# vtapi3 - VirusTotal in Python

***Release 1.0.x***

**Mar 19, 2023**

# Contents

This guide describes how to use the vtapi3 Python module.

CHAPTER 1

# Introduction

## 1.1 Overview

vtapi3 is a Python module that implements the service API functions www.virustotal.com (3 versions) are available using the public key. For a detailed description of the API, see: https://developers.virustotal.com/v3.0/reference.

The vtapi3 module implements the following VirusTotal API functions:

**For files:**

- POST /files
- GET /files/upload_url
- GET /files/{id}
- POST /files/{id}/analyse
- GET /files/{id}/comments
- POST /files/{id}/comments
- GET /files/{id}/votes
- POST /files/{id}/votes
- GET /files/{id}/{relationship}
- GET /file_behaviours/{sandbox_id}/pcap

- GET /files/{id}/download_url (Added in version 1.2.0, requires a private key to access API functions)
- GET /files/{id}/download (Added in version 1.2.0, requires a private key to access API functions)

**For URLs:**

- POST /urls
- GET /urls/{id}
- POST /urls/{id}/analyse
- GET /urls/{id}/comments
- POST /urls/{id}/comments
- GET /urls/{id}/votes
- POST /urls/{id}/votes
- GET /urls/{id}/network_location

**For domains:**

- GET /domains/{domain}
- GET /domains/{domain}/comments
- POST /domains/{domain}/comments
- GET /domains/{domain}/{relationship}
- GET /domains/{domain}/votes
- POST /domains/{domain}/votes

**For IP-addresses:**

- GET /domains/{domain}
- GET /domains/{domain}/comments
- POST /domains/{domain}/comments
- GET /domains/{domain}/{relationship}
- GET /domains/{domain}/votes

- POST /domains/{domain}/votes

**File and URL analysis:**

- GET /analyses/{id}

### 1.1.1 Installation

```
$ pip install vtapi3
```

### 1.1.2 Usage

#### In python programs

#### Code

```
import json
from vtapi3 import VirusTotalAPIFiles, VirusTotalAPIError
   ...
vt_files = VirusTotalAPIFiles('<API key>')
try:
    result = vt_files.upload('<file path>')
except VirusTotalAPIError as err:
    print(err, err.err_code)
else:
    if vt_files.get_last_http_error() == vt_files.HTTP_OK:
        result = json.loads(result)
        result = json.dumps(result, sort_keys=False, indent=4)
        print(result)
    else:
        print('HTTP Error [' + str(vt_files.get_last_http_error()) +']')
    ...
```

#### Output

```
{
  "data": {
    "type": "analysis",
    "id": "NjY0MjRlOTFjMDIyYTkyNWM0NjU2NWQzYWNlMzFmZmI6MTQ3NTA0ODI3Nw=="
  }
}
```

#### From command line (added in version 1.1.0)

Before using the package from the command line, you must create an environment variable `VT_API_KEY` in which to place the value of the access key to the VirusTotal API functions.

```
$ python -m vtapi3  [-h] [-fid] [-fsr] [-far] [-hr] [-uid] [-usr] [-uar] [-ipr]
                    [-dr]
                    resource
```

#### Positional arguments

- `resource` - Object that you want to analyse in VirusTotal (file, URL, IP address or domain).

#### Optional arguments

- `-h`, `--help` - Show help message and exit.
- `-fid`, `--file-id` - Getting the identifier of the file for further analysis.
- `-fsr`, `--file-scan-report` - Getting a report on the results of scanning a file.
- `-far`, `--file-analyse-report` - Getting a report on the results of file analysis (enabled by default).
- `-hr`, `--hash-report` - Getting a report on the results of analyzing a file by its hash (SHA256, SHA1 or MD5).
- `-uid`, `--url-id` - Getting the identifier of the URL for further analysis.
- `-usr`, `--url-scan-report` - Getting a report on the results of scanning a URL.
- `-uar`, `--url-analyse-report` - Getting a report on the results of URL analysis.
- `-ipr`, `--ip-report` - Getting a report on the results of IP address analysis.
- `-dr`, `--domain-report` - Getting a report on the results of domain analysis.

### 1.1.3 License



### 1.1.4 Documentation

Documentation for using this package: https://virustotalapi3.readthedocs.io

# CHAPTER 2

# Release History

### 1.2.1 (10.04.2020)

- Fixed several bugs
- hanged the structure and composition of tests (the value of code coverage by tests is 99%).

### 1.2.0 (11.02.2020)

- hanged the structure of files and directories of the module.
- Added the `get_download_url()` and `get_download()` functions (`VirusTotalAPIFiles` class).
- The `main()` function was refactored in `__main__.py`.
- hanged the structure and composition (added tests for checking functions when the "Connection Error" error occurs) of tests (the value of code coverage by tests is 93%).

### 1.1.3 (7.02.2020)

- Fixed several bugs in `__main__.ru`

### 1.1.2 (5.02.2020)

- Fixed `__init__.py` (to ensure correct implementation of import).
- Added `__main__.py` (to improve the command line experience).

### 1.1.1 (4.02.2020)

- Fixed several errors in the `get_file_id_to_analyse()` and `get_url_id_to_analyse functions()`.

- Added VirusTotalAPIError(IO Error) exception in the `get_file_id()` and `upload()` functions of the VirusTotalAPIFiles class.

### 1.1.0 (3.02.2020)

- Added the ability to performance the package from the command line.

### 1.0.4 (1.02.2020)

- Fixing README.rst for better PYPI presentation.

### 1.0.3 (26.01.2020)

- Added a new attribute `_last_result` to the VirustotalAPI base class.
- Added a new method `get_last_result` to the VirustotalAPI base class.

### 1.0.2 (12.01.2020)

- Fixed errors in source comments.
- Fixing README.rst for better PYPI presentation.
- Fixing setup.py for better PYPI presentation.
- README.rst translated into English.
- Added two tests (`test_get_version_avi()` and `test_get_lost_http_error ()`) in test_vt_3.py

### 1.0.1 (08.01.2020)

- First release of vtapi3

# CHAPTER 3

# VirusTotalAPI

A base class for subclasses that implement methods for working with files, URLs, domain names, and IP addresses.

## 3.1 Attributes

### 3.1.1 base_url

The base URL for sending requests (str). Has the value: `https://www.virustotal.com/api/v3`.

### 3.1.2 headers

Request header containing API key (dict).

### 3.1.3 timeout

Server response timeout. A tuple that includes a timeout value for `connect` and a timeout value for `read`. If specify a single timeout value, it will be applied to both timeout `connect` and timeout `read`.

### 3.1.4 proxies

The Protocol and the URL of the proxy server (dict).

### 3.1.5 _version_api

VirusTotal API version (str). Has the value: `version 3`.

### 3.1.6 _last_http_error

HTTP status code of last operation (int).

### 3.1.7 _last_result

Result of the last execution of a subclass method of this class (added in version 1.0.3).

---

## 3.2 Constants

**HTTP error codes constants:**

- **HTTP_OK** - Function completed successfully.
- **HTTP_BAD_REQUEST_ERROR** - The API request is invalid or malformed. The message usually provides details about why the request is not valid.
- **HTTP_AUTHENTICATION_REQUIRED_ERROR** - The operation requires an authenticated user. Verify that you have provided your API key.
- **HTTP_FORBIDDEN_ERROR** - You are not allowed to perform the requested operation.
- **HTTP_NOT_FOUND_ERROR** - The requested resource was not found.
- **HTTP_ALREADY_EXISTS_ERROR** - The resource already exists.
- **HTTP_QUOTA_EXCEEDED_ERROR** - You have exceeded one of your quotas (minute, daily or monthly). Daily quotas are reset every day at 00:00 UTC.
- **HTTP_TRANSIENT_ERROR** - Transient server error. Retry might work.

---

## 3.3 Methods:

### 3.3.1 __init__(api_key, timeout, proxies)

Inits VirusTotalAPI.

