Censys Python

Release 2.2.19

Censys, Inc.

Dec 11, 2025

CONTENTS

1 The User Guide 3

2 The API Documentation 29

3 The Contributor Guide 101

Python Module Index 103

Index 105

Release v2.2.19. (*Quick Start*)

Censys Python is an easy-to-use and lightweight API wrapper for Censys APIs.

CHAPTER

ONE

THE USER GUIDE

1.1 Quick Start

Assuming you have Python already, install the package:

```
$ pip install censys
```

If you do not have pip installed, get it here.

Configure your credentials:

Search API

ASM API

```
$ censys config
```

Or you can set the environment variables:

```
$ export CENSYS_API_ID=<your-api-id>
$ export CENSYS_API_SECRET=<your-api-secret>
```

Find your credentials on the Account page.

```
$ censys asm config
```

Or you can set the environment variables:

```
$ export CENSYS_ASM_API_KEY=<your-api-key>
```

Find your credentials on the Integrations page.

Optionally, you can enable tab completion for the CLI by adding this line to your *~/.bashrc*, *~/.zshrc*, or equivalent.

```
$ eval "$(register-python-argcomplete censys)"
```

Note: Please note that autocomplete is supported for field names in the *search* command.

1.2 CLI Usage

Before continuing please ensure you have successfully configured your credentials.

```
$ censys config
```

The configuration file by default is written to `~/.config/censys/censys.cfg`, but you can change this by setting the `CENSYS_CONFIG_PATH` environment variable.

```
$ export CENSYS_CONFIG_PATH=/path/to/config/file
```

Optionally, you can enable tab completion for the CLI by adding this line to your *~/.bashrc*, *~/.zshrc*, or equivalent.

```
$ eval "$(register-python-argcomplete censys)"
```

Note: Please note that autocomplete is supported for field names in the *search* command.

1.2.1 search

Below we show an example of searching hosts from the CLI.

```
$ censys search 'services.http.response.html_title: "Dashboard"'
```

By combining the `search` command with `jq` we can easily manipulate the output to get the desired fields.

```
$ censys search 'services.service_name: ELASTICSEARCH' | jq -c '.[] | ip: .ip'
```

By setting the `--pages` flag to `-1` we can get all pages of results.

```
$ censys search 'ip: 8.8.8.0/16' --pages -1 | jq -c '[.[] | .ip]'
```

By settings the `--index-type` flag we can search other indexes such as `certificates`.

```
$ censys search 'parsed.subject_dn: "censys.io"' --index-type certificates
```

For the `certificates` index we can also add the `--fields` flag to specify which fields we want returned.

```
$ censys search 'parsed.subject.country: AU' --index-type certificates --fields parsed.issuer.organizati
```

1.2.2 view

Below we show an example of viewing a host from the CLI.

```
$ censys view 8.8.8.8
```

Below we show an example of viewing a certificate from the CLI.

```
$ censys view 9b267decc8d23586dc4c56dd0789574cab0f28581ef354ff2fcec8ca6d992fc2 --index-type certificates
```

You can save results to a file using the `-o` argument.

```
$ censys view 8.8.8.8 -o google.json
```

We can then parse this json with something like `jq`.

```
$ cat google.json | jq '[.services[] | port: .port, protocol: .service_name]'
```

If you have access to historical `hosts` data you can view the data at a specific point in time using the `--at-time` argument.

```
$ censys view 1.1.1.1 --at-time 2023-01-01
```

Note: The `--at-time` argument is only available for the `hosts` index.

1.2.3 subdomains

Below we show an example of subdomain enumeration from the CLI.

```
$ censys subdomains censys.io
```

You can limit the number of results by setting the `--max-records` flag.

```
$ censys subdomains censys.io --max-records 10
```

We can then output the results in JSON format using the `--json` flag.

```
$ censys subdomains censys.io --json
```

1.2.4 account

Below we show an example of viewing your account information from the CLI.

```
$ censys account
```

You can also request the JSON version of your account information.

```
$ censys account --json
```

1.2.5 asm

add-seeds

See CLI command *asm add-seeds* for detail documentation of parameters.

Below we show an example of adding seeds from the CLI.

```
$ censys asm add-seeds -j '["1.1.1.1"]'
```

You can add seeds from JSON or CSV files. JSON is assumed unless `--csv` is specified. The CSV file option is shown here.

```
$ censys asm add-seeds --csv -i 'good_seeds.csv'
```

Table 1: CSV File Format

type	value	label
IP_ADDRESS	1.1.1.1	Example Label
DOMAIN_NAME	one.one.one.one	Example Label

You can also add seeds from STDIN using the `-i -` argument. In the example below we are adding IPs from a Censys Search.

```
$ censys search 'services.tls.certificates.leaf_data.issuer.common_name: "Roomba CA"' | jq '[.[] | .ip]
```

You can also add seeds from an nmap XML file using the `--nmap-xml` argument. In the example below we are adding IPs from a nmap scan on `censys.io`.

```
$ nmap censys.io -oX censys.xml
$ censys asm add-seeds --nmap-xml censys.xml
```

delete-seeds

See CLI command *asm delete-seeds* for detail documentation of parameters.

