) 2017

**Contact:** henriquemoura@hotmail.com

**Since:** July 2015

**Status:** in development

**Requires:** construct 2.5.2

### 45.1 Functions

|                                                                                                                                                                                                      |
|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b><i>snr_threshold_reached</i></b> ( <i>server</i> , <i>id</i> =0, <i>sta_ip</i> =None, <i>sta_port</i> =0,<br><i>sta_mac</i> =None, <i>intf_name</i> =None, <i>mac_ap</i> =None, <i>snr</i> =None) |
|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

send information to controller. this implementation will 'never' be used in its python form

**Parameters**

**server:** tuple (ip, port\_num)

**id:** message id

|                                                                               |
|-------------------------------------------------------------------------------|
| <b><i>process_snr_threshold</i></b> ( <i>received_msg</i> , <i>fromaddr</i> ) |
|-------------------------------------------------------------------------------|

|                                                                         |
|-------------------------------------------------------------------------|
| <b><i>bogus_snr_threshold_reached_on_change</i></b> (** <i>kwargs</i> ) |
|-------------------------------------------------------------------------|

```
snr_threshold_interval_reached(server, id=0, sta_ip=None,
sta_port=0, intf_name=None, interval=10)
```

set the time between SNR scans in the station.

#### Parameters

**server:** tuple (ip, port\_num)  
**id:** message id  
**interval:** interval in milliseconds  
*(type=int)*

```
set_snr_threshold(server, id=0, sta_ip=None, sta_port=0,
intf_name=None, threshold=10)
```

set the SNR threshold in dBm. Send message to a station.

#### Parameters

**server:** tuple (ip, port\_num)  
**id:** message id  
**threshold:** SNR threshold in dBm

## 45.2 Variables

| Name                         | Description                                                                                                                                                                                                      |
|------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| events_snr_threshold_reached | to handle a receiving snr_threshold_reached message, just add your function to events_snr_threshold_reached your function must use 'def my_func(**kwargs)' signature for compatibility<br><b>Value:</b> Events() |
| field_mac_ap                 | handles a mac address field for the new ap (a C char * field)<br><b>Value:</b> Struct('mac_ap', SLInt32('mac_ap_size'), If(lambda ctx: c...                                                                      |
| msg_snr_threshold_reached    | message structure<br>MSG_SET_SNR_THRESHOLD_REACHED<br><b>Value:</b> Struct('msg_snr_threshold_reached', Embed(msg_default), E...                                                                                 |

*continued on next page*

| Name              | Description                                                                                                                 |
|-------------------|-----------------------------------------------------------------------------------------------------------------------------|
| msg_snr_interval  | message structure<br>MSG_SET_SNR_INTERVAL<br><b>Value:</b> Struct('msg_snr_interval',<br>Embed(msg_default), Embed(fiel...  |
| msg_snr_threshold | message structure<br>MSG_SET_SNR_THRESHOLD<br><b>Value:</b> Struct('msg_snr_threshold',<br>Embed(msg_default), Embed(fie... |

## 46 Module `ethanol.ssl_message.msg_hello`

basic hello message. Hello carries information about the ap or station to the controller

**Author:** Henrique Duarte Moura

**Organization:** WINET/DCC/UFGM

**Copyright:** h3dema (c) 2017

**Contact:** henriquemoura@hotmail.com

**Since:** July 2015

**Status:** in development

**Requires:** construct 2.5.2

### 46.1 Functions

|                                                                                                                                                    |
|----------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>send_msg_hello(server, m_id=0)</code>                                                                                                        |
| <b>Parameters</b><br><code>server</code> : tuple (ip, port_num)<br><code>m_id</code> : message id<br><b>Return Value</b><br>msg - received message |

|                                                                                                                                                                              |
|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>process_hello(received_msg, fromaddr)</code>                                                                                                                           |
| returns the message to the ssl server process<br><b>Parameters</b><br><code>received_msg</code> :<br><code>fromaddr</code> : ip address of the device that sent this message |

|                                              |
|----------------------------------------------|
| <code>bogus_hello_on_change(**kwargs)</code> |
|----------------------------------------------|

### 46.2 Variables



| Name         | Description                                                                                                                                                                      |
|--------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| events_hello | to handle a receiving hello message, just add your function to events_hello your function must use 'def my_func(**kwargs)' signature for compatibility<br><b>Value:</b> Events() |
| msg_hello    | <b>Value:</b> Struct('msg_hello',<br>Embed(msg_default),<br>SLInt32('device_t...'))                                                                                              |

## 47 Module *ethanol.ssl\_message.msg\_hostapd\_conf*

configure hostapd. Implements:

\* *get\_hostapd\_conf()*

\* *set\_hostapd\_conf()*

no process is implemented: the controller is not supposed to respond to these message

**Author:** Henrique Duarte Moura

**Organization:** WINET/DCC/UFGM

**Copyright:** h3dema (c) 2017

**Contact:** henriquemoura@hotmail.com

**Since:** July 2015

**Status:** in development

**Requires:** construct 2.5.2

### 47.1 Functions

|                                                                                                                 |
|-----------------------------------------------------------------------------------------------------------------|
| <b><i>get_hostapd_conf</i></b> ( <i>server</i> , <i>id</i> =0, <i>intf_name</i> =None, <i>conf_param</i> =None) |
| get beacon interval in milliseconds for the interface <i>intf_name</i>                                          |
| <b>Parameters</b>                                                                                               |
| <i>server</i> : tuple (ip, port_num)                                                                            |
| <i>id</i> : message id                                                                                          |
| <i>intf_name</i> : name of the wireless interface<br>( <i>type</i> =str)                                        |
| <b>Return Value</b>                                                                                             |
| -1 if an error occurs                                                                                           |

```
set_hostapd_conf(server, id=0, intf_name=None, conf_param=None,
conf_value=None)
```

set the beacon interval (in ms) default = 100ms different brands and models offer different allowable beacon interval ranges

#### Parameters

**server:** tuple (ip, port\_num)  
**id:** message id  
**intf\_name:** name of the wireless interface  
*(type=str)*

## 47.2 Variables

| Name                  | Description                                                                                          |
|-----------------------|------------------------------------------------------------------------------------------------------|
| field_parameter       | name of the parameter<br><b>Value:</b> Struct('param_name', SLInt32('param_name_size'), If(lambd...  |
| field_parameter_value | value of the parameter<br><b>Value:</b> Struct('param_name_value', SLInt32('param_name_value_size... |
| msg_hostapd_conf      | <b>Value:</b> Struct('msg_hostapd_conf', Embed(msg_default), Embed(fiel...                           |

## 48 Module `ethanol.ssl_message.msg_interfaces`

implements the following messages:

- \* `get_one_intf`
- \* `get_interfaces`

no process is implemented: the controller is not supposed to respond to these message

**Author:** Henrique Duarte Moura

**Organization:** WINET/DCC/UFGM

**Copyright:** h3dema (c) 2017

**Contact:** henriquemoura@hotmail.com

**Since:** July 2015

**Status:** in development

**Requires:** construct 2.5.2

### 48.1 Functions

```
get_one_intf(server, m_id=0, intf_name=None, sta_ip=None,
sta_port=0)
```

MSG\_GET\_ONE\_INTF: eturns info of interface "intf\_name"

#### Parameters

- server:** tuple (ip, port\_num)
- m\_id:** message id
- intf\_name:** name of the wireless interface  
(*type=str*)
- sta\_ip:** ip address of the station that this message should be  
relayed to, if sta\_ip is different from None  
(*type=str*)
- sta\_port:** socket port number of the station  
(*type=int*)