**Arguments:**

- `api_key` : Your API key to access the functions of the service VirusTotal (str). How to get the api key is described in: https://developers.virustotal.com/v3.0/reference#getting-started.
- *timeout* : Server response timeout (int). Optional.
- *proxies* : The protocol and the URL of the proxy server (dict). Optional.

---

### 3.3.2 get_version_api()

Return the API version values.

**Arguments:**

None.

**Return value:**

String containing API version (`version 3`).

**Usage:**

```
import vtapi3
   ...
vt_api = vtapi3.VirusTotalAPI('<API key>')
version_api = vt_api.get_version_api()
print(version_api)
   ...
```

---

### 3.3.3 get_last_http_error()

Return the HTTP status code of last operation.

**Arguments:**

None.

**Return value:**

HTTP status code of last operation.

**Usage:**

```
import vtapi3
   ...
vt_api = vtapi3.VirusTotalAPI('<API key>')
http_error = vt_api.get_last_http_error()
print(http_error)
   ...
```

### 3.3.4 get_last_result()

Return the result of executing methods of subclasses of this class (added in version 1.0.3).

**Arguments:**

None.

**Return value:**

Result of the last execution of a subclass method of this class.

**Usage:**

```
import vtapi3
    ...
vt_api = vtapi3.VirusTotalAPI('<API key>')
result = vt_api.get_last_result()
print(result)
    ...
```

CHAPTER 4

# VirusTotalAPIFiles

The analysis new files and retrieving information about any file from the VirusTotal database methods are defined in the class.

## 4.1 Methods:

### 4.1.1 get_file_id(file_path, hash_alg)

Get SHA256, SHA1 or MD5 file identifier.

**Arguments:**

- `file_path` : Path to the file to be scanned (str).
- `hash_alg` : Necessary identifier (`sha256`, `sha1` or `md5`). The default value is `sha256`.

**Return value:**

The SHA256, SHA1 or MD5 identifier of the file (str).

**Exception:**

- *VirusTotalAPIError* (File not found): In case the file you want to upload to the server is not found.
- *VirusTotalAPIError* (Permission error): In case do not have access rights to the file.

**Usage:**

```
from vtapi3 import VirusTotalAPIFiles, VirusTotalAPIError
    ...
try:
    file_id = VirusTotalAPIFiles.get_file_id('<file path>')
except VirusTotalAPIError as err:
    print(err, err.err_code)
```

(continues on next page)

(continued from previous page)

```
else:
    print(file_id)
    ...
```

### 4.1.2 upload(file_path)

Upload and analyse a file.

**Arguments:**

- `file_path` : Path to the file to be scanned (str).

**Return value:**

The response from the server as a byte sequence.

**Exception:**

- *VirusTotalAPIError* (Connection error): In case of server connection errors.
- *VirusTotalAPIError* (Timeout error): If the response timeout from the server is exceeded.
- *VirusTotalAPIError* (File not found): In case the file you want to upload to the server is not found.
- *VirusTotalAPIError* (Permission error): In case do not have access rights to the file.

Usage:

```
from vtapi3 import VirusTotalAPIFiles, VirusTotalAPIError
   ...
vt_api_files = VirusTotalAPIFiles('<API key>')
try:
    result = vt_api_files.upload('<file path>')
except VirusTotalAPIError as err:
    print(err, err.err_code)
else:
    if vt_api_files.get_last_http_error() == vt_api_files.HTTP_OK:
        result = json.loads(result)
        result = json.dumps(result, sort_keys=False, indent=4)
        print(result)
    else:
        print('HTTP Error [' + str(vt_api_files.get_last_http_error()) +']')
    ...
```

**Note:** The total payload size can not exceed 32 MB. For uploading larger files see the *get_upload_url()* .

**Example response:**

When `_last_http_error` = `HTTP_OK` and after conversion to JSON, the response will look like this:

```
{
  "data": {
    "type": "analysis",
    "id": "NjY0MjRlOTFjMDIyYTkyNWM0NjU2NWQzYWNlMzFmZmI6MTQ3NTA0ODI3Nw=="
  }
}
```

### 4.1.3 get_upload_url()

Get a URL for uploading files larger than 32 MB.

**Arguments:**

None.

**Return value:**

The response from the server as a byte sequence.

**Exception:**

- *VirusTotalAPIError* (Connection error): In case of server connection errors.
- *VirusTotalAPIError* (Timeout error): If the response timeout from the server is exceeded.

**Usage:**

```
from vtapi3 import VirusTotalAPIFiles, VirusTotalAPIError
    ...
vt_api_files = VirusTotalAPIFiles('<API key>')
try:
    result = vt_api_files.get_upload_url()
except VirusTotalAPIError as err:
    print(err, err.err_code)
else:
    if vt_api_files.get_last_http_error() == vt_api_files.HTTP_OK:
        result = json.loads(result)
        result = json.dumps(result, sort_keys=False, indent=4)
        print(result)
    else:
        print('HTTP Error [' + str(vt_api_files.get_last_http_error()) +']')
    ...
```

**Example response:**

When `_last_http_error` = `HTTP_OK` and after conversion to JSON, the response will look like this:

```
{
  "data": "http://www.virustotal.com/_ah/upload/AMmfu6b-_DXUeFe36Sb3b0F4B8mH9Nb-
↪CHbRoUNVOPwG/"
}
```

### 4.1.4 get_report(file_id)

Retrieve information about a file.

**Arguments:**

- `fle_id` : SHA-256, SHA-1 or MD5 identifying the file (str).

**Return value:**

The response from the server as a byte sequence.

**Exception:**

- *VirusTotalAPIError* (Connection error): In case of server connection errors.
- *VirusTotalAPIError* (Timeout error): If the response timeout from the server is exceeded.

**Usage:**

```
from vtapi3 import VirusTotalAPIFiles, VirusTotalAPIError
    ...
vt_api_files = VirusTotalAPIFiles('<API key>')
try:
    result = vt_api_files.get_report('<file id>')
except VirusTotalAPIError as err:
    print(err, err.err_code)
else:
    if vt_api_files.get_last_http_error() == vt_api_files.HTTP_OK:
        result = json.loads(result)
        result = json.dumps(result, sort_keys=False, indent=4)
        print(result)
    else:
        print('HTTP Error [' + str(vt_api_files.get_last_http_error()) +']')
    ...
```

---

### 4.1.5 analyse(file_id)

Reanalyse a file already in VirusTotal.

**Arguments:**

- `fle_id` : SHA-256, SHA-1 or MD5 identifying the file (str).

**Return value:**

The response from the server as a byte sequence.

**Exception:**

- *VirusTotalAPIError* (Connection error): In case of server connection errors.
- *VirusTotalAPIError* (Timeout error): If the response timeout from the server is exceeded.

**Usage:**

```
from vtapi3 import VirusTotalAPIFiles, VirusTotalAPIError
    ...
vt_api_files = VirusTotalAPIFiles('<API key>')
try:
    result = vt_api_files.analyse('<file id>')
except VirusTotalAPIError as err:
    print(err, err.err_code)
else:
    if vt_api_files.get_last_http_error() == vt_api_files.HTTP_OK:
        result = json.loads(result)
        result = json.dumps(result, sort_keys=False, indent=4)
        print(result)
    else:
        print('HTTP Error [' + str(vt_api_files.get_last_http_error()) +']')
    ...
```

**Example response:**

When `_last_http_error` = `HTTP_OK` and after conversion to JSON, the response will look like this:

```
{
  "data": {
    "type": "analysis",
    "id": "NjY0MjRlOTFjMDIyYTkyNWM0NjU2NWQzYWNlMzFmZmI6MTQ3NTA0ODI3Nw=="
  }
}
```

---

### 4.1.6 get_comments(file_id, limit, cursor)

Retrieve comments for a file.

**Arguments:**

- `fle_id` : SHA-256, SHA-1 or MD5 identifying the file (str).
- `limit` : Maximum number of comments to retrieve (int). The default value is 10.
- `cursor` : Continuation cursor (str). The default value is ‘’.

**Return value:**

The response from the server as a byte sequence.