Below we show an example of deleting seeds from the CLI.

```
$ censys asm delete-seeds -j '["1.1.1.1"]'
```

You can delete seeds using file input as well, including CSV files.

```
$ censys asm delete-seeds --csv -i 'bad_seeds.csv'
```

Table 2: CSV File Format

id	type	value	label
1	IP_ADDRESS	1.1.1.1	Example Label
2	DOMAIN_NAME	one.one.one.one	Example Label

delete-all-seeds

See CLI command *asm delete-all-seeds* for detail documentation of parameters.

Below we show an example of deleting all seeds from the CLI. You will be prompted to confirm.

```
$ censys asm delete-all-seeds
```

If you want to delete all seeds without a prompt, you can use the `--force` parameter.

```
$ censys asm delete-all-seeds --force
```

delete-labeled-seeds

See CLI command *asm delete-labeled-seeds* for detail documentation of parameters.

Below we show an example of deleting all seeds with a given label from the CLI.

```
$ censys asm delete-labeled-seeds -l "Some Label"
```

replace-labeled-seeds

See CLI command *asm replace-labeled-seeds* for detail documentation of parameters.

Below we show an example of replacing labeled seeds from the CLI, which will replace all existing seeds that have the specified label with the provided seeds, which will also have that label applied.

```
$ censys asm replace-labeled-seeds -l "Some Label" -j '["1.1.1.1"]'
```

You can also use a variety of methods to specific the new seeds, including providing them in a CSV file.

```
$ censys asm replace-labeled-seeds -l "Some Label" --csv -i 'new_seeds.csv'
```

list-seeds

See CLI command *asm list-seeds* for detail documentation of parameters.

Below we show an example of listing all seeds in CSV file format and appending it to a file.

```
$ censys asm list-seeds --csv >> seeds.csv
```

You can also filter the seeds by type (`-t`) and/or label (`-l`).

```
$ censys asm list-seeds -t 'IP_ADDRESS' -l 'Some Label' >> filtered_seeds.json
```

list-saved-queries

See CLI command *asm list-saved-queries* for detail documentation of parameters.

Below we show an example of listing all saved queries in CSV file format and appending it to a file.

```
$ censys asm list-saved-queries --csv >> saved_queries.csv
```

You can also filter the saved queries by query name prefix (`--query-name-prefix`) and/or filter term (`--filter-term`).

```
$ censys asm list-saved-queries --query-name-prefix 'Some Prefix' --filter-term 'Some Term' >> filtered
```

add-saved-query

See CLI command *asm add-saved-query* for detail documentation of parameters.

Below we show an example of adding a saved query from the CLI.

```
$ censys asm add-saved-query --query-name 'Some Query' --query 'services.http.response.html_title: "Dash
```

get-saved-query-by-id

See CLI command *asm get-saved-query-by-id* for detail documentation of parameters.

Below we show an example of getting a saved query by ID from the CLI.

```
$ censys asm get-saved-query-by-id --query-id 'Some ID'
```

edit-saved-query-by-id

See CLI command *asm edit-saved-query-by-id* for detail documentation of parameters.

Below we show an example of editing a saved query by ID from the CLI.

```
$ censys asm edit-saved-query-by-id --query-id 'Some ID' --query-name 'Some Query' --query 'services.ht
```

delete-saved-query-by-id

See CLI command *asm delete-saved-query-by-id* for detail documentation of parameters.

Below we show an example of deleting a saved query by ID from the CLI.

```
$ censys asm delete-saved-query-by-id --query-id 'Some ID'
```

execute-saved-query-by-name

See CLI command *asm execute-saved-query-by-name* for detail documentation of parameters.

Below we show an example of executing a saved query by name from the CLI.

```
$ censys asm execute-saved-query-by-name --query-name 'Some query name'
```

execute-saved-query-by-id

See CLI command *asm execute-saved-query-by-id* for detail documentation of parameters.

Below we show an example of executing a saved query by ID from the CLI.

```
$ censys asm execute-saved-query-by-id --query-id 'Some query ID'
```

search

See CLI command *asm search* for detail documentation of parameters.

Below we show an example of executing an inventory search query from the CLI.

```
$ censys asm search --query 'Some query'
```

1.3 Usage v2

The Censys Search API provides functionality for interacting with Censys resources such as Hosts.

There are three main API endpoints that this library provides access to:

- *search* - Allows searches against the Hosts index using the same search syntax as the web app.
- *view* - Returns the structured data we have about a specific Host, given the resource's natural ID.
- *aggregate* - Allows you to view resources as a spectrum based on attributes of the resource, similar to the Report Builder page on the web app.