#### Return Value

msg - received message

```
get_interfaces(server, m_id=0, sta_ip=None, sta_port=0)
```

MSG\_GET\_ALL\_INTF: returns all interfaces

#### Parameters

**server:** tuple (ip, port\_num)  
**m\_id:** message id  
**intf\_name:** name of the wireless interface  
*(type=str)*  
**sta\_ip:** ip address of the station that this message should be relayed to, if sta\_ip is different from None  
*(type=str)*  
**sta\_port:** socket port number of the station  
*(type=int)*

#### Return Value

msg - received message

## 48.2 Variables

| Name     | Description                                                                      |
|----------|----------------------------------------------------------------------------------|
| intfs    | <b>Value:</b> Struct('intfs',<br>SLInt64('ifindex'),<br>Embed(field_intf_name... |
| msg_intf | <b>Value:</b> Struct('msg_intf',<br>Embed(msg_default),<br>Embed(field_statio... |

## 49 Module *ethanol.ssl\_message.msg\_log*

defines if our modules will use *pox.log* facility or *python log* facility

**Author:** Henrique Duarte Moura

**Organization:** WINET/DCC/UFMG

**Copyright:** h3dema (c) 2017

**Contact:** henriquemoura@hotmail.com

**Since:** July 2015

**Status:** in development

**Requires:** *construct* 2.5.2

### 49.1 Variables

| Name        | Description                                                                     |
|-------------|---------------------------------------------------------------------------------|
| USING_POX   | if true, then <i>pox</i> logs our module messages<br><b>Value:</b> <i>False</i> |
| __package__ | <b>Value:</b> <i>'ethanol.ssl_message'</i>                                      |

## 50 Module `ethanol.ssl_message.msg_mean_sta_stats`

implements the following messages:

- \* `send_msg_mean_sta_statistics`
- \* `send_msg_mean_sta_statistics_interface_add`
- \* `send_msg_mean_sta_statistics_interface_remove`
- \* `send_msg_mean_sta_statistics_alpha`
- \* `send_msg_mean_sta_statistics_time`

no process is implemented: the controller is not supposed to respond to these message

**Author:** Henrique Duarte Moura

**Organization:** WINET/DCC/UFMG

**Copyright:** h3dema (c) 2017

**Contact:** henriquemoura@hotmail.com

**Since:** July 2015

**Status:** in development

**Requires:** construct 2.5.2

### 50.1 Functions

```
send_msg_mean_sta_statistics(server, id=0, sta_ip=None, sta_port=0)
```

#### Parameters

- server:** tuple (ip, port\_num)
- id:** message id
- sta\_ip:** ip address of the station that this message should be relayed to, if sta\_ip is different from None  
(*type=*str)
- sta\_port:** socket port number of the station  
(*type=*int)

#### Return Value

- msg - received message

```
send_msg_mean_sta_statistics_interface_add(server, id=0,  
sta_ip=None, sta_port=0, intf_name=None)
```

**Parameters**

**server:** tuple (ip, port\_num)  
**id:** message id  
**sta\_ip:** ip address of the station that this message should be relayed to, if sta\_ip is different from None  
(*type=*str)  
**sta\_port:** socket port number of the station  
(*type=*int)  
**intf\_name:** name of the wireless interface you want to get statistics from  
(*type=*str)

**Return Value**

msg - received message

```
send_msg_mean_sta_statistics_interface_remove(server, id=0,  
sta_ip=None, sta_port=0, intf_name=None)
```

**Parameters**

**server:** tuple (ip, port\_num)  
**id:** message id  
**sta\_ip:** ip address of the station that this message should be relayed to, if sta\_ip is different from None  
(*type=*str)  
**sta\_port:** socket port number of the station  
(*type=*int)  
**intf\_name:** name of the wireless interface you want to remove from pool  
(*type=*str)

**Return Value**

msg - received message



```
send_msg_mean_sta_statistics_alpha(server, id=0, sta_ip=None,
sta_port=0, alpha=0.1)
```

**Parameters**

**server:** tuple (ip, port\_num)  
**id:** message id  
**sta\_ip:** ip address of the station that this message should be relayed to, if sta\_ip is different from None  
*(type=str)*  
**sta\_port:** socket port number of the station  
*(type=int)*  
**alpha:** alpha from EWMA  
*(type=float)*

**Return Value**

msg - received message

```
send_msg_mean_sta_statistics_time(server, id=0, sta_ip=None,
sta_port=0, msec=100)
```

**Parameters**

**server:** tuple (ip, port\_num)  
**id:** message id  
**sta\_ip:** ip address of the station that this message should be relayed to, if sta\_ip is different from None  
*(type=str)*  
**sta\_port:** socket port number of the station  
*(type=int)*  
**msec:** statistics are collected during "msec" interval  
*(type=int)*

**Return Value**

msg - received message

**50.2 Variables**

| Name                | Description                                                                |
|---------------------|----------------------------------------------------------------------------|
| mean_net_statistics | <b>Value:</b> Struct('mean_net_statistics', LFloat64('collisions'), LFl... |

*continued on next page*

| Name                                   | Description                                                                      |
|----------------------------------------|----------------------------------------------------------------------------------|
| msg_mean_statistics                    | <b>Value:</b> Struct('msg_mean_statistics',<br>Embed(msg_default), Embed(f...    |
| msg_mean_sta_statistics-<br>_interface | <b>Value:</b><br>Struct('msg_mean_sta_statistics_interface',<br>Embed(msg_def... |
| msg_mean_sta_statistics-<br>_alpha     | <b>Value:</b><br>Struct('msg_mean_sta_statistics_alpha',<br>Embed(msg_default... |
| msg_mean_sta_statistics-<br>_time      | <b>Value:</b><br>Struct('msg_mean_sta_statistics_time',<br>Embed(msg_default)... |

## 51 Module *ethanol.ssl\_message.msg\_memcpu*

implements the following messages:

\* `get_memory_usage`

\* `get_cpu_usage`

no process is implemented: the controller is not supposed to respond to these message

**Note:** see `msg_cpu.h` and `msg_memory.h` in `hostapd/src/messaging`

**Author:** Henrique Duarte Moura

**Organization:** WINET/DCC/UFMG

**Copyright:** h3dema (c) 2017

**Contact:** `henriquemoura@hotmail.com`

**Since:** July 2015

**Status:** in development

**Requires:** `construct 2.5.2`

### 51.1 Functions

|                                                                                  |
|----------------------------------------------------------------------------------|
| <code>get_memory_usage(server, id=0, sta_ip=None, sta_port=0)</code>             |
| requests the memory usage (in percent) implements <code>MSG_GET_MEMORY</code>    |
| <b>Parameters</b>                                                                |
| <b>server:</b> tuple (ip, port_num)                                              |
| <b>id:</b> message id                                                            |
| <b>sta_ip:</b> ip address of a station that this message should be<br>relayed to |
| <b>sta_port:</b> socket port of the station                                      |
| <b>Return Value</b>                                                              |
| msg, memory usage in percent                                                     |

**get\_cpu\_usage**(*server*, *id*=0, *sta\_ip*=None, *sta\_port*=0)

requests the memory usage (in percent) implements MSG\_GET\_CPU

**Parameters**

**server:** tuple (ip, port\_num)

**id:** message id

**sta\_ip:** ip address of a station that this message should be relayed to

**sta\_port:** socket port of the station

**Return Value**

msg, cpu usage in percent

## 51.2 Variables

| Name       | Description                                                                                                                                                           |
|------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| msg_memcpu | format the MSG_GET_CPU and MSG_GET_MEMORY data structure to be sent by ethanol protocol<br><b>Value:</b> Struct('msg_memcpu', Embed(msg_default), Embed(field_stat... |