**Exception:**

- *VirusTotalAPIError* (Connection error): In case of server connection errors.
- *VirusTotalAPIError* (Timeout error): If the response timeout from the server is exceeded.

**Usage:**

```
from vtapi3 import VirusTotalAPIFiles, VirusTotalAPIError
   ...
vt_api_files = VirusTotalAPIFiles('<API key>')
try:
    result = vt_api_files.get_comments('<file id>', 5)
except VirusTotalAPIError as err:
    print(err, err.err_code)
else:
    if vt_api_files.get_last_http_error() == vt_api_files.HTTP_OK:
        result = json.loads(result)
        result = json.dumps(result, sort_keys=False, indent=4)
        print(result)
    else:
        print('HTTP Error [' + str(vt_api_files.get_last_http_error()) +']')
    ...
```

---

### 4.1.7 put_comments(file_id, text)

Add a comment to a file.

**Arguments:**

- `fle_id` : SHA-256, SHA-1 or MD5 identifying the file (str).
- `text` : Text of the comment (str). Any word starting with # in your comment's text will be considered a tag, and added to the comment's tag attribute.

**Return value:**

The response from the server as a byte sequence.

**Exception:**

- *VirusTotalAPIError* (Connection error): In case of server connection errors.
- *VirusTotalAPIError* (Timeout error): If the response timeout from the server is exceeded.

**Usage:**

```
from vtapi3 import VirusTotalAPIFiles, VirusTotalAPIError
    ...
vt_api_files = VirusTotalAPIFiles('<API key>')
try:
    result = vt_api_files.put_comment('<file id>', '<text of the comment>')
except VirusTotalAPIError as err:
    print(err, err.err_code)
else:
    if vt_api_files.get_last_http_error() == vt_api_files.HTTP_OK:
        result = json.loads(result)
        result = json.dumps(result, sort_keys=False, indent=4)
        print(result)
    else:
        print('HTTP Error [' + str(vt_api_files.get_last_http_error()) +']')
    ...
```

**Example response:**

When `_last_http_error` = `HTTP_OK` and after conversion to JSON, the response will look like this:

```
{
  "data": {
    "type": "comment",
    "id": "<comment's ID>",
    "links": {
      "self": "https://www.virustotal.com/api/v3/comments/<comment's ID>"
    },
    "attributes": {
      "date": 1521725475,
      "tags": ["ipsum"],
      "html": "Lorem #ipsum dolor sit ...",
      "text": "Lorem #ipsum dolor sit ...",
      "votes": {
        "abuse": 0,
        "negative": 0,
        "positive": 0
      }
    }
  }
}
```

### 4.1.8 get_votes(file_id, limit, cursor)

Retrieve votes for a file.

**Arguments:**

- `fle_id` : SHA-256, SHA-1 or MD5 identifying the file (str).
- `limit` : Maximum number of vites to retrieve (int). The default value is 10.
- `cursor` : Continuation cursor (str). The default value is ‘’.

**Return value:**

The response from the server as a byte sequence.

**Exception:**

- *VirusTotalAPIError* (Connection error): In case of server connection errors.
- *VirusTotalAPIError* (Timeout error): If the response timeout from the server is exceeded.

**Usage:**

```
from vtapi3 import VirusTotalAPIFiles, VirusTotalAPIError
    ...
vt_api_files = VirusTotalAPIFiles('<API key>')
try:
    result = vt_api_files.get_votes('<file id>', 5)
except VirusTotalAPIError as err:
    print(err, err.err_code)
else:
    if vt_api_files.get_last_http_error() == vt_api_files.HTTP_OK:
        result = json.loads(result)
        result = json.dumps(result, sort_keys=False, indent=4)
        print(result)
    else:
        print('HTTP Error [' + str(vt_api_files.get_last_http_error()) +']')
    ...
```

---

### 4.1.9 put_votes(file_id, malicious)

Add a vote to a file.

**Arguments:**

- `fle_id` : SHA-256, SHA-1 or MD5 identifying the file (str).
- `malicious` : Determines a malicious (True) or harmless (False) file (bool). The default value is `False`.

**Return value:**

The response from the server as a byte sequence.

**Exception:**

- *VirusTotalAPIError* (Connection error): In case of server connection errors.
- *VirusTotalAPIError* (Timeout error): If the response timeout from the server is exceeded.

**Usage:**

```
from vtapi3 import VirusTotalAPIFiles, VirusTotalAPIError
   ...
vt_api_files = VirusTotalAPIFiles('<API key>')
try:
    result = vt_api_files.put_votes('<file id>', True)
except VirusTotalAPIError as err:
    print(err, err.err_code)
else:
    if vt_api_files.get_last_http_error() == vt_api_files.HTTP_OK:
        result = json.loads(result)
        result = json.dumps(result, sort_keys=False, indent=4)
        print(result)
    else:
        print('HTTP Error [' + str(vt_api_files.get_last_http_error()) +']')
    ...
```

### 4.1.10 get_relationship(file_id, relationship, limit, cursor)

Retrieve objects related to a file.

**Arguments:**

- `fle_id` : SHA-256, SHA-1 or MD5 identifying the file (str).
- `relationship` : Relationship name (str). The default value is `/behaviours`. For more information, see https://developers.virustotal.com/v3.0/reference#files-relationships.
- `limit` : Maximum number of related objects to retrieve (int). The default value is 10.
- `cursor` : Continuation cursor (str). The default value is ‘’.

**Return value:**

The response from the server as a byte sequence.

**Exception:**

- *VirusTotalAPIError* (Connection error): In case of server connection errors.
- *VirusTotalAPIError* (Timeout error): If the response timeout from the server is exceeded.

**Usage:**

```
from vtapi3 import VirusTotalAPIFiles, VirusTotalAPIError
   ...
vt_api_files = VirusTotalAPIFiles('<API key>')
try:
    result = vt_api_files.get_relationship('<file id>', 'bundled_files')
except VirusTotalAPIError as err:
    print(err, err.err_code)
else:
    if vt_api_files.get_last_http_error() == vt_api_files.HTTP_OK:
        result = json.loads(result)
        result = json.dumps(result, sort_keys=False, indent=4)
        print(result)
    else:
        print('HTTP Error [' + str(vt_api_files.get_last_http_error()) +']')
    ...
```

### 4.1.11 get_behaviours(sandbox_id)

Get the PCAP for the sandbox.

**Arguments:**

- `sandbox_id` : Identifier obtained using the *get_relationship(file_id, relationship, limit, cursor)* method with the value of the `relationship` argument equal to `behaviours` (str).

**Return value:**

The response from the server as a byte sequence.

**Exception:**

- *VirusTotalAPIError* (Connection error): In case of server connection errors.
- *VirusTotalAPIError* (Timeout error): If the response timeout from the server is exceeded.

**Usage:**

```
from vtapi3 import VirusTotalAPIFiles, VirusTotalAPIError
    ...
vt_api_files = VirusTotalAPIFiles('<API key>')
try:
    result = vt_api_files.get_relationship('<file id>', 'bundled_files')
except VirusTotalAPIError as err:
    print(err, err.err_code)
else:
    if vt_api_files.get_last_http_error() == vt_api_files.HTTP_OK:
        result = json.loads(result)
        result = json.dumps(result, sort_keys=False, indent=4)
        print(result)
    else:
        print('HTTP Error [' + str(vt_api_files.get_last_http_error()) +']')
    ...
```

### 4.1.12 get_download_url(file_id)

Get a download URL for a file (added in version 1.2.0).

**Warning:** This function is only available for users with special privileges. You need a private key to access the VirusTotal API.

**Arguments:**

- `file_id` : SHA-256, SHA-1 or MD5 identifying the file (str).

**Return value:**

The response from the server as a byte sequence.

**Exception:**

- *VirusTotalAPIError* (Connection error): In case of server connection errors.
- *VirusTotalAPIError* (Timeout error): If the response timeout from the server is exceeded.

