TeamspeakStats Documentation

Release 1.4.1

Thor77

Apr 19, 2017

Contents

CHAPTER 1

Command Line Interface

```
$ tsstats --help
usage: tsstats [-h] [-c CONFIG] [--idmap IDMAP] [-l LOG] [-o OUTPUT] [-d]
               [-ds] [-nod] [-t TEMPLATE] [-dtf DATETIMEFORMAT]
               [-otth ONLINETIMETHRESHOLD]

A simple Teamspeak stats-generator, based solely on server-logs

optional arguments:
  -h, --help            show this help message and exit
  -c CONFIG, --config CONFIG
                        path to config
  --idmap IDMAP         path to id_map
  -l LOG, --log LOG     path to your logfile(s). pass a directory to use all
                        logfiles inside it
  -o OUTPUT, --output OUTPUT
                        path to the output-file
  -d, --debug           debug mode
  -ds, --debugstdout    write debug output to stdout
  -nod, --noonlinedc    don't add connect until now to onlinetime
  -t TEMPLATE, --template TEMPLATE
                        path to custom template
  -dtf DATETIMEFORMAT, --datetimeformat DATETIMEFORMAT
                        format of date/time-values (datetime.strftime)
  -otth ONLINETIMETHRESHOLD, --onlinetimethreshold ONLINETIMETHRESHOLD
                        threshold for displaying onlinetime (in seconds)
```

CHAPTER 2

Config

The configfile is using the .ini-format. Currently all settings are read from the `[General]`-section.

Key	Description
log	Path to TS3Server-logfile(s) (supports globbing)
output	Path to the location, where the generator will put the generated .html-file
idmap	Path to IdentMap
debug	debug mode
onlinedc	Add timedelta from last-connect until now to onlinetime for connected clients
template	Path to a custom template file (relative from `tsstats/` or absolute)
datetimeformat	Format of date/time-values used for render-timestamp and last online (using datetime.strftime)
onlinetimethreshold	Clients with an onlinetime below that threshold (in seconds) are hidden in the onlinetime-section

Example

`config.ini`

```
[General]
log = /usr/local/bin/teamspeak-server/logs/ts3server*_1.log
output = /var/www/html/stats.html
```

```
$ tsstats -c config.ini
```

CHAPTER 3

IdentMap

An IdentMap is used to map multiple (U)ID's of one client to one client. This can be useful, if a user creates multiple identities and you want to summarize all actions from all identities. To pass an IdentMap to TeamspeakStats, create your IdentMap as shown above and pass it to the cli:

```
tsstats --idmap <path to idmap.json>
```

TeamspeakStats' IdentMap-file is saved in json-format:

```
[
  {
    "primary_id": "1",
    "alternate_ids": ["2", "3", "4"]
  }
]
```

If you would pass this IdentMap to TeamspeakStats and your log would contain entries for clients with id's 1, 2, 3 and 4, your output will just show data for one client (1).

The format is flexible enough to support other arbitrary data to assist you in maintaining your IdentMap:

```
[
  {
    "name": "Friend 1",
    "primary_id": "1",
    "alternate_ids": ["2", "3", "4"]
  }
]
```

The parser will ignore all nodes other than the "primary_id" and "alternate_ids" nodes, allowing you to use them for whatever you want.

The original IdentMap format is still supported:

```
{
  '2': '1',
```

```
  '3': '1',
  '4': '1'
}
```

While it is less expressive, it is also less verbose.

CHAPTER 4

API

Log

class tsstats.log.**TimedLog**(*path*, *timestamp*)

path
Alias for field number 0

timestamp
Alias for field number 1

class tsstats.log.**Server**(*sid*, *clients*)

clients
Alias for field number 1

sid
Alias for field number 0

tsstats.log.**parse_logs**(*log_glob*, *ident_map=None*, *online_dc=True*, **args*, ***kwargs*)
parse logs from *log_glob*

Parameters

- **log_glob** (*str*) – path to server-logs (supports globbing)
- **ident_map** (*dict*) – identmap used for Client-initializations

Returns clients bundled by virtual-server

Return type *tsstats.log.Server*

Client

class tsstats.client.**Client**(*identifier*, *nick=None*)
Client provides high-level-access to a Teamspeak-Client

__init__(*identifier*, *nick=None*)
Initialize a new Client

Parameters **identifier** (*int or str*) – Identifier of the client

connect(*timestamp*)
Connect client at *timestamp*

Parameters **timestamp** (*int*) – time of connect

disconnect(*timestamp*)
Disconnect client at *timestamp*

Parameters **timestamp** (*int*) – time of disconnect

kick(*target*)
Let client kick *target*

Parameters **target** (Client) – client to kick

ban(*target*)
Let client ban *target*

Parameters **target** (Client) – client to ban

class tsstats.client.**Clients**(*ident_map=None*, **args*, ***kwargs*)
A high-level-interface to multiple Client-objects

__init__(*ident_map=None*, **args*, ***kwargs*)
Initialize a new Client-collection

Parameters **ident_map** (*dict*) – Identity-map (see *IdentMap*)