More details about each option can be found in the Censys API documentation. A list of index fields can be found in the Censys API definitions page.

Python class objects must be initialized for each resource index (Hosts).

- *CensysHosts*
- *CensysCerts*

1.3.1 search

Below we show an example using the *CensysHosts* index.

```
"""Search hosts data set."""

from censys.search import CensysHosts

h = CensysHosts()

# Single page of search results
query = h.search("services.service_name: HTTP", per_page=5)
print(query())

# Multiple pages of search results
query = h.search("services.service_name: HTTP", per_page=5, pages=2)
for page in query:
    for host in page:
        print(host)

# View all results (this will do a lookup for each host returned by the search)
query = h.search("services.service_name: HTTP", per_page=5, pages=2)
print(query.view_all())

# Search for virtual hosts
query = h.search("not services.service_name: HTTP", per_page=5, virtual_hosts="ONLY")
print(query())

# Search including specific fields
query = h.search(
    "not services.service_name: HTTP",
    per_page=5,
    fields=["ip", "services.port", "services.service_name"],
)
print(query())
```

1.3.2 view

Below we show an example using the *CensysHosts* index.

```
"""View specific host."""

from censys.search import CensysHosts

h = CensysHosts()

# Fetch a specific host and its services
host = h.view("8.8.8.8")
print(host)

# You can optionally pass in a RFC3339 timestamp to
# fetch a host at the given point in time.
```

(continues on next page)

(continued from previous page)

```python
# Please note historical API access is required.
host = h.view("8.8.8.8", at_time="2021-03-01T17:49:05Z")
print(host)

# You can also pass in a date or datetime object.
from datetime import date

host = h.view("8.8.8.8", at_time=date(2021, 3, 1))
print(host)
# {
#     "ip": "8.8.8.8",
#     "services": [
#         {
#             "dns": {
#                 "server_type": "FORWARDING",
#                 "resolves_correctly": False,
#                 "r_code": "UNKNOWN_CODE",
#             },
#             "extended_service_name": "DNS",
#             "observed_at": "2021-02-28T23:58:55.705035895Z",
#             "perspective_id": "PERSPECTIVE_TELIA",
#             "port": 53,
#             "service_name": "DNS",
#             "source_ip": "74.120.14.37",
#             "transport_protocol": "UDP",
#             "truncated": False,
#         },
#         ...,
#     ],
#     "location": {
#         "continent": "North America",
#         "country": "United States",
#         "country_code": "US",
#         "postal_code": "",
#         "timezone": "America/Chicago",
#         "coordinates": {"latitude": 37.751, "longitude": -97.822},
#         "registered_country": "United States",
#         "registered_country_code": "US",
#     },
#     "location_updated_at": "2022-01-20T16:08:47.237933Z",
#     "autonomous_system": {
#         "asn": 15169,
#         "description": "GOOGLE",
#         "bgp_prefix": "8.8.8.0/24",
#         "name": "GOOGLE",
#         "country_code": "US",
#     },
#     "autonomous_system_updated_at": "2022-01-20T16:08:47.237933Z",
#     "dns": {},
#     "last_updated_at": "2021-02-28T23:58:55.765Z",
# }
```

1.3.3 bulk_view

Below we show an example using the *CensysHosts* index.

```
"""Bulk IP Lookup Example."""

from censys.search import CensysHosts

h = CensysHosts()

IPS = [
    "1.1.1.1",
    "1.1.1.2",
    "1.1.1.3",
]

hosts = h.bulk_view(IPS)
print(hosts)
# {
#     "1.1.1.1": {...},
#     "1.1.1.2": {...},
#     "1.1.1.3": {...},
# }
```

1.3.4 aggregate

Below we show an example using the *CensysHosts* index.

```
"""Aggregate hosts data set."""

from censys.search import SearchClient

c = SearchClient()

# The aggregate method constructs a report using a query, an aggregation field, and the
# number of buckets to bin.
report = c.v2.hosts.aggregate(
    "services.service_name: HTTP",
    "services.port",
    num_buckets=5,
)
print(report)
# {
#     "total": 987342156,
#     "total_omitted": 836949090,
#     "potential_deviation": 3965103,
#     "buckets": [
#         {"key": "80", "count": 58727150},
#         {"key": "443", "count": 46716751},
#         {"key": "7547", "count": 19185117},
#         {"key": "22", "count": 13276559},
#         {"key": "30005", "count": 12487489},
```

(continues on next page)

(continued from previous page)

```
#     ],
#     "query": "services.service_name: HTTP",
#     "field": "services.port",
# }

# You can also specify whether to include virtual hosts in the report.
report = c.v2.hosts.aggregate(
    "services.service_name: HTTP",
    "services.port",
    num_buckets=5,
    virtual_hosts="INCLUDE",
)
print(report)
```

1.3.5 metadata

Please note this method is only available only for the CensysHosts index.

