
openvpn-status Documentation

Release 0.2.2

Jiangge Zhang

Jan 11, 2023

Contents

1	Table Of Contents	3
1.1	User's Guide	3
1.2	API Reference	4
1.3	Changelog	7
1.4	Authors	7
	Python Module Index	9
	Index	11

openvpn-status is a Python library. It parses OpenVPN status log and turns it into Python data structure for you.

CHAPTER 1

Table Of Contents

1.1 User's Guide

It is compatible with Python 2.7, 3.6 to 3.10, and PyPy.

1.1.1 Installation

```
pip install openvpn-status
```

Don't forget to put it in `setup.py / requirements.txt`.

1.1.2 Getting Started

You could configure your OpenVPN server to log for client status. In usual it could be achieved by adding `status /path/to/openvpn-status.log` line to `/etc/openvpn/openvpn.conf`. For example:

```
proto udp
port 1194
dev tun0
status /var/run/openvpn-status.log
```

Once OpenVPN server running, the log file will be created and written. It looks like:

```
OpenVPN CLIENT LIST
Updated,Thu Jun 18 08:12:15 2015
Common Name,Real Address,Bytes Received,Bytes Sent,Connected Since
foo@example.com,10.10.10.10:49502,334948,1973012,Thu Jun 18 04:23:03 2015
bar@example.com,10.10.10.10:64169,1817262,28981224,Thu Jun 18 04:08:39 2015
ROUTING TABLE
Virtual Address,Common Name,Real Address,Last Ref
192.168.255.134,foo@example.com,10.10.10.10:49502,Thu Jun 18 08:12:09 2015
```

(continues on next page)

(continued from previous page)

```
192.168.255.126,bar@example.com,10.10.10.10:64169,Thu Jun 18 08:11:55 2015
GLOBAL STATS
Max bcst/mcast queue length,0
END
```

Now we could parse log file with this library:

```
from openvpn_status import parse_status

with open('/var/run/openvpn-status.log') as logfile:
    status = parse_status(logfile.read())

print(status.updated_at) # datetime.datetime(2015, 6, 18, 8, 12, 15)

foo_client = status.client_list['169.254.0.1']
print(foo_client.common_name) # foo@example.com
print(foo_client.bytes_received) # 334.9 kB
print(foo_client.bytes_sent) # 2.0 MB
print(int(foo_client.bytes_sent)) # 2097152
```

More details are in the [API reference](#).

1.1.3 Contributing

If you want to report bugs or request features, please feel free to open issues on [GitHub](#).

Of course, pull requests are always welcome.

1.2 API Reference

1.2.1 Models

class openvpn_status.models.Status

The OpenVPN status model.

client_list

(OrderedDict)

The list of connected clients. The dictionary items have form of (*client.real_address*, *client*). See also *Client*.

routing_table

Type OrderedDict

The list of routing table. The dictionary items have form of (*routing.virtual_address*, *routing*). See also *Routing*.

global_stats

Type GlobalStats

updated_at

Type datetime.datetime

The last updated time of log file in UTC.

class `openvpn_status.models.Client`

The OpenVPN client model.

common_name

Type `str`

The common name of OpenVPN client certificate. (e.g. *foo@example.com*)

real_address

Type `PeerAddress`

The real IP address and port of client.

bytes_received

Type `FileSize`

bytes_sent

Type `FileSize`

connected_since

Type `datetime.datetime`

The time in UTC since last connection established.

class `openvpn_status.models.Routing`

The OpenVPN routing model.

virtual_address

Type

- `ipaddress.IPv4Address` or `ipaddress.IPv6Address` for TUN mode
- `netaddr.EUI` (MAC address) for TAP mode
- `ipaddress.IPv4Network` or `ipaddress.IPv6Network` for *client-config-dir* and *iroute* enabled servers.

Read more about TUN and TAP: [Bridging vs. routing](#).

Read more about *client-config-dir* (CCD) and *iroute*: [Lans behind OpenVPN](#).

common_name

Same as `Client.common_name`

real_address

Same as `Client.real_address`

last_ref

Type `datetime.datetime`

class `openvpn_status.models.GlobalStats`

The OpenVPN global stats model.

max_bcast_mcast_queue_len

Type `int`

1.2.2 Parser

class `openvpn_status.parser.LogParser` (*lines*)

The parser for parsing OpenVPN status log.