**Usage:**

```
from vtapi3 import VirusTotalAPIFiles, VirusTotalAPIError
    ...
vt_api_files = VirusTotalAPIFiles('<API key>')
try:
    result = vt_api_files.get_download_url('<file id>')
except VirusTotalAPIError as err:
    print(err, err.err_code)
else:
    if vt_api_files.get_last_http_error() == vt_api_files.HTTP_OK:
        result = json.loads(result)
        result = json.dumps(result, sort_keys=False, indent=4)
        print(result)
    else:
        print('HTTP Error [' + str(vt_api_files.get_last_http_error()) +']')
    ...
```

---

### 4.1.13 get_download(file_id)

Download a file (added in version 1.2.0).

**Warning:** This function is only available for users with special privileges. You need a private key to access the VirusTotal API.

**Arguments:**

- `file_id` : SHA-256, SHA-1 or MD5 identifying the file (str).

**Return value:**

The response from the server as a byte sequence.

**Exception:**

- *VirusTotalAPIError* (Connection error): In case of server connection errors.
- *VirusTotalAPIError* (Timeout error): If the response timeout from the server is exceeded.

**Usage:**

```
from vtapi3 import VirusTotalAPIFiles, VirusTotalAPIError
    ...
vt_api_files = VirusTotalAPIFiles('<API key>')
try:
    result = vt_api_files.get_download('<file id>')
except VirusTotalAPIError as err:
    print(err, err.err_code)
else:
    if vt_api_files.get_last_http_error() == vt_api_files.HTTP_OK:
        result = json.loads(result)
        result = json.dumps(result, sort_keys=False, indent=4)
```

(continues on next page)

(continued from previous page)

```
        print(result)
    else:
        print('HTTP Error [' + str(vt_api_files.get_last_http_error()) +']')
    ...
```

CHAPTER 5

# VirusTotalAPIUrls

The analysis new URLs and retrieving information about any URLs from the VirusTotal database methods are defined in the class.

## 5.1 Methods:

### 5.1.1 get_url_id_base64(url)

Get base64 encoded URL identifier.

**Arguments:**

- `url` : The URL for which you want to get the identifier (str).

**Return value:**

The identifier of the url, base64 encoded (str).

**Usage:**

```
from vtapi3 import VirusTotalAPIUrls
    ...
url_id = VirusTotalAPIUrls.get_url_id_base64('<url>')
print(url_id)
    ...
```

### 5.1.2 get_url_id_sha256(url)

Get the URL identifier as a SHA256 hash.

**Arguments:**

- `url` : The URL for which you want to get the identifier (str).

**Return value:**

The identifier of the url, SHA256 encoded (str).

**Usage:**

```
from vtapi3 import VirusTotalAPIUrls
    ...
url_id = VirusTotalAPIUrls.get_url_id_sha256('<url>')
print(url_id)
    ...
```

---

### 5.1.3 upload(url)

Upload URL for analysis.

**Arguments:**

- `url` : URL to be analyzed (str).

**Return value:**

The response from the server as a byte sequence.

**Exception:**

- *VirusTotalAPIError* (Connection error): In case of server connection errors.
- *VirusTotalAPIError* (Timeout error): If the response timeout from the server is exceeded.

**Usage:**

```
from vtapi3 import VirusTotalAPIUrls, VirusTotalAPIError
   ...
vt_api_urls = VirusTotalAPIUrls('<API key>')
try:
    result = vt_api_urls.upload('<url>')
except VirusTotalAPIError as err:
    print(err, err.err_code)
else:
    if vt_api_urls.get_last_http_error() == vt_api_urls.HTTP_OK:
        result = json.loads(result)
        result = json.dumps(result, sort_keys=False, indent=4)
        print(result)
    else:
        print('HTTP Error [' + str(vt_api_urls.get_last_http_error()) +']')
    ...
```

**Response structure:**

When `_last_http_error` = `HTTP_OK` and after conversion to JSON, the response structure will look like this:

```
{
  "data": {"id": "<string>", "type": "analysis"}
}
```

### 5.1.4 get_report(url_id)

Retrieve information about an URL.

**Arguments:**

- `url_id` : URL identifier (str). This identifier can adopt two forms: the SHA-256 of the canonized URL (method *get_url_id_sha256(url)* ), the string resulting from encoding the URL in base64 without the "=" padding (method *get_url_id_base64(url)* ).

**Return value:**

The response from the server as a byte sequence.

**Exception:**

- *VirusTotalAPIError* (Connection error): In case of server connection errors.
- *VirusTotalAPIError* (Timeout error): If the response timeout from the server is exceeded.

**Usage:**

```
from vtapi3 import VirusTotalAPIUrls, VirusTotalAPIError
    ...
vt_api_urls = VirusTotalAPIUrls('<API key>')
try:
    result = vt_api_urls.get_report('<url id>')
except VirusTotalAPIError as err:
    print(err, err.err_code)
else:
    if vt_api_urls.get_last_http_error() == vt_api_urls.HTTP_OK:
        result = json.loads(result)
        result = json.dumps(result, sort_keys=False, indent=4)
        print(result)
    else:
        print('HTTP Error [' + str(vt_api_urls.get_last_http_error()) +']')
    ...
```

**Response structure:**

When `_last_http_error` = `HTTP_OK` and after conversion to JSON, the response structure will look like this (for more information, see https://developers.virustotal.com/v3.0/reference#ip-object):

```
{
  "data": "<URL OBJECT>"
}
```

### 5.1.5 analyse(url_id)

Analyse an URL.

**Arguments:**

- `url_id` : URL identifier (str). This identifier can adopt two forms: the SHA-256 of the canonized URL (method *get_url_id_sha256(url)* ), the string resulting from encoding the URL in base64 without the "=" padding (method *get_url_id_base64(url)* ).

**Return value:**

The response from the server as a byte sequence.

**Exception:**

- *VirusTotalAPIError* (Connection error): In case of server connection errors.
- *VirusTotalAPIError* (Timeout error): If the response timeout from the server is exceeded.

**Usage:**

```
from vtapi3 import VirusTotalAPIUrls, VirusTotalAPIError
   ...
vt_api_urls = VirusTotalAPIUrls('<API key>')
try:
    result = vt_api_urls.analyse('<url id>')
except VirusTotalAPIError as err:
    print(err, err.err_code)
else:
    if vt_api_urls.get_last_http_error() == vt_api_urls.HTTP_OK:
        result = json.loads(result)
        result = json.dumps(result, sort_keys=False, indent=4)
        print(result)
    else:
        print('HTTP Error [' + str(vt_api_urls.get_last_http_error()) +']')
    ...
```

**Response structure:**

When `_last_http_error` = `HTTP_OK` and after conversion to JSON, the response structure will look like this:

```
{
  "data": {"id": "<string>", "type": "analysis"}
}
```

---

### 5.1.6 get_comments(url_id, limit, cursor)

Retrieve comments for an URL.

**Arguments:**

- `url_id` : URL identifier (str). This identifier can adopt two forms: the SHA-256 of the canonized URL (method *get_url_id_sha256(url)* ), the string resulting from encoding the URL in base64 without the "=" padding (method *get_url_id_base64(url)* ).
- `limit` : Maximum number of comments to retrieve (int). The default value is 10.
- `cursor` : Continuation cursor (str). The default value is ''.

**Return value:**

The response from the server as a byte sequence.

**Exception:**

- *VirusTotalAPIError* (Connection error): In case of server connection errors.
- *VirusTotalAPIError* (Timeout error): If the response timeout from the server is exceeded.

**Usage:**

```
from vtapi3 import VirusTotalAPIUrls, VirusTotalAPIError
    ...
vt_api_urls = VirusTotalAPIUrls('<API key>')
try:
    result = vt_api_urls.get_comments('<url id>', 5)
except VirusTotalAPIError as err:
    print(err, err.err_code)
else:
    if vt_api_urls.get_last_http_error() == vt_api_urls.HTTP_OK:
        result = json.loads(result)
        result = json.dumps(result, sort_keys=False, indent=4)
        print(result)
    else:
        print('HTTP Error [' + str(vt_api_urls.get_last_http_error()) +']')
    ...
```

---

### 5.1.7 put_comments(url_id, text)

Add a comment to a URL.