Below we show an example using the *CensysHosts* index.

```
"""Metadata on the hosts index."""

from censys.search import CensysHosts

h = CensysHosts()

# Fetch metadata about hosts.
meta = h.metadata()
print(meta)
# {
#     "services": [
#         "DNS",
#         "FTP",
#         "HTTP",
#         "POP3",
#         "SMTP",
#         "SSH",
#         ...
#     ]
# }
```

1.3.6 view_host_names

Please note this method is only available only for the CensysHosts index.

Below we show an example using the *CensysHosts* index.

```
"""View host names."""

from censys.search import CensysHosts
```

(continues on next page)

(continued from previous page)

```
h = CensysHosts()

# Fetch a list of host names for the specified IP address.
names = h.view_host_names("1.1.1.1")
print(names)
# [
#     "one.one.one.one",
#     ...
# ]
```

1.3.7 view_host_events

Please note this method is only available only for the CensysHosts index.

Below we show an example using the *CensysHosts* index.

```
"""View host events."""

from censys.search import CensysHosts

h = CensysHosts()

# Fetch a list of events for the specified IP address.
events = h.view_host_events("1.1.1.1")
print(events)

# You can also pass in a date or datetime objects.
from datetime import date

events = h.view_host_events(
    "1.1.1.1", per_page=1, start_time=date(2022, 1, 1), end_time=date(2022, 1, 31)
)
print(events)
# {
#     'ip': '1.1.1.1',
#     'events': [
#         {
#             'timestamp': '2022-01-01T00:00:01.713Z',
#             'service_observed': {
#                 'id': {'port': 80, 'service_name': 'HTTP', 'transport_protocol': 'TCP'},
#                 'observed_at': '2021-12-31T23:59:39.910804158Z',
#                 'perspective_id': 'PERSPECTIVE_NTT',
#                 'changed_fields': [{'field_name': 'http.request.uri'}, {'field_name': 'http.
↪response.headers.Cf-Ray.headers'}, {'field_name': 'http.response.headers.Location.headers
↪'}, {'field_name': 'banner'}, {'field_name': 'banner_hashes'}]
#             },
#             '_event': 'service_observed'
#         }
#     ],
#     'links': {
#         'next': 'AS-RtkcKDRPshfT6ojz5ubSuyen_J_J2s9VLmJf9WCg7_jGt0KdU2JvoYW9QXof1Cskvm-
```

(continues on next page)

(continued from previous page)

```
↪b41QyRiR38kWADJuUA_w8rAA5ZNv9llYarhmPv22nIf88JFGGhH0h6dRZ7kDy5RfsiUxNFXeMQQXz0BWYrcQ=='
#     }
# }
```

1.3.8 view_host_diff

Please note this method is only available only for the CensysHosts index.

Below we show an example using the *CensysHosts* index.

```
"""View Host Diff."""

from datetime import date

from censys.search import CensysHosts

h = CensysHosts()

# Compare a single host between two timestamps
diff = h.view_host_diff("1.1.1.1", at_time=date(2022, 1, 1), at_time_b=date(2022, 1, 2))
print(diff)

# Compare a single host between its current timestamp and a timestamp
diff = h.view_host_diff("1.1.1.2", at_time=date(2022, 1, 2))
print(diff)

# Compare two hosts
diff = h.view_host_diff("1.1.1.1", ip_b="1.1.1.2")
print(diff)

# Compare two hosts between two timestamps
diff = h.view_host_diff(
    ip="1.1.1.1",
    ip_b="1.1.1.2",
    at_time=date(2022, 1, 1),
    at_time_b=date(2022, 1, 2),
)
print(diff)
```

1.3.9 get_hosts_by_cert

Please note this method is only available only for the CensysCerts index

Below we show an example using the *CensysCerts* index.

```
from censys.search import CensysCerts

c = CensysCerts()

# Fetch a list of events for the specified IP address.
hosts, links = c.get_hosts_by_cert(
```

(continues on next page)

(continued from previous page)

```
    "fb444eb8e68437bae06232b9f5091bccff62a768ca09e92eb5c9c2cf9d17c426"
)
print(hosts)
```

1.3.10 Comments

get_comments

Below we show an example using the *CensysCerts* index.

```
from censys.search import CensysCerts

c = CensysCerts()

# Fetch a list of comments for the specified certificate.
comments = c.get_comments(
    "fb444eb8e68437bae06232b9f5091bccff62a768ca09e92eb5c9c2cf9d17c426"
)
print(comments)
```

add_comment

Below we show an example using the *CensysHosts* index.

```
from censys.search import CensysHosts

h = CensysHosts()

# Add a comment to a host.
comment = h.add_comment("1.1.1.1", "This is a test comment")
print(comment)
```

update_comment

Below we show an example using the *CensysHosts* index.

```
from censys.search import CensysHosts

h = CensysHosts()

# Update a comment to a host.
comment = h.update_comment("1.1.1.1", 101, "This is an updated test comment")
```

delete_comment

Below we show an example using the *CensysCerts* index.

```
from censys.search import CensysCerts

c = CensysCerts()

# Delete a comment for a certificate.
c.delete_comment(
    "fb444eb8e68437bae06232b9f5091bccff62a768ca09e92eb5c9c2cf9d17c426", 102
)
```

