Pulsedive Documentation

Release 0.1.0

Pepe Berba

Jan 10, 2019

Contents

A low-level client for Pulsedive that aims provide an easy and idiomatic way to interact with the Pulsedive API.

CHAPTER 1

Installation

Install the `pulsedive` package with pip:

```
pip install pulsedive
```

CHAPTER 2

Example use

Sample Code:

```
import pulsedive
# pud = pulsedive.Pulsedive('<API KEY>')
pud = pulsedive.Pulsedive()

# Getting a specific indicator
ind = pud.indicator(value='pulsedive.com')
pud.indicator.links(ind['iid'])

# Searching for indicators
pud.search('pulsedive', risk=['high', 'critical'], indicator_type=['ip'])

# Pulling from feeds or threats
pud.feed.links(1)
pud.threat.links(1)

# Searching for threats and feeds
pud.search.threat('Zeus', risk=['high', 'critical'])
pud.search.feed('Zeus')

# Exporting a search
pud.search.to_csv(filename="zues.csv", threat=['Zeus'], indicator_type=['ip'])

# Analyzing
# q = pud.analyze.encoded('Z29vZ2xlLmNvbQ==')
q = pud.analyze('google.com')
pud.analyze.results(q['qid'])
```

CHAPTER 3

Contents

3.1 API Documentation

The API calls tries to map the raw API with reasonable default values to make it easier for the developers. Some of the default values in the API are not the default behavior in the actual API. This most evident in the *SearchClient*

For example, the default behavior of `limit='hundred'` is not the default behavior when no limit is defined. To get all the results set this to None.

Other fields that have default values are `risk`, `category`, `indicator_type`, and `sanitize`.

Note: For compatibility with the Python ecosystem we use `properties` instead of `property`

3.1.1 Global parameters

The `raw`, `sanitize` and `pretty` parameters can be set on the instantiation of a *Pulsedive*. This will be set as the default but they can be overridden in the individual methods calls.

Raw Requests

For most API calls, the client will return the json of the response. However, if there is any need for you to handle the raw response, you can set the *raw* parameter:

```
from pulsedive import Pulsedive
pud = Pulsedive()

pud.indicator('1', raw=True)
# <Response [200]>
```

HTML-Ready Output

You can return HTML-ready results from the API by including and setting the `sanitize` parameter to 1. By default sanitize is set to 1.

Pretty-Printing

You can pretty-print results from the API by including and setting the `pretty` parameter to 1:

```
from pulsedive import Pulsedive
pud = Pulsedive()

pud.indicator('1').text
#{"page_current": 0,"results": [...

pud.indicator('1', pretty=True).text
#{
#    "page_current": 0,
#    "results": [
#        {
#            "iid": "1",
#            "indicator": "afobal.cl",
# ....
```

Pagination

Some requests might require going through pages of results. Described here.

In that case, you can set the `page` keyword in the method calls:

```
from pulsedive import Pulsedive
pud = Pulsedive(raw=True)

pud.search(risk=['high', 'critical'], limit=None)
#{'page_current': 0,
# 'page_next': 1,
# 'results': [{'attributes': ['25',
#     '80',
# ...

pud.search(risk=['high', 'critical'], limit=None, page=1)
```

3.1.2 Pulsedive

class pulsedive.**Pulsedive**(*api_key=None*, *sanitize=True*, *pretty=False*, *raw=False*, ***kwargs*)

Pulsedive low-level client. Provides a straightforward mapping from Python to the Pulsedive API.

https://pulsedive.com/api

The instance has attributes `indicator`, `threat`, `feed`, `search`, and `analyze` that provide access to instances of *`IndicatorClient`*, *`ThreatClient`*, *`FeedClient`*, *`SearchClient`*, and *`AnalyzeClient`*, respectively. This is the preferred (and only supported) way to get access to those classes and their methods.

Parameters

- **api_key** – This parameter is optional. Pulsedive allows access to the API without a key
- **sanitize** – Sets the default *sanitize* option for all requests
- **pretty** – Sets the default *pretty* option for all requests
- **raw** – If set to True, the raw requests
- **kwargs** – Other parameters that will be passed on to all calls to `Request.get()` and `Request.post()`. Some Request keyword examples are *proxies* and *cert*

3.1.3 Indicators

class pulsedive.client.**IndicatorClient**(*pulsedive_client*)

This exposes the Pulsedive Indicator API

get(*iid=None*, *value=None*, *schema=False*, ***kwargs*)

Queries for an indicator by either indicator id or by value.

Parameters

- **iid** – Used when retrieving by indicator ID
- **value** – Used when retrieving by value
- **schema** – *schema* is an optional boolean parameter. It's a flag to return associated attributes of the indicator. Default: False

links(*iid*, ***kwargs*)

Returns historical links of indicator

Parameters **iid** – Indicator ID

properties(*iid*, ***kwargs*)

Returns historical properties of indicator

Parameters **iid** – Indicator ID

3.1.4 Threats

class pulsedive.client.**ThreatClient**(*pulsedive_client*)

This exposes the Pulsedive Threat API

get(*tid=None*, *name=None*, ***kwargs*)

Queries threats by either threat id or by name.

Parameters

- **tid** – Used when retrieving by threat ID
- **name** – Used when retrieving by threat name

links(*tid*, ***kwargs*)

Returns the linked indicators for the threat

Parameters **tid** – Threat ID

summary(*tid*, *splitrisk=False*, ***kwargs*)

Gives a summary of a threat that gives counts of indicators per feed, attribute, etc.