**Arguments:**

- `url_id` : URL identifier (str). This identifier can adopt two forms: the SHA-256 of the canonized URL (method *get_url_id_sha256(url)* ), the string resulting from encoding the URL in base64 without the "=" padding (method *get_url_id_base64(url)* ).
- `text` : Text of the comment (str). Any word starting with # in your comment's text will be considered a tag, and added to the comment's tag attribute.

**Return value:**

The response from the server as a byte sequence.

**Exception:**

- *VirusTotalAPIError* (Connection error): In case of server connection errors.
- *VirusTotalAPIError* (Timeout error): If the response timeout from the server is exceeded.

**Usage:**

```
from vtapi3 import VirusTotalAPIUrls, VirusTotalAPIError
    ...
vt_api_urls = VirusTotalAPIUrls('<API key>')
try:
    result = vt_api_urls.put_comment('<url id>', '<text of the comment>')
except VirusTotalAPIError as err:
    print(err, err.err_code)
else:
    if vt_api_urls.get_last_http_error() == vt_api_urls.HTTP_OK:
        result = json.loads(result)
        result = json.dumps(result, sort_keys=False, indent=4)
```

(continues on next page)

(continued from previous page)

```
        print(result)
    else:
        print('HTTP Error [' + str(vt_api_urls.get_last_http_error()) +']')
    ...
```

### 5.1.8 get_votes(url_id, limit, cursor)

Retrieve votes for a URL.

**Arguments:**

- `url_id` : URL identifier (str). This identifier can adopt two forms: the SHA-256 of the canonized URL (method *get_url_id_sha256(url)* ), the string resulting from encoding the URL in base64 without the "=" padding (method *get_url_id_base64(url)* ).
- `limit` : Maximum number of vites to retrieve (int). The default value is 10.
- `cursor` : Continuation cursor (str). The default value is ''.

**Return value:**

The response from the server as a byte sequence.

**Exception:**

- *VirusTotalAPIError* (Connection error): In case of server connection errors.
- *VirusTotalAPIError* (Timeout error): If the response timeout from the server is exceeded.

**Usage:**

```
from vtapi3 import VirusTotalAPIUrls, VirusTotalAPIError
   ...
vt_api_urls = VirusTotalAPIUrls('<API key>')
try:
    result = vt_api_urls.get_votes('<url id>', 5)
except VirusTotalAPIError as err:
    print(err, err.err_code)
else:
    if vt_api_urls.get_last_http_error() == vt_api_urls.HTTP_OK:
        result = json.loads(result)
        result = json.dumps(result, sort_keys=False, indent=4)
        print(result)
    else:
        print('HTTP Error [' + str(vt_api_urls.get_last_http_error()) +']')
    ...
```

### 5.1.9 put_votes(url_id, malicious)

Add a vote to a URL.

**Arguments:**

- `url_id` : URL identifier (str). This identifier can adopt two forms: the SHA-256 of the canonized URL (method *get_url_id_sha256(url)* ), the string resulting from encoding the URL in base64 without the "=" padding (method *get_url_id_base64(url)* ).
- `malicious` : Determines a malicious (True) or harmless (False) URL (bool). The default value is `False`.

**Return value:**

The response from the server as a byte sequence.

**Exception:**

- *VirusTotalAPIError* (Connection error): In case of server connection errors.
- *VirusTotalAPIError* (Timeout error): If the response timeout from the server is exceeded.

**Usage:**

```
from vtapi3 import VirusTotalAPIUrls, VirusTotalAPIError
    ...
vt_api_urls = VirusTotalAPIUrls('<API key>')
try:
    result = vt_api_urls.put_votes('<url id>', True)
except VirusTotalAPIError as err:
    print(err, err.err_code)
else:
    if vt_api_urls.get_last_http_error() == vt_api_urls.HTTP_OK:
        result = json.loads(result)
        result = json.dumps(result, sort_keys=False, indent=4)
        print(result)
    else:
        print('HTTP Error [' + str(vt_api_urls.get_last_http_error()) +']')
    ...
```

---

### 5.1.10 get_network_location(url_id)

Get the domain or IP address for a URL.

**Arguments:**

- `url_id` : URL identifier (str). This identifier can adopt two forms: the SHA-256 of the canonized URL (method *get_url_id_sha256(url)* ), the string resulting from encoding the URL in base64 without the "=" padding (method *get_url_id_base64(url)* ).

**Return value:**

The response from the server as a byte sequence.

**Exception:**

- *VirusTotalAPIError* (Connection error): In case of server connection errors.
- *VirusTotalAPIError* (Timeout error): If the response timeout from the server is exceeded.

**Usage:**

```
from vtapi3 import VirusTotalAPIUrls, VirusTotalAPIError
    ...
vt_api_urls = VirusTotalAPIUrls('<API key>')
try:
```

(continues on next page)

(continued from previous page)

```
    result = vt_api_urls.get_network_location('<url id>')
except VirusTotalAPIError as err:
    print(err, err.err_code)
else:
    if vt_api_urls.get_last_http_error() == vt_api_urls.HTTP_OK:
        result = json.loads(result)
        result = json.dumps(result, sort_keys=False, indent=4)
        print(result)
    else:
        print('HTTP Error [' + str(vt_api_urls.get_last_http_error()) +']')
    ...
```

**Response structure:**

When `_last_http_error` = `HTTP_OK` and after conversion to JSON, the response structure will look like this:

```
{
  "data": "<DOMAIN OBJECT> or <IP OBJECT>",
  "links": {"self": "<string>"}
}
```

### 5.1.11 get_relationship(url_id, relationship, limit, cursor)

Retrieve objects related to an URL.

**Arguments:**

- `url_id` : URL identifier (str). This identifier can adopt two forms: the SHA-256 of the canonized URL (method *get_url_id_sha256(url)* ), the string resulting from encoding the URL in base64 without the "=" padding (method *get_url_id_base64(url)* ).
- `relationship` : Relationship name (str). The default value is `/last_serving_ip_address`. For more information, see https://developers.virustotal.com/v3.0/reference#urls-relationships.
- `limit` : Maximum number of related objects to retrieve (int). The default value is 10.
- `cursor` : Continuation cursor (str). The default value is ''.

**Return value:**

The response from the server as a byte sequence.

**Exception:**

- *VirusTotalAPIError* (Connection error): In case of server connection errors.
- *VirusTotalAPIError* (Timeout error): If the response timeout from the server is exceeded.

**Usage:**

```
from vtapi3 import VirusTotalAPIUrls, VirusTotalAPIError
    ...
vt_api_urls = VirusTotalAPIUrls('<API key>')
try:
    result = vt_api_urls.get_relationship('<url id>', 'graphs')
except VirusTotalAPIError as err:
```

(continues on next page)

(continued from previous page)

```
    print(err, err.err_code)
else:
    if vt_api_urls.get_last_http_error() == vt_api_urls.HTTP_OK:
        result = json.loads(result)
        result = json.dumps(result, sort_keys=False, indent=4)
        print(result)
    else:
        print('HTTP Error [' + str(vt_api_urls.get_last_http_error()) +']')
    ...
```

CHAPTER 6

# VirusTotalAPIDomains

The retrieving information about any domain from the VirusTotal database methods are defined in the class.

## 6.1 Methods:

### 6.1.1 get_report(domain)

Retrieve information about an Internet domain.

**Arguments:**

- `domain` : Domain name (str).

**Return value:**

The response from the server as a byte sequence.

**Exception:**

- *VirusTotalAPIError* (Connection error): In case of server connection errors.
- *VirusTotalAPIError* (Timeout error): If the response timeout from the server is exceeded.