## 52 Module *ethanol.ssl\_message.msg\_metric*

implements:

- \* the default process function used by the controller
- \* `register_metric()` used in VAP
- \* `set_metric()`

omitted fieldlist **Author:** Henrique Duarte Moura

**Organization:** WINET/DCC/UFGM

**Copyright:** h3dema (c) 2017

**Contact:** henriquemoura@hotmail.com

**Since:** July 2015

**Status:** in development

**Requires:** construct 2.5.2

### 52.1 Functions

|                                                                                                              |
|--------------------------------------------------------------------------------------------------------------|
| <b>set_metric</b> ( <i>server</i> , <i>id</i> =0, <i>metric</i> =0, <i>enable</i> =True, <i>period</i> =100) |
|--------------------------------------------------------------------------------------------------------------|

|                                                  |
|--------------------------------------------------|
| only for tests. the controller don't use this!!! |
|--------------------------------------------------|

|                                                       |
|-------------------------------------------------------|
| <b>register_metric</b> ( <i>mac</i> , <i>device</i> ) |
|-------------------------------------------------------|

|                                                                                                                                                                                   |
|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| use this function to register the device object process_association will call the object's methods to deal with each one of the association steps mac is the device's mac address |
|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

|                                                                 |
|-----------------------------------------------------------------|
| <b>process_metric</b> ( <i>received_msg</i> , <i>fromaddr</i> ) |
|-----------------------------------------------------------------|

|                           |
|---------------------------|
| calls the device evMetric |
|---------------------------|

### 52.2 Variables

| Name                              | Description                                                                                                            |
|-----------------------------------|------------------------------------------------------------------------------------------------------------------------|
| <code>msg_metric</code>           | all metric message types are the same<br><b>Value:</b> Struct('msg_metric',<br>Embed(msg_default), SLInt8('enable')... |
| <code>registered_functions</code> | <b>Value:</b> {}                                                                                                       |

*continued on next page*

| Name                | Description                                                                                                                     |
|---------------------|---------------------------------------------------------------------------------------------------------------------------------|
| msg_metric_received | all received metric message types are the same<br><b>Value:</b> Struct('msg_metric',<br>Embed(msg_default), Embed(field_mac_... |

## 53 Module *ethanol.ssl\_message.msg\_mlme*

implements the following MLME messages:

\* qos\_map\_request \* scan\_request \* channel\_measurement \* channel\_switch \* neighbor\_report \* link\_measurement \* bss\_transition

no process is implemented: the controller is not supposed to respond to these message

**Author:** Henrique Duarte Moura

**Organization:** WINET/DCC/UFGM

**Copyright:** h3dema (c) 2017

**Contact:** henriquemoura@hotmail.com

**Since:** July 2015

**Status:** in development

**Requires:** construct 2.5.2

### 53.1 Functions

|                                                                                                                                                     |
|-----------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>qos_map_request</b> ( <i>server</i> , <i>id</i> =0, <i>intf_name</i> =None, <i>bssid</i> =None, <i>mac_station</i> =None, <i>mappings</i> =None) |
|-----------------------------------------------------------------------------------------------------------------------------------------------------|

|                                                                                               |
|-----------------------------------------------------------------------------------------------|
| MLME-QOS-MAP.request AP to transmit an unsolicited QoS Map Configure frame to a specified STA |
|-----------------------------------------------------------------------------------------------|

|                                                                                                                                                 |
|-------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>scan_request</b> ( <i>server</i> , <i>id</i> =0, <i>intf_name</i> =None, <i>bssid</i> =None, <i>mac_station</i> =None, <i>configs</i> =None) |
|-------------------------------------------------------------------------------------------------------------------------------------------------|

|                                                                                                                   |
|-------------------------------------------------------------------------------------------------------------------|
| MLME-SCAN.request This primitive requests a survey of potential BSSs that the STA can later elect to try to join. |
|-------------------------------------------------------------------------------------------------------------------|

|                                                                                                                                                                                                                                                     |
|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| config is a dictionary with the following fields (optional): BSSType, BSSID, SSID, ScanType, ProbeDelay, ChannelList, MinChannelTime, MaxChannelTime, RequestInformation, SSID List, ChannelUsage, AccessNetworkType, HESSID, MeshID, DiscoveryMode |
|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

|                                                                                                                                  |
|----------------------------------------------------------------------------------------------------------------------------------|
| <b>channel_measurement</b> ( <i>server</i> , <i>id</i> =0, <i>intf_name</i> =None, <i>bssid</i> =None, <i>mac_station</i> =None) |
|----------------------------------------------------------------------------------------------------------------------------------|

|                      |
|----------------------|
| MLME-MEASURE.request |
|----------------------|

**channel\_switch**(*server*, *id*=0, *intf\_name*=None, *bssid*=None, *mac\_station*=None, *configs*=None)

MLME-CHANNELSWITCH.request requests a switch to a new operating channel. waits the response or timeout

config is a dictionary with the following fields. All optional, except "Channel Number" Mode, Channel Number, Secondary Channel Offset, Channel Switch Count, Mesh Channel Switch Parameters, Wide Bandwidth Channel Switch, New Transmit Power Envelope

**neighbor\_report**(*server*, *id*=0, *intf\_name*=None, *bssid*=None, *mac\_station*=None)

start with MLME-NEIGHBORPREPREQ.request and manages the whole process, until MLME-NEIGHBORRESP.indication

requests that a Neighbor Report Request frame be sent to the AP with which the STA is associated.

waits the response or timeout

**link\_measurement**(*server*, *id*=0, *intf\_name*=None, *bssid*=None, *mac\_station*=None, *configs*=None)

start with MLME-LINKMEASURE.request

supports the measurement of link path loss and the estimation of link margin between peer entities. waits the response or timeout

configs is a dictionary with the following fields Transmit Power, Max Transmit Power

**bss\_transition**(*server*, *id*=0, *intf\_name*=None, *bssid*=None, *mac\_station*=None, *new\_ap*=None)

start with MLME-BTM.request

requests transmission of a BSS Transition Management Request frame to a non-AP STA. waits the response or timeout. generated by the SME to request that a BSS Transition Management Request frame be sent to an associated non-AP STA. This request is sent autonomously.

## 53.2 Variables



| Name        | Description        |
|-------------|--------------------|
| __package__ | <b>Value:</b> None |

## 54 Module *ethanol.ssl\_message.msg\_mtu\_qlen*

implements: \* *set\_txqueuelen* \* *set\_mtu*

**Author:** Henrique Duarte Moura

**Organization:** WINET/DCC/UFGM

**Copyright:** h3dema (c) 2017

**Contact:** henriquemoura@hotmail.com

**Since:** July 2015

**Status:** in development

**Requires:** *construct* 2.5.2

### 54.1 Functions

```
set_msg_mtu_qlen(server, m_type, m_id=0, sta_ip=None, sta_port=0,
intf_name=None, value=None)
```

sets the MTU or Queue Len values

#### Parameters

**server:** tuple (ip, port\_num)  
**id:** message id  
**sta\_ip:** ip address of the station that this message should be relayed to, if *sta\_ip* is different from *None*  
*(type=str)*  
**sta\_port:** socket port number of the station  
*(type=int)*  
**intf\_name:** name of the interface

```
set_mtu(server, m_id=0, sta_ip=None, sta_port=0, intf_name=None,
mtu=None)
```

```
set_txqueuelen(server, m_id=0, sta_ip=None, sta_port=0,
intf_name=None, txqueuelen=None)
```