1.3.11 Tags

list_all_tags

Below we show an example using the *CensysHosts* index.

```
from censys.search import CensysHosts

h = CensysHosts()

# Fetch a list of all tags.
tags = h.list_all_tags()
print(tags)
```

create_tag

Below we show an example using the *CensysCerts* index.

```
from censys.search import CensysCerts

c = CensysCerts()

# Create a new tag.
tag = c.create_tag("test-tag")
print(tag)

# Optionally you can specify a color for the tag.
tag = c.create_tag("test-tag", color="#00FF00")
print(tag)
```

get_tag

Below we show an example using the *CensysHosts* index.

```
from censys.search import CensysHosts

h = CensysHosts()

# Fetch a tag.
tag = h.get_tag("123")
print(tag)
```

update_tag

Below we show an example using the *CensysCerts* index.

```
from censys.search import CensysCerts

c = CensysCerts()

# Update a tag.
tag = c.update_tag("123", "test-tag")
print(tag)

# Optionally you can specify a color for the tag.
tag = c.update_tag("123", "test-tag", color="#00FF00")
print(tag)
```

delete_tag

Below we show an example using the *CensysHosts* index.

```
from censys.search import CensysHosts

h = CensysHosts()

# Delete a tag.
h.delete_tag("123")
```

list_tags_on_document

Below we show an example using the *CensysCerts* index.

```
from censys.search import CensysCerts

c = CensysCerts()

# Fetch a list of tags for a document.
tags = c.list_tags_on_document(
    "fb444eb8e68437bae06232b9f5091bccff62a768ca09e92eb5c9c2cf9d17c426"
```

(continues on next page)

(continued from previous page)

```
)
print(tags)
```

add_tag_to_document

Below we show an example using the *CensysHosts* index.

```
from censys.search import CensysHosts

h = CensysHosts()

# Add a tag to a document.
h.add_tag_to_document("123")
```

remove_tag_from_document

Below we show an example using the *CensysCerts* index.

```
from censys.search import CensysCerts

c = CensysCerts()

# Remove a tag from a document.
c.remove_tag_from_document(
    "fb444eb8e68437bae06232b9f5091bccff62a768ca09e92eb5c9c2cf9d17c426"
)
```

list_certs_with_tag

Please note this method is only available only for the CensysCerts index

Below we show an example using the *CensysCerts* index.

```
from censys.search import CensysCerts

c = CensysCerts()

# Fetch a list of certs with the specified tag.
certs = c.list_certs_with_tag("123")
print(certs)
```

list_hosts_with_tag

Please note this method is only available only for the CensysHosts index.

Below we show an example using the *CensysHosts* index.

```
from censys.search import CensysHosts

h = CensysHosts()

# Fetch a list of hosts with the specified tag.
hosts = h.list_hosts_with_tag("123")
print(hosts)
```

1.4 Usage v1

The Censys Search API provides functionality for interacting with Censys resources such as Data, and for viewing Account information such as query quota.

There is one API options that this library provides access to:

- *data* - Returns collections of scan series whose metadata includes a description of the data collected in the series and links to the individual scan results.

More details about each option can be found in the Censys API documentation. A list of index fields can be found in the Censys API definitions page.

Note: Please note that the Censys Search Certificates v1 API is being deprecated. Please use the *CensysCerts (v2) index* for this functionality.

1.4.1 data

Below we show an example using the *CensysData* index.

```
from censys.search import CensysData

c = CensysData()

# View a specific result from a specific series
result = c.view_result("ipv4_2018", "20200818")
print(result)
```

1.4.2 account

Below we show an example using the *CensysHosts* index.

```
from censys.search import CensysHosts

c = CensysHosts()

# Gets account data
account = c.account()
print(account)

# Gets account quota
quota = c.quota()
print(quota)
```

1.5 ASM Usage

The Censys ASM API provides functionality for interacting with Censys ASM endpoints such as Seeds, Assets, Logbook Events, Risks, Inventory Search, and Saved Queries.

The following API clients provided are:

- *seeds* - Provides programmatic management of seeds in the ASM platform.
- *assets* - Returns asset data for hosts, certificates, and domains. This option also allows the user to manage tags and comments on assets.
- *logbook* - Returns logbook events. Can be used to execute targeted searches for events based on start id or date, and event type filters.
- *risks* - Returns risk data for hosts, certificates, and domains. This option also allows the user to get more information about a specific risk.
- *inventory* - Returns inventory data for hosts, certificates, and domains. This option also allows the user to Search for assets based on a variety of criteria.
- *web entities* - Returns web entities instances. This option also allows the user to manage tags and comments on web entities.
- *saved queries* - Returns saved queries. This option also allows the user to manage saved queries.