**Usage:**

```
from vtapi3 import VirusTotalAPIDomains, VirusTotalAPIError
    ...
vt_api_domains = VirusTotalAPIDomains('<API key>')
try:
    result = vt_api_domains.get_report('<domain>')
except VirusTotalAPIError as err:
    print(err, err.err_code)
else:
    if vt_api_domains.get_last_http_error() == vt_api_domains.HTTP_OK:
```

(continues on next page)

(continued from previous page)

```
        result = json.loads(result)
        result = json.dumps(result, sort_keys=False, indent=4)
        print(result)
    else:
        print('HTTP Error [' + str(vt_api_domains.get_last_http_error()) +']')
    ...
```

**Example response:**

When `_last_http_error` = `HTTP_OK` and after conversion to JSON, the response will look like this:


```
{
  "data": {
    "type": "domain",
    "id": "virustotal.com",
    "links": {
      "self": "https://virustotal.com/api/v3/domains/virustotal.com"
    },
    "attributes": {
      "categories": {
        "Alexa": "services",
        "BitDefender": "computersandsoftware",
        "TrendMicro": "computers internet",
        "Websense ThreatSeeker": "computer security"
      },
      "creation_date": 1032308169,
      "last_update_date": 1389199030,
      "registrar": "MarkMonitor Inc.",
      "reputation": 13,
      "total_votes": {
        "harmless": 2,
        "malicious": 0
      },
        "whois": "Domain Name: VIRUSTOTAL.COM\r\n   Registry Domain ID: ...",
      "whois_date": 1560599498
    }
  }
}
```


### 6.1.2 get_comments(domain, limit, cursor)

Retrieve comments for an Internet domain.

**Arguments:**

- `domain` : Domain name (str).
- `limit` : Maximum number of comments to retrieve (int). The default value is 10.
- `cursor` : Continuation cursor (str). The default value is ‘’.

**Return value:**

The response from the server as a byte sequence.

**Exception:**

- *VirusTotalAPIError* (Connection error): In case of server connection errors.
- *VirusTotalAPIError* (Timeout error): If the response timeout from the server is exceeded.

**Usage:**

```
from vtapi3 import VirusTotalAPIDomains, VirusTotalAPIError
    ...
vt_api_domains = VirusTotalAPIDomains('<API key>')
try:
    result = vt_api_domains.get_comments('<domain>', 5)
except VirusTotalAPIError as err:
    print(err, err.err_code)
else:
    if vt_api_domains.get_last_http_error() == vt_api_domains.HTTP_OK:
        result = json.loads(result)
        result = json.dumps(result, sort_keys=False, indent=4)
        print(result)
    else:
        print('HTTP Error [' + str(vt_api_domains.get_last_http_error()) +']')
    ...
```

### 6.1.3 put_comments(domain, text)

Add a comment to an Internet domain..

**Arguments:**

- `domain` : Domain name (str).
- `text` : Text of the comment (str). Any word starting with # in your comment's text will be considered a tag, and added to the comment's tag attribute.

**Return value:**

The response from the server as a byte sequence.

**Exception:**

- *VirusTotalAPIError* (Connection error): In case of server connection errors.
- *VirusTotalAPIError* (Timeout error): If the response timeout from the server is exceeded.

**Usage:**

```
from vtapi3 import VirusTotalAPIDomains, VirusTotalAPIError
    ...
vt_api_domainss = VirusTotalAPIDomains('<API key>')
try:
    result = vt_api_domains.put_comment('<domain>', '<text of the comment>')
except VirusTotalAPIError as err:
    print(err, err.err_code)
else:
    if vt_api_domains.get_last_http_error() == vt_api_domains.HTTP_OK:
        result = json.loads(result)
        result = json.dumps(result, sort_keys=False, indent=4)
        print(result)
    else:
```

(continues on next page)

(continued from previous page)

```
        print('HTTP Error [' + str(vt_api_domains.get_last_http_error()) +']')
    ...
```

### 6.1.4 get_relationship(domain, relationship, limit, cursor)

Retrieve objects related to an Internet domain.

**Arguments:**

- `domain` : Domain name (str).
- `relationship` : Relationship name (str). The default value is `/resolutions`. For more information, see https://developers.virustotal.com/v3.0/reference#domains-relationships.
- `limit` : Maximum number of related objects to retrieve (int). The default value is 10.
- `cursor` : Continuation cursor (str). The default value is ‘’.

**Return value:**

The response from the server as a byte sequence.

**Exception:**

- *VirusTotalAPIError* (Connection error): In case of server connection errors.
- *VirusTotalAPIError* (Timeout error): If the response timeout from the server is exceeded.

**Usage:**

```
from vtapi3 import VirusTotalAPIDomains, VirusTotalAPIError
   ...
vt_api_domains = VirusTotalAPIDomains('<API key>')
try:
    result = vt_api_domains.get_relationship('<domain>', 'downloaded_files')
except VirusTotalAPIError as err:
    print(err, err.err_code)
else:
    if vt_api_domains.get_last_http_error() == vt_api_domains.HTTP_OK:
        result = json.loads(result)
        result = json.dumps(result, sort_keys=False, indent=4)
        print(result)
    else:
        print('HTTP Error [' + str(vt_api_domains.get_last_http_error()) +']')
    ...
```

### 6.1.5 get_votes(domain, limit, cursor)

Retrieve votes for a hostname or domain.

**Arguments:**

- `domain` : Domain name (str).
- `limit` : Maximum number of vites to retrieve (int). The default value is 10.

- `cursor` : Continuation cursor (str). The default value is ‘’.

**Return value:**

The response from the server as a byte sequence.

**Exception:**

- *VirusTotalAPIError* (Connection error): In case of server connection errors.
- *VirusTotalAPIError* (Timeout error): If the response timeout from the server is exceeded.

**Usage:**

```
from vtapi3 import VirusTotalAPIDomains, VirusTotalAPIError
   ...
vt_api_domains = VirusTotalAPIDomains('<API key>')
try:
    result = vt_api_domains.get_votes('<domain>', 5)
except VirusTotalAPIError as err:
    print(err, err.err_code)
else:
    if vt_api_domains.get_last_http_error() == vt_api_domains.HTTP_OK:
        result = json.loads(result)
        result = json.dumps(result, sort_keys=False, indent=4)
        print(result)
    else:
        print('HTTP Error [' + str(vt_api_domains.get_last_http_error()) +']')
    ...
```

### 6.1.6 put_votes(domain, malicious)

Add a vote for a hostname or domain.

**Arguments:**

- `domain` : Domain name(str).
- `malicious` : Determines a malicious (True) or harmless (False) file (bool). The default value is `False`.

**Return value:**

The response from the server as a byte sequence.

**Exception:**

- *VirusTotalAPIError* (Connection error): In case of server connection errors.
- *VirusTotalAPIError* (Timeout error): If the response timeout from the server is exceeded.

**Usage:**

```
from vtapi3 import VirusTotalAPIDomains, VirusTotalAPIError
   ...
vt_api_domains = VirusTotalAPIDomains('<API key>')
try:
    result = vt_api_domains.put_votes('<domain>', True)
except VirusTotalAPIError as err:
    print(err, err.err_code)
else:
```

(continues on next page)

(continued from previous page)

```
    if vt_api_domains.get_last_http_error() == vt_api_domains.HTTP_OK:
        result = json.loads(result)
        result = json.dumps(result, sort_keys=False, indent=4)
        print(result)
    else:
        print('HTTP Error [' + str(vt_api_domains.get_last_http_error()) +']')
    ...
```

CHAPTER 7

# VirusTotalAPIIPAddresses

The retrieving information about any IP addresses from the VirusTotal database methods are defined in the class.

## 7.1 Methods:

### 7.1.1 get_report(ip_address)

Retrieve information about an IP address.

**Arguments:**

- `ip_address` : IP address (str).

**Return value:**

The response from the server as a byte sequence.

**Exception:**

- *VirusTotalAPIError* (Connection error): In case of server connection errors.
- *VirusTotalAPIError* (Timeout error): If the response timeout from the server is exceeded.