### 54.2 Variables

| Name         | Description                                                                                        |
|--------------|----------------------------------------------------------------------------------------------------|
| msg_mtu_qlen | message structure<br><b>Value:</b> Struct('msg_mtu_qlen',<br>Embed(msg_default), Embed(field_st... |

## 55 Module `ethanol.ssl_message.msg_ping`

implements:

\* `process_msg_ping()`: generates a pong message in response to a received ping message

\* `send_msg_ping()`: send a ping to another device

**Note:** see `msg_ping.h` in `hostapd/src/messaging`

**Author:** Henrique Duarte Moura

**Organization:** WINET/DCC/UFGM

**Copyright:** h3dema (c) 2017

**Contact:** [henriquemoura@hotmail.com](mailto:henriquemoura@hotmail.com)

**Since:** July 2015

**Status:** in development

**Requires:** `construct 2.5.2`

### 55.1 Functions

|                                            |
|--------------------------------------------|
| <code>generate_ping_data(p_size=64)</code> |
|--------------------------------------------|

|                                        |
|----------------------------------------|
| <code>verify_data(data, p_size)</code> |
|----------------------------------------|

|                                          |
|------------------------------------------|
| check if the payload received is correct |
|------------------------------------------|

|                                    |
|------------------------------------|
| <code>send_msg(server, msg)</code> |
|------------------------------------|

|                                        |
|----------------------------------------|
| sends a message PING msg to the server |
|----------------------------------------|

**Parameters**

**server:** tuple (ip, port) used to socket connect to the client

**msg:** message to be sent (ping or pong)

**send\_msg\_ping**(*server*, *id*=0, *num\_tries*=1, *p\_size*=64)

send a ping message to other ethanol device (mainly to the controller) and receives a pong response

**Parameters**

**server:** tuple (ip, port\_num)

**id:** message id

**num\_tries:** number of message retries before quitting

**p\_size:** payload size (extra size in bytes added to the message)

**Return Value**

all messages sent

**process\_msg\_ping**(*received\_msg*, *fromaddr*)

grabs the ping message, verifies the data field and returns a pong message

## 55.2 Variables

| Name         | Description                                                                                                      |
|--------------|------------------------------------------------------------------------------------------------------------------|
| msg_ping     | ping message data structure<br><b>Value:</b> Struct('msg_ping',<br>Embed(msg_default),<br>SLInt32('data_size...' |
| msg_pong     | pong message data structure<br><b>Value:</b> Struct('msg_pong',<br>Embed(msg_default), LFloat32('rtt'),<br>S...  |
| BYTE_INICIAL | <b>Value:</b> 48                                                                                                 |

## 56 Module *ethanol.ssl\_message.msg\_powersave*

implements the following messages:

\* `get_powersave_mode(intf_name)`

\* `set_powersave_mode(intf_name, powersave_mode)`

no process is implemented: the controller is not supposed to respond to these message

**Author:** Henrique Duarte Moura

**Organization:** WINET/DCC/UFGM

**Copyright:** h3dema (c) 2017

**Contact:** henriquemoura@hotmail.com

**Since:** July 2015

**Status:** in development

**Requires:** construct 2.5.2

### 56.1 Functions

|                                                                                                                            |
|----------------------------------------------------------------------------------------------------------------------------|
| <b>get_powersave_mode</b> ( <i>server</i> , <i>id</i> =0, <i>intf_name</i> =None, <i>sta_ip</i> =None, <i>sta_port</i> =0) |
|----------------------------------------------------------------------------------------------------------------------------|

get if the powersave is set or not

**Parameters**

**server:** tuple (ip, port\_num)

**id:** message id

**intf\_name:** name of the wireless interface

(*type=str*)

**sta\_ip:** ip address of the station that this message should be relayed to, if *sta\_ip* is different from None

(*type=str*)

**sta\_port:** socket port number of the station

(*type=int*)

**Return Value**

msg - received message

```
set_powersave_mode(server, id=0, powersave=True, intf_name=None,
sta_ip=None, sta_port=0)
```

#### Parameters

**server:** tuple (ip, port\_num)

**id:** message id

**intf\_name:** name of the wireless interface  
(*type=*str)

**sta\_ip:** ip address of the station that this message should be relayed to, if sta\_ip is different from None  
(*type=*str)

**sta\_port:** socket port number of the station  
(*type=*int)

## 56.2 Variables

| Name          | Description                                                                |
|---------------|----------------------------------------------------------------------------|
| msg_powersave | <b>Value:</b> Struct('msg_powersave', Embed(msg_default), Embed(field_i... |

## 57 Module *ethanol.ssl\_message.msg\_preamble*

implements: \* *get\_preamble* \* *set\_preamble*

**Author:** Henrique Duarte Moura

**Organization:** WINET/DCC/UFGM

**Copyright:** h3dema (c) 2017

**Contact:** henriquemoura@hotmail.com

**Since:** July 2015

**Status:** in development

**Requires:** *construct* 2.5.2

### 57.1 Functions

```
get_preamble(server, id=0, intf_name=None)
```

gets if the configured preamble is long or short

**Parameters**

**server:** tuple (ip, port\_num)  
**id:** message id  
**intf\_name:** name of the wireless interface  
*(type=str)*

**Return Value**

msg - received message

```
set_preamble(server, id=0, intf_name=None, preamble=0)
```

set the preamble used in some interface

0 = preamble LONG | 1 = preamble SHORT

@param server: tuple (ip, port\_num)

@param id: message id

@param intf\_name: name of the wireless interface

@type intf\_name: str

@param preamble:

@type sta\_ip: bool

@return: msg - received message



## 57.2 Variables

| Name         | Description                                                                   |
|--------------|-------------------------------------------------------------------------------|
| msg_preamble | <b>Value:</b> Struct('msg_preamble',<br>Embed(msg_default), Embed(field_in... |

## 58 Module *ethanol.ssl\_message.msg\_radio\_wlans*

implements the following messages:

\* `get_radio_wlans()` : MSG\_GET\_RADIO\_WLANS

no process is implemented: the controller is not supposed to respond to these message

**Author:** Henrique Duarte Moura

**Organization:** WINET/DCC/UFGM

**Copyright:** h3dema (c) 2017

**Contact:** henriquemoura@hotmail.com

**Since:** July 2015

**Status:** in development

**Requires:** construct 2.5.2

### 58.1 Functions

```
get_radio_wlans(server, id=0, intf_name=None, sta_ip=None,  
sta_port=0)
```

requests the radio wlans, if *intf\_name* is not None, only this interface is considered, otherwise returns all wireless interfaces

#### Parameters

**server:** tuple (ip, port\_num)  
**id:** message id  
**intf\_name:** name of the wireless interface  
(*type=*str)  
**sta\_ip:** ip address of the station that this message should be relayed to, if *sta\_ip* is different from None  
(*type=*str)  
**sta\_port:** socket port number of the station  
(*type=*int)

#### Return Value

msg - received message

## 58.2 Variables

| Name                | Description                                                                                        |
|---------------------|----------------------------------------------------------------------------------------------------|
| list_of_radio_wlans | message structure<br><b>Value:</b> Struct('list_of_radio_wlans',<br>Embed(field_intf_name), Emb... |
| msg_radio_wlans     | <b>Value:</b> Struct('msg_radio_wlans',<br>Embed(msg_default), Embed(field...                      |

## 59 Module `ethanol.ssl_message.msg_sent_received`

implements the following messages:

- \* `send_msg_get_bytesreceived`
- \* `send_msg_get_bytessent`
- \* `send_msg_get_byteslost`
- \* `send_msg_get_packetsreceived`
- \* `send_msg_get_packetssent`
- \* `send_msg_get_packetslost`

no process is implemented: the controller is not supposed to respond to these message

**Author:** Henrique Duarte Moura

**Organization:** WINET/DCC/UFGM

**Copyright:** h3dema (c) 2017

**Contact:** henriquemoura@hotmail.com

**Since:** July 2015

**Status:** in development

**Requires:** construct 2.5.2

## 59.1 Functions

**send\_msg\_sent\_received**(*server*, *id*=0, *type*=None, *intf\_name*=None, *sta\_ip*=None, *sta\_port*=0)

INTERNAL FUNCTION: don't call this function

### Parameters

**server:** tuple (ip, port\_num)  
**id:** message id  
**intf\_name:** name of the wireless interface  
*(type=str)*  
**sta\_ip:** ip address of the station that this message should be relayed to, if *sta\_ip* is different from None  
*(type=str)*  
**sta\_port:** socket port number of the station  
*(type=int)*

### Return Value

msg - received message value (bytes or packets received or sent or lost)

**send\_msg\_get\_bytesreceived**(*server*, *id*=0, *intf\_name*=None, *sta\_ip*=None, *sta\_port*=0)

requests number of bytes received. this number is always incremented since the interface activation

### Parameters

**server:** tuple (ip, port\_num)  
**id:** message id  
**intf\_name:** name of the wireless interface  
*(type=str)*  
**sta\_ip:** ip address of the station that this message should be relayed to, if *sta\_ip* is different from None  
*(type=str)*  
**sta\_port:** socket port number of the station  
*(type=int)*

### Return Value

msg - received message

---

**send\_msg\_get\_bytesent**(*server*, *id*=0, *intf\_name*=None, *sta\_ip*=None, *sta\_port*=0)

---

requests number of bytes sent by the interface. this number is always incremented since the interface activation

**Parameters**

**server:** tuple (ip, port\_num)  
**id:** message id  
**intf\_name:** name of the wireless interface  
(*type=str*)  
**sta\_ip:** ip address of the station that this message should be relayed to, if *sta\_ip* is different from None  
(*type=str*)  
**sta\_port:** socket port number of the station  
(*type=int*)

**Return Value**

msg - received message

---

**send\_msg\_get\_byteslost**(*server*, *id*=0, *intf\_name*=None, *sta\_ip*=None, *sta\_port*=0)

---

requests number of bytes sent by the interface. this number is always incremented since the interface activation

**Parameters**

**server:** tuple (ip, port\_num)  
**id:** message id  
**intf\_name:** name of the wireless interface  
(*type=str*)  
**sta\_ip:** ip address of the station that this message should be relayed to, if *sta\_ip* is different from None  
(*type=str*)  
**sta\_port:** socket port number of the station  
(*type=int*)

**Return Value**

msg - received message

```
send_msg_get_packetsreceived(server, id=0, intf_name=None,  
sta_ip=None, sta_port=0)
```

requests number of packets received by the interface. this number is always incremented since the interface activation

**Parameters**

**server:** tuple (ip, port\_num)  
**id:** message id  
**intf\_name:** name of the wireless interface  
(*type=*str)  
**sta\_ip:** ip address of the station that this message should be relayed to, if sta\_ip is different from None  
(*type=*str)  
**sta\_port:** socket port number of the station  
(*type=*int)

**Return Value**

msg - received message

```
send_msg_get_packetssent(server, id=0, intf_name=None,  
sta_ip=None, sta_port=0)
```

requests number of packets sent by the interface. this number is always incremented since the interface activation

**Parameters**

**server:** tuple (ip, port\_num)  
**id:** message id  
**intf\_name:** name of the wireless interface  
(*type=*str)  
**sta\_ip:** ip address of the station that this message should be relayed to, if sta\_ip is different from None  
(*type=*str)  
**sta\_port:** socket port number of the station  
(*type=*int)

**Return Value**

msg - received message

```
send_msg_get_packetslost(server, id=0, intf_name=None, sta_ip=None,
sta_port=0)
```

requests number of packets lost by the interface. this number is always incremented since the interface activation

#### Parameters

**server:** tuple (ip, port\_num)  
**id:** message id  
**intf\_name:** name of the wireless interface  
*(type=str)*  
**sta\_ip:** ip address of the station that this message should be relayed to, if sta\_ip is different from None  
*(type=str)*  
**sta\_port:** socket port number of the station  
*(type=int)*

#### Return Value

msg - received message

## 59.2 Variables

| Name               | Description                                                                                                                                                               |
|--------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| msg_sent_received  | message structure common to all supported_messages messages<br><b>Value:</b> Struct('msg_sent_received', Embed(msg_default), Embed(fie...                                 |
| supported_messages | this module deals with multiple message types. these types are listed in supported_messages<br><b>Value:</b> [MSG_TYPE.MSG_GET_BYTESRECEIVED, MSG_TYPE.MSG_GET_BYTESSE... |



## 60 Module `ethanol.ssl_message.msg_server`

this is creates the server, that deals with clients (aps and stations) messages the messages implemented are mapped in `map_msg_to_procedure` main entry to this module is: `call run(server)`

**Author:** Henrique Duarte Moura

**Organization:** WINET/DCC/UFMG

**Copyright:** h3dema (c) 2017

**Contact:** henriquemoura@hotmail.com

**Since:** July 2015

**Status:** in development

**Requires:** `construct 2.5.2`

### 60.1 Functions

**`deal_with_client`**(*connstream*, *fromaddr*)

this function is called as a Thread to manage each connection

**Parameters**

**`connstream`:**

**`fromaddr`:**

**`run`**(*server*)

to use this module only call this method, providing a tuple with (server ip address, server port)

**Parameters**

**`server`:** (ip, port) tuple

### 60.2 Variables

| Name                              | Description                                                                                                  |
|-----------------------------------|--------------------------------------------------------------------------------------------------------------|
| <code>map_msg_to_procedure</code> | all message types supported<br><b>Value:</b> {MSG_TYPE.MSG_ASSOCIATION:<br>process_association, MSG_TYPE.... |

*continued on next page*

| Name              | Description                                                                                                                                   |
|-------------------|-----------------------------------------------------------------------------------------------------------------------------------------------|
| DEFAULT_CERT_PATH | path to the ssl certificate used in the secure socket connections<br><b>Value:</b><br><code>os.path.dirname(os.path.abspath(__file__))</code> |
| SSL_CERTIFICATE   | path and default name of the ssl certificate<br><b>Value:</b> <code>DEFAULT_CERT_PATH+ '/mycert.pem'</code>                                   |

## 61 Module `ethanol.ssl_message.msg_snr_power`

implements the following messages:

- \* `get_snr`: `MSG_GET_SNR`
- \* `get_txpower`: `MSG_GET_TXPOWER`
- \* `set_txpower`: `MSG_SET_TXPOWER`

no process is implemented: the controller is not supposed to respond to these message

**Author:** Henrique Duarte Moura

**Organization:** WINET/DCC/UFMG

**Copyright:** h3dema (c) 2017

**Contact:** henriquemoura@hotmail.com

**Since:** July 2015

**Status:** in development

**Requires:** construct 2.5.2

### 61.1 Functions

```
get_snr_power(server, id=0, intf_name=None, sta_ip=None, sta_port=0,
m_type=None)
```

INTERVAL FUNCTION: DON'T CALL THIS METHOD.