More details about each option can be found in the Censys ASM API documentation. Users can also test example requests from the API documentation page.

Python class objects can be used individually, but must be initialized for each resource type (Seeds, Assets, Events, Risks, Inventory, Clouds, SavedQueries).

- *Seeds*
- *Assets*
  - *CertificatesAssets*
  - *DomainsAssets*
  - *HostsAssets*
  - *ObjectStoragesAssets*

  - *SubdomainsAssets*
  - *WebEntitiesAssets*
- *Logbook*
- *Risks*
- *InventorySearch*
- *SavedQueries*

Alternatively, all class objects can be used together by initializing an AsmClient object. This client wraps the APIs under one object for ease of use.

- *AsmClient*

1.5.1 Seeds

Below we show examples for **listing seeds** from the Censys ASM platform.

```
from censys.asm import Seeds

s = Seeds()

# Get all seeds
seeds = s.get_seeds()
print(seeds)

# Get a specific type of seed. Optional seed types are ["IP_ADDRESS", "DOMAIN_NAME",
↪"CIDR", "ASN"]
# Here we get IP address seeds.
seeds = s.get_seeds("IP_ADDRESS")
print(seeds)

# Get a single seed by its ID (here we get seed with ID=3)
seeds = s.get_seeds(3)
print(seeds)
```

Below we show examples for **adding seeds** to the Censys ASM platform.

```
from censys.asm import Seeds

s = Seeds()

# Add a list of seeds. To add a single seed, just pass a list containing one seed.
# Here, we add two ASN seeds.
seed_list = [
    {"type": "ASN", "value": 99998, "label": "seed-test-label"},
    {"type": "ASN", "value": 99999, "label": "seed-test-label"},
]
s.add_seeds(seed_list)

# Add a list of seeds, replacing existing seeds with a specified label
# Here, all seeds with label="seed-test-label" will be removed and then
# Seeds 99996 and 99997 will be added.
```

(continues on next page)

(continued from previous page)

```
seed_list = [{"type": "ASN", "value": 99996}, {"type": "ASN", "value": 99997}]
s.replace_seeds_by_label("seed-test-label", seed_list)
```

Below we show examples for **deleting seeds** from the Censys ASM platform.

```
from censys.asm import Seeds

s = Seeds()

# Delete all seeds with a specified label
# Here we delete all seeds with label="seed-test-label"
s.delete_seeds_by_label("seed-test-label")

# Delete a seed by its ID
# Here, a seed with ID=224 will be deleted.
s.delete_seed_by_id(224)
```

1.5.2 Assets

There are four types of assets (Hosts, Certificates, Domains, and Subdomains). Each asset type shares the same API interface so we will use a mixture of asset types in the following examples.

Below we show examples for **viewing assets** on the Censys ASM platform.

```
from censys.asm import HostsAssets

h = HostsAssets()

# Get a generator that returns hosts
hosts = h.get_assets()
print(next(hosts))

# Get a single host by ID (here we get host with ID="0.0.0.0")
host = h.get_asset_by_id("0.0.0.0")
print(host)
```

Below we show examples for **managing asset comments** via the ASM API.

```
from censys.asm import DomainsAssets

d = DomainsAssets()

# Get a generator that returns all comments on a specific domain asset
# Here we get all comments on the domain with ID="my_domain.com"
comments = d.get_comments("my_domain.com")
print(next(comments))

# Get a single comment on a specific domain by comment ID
# Here we look at domain with ID="my_domain.com" and get comment with ID=3
comment = d.get_comment_by_id("my_domain.com", 3)
print(comment)
```

(continues on next page)

(continued from previous page)

```
# Add a comment to a domain asset
# Here we add comment "hello world" to domain with ID="my_domain.com"
d.add_comment("my_domain.com", "hello world")
```

Below we show examples for **managing asset tags** via the ASM API.

```
from censys.asm import CertificatesAssets

c = CertificatesAssets()
cert_sha = "0006afc1ddc8431aa57c812adf028ab4f168b25bf5f06e94af86edbafa88dfe0"

# Add a tag to a certificate asset
# Here we add tag "New" to certificate with ID=cert_sha
c.add_tag(cert_sha, "New")

# We can optionally give the tag a hexadecimal color where the default=#ffffff (white)
# Here we add a blue tag "New-2" to certificate with ID=cert_sha
c.add_tag(cert_sha, "New-2", color="#0011ff")

# Delete a tag by tag name
# Here we delete tag name="New" from certificate with ID=cert_sha
c.delete_tag(cert_sha, "New")
```

Below we show examples for **subdomain asset tags** via the ASM API.

```
from censys.asm import AsmClient

client = AsmClient()

sub = client.get_subdomains("my_domain.com")

# Add a tag to a subdomain under my_domain.com
sub.add_tag("sub.my_domain.com", "New")
```