**Usage:**

```
from vtapi3 import VirusTotalAPIIPAddresses, VirusTotalAPIError
    ...
vt_api_ip_addresses = VirusTotalAPIIPAddresses('<API key>')
try:
    result = vt_api_ip_addresses.get_report('<ip_address>')
except VirusTotalAPIError as err:
    print(err, err.err_code)
else:
    if vt_api_ip_addresses.get_last_http_error() == vt_api_ip_addresses.HTTP_OK:
```

(continues on next page)

(continued from previous page)

```
        result = json.loads(result)
        result = json.dumps(result, sort_keys=False, indent=4)
        print(result)
    else:
        print('HTTP Error [' + str(vt_api_ip_addresses.get_last_http_error()) +']')
    ...
```

**Example response:**

When `_last_http_error` = `HTTP_OK` and after conversion to JSON, the response will look like this:

```
{
  "type": "ip_address",
  "id": "8.8.8.8",
  "links": {
    "self": "https://www.virustotal.com/api/v3/ip_addresses/8.8.8.8"
  },
  "data": {
    "attributes": {
      "as_owner": "Google Inc.",
      "asn": 15169,
      "country": "US"
    }
  }
}
```

### 7.1.2 get_comments(ip_address, limit, cursor)

Retrieve comments for an IP address.

**Arguments:**

- `ip_address` : IP address (str).
- `limit` : Maximum number of comments to retrieve (int). The default value is 10.
- `cursor` : Continuation cursor (str). The default value is ‘’.

**Return value:**

The response from the server as a byte sequence.

**Exception:**

- *VirusTotalAPIError* (Connection error): In case of server connection errors.
- *VirusTotalAPIError* (Timeout error): If the response timeout from the server is exceeded.

**Usage:**

```
from vtapi3 import VirusTotalAPIIPAddresses, VirusTotalAPIError
    ...
vt_api_ip_addresses = VirusTotalAPIIPAddresses('<API key>')
try:
    result = vt_api_ip_addresses.get_comments('<ip_address>', 5)
except VirusTotalAPIError as err:
    print(err, err.err_code)
```

(continues on next page)

(continued from previous page)

```
else:
    if vt_api_ip_addresses.get_last_http_error() == vt_api_ip_addresses.HTTP_OK:
        result = json.loads(result)
        result = json.dumps(result, sort_keys=False, indent=4)
        print(result)
    else:
        print('HTTP Error [' + str(vt_api_ip_addresses.get_last_http_error()) +']')
    ...
```

### 7.1.3 put_comments(ip_address, text)

Add a comment to an IP address.

**Arguments:**

- `ip_address` : IP address (str).
- `text` : Text of the comment (str). Any word starting with # in your comment's text will be considered a tag, and added to the comment's tag attribute.

**Return value:**

The response from the server as a byte sequence.

**Exception:**

- *VirusTotalAPIError* (Connection error): In case of server connection errors.
- *VirusTotalAPIError* (Timeout error): If the response timeout from the server is exceeded.

**Usage:**

```
from vtapi3 import VirusTotalAPIIPAddresses, VirusTotalAPIError
   ...
vt_api_ip_addresses = VirusTotalAPIIPAddresses('<API key>')
try:
    result = vt_api_ip_addresses.put_comment('<ip_address>', '<text of the comment>')
except VirusTotalAPIError as err:
    print(err, err.err_code)
else:
    if vt_api_ip_addresses.get_last_http_error() == vt_api_ip_addresses.HTTP_OK:
        result = json.loads(result)
        result = json.dumps(result, sort_keys=False, indent=4)
        print(result)
    else:
        print('HTTP Error [' + str(vt_api_ip_addresses.get_last_http_error()) +']')
    ...
```

### 7.1.4 get_relationship(ip_address, relationship, limit, cursor)

Retrieve objects related to an IP address.

**Arguments:**

- `ip_address` : IP address (str).

- `relationship` : Relationship name (str). The default value is `/resolutions`. For more information, see https://developers.virustotal.com/v3.0/reference#ip-relationships.
- `limit` : Maximum number of related objects to retrieve (int). The default value is 10.
- `cursor` : Continuation cursor (str). The default value is ‘’.

**Return value:**

The response from the server as a byte sequence.

**Exception:**

- *VirusTotalAPIError* (Connection error): In case of server connection errors.
- *VirusTotalAPIError* (Timeout error): If the response timeout from the server is exceeded.

**Usage:**

```
from vtapi3 import VirusTotalAPIIPAddresses, VirusTotalAPIError
    ...
vt_api_ip_addresses = VirusTotalAPIIPAddresses('<API key>')
try:
    result = vt_api_ip_addresses.get_relationship('<ip_address>', 'downloaded_files')
except VirusTotalAPIError as err:
    print(err, err.err_code)
else:
    if vt_api_ip_addresses.get_last_http_error() == vt_api_ip_addresses.HTTP_OK:
        result = json.loads(result)
        result = json.dumps(result, sort_keys=False, indent=4)
        print(result)
    else:
        print('HTTP Error [' + str(vt_api_ip_addresses.get_last_http_error()) +']')
    ...
```

---

### 7.1.5 get_votes(ip_address, limit, cursor)

Retrieve votes for an IP address.

**Arguments:**

- `ip_address` : IP address (str).
- `limit` : Maximum number of vites to retrieve (int). The default value is 10.
- `cursor` : Continuation cursor (str). The default value is ‘’.

**Return value:**

The response from the server as a byte sequence.

**Exception:**

- *VirusTotalAPIError* (Connection error): In case of server connection errors.
- *VirusTotalAPIError* (Timeout error): If the response timeout from the server is exceeded.

**Usage:**

```
from vtapi3 import VirusTotalAPIIPAddresses, VirusTotalAPIError
   ...
vt_api_ip_addresses = VirusTotalAPIIPAddresses('<API key>')
try:
    result = vt_api_ip_addresses.get_votes('<ip_address>', 5)
except VirusTotalAPIError as err:
    print(err, err.err_code)
else:
    if vt_api_ip_addresses.get_last_http_error() == vt_api_ip_addresses.HTTP_OK:
        result = json.loads(result)
        result = json.dumps(result, sort_keys=False, indent=4)
        print(result)
    else:
        print('HTTP Error [' + str(vt_api_ip_addresses.get_last_http_error()) +']')
    ...
```

### 7.1.6 put_votes(ip_address, malicious)

Add a vote for an IP address.

**Arguments:**

- `ip_address` : IP address (str).
- `malicious` : Determines a malicious (True) or harmless (False) file (bool). The default value is `False`.

**Return value:**

The response from the server as a byte sequence.

**Exception:**

- *VirusTotalAPIError* (Connection error): In case of server connection errors.
- *VirusTotalAPIError* (Timeout error): If the response timeout from the server is exceeded.

**Usage:**

```
from vtapi3 import VirusTotalAPIIPAddresses, VirusTotalAPIError
   ...
vt_api_ip_addresses = VirusTotalAPIIPAddresses('<API key>')
try:
    result = vt_api_ip_addresses.put_votes('<ip_address>', True)
except VirusTotalAPIError as err:
    print(err, err.err_code)
else:
    if vt_api_ip_addresses.get_last_http_error() == vt_api_ip_addresses.HTTP_OK:
        result = json.loads(result)
        result = json.dumps(result, sort_keys=False, indent=4)
        print(result)
    else:
        print('HTTP Error [' + str(vt_api_ip_addresses.get_last_http_error()) +']')
    ...
```

CHAPTER 8

# VirusTotalAPIAnalyses

The retrieving information about analysis of the file or URL method are defined in the class.

## 8.1 Methods:

### 8.1.1 get_report(object_id)

Retrieve information about a file or URL analysis.

**Arguments:**

- `object_id` : Analysis identifier (str).

**Return value:**

The response from the server as a byte sequence.

**Exception:**

- *VirusTotalAPIError* (Connection error): In case of server connection errors.
- *VirusTotalAPIError* (Timeout error): If the response timeout from the server is exceeded.