#### Parameters

- server:** tuple (ip, port\_num)
- id:** message id
- intf\_name:** name of the wireless interface  
(*type=*str)
- sta\_ip:** ip address of the station that this message should be  
relayed to, if sta\_ip is different from None  
(*type=*str)
- sta\_port:** socket port number of the station  
(*type=*int)

#### Return Value

- msg - received message

---

**get\_snr**(*server*, *id*=0, *intf\_name*=None, *sta\_ip*=None, *sta\_port*=0)

obtain SNR

**Parameters**

**server:** tuple (ip, port\_num)  
**id:** message id  
**intf\_name:** name of the wireless interface  
(*type=str*)  
**sta\_ip:** ip address of the station that this message should be relayed to, if *sta\_ip* is different from None  
(*type=str*)  
**sta\_port:** socket port number of the station  
(*type=int*)

**Return Value**

msg - received message

---

**get\_txpower**(*server*, *id*=0, *intf\_name*=None, *sta\_ip*=None, *sta\_port*=0)

obtain txpower

**Parameters**

**server:** tuple (ip, port\_num)  
**id:** message id  
**intf\_name:** name of the wireless interface  
(*type=str*)  
**sta\_ip:** ip address of the station that this message should be relayed to, if *sta\_ip* is different from None  
(*type=str*)  
**sta\_port:** socket port number of the station  
(*type=int*)

**Return Value**

msg - received message

```
set_txpower(server, id=0, intf_name=None, sta_ip=None, sta_port=0,
txpower=None)
```

set the txpower for the wireless interface

#### Parameters

**server:** tuple (ip, port\_num)  
**id:** message id  
**intf\_name:** name of the wireless interface  
*(type=str)*  
**sta\_ip:** ip address of the station that this message should be  
relayed to, if sta\_ip is different from None  
*(type=str)*  
**sta\_port:** socket port number of the station  
*(type=int)*

## 61.2 Variables

| Name          | Description                                                                   |
|---------------|-------------------------------------------------------------------------------|
| msg_snr_power | <b>Value:</b> Struct('msg_snr_power',<br>Embed(msg_default), Embed(field_i... |

## 62 Module *ethanol.ssl\_message.msg\_ssid*

implements the following messages:

\* `get_ssid`

no process is implemented: the controller is not supposed to respond to these message

**Author:** Henrique Duarte Moura

**Organization:** WINET/DCC/UFMG

**Copyright:** h3dema (c) 2017

**Contact:** henriquemoura@hotmail.com

**Since:** July 2015

**Status:** in development

**Requires:** construct 2.5.2

### 62.1 Functions

`get_ssid(server, id=0, intf_name=[], sta_ip=None, sta_port=0)`

returns the value None equals an error has ocured (or no interface found)

#### Parameters

|                   |                                                                                                                              |
|-------------------|------------------------------------------------------------------------------------------------------------------------------|
| <b>server:</b>    | tuple (ip, port_num)                                                                                                         |
| <b>id:</b>        | message id                                                                                                                   |
| <b>intf_name:</b> | names of the wireless interface<br>( <i>type=list of str</i> )                                                               |
| <b>sta_ip:</b>    | ip address of the station that this message should be<br>relayed to, if sta_ip is different from None<br>( <i>type=str</i> ) |
| <b>sta_port:</b>  | socket port number of the station<br>( <i>type=int</i> )                                                                     |

#### Return Value

msg - received message

### 62.2 Variables

| Name      | Description                                                                      |
|-----------|----------------------------------------------------------------------------------|
| ssid_info | <b>Value:</b> Struct('ssid_info',<br>Embed(field_intf_name),<br>Embed(field_s... |
| msg_ssid  | <b>Value:</b> Struct('msg_ssid',<br>Embed(msg_default),<br>Embed(field_statio... |

## 63 Module `ethanol.ssl_message.msg_sta_link_information`

implements the following messages:

\* `get_sta_link_info`

no process is implemented: the controller is not supposed to respond to these message

**Author:** Henrique Duarte Moura

**Organization:** WINET/DCC/UFMG

**Copyright:** h3dema (c) 2017

**Contact:** henriquemoura@hotmail.com

**Since:** July 2015

**Status:** in development

**Requires:** construct 2.5.2

### 63.1 Functions

```
get_sta_link_info(server, id=0, sta_ip=None, sta_port=0,  
intf_name=None)
```

returns three values: mac\_addr, ssid, frequency None equals an error has occurred (or no interface found)

#### Parameters

**server:** tuple (ip, port\_num)  
**id:** message id  
**intf\_name:** names of the wireless interface  
(*type=list of str*)  
**sta\_ip:** ip address of the station that this message should be relayed to, if sta\_ip is different from None  
(*type=str*)  
**sta\_port:** socket port number of the station  
(*type=int*)

#### Return Value

msg - received message

**To Do:** Nao eh necessario retornar intf\_name



## 63.2 Variables

| Name              | Description                                                                   |
|-------------------|-------------------------------------------------------------------------------|
| msg_sta_link_info | <b>Value:</b> Struct('msg_sta_link_info',<br>Embed(msg_default), Embed(fie... |

## 64 Module `ethanol.ssl_message.msg_sta_statistics`

implements the following messages:

\* `get_ssid`

no process is implemented: the controller is not supposed to respond to these message

**Author:** Henrique Duarte Moura

**Organization:** WINET/DCC/UFGM

**Copyright:** h3dema (c) 2017

**Contact:** henriquemoura@hotmail.com

**Since:** July 2015

**Status:** in development

**Requires:** construct 2.5.2

### 64.1 Functions

```
get_sta_statistics(server, id=0, intf_name=None, sta_ip=None,
sta_port=0)
```

returns the value None equals an error has occurred (or no interface found)

#### Parameters

**server:** tuple (ip, port\_num)  
**id:** message id  
**intf\_name:** names of the wireless interface  
(*type=list of str*)  
**sta\_ip:** ip address of the station that this message should be  
relayed to, if sta\_ip is different from None  
(*type=str*)  
**sta\_port:** socket port number of the station  
(*type=int*)

#### Return Value

msg - received message

### 64.2 Variables

| Name               | Description                                                                      |
|--------------------|----------------------------------------------------------------------------------|
| field_time_stamp   | <b>Value:</b> Struct('time_stamp',<br>SLInt32('time_stamp_size'), If(lambd...    |
| stats_field        | <b>Value:</b> Struct('stats',<br>Embed(field_mac_addr),<br>Embed(field_intf_n... |
| msg_sta_statistics | <b>Value:</b> Struct('msg_sta_statistics',<br>Embed(msg_default), Embed(fi...    |

## 65 Module *ethanol.ssl\_message.msg\_station\_trigger\_transition*

implements the following messages:

\* *station\_trigger\_transition*

no process is implemented: the controller is not supposed to respond to these message

**Author:** Henrique Duarte Moura

**Organization:** WINET/DCC/UFMG

**Copyright:** h3dema (c) 2017

**Contact:** henriquemoura@hotmail.com

**Since:** July 2015

**Status:** in development

**Requires:** construct 2.5.2

### 65.1 Functions

```
station_trigger_transition(server, id=0, sta_ip=None, sta_port=0,
sta_mac=None, intf_name=None, mac_new_ap=None)
```

sendo command to station to change to a new ap

**Parameters**

**server:** tuple (ip, port\_num)

**id:** message id

### 65.2 Variables

| Name                                        | Description                                                                                                                                 |
|---------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------|
| <code>field_mac_new_ap</code>               | handles a mac address field for the new ap (a C char * field)<br><b>Value:</b> Struct('mac_new_ap', SLInt32('mac_new_ap_size'), If(lambd... |
| <code>msg_station_trigger_transition</code> | message structure common to all supported_messages messages<br><b>Value:</b> Struct('msg_station_trigger_transition', Embed(msg_defaul...   |