1.5.3 Logbook

Note: Note that all timestamp fields in logbook operations use **ISO-8601** format. This is the full list of event types that can be used as filters:

- CERT
- CERT_RISK
- DOMAIN
- DOMAIN_EXPIRATION_DATE
- DOMAIN_MAIL_EXCHANGE_SERVER
- DOMAIN_NAME_SERVER
- DOMAIN_REGISTRAR
- DOMAIN_RISK

- DOMAIN_SUBDOMAIN
- HOST
- HOST_CERT
- HOST_PORT
- HOST_PROTOCOL
- HOST_RISK
- HOST_SOFTWARE
- HOST_VULNERABILITY

Below we show examples for **creating a logbook cursor** for retrieving filtered events.

```
from censys.asm import Logbook

l = Logbook()

# Get a logbook cursor beginning at timestamp "2020-04-22T06:55:01.000Z"
cursor = l.get_cursor("2020-04-22T06:55:01.000Z")
print(cursor)

# Get a logbook cursor beginning at event ID=10
cursor = l.get_cursor(10)
print(cursor)

# Get a logbook cursor that filters on events of type "CERT" and "CERT_RISK"
cursor = l.get_cursor(filters=["CERT", "CERT_RISK"])
print(cursor)

# Get a logbook cursor combining previous start ID and filters
cursor = l.get_cursor(10, filters=["CERT", "CERT_RISK"])
print(cursor)
```

Below we show examples for **getting logbook events.**

```
from censys.asm import Logbook

l = Logbook()

# Get a generator that returns all events
events = l.get_events()
print(next(events))

# Get events based off cursor specifications
events = l.get_events(cursor)
print(next(events))
```

1.5.4 Risks

Below we show an example of **getting risk instances**.

```
from censys.asm import Risks

r = Risks()

# Get risk events
risk_events = r.get_risk_events()
print(risk_events)

# Get a dict that returns all risk instances
risk_instances = r.get_risk_instances()
print(risk_instances)

# Get a single risk instance by ID
risk_instance = r.get_risk_instance(1)
print(risk_instance)

# Get risk types
risk_types = r.get_risk_types()
print(risk_types)

# Get a single risk type by ID
risk_type = r.get_risk_type("missing-common-security-headers")
print(risk_type)
```

1.5.5 InventorySearch

Below we show an example of **searching for assets**.

```
from censys.asm import InventorySearch

i = InventorySearch()

# Get a dict that contains a list of hits for a search query with pagination
assets = i.search(workspaces=["my_workspace"], query="host.services.http.response.body: /
↪.*test.*/")
print(assets)

# Aggregate search results by a field
aggregation = i.aggregate(workspaces=["my_workspace"], query="host.location.country_
↪code: 'US'", aggregation={"term": {"field": "host.autonomous_system.bgp_prefix",
↪"numberOfBuckets": 50}})
print(aggregation)

# Get list of all available fields
fields = i.fields()
print(fields)
```

1.5.6 SavedQueries

Below we show an example of **managing saved queries**.

```
from censys.asm import SavedQueries

s = SavedQueries()

# Get a dict that contains a list of saved queries
saved_queries = s.get_saved_queries()
print(saved_queries)

# Get a single saved query by ID
saved_query = s.get_saved_query_by_id("query_id")
print(saved_query)

# Add a saved query
saved_query = s.add_saved_query("my_saved_query", "host.services.http.response.body: /.
↪*test.*/")
print(saved_query)

# Update a saved query
saved_query = s.edit_saved_query_by_id("query_id", "my_updated_saved_query", "host.
↪services.http.response.body: /.*test.*/")
print(saved_query)

# Delete a saved query
s.delete_saved_query_by_id("query_id")
```

1.5.7 AsmClient

The Censys AsmClient wraps the Seeds, Assets, and Events classes into a single object. It can be used as a single point of interaction for all three APIs.

Below we show how to initialize the AsmClient class object as well as a couple examples of its use. Note that with the AsmClient object, all Seeds, Assets, and Event operations can be accessed in a similar way as the individual APIs above.

```
from censys.asm import AsmClient

client = AsmClient()

# Get all seeds
seeds = client.seeds.get_seeds()
print(seeds)

# Get all domain assets
domains = client.domains.get_assets()
print(next(domains))

# Get all logbook events
logbook_events = client.logbook.get_events()
print(next(logbook_events))
```

1.5.8 Exceptions

1.6 Advanced Usage

1.6.1 Proxies

If you need to use a proxy, you can configure resource indexes with the proxies argument:

```
from censys.search import CensysHosts

proxies = {
    "https": "http://10.10.1.10:1080",
}

c = CensysHosts(proxies=proxies)

c.account()
```

Note: HTTP proxies will be ignored in favor of HTTPS proxies.

See Requests Proxies for more information on the format of proxies.