Parameters

- **tid** – Threat ID

- **splitrisk** – Whether to split each indicator count by risk categories (none, low, medium, etc.). Default: False

3.1.5 Feeds

class pulsedive.client.**FeedClient**(*pulsedive_client*)

This exposes the Pulsedive Feed API

get(*fid=None*, *feed=None*, *organization=None*, ***kwargs*)

Gets data of a feed through its feed ID.

This is aliased by the __call__ method so the following lines are equivalent:

```
pud.feed.get(1)
pud.feed(1)
pud.feed.get(feed='zeus bad domains')
pud.feed.get(
    feed='Zeus Bad Domains',
    organization='abuse.ch'
)
```

Parameters

- **fid** – Feed ID used when retrieving by fid
- **feed** – name of the feed used when retrieving by name. This has to be the complete name of the field but is not case sensitive.
- **organization** – optional field when retrieving by name

links(*fid*, ***kwargs*)

Returns the linked indicators for the feed

Parameters **fid** – Feed ID

3.1.6 Search

class pulsedive.client.**SearchClient**(*pulsedive_client*)

Exposes the Pulsedive Search API

> **Warning:** The use of properties is not yet stable, and untested.

feed(*value=''*, *category=['general', 'abuse', 'apt', 'attack', 'botnet', 'crime', 'exploitkit', 'fraud', 'group', 'malware', 'proxy', 'pup', 'reconnaissance', 'spam', 'terrorism', 'phishing', 'vulnerability']*, *splitrisk=False*, ***kwargs*)

Searches for feeds, similar to how searches are done in the Pulsedive Site.

Parameters

- **value** – Search value for the feed
- **category** – Either a value or list of values from the threat categories to include in the search. Defaults to all threat categories.
- **splitrisk** – Whether to split each indicator count by risk categories (none, low, medium, etc.). Defaults to False

indicator(*value='', risk=['unknown', 'none', 'low', 'medium', 'high', 'critical', 'retired'], indicator_type=['ip', 'ipv6', 'url', 'domain', 'artifact'], lastseen=None, latest=None, limit='hundred', export=False, properties=None, attribute=None, feed=None, threat=None, **kwargs*)

Searches for indicators, similar to how searches are done in the Pulsedive Site.

This is aliased by the `__call__` method so the following lines are equivalent:

```
pud.search.indicator('Zues')
pud.search('Zeus')
```

Parameters

- **value** – Search value for the indicator
- **risk** – Either a value or list of values from the risk types to include to include in the search. Defaults to all risk types.
- **indicator_type** – Either a value or list of values from the indicator categories. Defaults to all categories.
- **lastseen** – Defaults none (All time). Valid values are: 'day', 'week', 'month' or None (All time)
- **latest** – Reflects whether to search all properties or just the latest properties and can be either latest or historical. Defaults to _____??
- **properties** – List of properties to search for in the indicator. Defaults to none.
- **attribute** – List of attributes to search for in the indicator. Defaults to none
- **feed** – List of feeds to search in. Defaults none
- **threat** – List of threats to search in. Defaults none.
- **limit** – Limit on the number of indicators returned by search. Defaults to 'hundred'. Valid values are 'hundred', 'thousand', 'tenthousand', or None. If set to None, the results will be paged if too large for one request.
- **export** – Optional boolean value to convert results to CSV format. If set to true, this function will return a *str*, instead of a *dict* Defaults False.

threat(*value='', risk=['unknown', 'none', 'low', 'medium', 'high', 'critical', 'retired'], category=['general', 'abuse', 'apt', 'attack', 'botnet', 'crime', 'exploitkit', 'fraud', 'group', 'malware', 'proxy', 'pup', 'reconnaissance', 'spam', 'terrorism', 'phishing', 'vulnerability'], properties=None, attribute=None, splitrisk=False, **kwargs*)

Searches for threats, similar to how searches are done in the Pulsedive Site.

Parameters

- **value** – Search value for the threat
- **risk** – Either a value or list of values from the risk types to include to include in the search. Defaults to all risk types.
- **category** – Either a value or list of values from the threat categories to include in the search. Defaults to all threat categories.
- **properties** – List of properties to search for in the threats. Defaults to none
- **attribute** – List of attributes to search for in the threats. Defaults to none
- **splitrisk** – Whether to split each indicator count by risk categories (none, low, medium, etc.). Defaults to False

to_csv(*value=''*, *filename=None*, ***kwargs*)
Searches for indicators and saves the result to `filename`.

All arguments aside from `filename` will be passed to `pud.search.indicator()` with `export=True`.

Parameters

- **value** – Search value for the indicator
- **filename** – Destination filename of the csv

3.1.7 Analyze

class pulsedive.client.**AnalyzeClient**(*pulsedive_client*)
This exposes the Pulsedive Analyze API

__call__(*value*, *enrich=True*, *probe=False*, ***kwargs*)
Encodes `value` in base64 and submits this encoded value to be added to the analyze queue for processing using the *pulsedive.client.AnalyzeClient.encoded()*

encrich and *probe* determine whether or note to probe the indicator and enrich with Shodan and VirusTotal.

Parameters

- **value** – Value to be encoded and processed
- **enrich** – Whether to enrich the indicator
- **probe** – Whether to probe the indicator

encoded(*value*, *enrich=True*, *probe=False*, ***kwargs*)
Submits `value`, a base64 encoding of the indicator, to be added to the analyze queue for processing.

encrich and *probe* determine whether or note to probe the indicator and enrich with Shodan and VirusTotal.

Parameters

- **value** – Value of your indicator in base64 encoding
- **enrich** – Whether to enrich the indicator
- **probe** – Whether to probe the indicator

results(*qid*, ***kwargs*)
Returns the result of the analysis when the indicator has been processed.

If the results are not yet ready, this will raise a `PulsediveException`

Parameters **qid** – Queue ID

Python Module Index

p

Index