## 66 Module *ethanol.ssl\_message.msg\_statistics*

implements the following messages:

\* `send_msg_get_statistics`

no process is implemented: the controller is not supposed to respond to these message

**Author:** Henrique Duarte Moura

**Organization:** WINET/DCC/UFMG

**Copyright:** h3dema (c) 2017

**Contact:** henriquemoura@hotmail.com

**Since:** July 2015

**Status:** in development

**Requires:** construct 2.5.2

### 66.1 Functions

```
send_msg_get_statistics(server, id=0, intf_name=None, sta_ip=None,
                        sta_port=0)
```

---

INTERNAL FUNCTION

returns the statistics using a `dict()` with 9 fields

**Parameters**

**server:** tuple (ip, port\_num)  
**id:** message id  
**intf\_name:** names of the wireless interface  
(*type=list of str*)  
**sta\_ip:** ip address of the station that this message should be  
relayed to, if `sta_ip` is different from `None`  
(*type=str*)  
**sta\_port:** socket port number of the station  
(*type=int*)

**Return Value**

`msg` - received message

## 66.2 Variables

| Name             | Description                                                                                                                                       |
|------------------|---------------------------------------------------------------------------------------------------------------------------------------------------|
| field_time_stamp | <b>Value:</b> Struct('time_stamp',<br>SLInt32('time_stamp_size'), If(lambd...                                                                     |
| msg_statistics   | message structure common to all supported<br>statistics messages<br><b>Value:</b> Struct('msg_statistics',<br>Embed(msg_default), Embed(field_... |

## 67 Module *ethanol.ssl\_message.msg\_tos*

implements the following messages:

\* *msg\_tos\_cleanall*

no process is implemented: the controller is not supposed to respond to these message

**Author:** Henrique Duarte Moura

**Organization:** WINET/DCC/UFMG

**Copyright:** h3dema (c) 2017

**Contact:** henriquemoura@hotmail.com

**Since:** July 2015

**Status:** in development

**Requires:** construct 2.5.2

### 67.1 Functions

|                                                            |
|------------------------------------------------------------|
| <b><i>tos_cleanall</i></b> ( <i>server</i> , <i>id</i> =0) |
| <i>msg_tos_cleanall</i> uptime                             |
| <b>Parameters</b>                                          |
| <i>server</i> : tuple (ip, port_num)                       |
| <i>id</i> : message id                                     |
| <b>Return Value</b>                                        |
| nothing                                                    |

|                                                                                                                                                                                                        |
|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b><i>tos_add</i></b> ( <i>server</i> , <i>msg_id</i> =0, <i>intf_name</i> =None, <i>proto</i> =None, <i>sip</i> =None, <i>sport</i> =None, <i>dip</i> =None, <i>dport</i> =None, <i>wmm_class</i> =0) |
| add TOS rule                                                                                                                                                                                           |
| <b>Parameters</b>                                                                                                                                                                                      |
| <i>server</i> : tuple (ip, port_num)                                                                                                                                                                   |
| <i>msg_id</i> : message id                                                                                                                                                                             |
| <b>Return Value</b>                                                                                                                                                                                    |
| nothing                                                                                                                                                                                                |

```
tos_replace(server, msg_id=0, rule_id=-1, intf_name=None, proto=None,
sip=None, sport=None, dip=None, dport=None, wmm_class=0)
```

`msg_tos_cleanall` uptime

#### Parameters

**server:** tuple (ip, port\_num)

**id:** message id

#### Return Value

nothing

## 67.2 Variables

| Name                          | Description                                                                                                               |
|-------------------------------|---------------------------------------------------------------------------------------------------------------------------|
| <code>msg_tos_cleanall</code> | message to clear mange rules<br><b>Value:</b> Struct('msg_tos_cleanall',<br>Embed(msg_default),)                          |
| <code>msg_tos</code>          | message to add or replace mange rules<br><b>Value:</b> Struct('msg_tos',<br>Embed(msg_default),<br>SLInt32('rule_id'),... |



## 68 Module *ethanol.ssl\_message.msg\_uptime*

implements the following messages:

\* *get\_uptime*

no process is implemented: the controller is not supposed to respond to these message

**Author:** Henrique Duarte Moura

**Organization:** WINET/DCC/UFMG

**Copyright:** h3dema (c) 2017

**Contact:** henriquemoura@hotmail.com

**Since:** July 2015

**Status:** in development

**Requires:** construct 2.5.2

### 68.1 Functions

***get\_uptime***(*server*, *id*=0)

get uptime

**Parameters**

**server:** tuple (ip, port\_num)

**id:** message id

**Return Value**

msg - received message value (bytes or packets received or sent or lost)

### 68.2 Variables

| Name       | Description                                                                                                                                |
|------------|--------------------------------------------------------------------------------------------------------------------------------------------|
| msg_uptime | message structure common to all supported_messages messages<br><b>Value:</b> Struct('msg_uptime', Embed(msg_default), LFloat64('uptime...' |

## 69 Module `ethanol.ssl_message.msg_wlan_info`

implements: \* `req_wlan_info()`: MSG\_WLAN\_INFO

**Author:** Henrique Duarte Moura

**Organization:** WINET/DCC/UFGM

**Copyright:** h3dema (c) 2017

**Contact:** henriquemoura@hotmail.com

**Since:** July 2015

**Status:** in development

**Requires:** construct 2.5.2

### 69.1 Functions

```
req_wlan_info(server, id=0, intf_name_list=None, sta_ip=None,
               sta_port=0)
```

#### Parameters

|                        |                                                                                                                              |
|------------------------|------------------------------------------------------------------------------------------------------------------------------|
| <b>server:</b>         | tuple (ip, port_num)                                                                                                         |
| <b>id:</b>             | message id                                                                                                                   |
| <b>intf_name_list:</b> | names of the wireless interface<br>( <i>type=list of str</i> )                                                               |
| <b>sta_ip:</b>         | ip address of the station that this message should<br>be relayed to, if sta_ip is different from None<br>( <i>type=str</i> ) |
| <b>sta_port:</b>       | socket port number of the station<br>( <i>type=int</i> )                                                                     |

#### Return Value

msg - received message

### 69.2 Variables

| Name       | Description                                                                                                         |
|------------|---------------------------------------------------------------------------------------------------------------------|
| wlan_entry | information about a wifi interface<br><b>Value:</b> Struct('wlan_entry',<br>SLInt32('ifindex'), Embed(field_intf... |

*continued on next page*

---

| Name          | Description                                                                   |
|---------------|-------------------------------------------------------------------------------|
| msg_wlan_info | <b>Value:</b> Struct('msg_wlan_info',<br>Embed(msg_default), Embed(field_s... |

## 70 Package ethanol.tos

This package contains some components to implement Ethanol API.  
ethanol should run as a pox module

sample command call:

```
python ./pox.py forwarding.l2_learning ethanol.server
```

ethanol.server is the ~/ethanol/python/server.py file

you must create a symbolic link inside pox subtree, like:

```
cd ~/ethanol/pox/pox
```

```
ln ~/ethanol/python ethanol
```

### 70.1 Modules

- **usecase\_\_tos**: This is a module that runs inside POX.  
(Section 71, p. 159)