This kind of parser is stateful. So the `LogParser.parse()` could be called once in the same instance of parser.

classmethod `fromstring` (*content*)

Creates a parser from content of log.

Parameters `content` (*str*) – The log content.

Returns The parser instance.

parse ()

Parses the status log.

Raises `ParsingError` – if syntax error found in the log.

Returns The `models.Status` with filled data.

exception `openvpn_status.parser.ParsingError`

1.2.3 Shortcuts

`openvpn_status.shortcuts.parse_status` (*status_log*, *encoding='utf-8'*)

Parses the status log of OpenVPN.

Parameters

- **status_log** (*str*) – The content of status log.
- **encoding** (*str*) – Optional. The encoding of status log.

Returns The instance of `models.Status`

1.2.4 Utilities

class `openvpn_status.utils.PeerAddress`

The address of peer entity.

Parameters

- **host** (`IPv4Address` or `IPv6Address`) – The host address of peer entity.
- **port** (*int*) – The port of peer entity.

class `openvpn_status.utils.FileSize`

The size of bytes.

humanize (***kwargs*)

Gets the human-friendly representation of file size.

Parameters `kwargs` – All keyword arguments will be passed to `humanize.filesize.naturalize()`.

1.3 Changelog

1.3.1 0.2.2 (2023-01-11)

- Fix GH-14: u'192.168.100.110C' does not appear to be an IPv4 or IPv6 address (GH-17, thanks to @mrevilme)
- Update for Python 3.10+ compatibility

1.3.2 0.2.1 (2019-08-27)

- Fix GH-11: Fix the parser which throws `ValueError` on interrupted stream.

1.3.3 0.2.0 (2017-10-20)

- Feature GH-1: Add support to TAP mode of OpenVPN servers by parsing virtual addresses as MAC and IP both.
- Feature GH-4: Add support to client-config-dir (ccd) and iroute.
- Fix GH-2: **BREAK-COMPATIBILITY** Use real or virtual addresses as the key of `client_list` and `routing_table`, instead of using common name.

1.3.4 0.1.1 (2016-06-29)

- Fix GH-3: The depended six must later than 1.9.0 because we need the “python_2_unicode_compatible” decorator.

1.3.5 0.1.0 (2015-06-19)

The first release.

1.4 Authors

- Jiangge Zhang <tonyseek@gmail.com>

O

`openvpn_status.models`, 4
`openvpn_status.parser`, 6
`openvpn_status.shortcuts`, 6
`openvpn_status.utils`, 6

B

bytes_received (*openvpn_status.models.Client* attribute), 5

bytes_sent (*openvpn_status.models.Client* attribute), 5

C

Client (*class in openvpn_status.models*), 4

client_list (*openvpn_status.models.Status* attribute), 4

common_name (*openvpn_status.models.Client* attribute), 5

common_name (*openvpn_status.models.Routing* attribute), 5

connected_since (*openvpn_status.models.Client* attribute), 5

F

FileSize (*class in openvpn_status.utils*), 6

fromstring() (*openvpn_status.parser.LogParser* class method), 6

G

global_stats (*openvpn_status.models.Status* attribute), 4

GlobalStats (*class in openvpn_status.models*), 5

H

humanize() (*openvpn_status.utils.FileSize* method), 6

L

last_ref (*openvpn_status.models.Routing* attribute), 5

LogParser (*class in openvpn_status.parser*), 6

M

max_bcast_mcast_queue_len (*openvpn_status.models.GlobalStats* attribute), 5

O

openvpn_status.models (*module*), 4

openvpn_status.parser (*module*), 6

openvpn_status.shortcuts (*module*), 6

openvpn_status.utils (*module*), 6

P

parse() (*openvpn_status.parser.LogParser* method), 6

parse_status() (*in module openvpn_status.shortcuts*), 6

ParsingError, 6

PeerAddress (*class in openvpn_status.utils*), 6

R

real_address (*openvpn_status.models.Client* attribute), 5

real_address (*openvpn_status.models.Routing* attribute), 5

Routing (*class in openvpn_status.models*), 5

routing_table (*openvpn_status.models.Status* attribute), 4

S

Status (*class in openvpn_status.models*), 4

U

updated_at (*openvpn_status.models.Status* attribute), 4

V

virtual_address (*openvpn_status.models.Routing* attribute), 5