__iter__()
Yield all Client-objects from the collection

Template

class tsstats.template.**SortedClients**(*onlinetime*, *kicks*, *pkicks*, *bans*, *pbans*)

bans
Alias for field number 3

kicks
Alias for field number 1

onlinetime
Alias for field number 0

pbans
Alias for field number 4

pkicks
Alias for field number 2

tsstats.template.**prepare_clients**(*clients*, *onlinetime_threshold=-1*)
Prepare *clients* for rendering

sort them, clean their nick-history and convert onlinetime to string

Parameters

- **clients** (tsstats.client.Clients) – List of clients to prepare
- **onlinetime_threshold** – threshold for clients onlinetime

Returns *clients* sorted by onlinetime, kics, pkicks, bans and pbans

Return type *tsstats.template.SortedClients*

tsstats.template.**render_servers**(*servers*, *output*, *title='TeamspeakStats'*, *template='index.jinja2'*, *datetime_fmt='%x %X %Z'*, *onlinetime_threshold=-1*)
Render *servers*

Parameters

- **servers** ([tsstats.log.Server]) – list of servers to render
- **output** (*str*) – path to output-file
- **template_name** (*str*) – path to template-file
- **title** (*str*) – title of the resulting html-document
- **template_path** (*str*) – path to template-file
- **datetime_fmt** (*str*) – custom datetime-format
- **onlinetime_threshold** (*int*) – threshold for clients onlinetime

Config

tsstats.config.**load**(*path=None*)
parse config at *config_path*

Parameters **config_path** (*str*) – path to config-file

Returns values of config

Return type tuple

Exceptions

exception tsstats.exceptions.**InvalidConfiguration**
The configuration is invalid (either config-file or cli-args)

exception tsstats.exceptions.**InvalidLog**
Something impossible appeared at the logs, for example a disconnect before a connect

Utils

tsstats.utils.**sort_clients**(*clients*, *key_l*)
sort *clients* by *key*

Parameters

- **clients** (tsstats.client.Clients) – clients to sort
- **key_l** (*function*) – lambda/function returning the value of *key* for a client

Returns sorted *clients*

Return type list

tsstats.utils.**seconds_to_text**(*seconds*)
convert *seconds* to a text-representation

Parameters **seconds** (*int*) – seconds to convert

Returns *seconds* as text-representation

Return type str

tsstats.utils.**filter_threshold**(*clients*, *threshold*)
Filter clients by threshold

Parameters **clients** (*list*) – List of clients as returned by tsstats.utils.sort_clients

Returns Clients matching given threshold

Return type list

class tsstats.utils.**UTC**
Reimplementation of *timezone.utc* for Python2-Compatibility

tsstats.utils.**tz_aware_datime**(*datetime*, *timezone=<tsstats.utils.UTC object>*)
Make *datetime* aware of it's timezone (UTC by default)

Parameters

- **datetime** (*datetime.datetime*) – Target datetime
- **timezone** (*datetime.timezone*) – Target timezone

tsstats.utils.**transform_pretty_identmap**(*pretty_identmap*)
Transforms a list of client ID mappings from a more descriptive format to the traditional format of alternative IDs to actual ID.

Parameters **pretty_identmap** (*list*) – ID mapping in "nice" form

Returns ID mapping in simple key/value pairs

Return type dict

CHAPTER 5

Development

Contributing

Contributions are very welcome!

Before developing a new (possibly breaking) feature, please open an Issue about it first so we can discuss your idea and possible implementations.

Please read this document carefully before submitting your Pull Request to avoid failing CI tests.

Style

Your contribution should pass flake8 as well as isort.

Testing

There are unit tests for all parts of the project built with py.test. Besides `py.test` tests require `BeautifulSoup` for template-testing. Those requirements are listed in `testing_requirements.txt`:

```
$ pip install -r testing_requirement.txt
$ py.test tsstats/tests/
```

Versioning

TeamspeakStats uses Semantic Versioning. Please don't bump versions in your Pull Requests, though, we will do that after merging.

Python Versions

To keep the tool accessible and maintainable at the same time at least `Python 2.7` is required, so keep this in mind when using fancy new features from a recent Python version.

CHAPTER 6

TeamspeakStats

A simple Teamspeak stat-generator - based solely on server-logs

CHAPTER 7

Installation

- Install the package via PyPi `pip install tsstats`
- Clone this repo `git clone https://github.com/Thor77/TeamspeakStats` and install with `python setup.py install`
- Just use the package as is via `python -m tsstats [-h]`

CHAPTER 8

Usage

- Run the script `tsstats [-h]`
- Optionally create a config-file (see Configuration)
- The package works entirely off your Teamspeak server's logs, so that no ServerQuery account is necessary

CHAPTER 9

Example

```
tsstats -l /var/log/teamspeak3-server/ts3server*.log -o /var/www/tsstats.html
```

Parse logs matching `ts3server*.log` in `/var/log/teamspeak3-server` and write output to `/var/www/tsstats.html`.

Python Module Index

t

Index

Symbols

B

C

D

F

I

K

L

O

P

R

S

T

U