### 70.2 Variables

| Name        | Description        |
|-------------|--------------------|
| __package__ | <b>Value:</b> None |

## 71 Module *ethanol.tos.usecase\_\_tos*

This is a module that runs inside POX.  
 It checks if ethanol is running,  
 if ethanol is running, this module can also run

To run this module use:

```
cd pox
```

```
python pox.py log --format="[(asctime)s] %(message)s" --datefmt="%Y-%m-%dT%H:%M:%S" et
```

### 71.1 Functions

```
set_app_paths(paths=[""])
```

set python's system path so we can call our functions :param list paths  
 contains the relative path for our functions

```
is_ethanol_loaded(module_name='pox.ethanol.server')
```

verifies if ethanol module is loaded

Keyword Arguments:

module\_name {str} -- name of the ethanol module (default: {'pox.ethanol.server'})

Returns:

[bool] -- True if the module is loaded

```
videocapture_traffic(sta, access_class=AC_BE, rule_id=1)
```

```
default_setup(sta, access_class=AC_BE)
```

call ethanol, to set new value

```
launch(ap_ip='150.164.10.90', sta_ip='150.164.10.22',  
sta_intf_name='wlan0', hostname='0.0.0.0', host_port=50000)
```

sta\_ip contains arpias' wlan0 ip socket\_address (video capture)

### 71.2 Variables

| Name  | Description                                                                         |
|-------|-------------------------------------------------------------------------------------|
| vap   | <b>Value:</b> None                                                                  |
| log   | this is the path of this python file<br><b>Value:</b> <code>core.getLogger()</code> |
| AC_BK | <b>Value:</b> 1                                                                     |
| AC_BE | <b>Value:</b> 3                                                                     |
| AC_VI | <b>Value:</b> 5                                                                     |
| AC_VO | <b>Value:</b> 7                                                                     |
| rule1 | <b>Value:</b> {'rule_id': 1, 'intf_name':<br>'wlan0', 'proto': 'tcp', 'sip...       |
| rule2 | <b>Value:</b> {'rule_id': 2, 'intf_name':<br>'wlan0', 'proto': 'udp', 'sip...       |

## **72   Script script-produce\_\_doc\_\_sh**

## Index

- ethanol (*package*), 2–5
  - ethanol.client\_test (*module*), 6
    - ethanol.client\_test.launch (*function*), 6
    - ethanol.client\_test.msg\_acs (*function*), 6
  - ethanol.ethanol (*package*), 7
    - ethanol.ethanol.ap (*module*), 8–12
    - ethanol.ethanol.device (*module*), 13–17
    - ethanol.ethanol.network (*module*), 18–20
    - ethanol.ethanol.radio (*module*), 21–24
    - ethanol.ethanol.station (*module*), 25–27
    - ethanol.ethanol.switch (*module*), 28–29
    - ethanol.ethanol.vap (*module*), 30–34
  - ethanol.events (*package*), 35–36
    - ethanol.events.events (*module*), 37–39
    - ethanol.events.tests (*package*), 40
  - ethanol.graph\_coloring (*package*), 47
    - ethanol.graph\_coloring.exact\_color (*module*), 48
  - ethanol.ovsdb (*package*), 49
    - ethanol.ovsdb.ovsdb (*module*), 50–51
  - ethanol.server (*module*), 52–53
    - ethanol.server.ethanol\_ap\_server (*class*), 52–53
    - ethanol.server.launch (*function*), 52
    - ethanol.server.run\_server (*function*), 52
  - ethanol.ssl\_message (*package*), 54–56
    - ethanol.ssl\_message.enum (*module*), 57
    - ethanol.ssl\_message.msg\_acs (*module*), 58–59
    - ethanol.ssl\_message.msg\_ap\_broadcastssid (*module*), 60–61
    - ethanol.ssl\_message.msg\_ap\_ctsprotection\_enabled (*module*), 62–63
    - ethanol.ssl\_message.msg\_ap\_dtiminterval (*module*), 64–65
    - ethanol.ssl\_message.msg\_ap\_frameburstenabled (*module*), 66–67
    - ethanol.ssl\_message.msg\_ap\_guardinterval (*module*), 68–69
    - ethanol.ssl\_message.msg\_ap\_in\_range (*module*), 70–71
    - ethanol.ssl\_message.msg\_ap\_interferencemap (*module*), 72
    - ethanol.ssl\_message.msg\_ap\_modes (*module*), 73
    - ethanol.ssl\_message.msg\_ap\_rtsthreshold (*module*), 74–75
    - ethanol.ssl\_message.msg\_ap\_ssid (*module*), 76–77
    - ethanol.ssl\_message.msg\_association (*module*), 78–79
    - ethanol.ssl\_message.msg\_beacon\_interval (*module*), 80–81
    - ethanol.ssl\_message.msg\_bitrates (*module*), 82–83
    - ethanol.ssl\_message.msg\_bye (*module*), 84–85
    - ethanol.ssl\_message.msg\_changed\_ap (*module*), 86–87
    - ethanol.ssl\_message.msg\_channelinfo (*module*), 88–89
    - ethanol.ssl\_message.msg\_channels (*module*), 90–92
    - ethanol.ssl\_message.msg\_common (*module*), 93–95
    - ethanol.ssl\_message.msg\_core (*module*), 96–97
    - ethanol.ssl\_message.msg\_enabled (*module*), 98–99
    - ethanol.ssl\_message.msg\_error (*module*), 100
    - ethanol.ssl\_message.msg\_frequency (*module*), 101–102
    - ethanol.ssl\_message.msg\_handle\_snr (*module*), 103–105
    - ethanol.ssl\_message.msg\_hello (*module*), 106–107
    - ethanol.ssl\_message.msg\_hostapd\_conf (*module*), 108–109
    - ethanol.ssl\_message.msg\_interfaces (*module*), 110–111



- ethanol.ssl\_message.msg\_log (*module*), 112
- ethanol.ssl\_message.msg\_mean\_sta\_stats (*module*), 113–116
- ethanol.ssl\_message.msg\_memcpu (*module*), 117–118
- ethanol.ssl\_message.msg\_metric (*module*), 119–120
- ethanol.ssl\_message.msg\_mlme (*module*), 121–123
- ethanol.ssl\_message.msg\_mtu\_qlen (*module*), 124–125
- ethanol.ssl\_message.msg\_ping (*module*), 126–127
- ethanol.ssl\_message.msg\_powersave (*module*), 128–129
- ethanol.ssl\_message.msg\_preamble (*module*), 130–131
- ethanol.ssl\_message.msg\_radio\_wlans (*module*), 132–133
- ethanol.ssl\_message.msg\_sent\_received (*module*), 134–138
- ethanol.ssl\_message.msg\_server (*module*), 139–140
- ethanol.ssl\_message.msg\_snr\_power (*module*), 141–143
- ethanol.ssl\_message.msg\_ssid (*module*), 144–145
- ethanol.ssl\_message.msg\_sta\_link\_information (*module*), 146–147
- ethanol.ssl\_message.msg\_sta\_statistics (*module*), 148–149
- ethanol.ssl\_message.msg\_station\_trigger\_transition (*module*), 150
- ethanol.ssl\_message.msg\_statistics (*module*), 151–152
- ethanol.ssl\_message.msg\_tos (*module*), 153–154
- ethanol.ssl\_message.msg\_uptime (*module*), 155
- ethanol.ssl\_message.msg\_wlan\_info (*module*), 156–157
- ethanol.tos (*package*), 158
- ethanol.tos.usecase\_tos (*module*), 159–160
- script-produce\_doc\_sh (*script*), 161