**Usage:**

```
from vtapi3 import VirusTotalAPIAnalyses, VirusTotalAPIError
    ...
vt_api_analyses = VirusTotalAPIAnalyses('<API key>')
try:
    result = vt_api_analyses.get_report('<object id>')
except VirusTotalAPIError as err:
    print(err, err.err_code)
else:
    if vt_api_analyses.get_last_http_error() == vt_api_analyses.HTTP_OK:
```

(continues on next page)

(continued from previous page)

```
        result = json.loads(result)
        result = json.dumps(result, sort_keys=False, indent=4)
        print(result)
    else:
        print('HTTP Error [' + str(vt_api_analyses.get_last_http_error()) +']')
    ...
```

# CHAPTER 9

# VirusTotalAPIError

A class that implements exceptions that may occur when module class methods are used.

## 9.1 Types of exceptions:

- **"Connection error":** This exception occurs when there is an error communicating with the server (Error code = errno.ECONNABORTED)
- **"Timeout error":** This exception occurs when the response time from the server is exceeded (Error code = errno.ETIMEDOUT).
- **"File not found":** This exception occurs when the file to be uploaded to the server is not found (Error code = errno.ENOENT).
- **"Permission error":** This exception occurs when the file to be uploaded to the server is not found (Error code = errno.EPERM).
- **"IO Error":** (Added in version 1.1.1) This exception occurs if there is an IO error during file operations (Error code = errno.EIO).
- **"API key environment error:" (Added in version 1.1.2) This exception occurs if the VT_API_KEY environment variable with the VirusTotal API function access key is missing (Error code = errno.EINVAL).

CHAPTER 10

# Command line option

This feature has been implemented since version 1.1.0. Using the command line options you can:

- upload the file to VirusTotal for scanning and get the file ID for later use with the `get_report()` function of the `VirusTotalAPIAnalyses` class;
- upload a file to VirusTotal for scanning and get a report on the results of its scanning;
- get a report on the results of analyzing a file that is available in the VirusTotal database;
- get a report on the results of analyzing a file that is available in the VirusTotal database by its hash ID (SHA1, SHA256 or MD5);
- upload a URL to VirusTotal for scanning and get the URL ID for later use using the `get_report()` function of the `VirusTotalAPIAnalyses` class;
- upload a URL to VirusTotal for scanning and get a report on the results of its scanning;
- get a report on the results of analyzing a URL that is available in the VirusTotal database;
- get a report on the results of IP address analysis;
- get a report on the results of domain analysis.

## 10.1 ommon format

```
$ python -m vtapi3 <resource> [-h], [-fid], [-fsr], [-far], [-hr], [-uid], [-usr], [-
↪uar], [-ipr] or [-dr]
```

## 10.2 Positional arguments

### 10.2.1 resource

Object that you want to analyse in VirusTotal (file, URL, IP address or domain). The file path, file hash (SHA1, SHA256, or MD5), URL, IP address, or domain name can be used.

---

## 10.3 Optional arguments

### 10.3.1 -h, –help

Show help message and exit.

---

### 10.3.2 -fid, –file-id

Getting the identifier of the file for further analysis.

---

### 10.3.3 -fsr, –file-scan-report

Getting a report on the results of scanning a file.

---

### 10.3.4 -far, –file-analyse-report

Getting a report on the results of file analysis (enabled by default).

#### Example JSON response


```
{
  "type": "file",
  "id": "8739c76e681f900923b900c9df0ef75cf421d39cabb54650c4b9ad19b6a76d85",
 "links": {
    "self": "https://www.virustotal.com/api/v3/files/
↪8739c76e681f900923b900c9df0ef75cf421d39cabb54650c4b9ad19b6a76d85"
  },
  "data": {
    "attributes": {
      "first_seen_itw_date": 1075654056,
      "first_submission_date": 1170892383,
      "last_analysis_date": 1502355193,
      "last_analysis_results": {
        "AVG": {
```


(continues on next page)

(continued from previous page)


```
        "category": "undetected",
        "engine_name": "AVG",
        "engine_update": "20170810",
        "engine_version": "8.0.1489.320",
        "method": "blacklist",
        "result": null
      }
      ...
    },
    "last_analysis_stats": {
      "harmless": 0,
      "malicious": 0,
      "suspicious": 0,
      "timeout": 0,
      "type-unsupported": 8,
      "undetected": 59
    },
    "last_submission_date": 1502355193,
    "magic": "data",
    "md5": "76cdb2bad9582d23c1f6f4d868218d6c",
    "names": [
      "zipnew.dat",
      "327916-1502345099.zip",
      "ac3plug.zip",
      "IMG_6937.zip",
      "DOC952.zip",
      "20170801486960.zip"
    ],
    "nsrl_info": {
      "filenames": [
        "WINDOWS DIALUP.ZIP",
        "kemsetup.ZIP",
        "Data_Linux.zip",
        "2003.zip",
        "_6A271FB199E041FC82F4D282E68B01D6"
      ],
      "products": [
        "Master Hacker Internet Terrorism (Core Publishing Inc.)",
        "Read Rabbits Math Ages 6-9 (Smart Saver)",
        "Neverwinter Nights Gold (Atari)",
        "Limited Edition Print Workshop 2004 (ValuSoft)",
        "Crysis (Electronic Arts Inc.)"
      ]
    },
    "reputation": -889,
    "sha1": "b04f3ee8f5e43fa3b162981b50bb72fe1acabb33",
    "sha256": "8739c76e681f900923b900c9df0ef75cf421d39cabb54650c4b9ad19b6a76d85",
    "size": 22,
    "ssdeep": "3:pjt/l:Nt",
    "tags": [
      "software-collection",
      "nsrl",
      "attachment",
      "trusted",
      "via-tor"
    ],
    "times_submitted": 26471,
```


(continues on next page)

(continued from previous page)


```
      "total_votes": {
        "harmless": 639,
        "malicious": 958
      },
      "trid": [
        {
          "file_type": "ZIP compressed archive (empty)",
          "probability": 100
        }
      ],
      "trusted_verdict": {
        "filename": "lprn_spotlightstory_015.zip",
        "link": "https://dl.google.com/dl/spotlight/test/lprn_spotlightstory/9/lprn_
↪spotlightstory_015.zip",
        "organization": "Google",
        "verdict": "goodware"
      },
      "type_description": "unknown",
      }
    }
  }
}
```


### 10.3.5 -hr, –hash-report

Getting a report on the results of analyzing a file by its hash (SHA256, SHA1 or MD5).

### 10.3.6 -uid, –url-id

Getting the identifier of the URL for further analysis.

### 10.3.7 -usr, –url-scan-report

Getting a report on the results of scanning a URL.

### 10.3.8 -uar, –url-analyse-report

Getting a report on the results of URL analysis.

### 10.3.9 -ipr, –ip-report

Getting a report on the results of IP address analysis.

#### Example JSON response


```
{
  "type": "ip_address",
  "id": "8.8.8.8",
  "links": {
    "self": "https://www.virustotal.com/api/v3/ip_addresses/8.8.8.8"
  },
  "data": {
    "attributes": {
      "as_owner": "Google Inc.",
      "asn": 15169,
      "country": "US"
    }
  }
}
```


### 10.3.10 -dr, –domain-report

Getting a report on the results of domain analysis.

#### Example JSON response


```
{
  "data": {
    "type": "domain",
    "id": "virustotal.com",
    "links": {
      "self": "https://virustotal.com/api/v3/domains/virustotal.com"
    },
    "attributes": {
      "categories": {
        "Alexa": "services",
        "BitDefender": "computersandsoftware",
        "TrendMicro": "computers internet",
        "Websense ThreatSeeker": "computer security"
      },
      "creation_date": 1032308169,
      "last_update_date": 1389199030,
      "registrar": "MarkMonitor Inc.",
      "reputation": 13,
      "total_votes": {
        "harmless": 2,
        "malicious": 0
      },
        "whois": "Domain Name: VIRUSTOTAL.COM\r\n   Registry Domain ID: ...",
      "whois_date": 1560599498
    }
  }
}
```


- genindex

# Index

## Symbols



## A



## B



## G



## H



## P



## R



## T



## U