CHAPTER

TWO

THE API DOCUMENTATION

2.1 API Reference

2.1.1 censys.asm package

An easy-to-use and lightweight API wrapper for Censys ASM (app.censys.io).

class censys.asm.**AsmClient**(*api_key: str | None = None*, ***kwargs*)

Bases: object

Client ASM API class.

class censys.asm.**Assets**(*asset_type: str*, **args*, ***kwargs*)

Bases: *CensysAsmAPI*

Assets API class.

add_comment(*asset_id: str*, *comment: str*) → dict

Adds a comment to a specified asset on the ASM platform.

Parameters

- **asset_id** (*str*) – Asset ID to add comment to.
- **comment** (*str*) – New comment text.

Returns
Added comment results.

Return type
dict

add_tag(*asset_id: str*, *name: str*, *color: str | None = None*) → dict

Adds a tag to a specified asset on the ASM platform.

Parameters

- **asset_id** (*str*) – Asset ID to add tag to.
- **name** (*str*) – New tag name.
- **color** (*str*) – Optional; New tag color (hex).

Returns
Added tag results.

Return type
dict

asset_type: *str*

delete_comment(*asset_id: str*, *comment_id: int*) → dict

Deletes a comment from a specified asset on the ASM platform by comment ID.

Parameters

- **asset_id** (*str*) – Asset ID to delete comment from.
- **comment_id** (*int*) – Comment ID to delete.

Returns
Deleted comment results.

Return type
dict

delete_tag(*asset_id: str*, *name: str*) → dict

Deletes a tag from a specified asset on the ASM platform by tag name.

Parameters

- **asset_id** (*str*) – Asset ID to delete tag from.
- **name** (*str*) – Tag name to delete.

Returns
Deleted tag results.

Return type
dict

get_asset_by_id(*asset_id: str*) → dict

Requests asset data by ID.

Parameters
asset_id (*str*) – Requested asset ID.

Returns
Asset search result.

Return type
dict

get_assets(*page_number: int = 1*, *page_size: int | None = None*, *tag: List[str] | None = None*, *tag_operator: str | None = None*, *source: List[str] | None = None*, *discovery_trail: bool | None = None*) → Iterator[dict]

Requests assets data.

Parameters

- **page_number** (*int*) – Optional; Page number to begin at when searching.
- **page_size** (*int*) – Optional; Page size for retrieving assets.
- **tag** (*list*) – Optional; List of tags to search for.
- **tag_operator** (*str*) – Optional; Operator to use when searching for tags.
- **source** (*list*) – Optional; List of sources to search for.
- **discovery_trail** (*bool*) – Optional; Bool indicating whether to return discovery trail.

Yields
dict – The assets result returned.

get_comment_by_id(*asset_id: str*, *comment_id: int*) → dict

Requests a comment on a specified asset by comment ID.

Parameters

- **asset_id** (*str*) – Asset ID for requested comments.
- **comment_id** (*int*) – Requested comment ID.

Returns
Comment search result.

Return type
dict

get_comments(*asset_id: str*, *page_number: int = 1*, *page_size: int | None = None*) → Iterator[dict]

Requests comments on a specified asset.

Parameters

- **asset_id** (*str*) – Asset ID for requested comments.
- **page_number** (*int*) – Optional; Page number to begin at when searching.
- **page_size** (*int*) – Optional; Page size for retrieving comments.

Returns
Comment search results.

Return type
generator

class censys.asm.**Beta**(*api_key: str | None = None*, ***kwargs*)

Bases: *CensysAsmAPI*

Beta API class.

add_cloud_assets(*cloud_connector_uid: str*, *cloud_assets: List[dict]*)

Add cloud assets.

Parameters

- **cloud_connector_uid** (*str*) – Cloud connector UID.
- **cloud_assets** (*List[dict]*) – Cloud assets.

Returns
Add cloud assets result.

Return type
dict

base_path = '/beta'

get_asset_counts(*since: str | date | datetime*, *environment: str*, *asset_type: str*)

Retrieve asset counts.

Parameters

- **since** (*Datetime*) – Date to include assets from.
- **environment** (*str*) – Environment to include assets from.
- **asset_type** (*str*) – Asset type to include.

Returns
Asset count result.

Return type
dict

`get_host_counts_by_country`(*since: str | date | datetime*, *environment: str*)

Retrieve host counts by country.

Parameters

- **since** (*Datetime*) – Date to include hosts from.
- **environment** (*str*) – Environment to include hosts from.

Returns
Host count result.

Return type
dict

`get_input_assets`(*page_number: int = 1*, *page_size: int | None = None*)

Retrieve input assets.

Parameters

- **page_number** (*int*) – Optional; Page number to begin at when searching.
- **page_size** (*int*) – Optional; Page size for retrieving assets.

Returns
Input assets result.

Return type
dict

`get_logbook_data`(*filters: dict | None = None*, *cursor: str | None = None*)

Retrieve logbook data.

Parameters

- **filters** (*dict*) – Optional; Filter parameters.
- **cursor** (*str*) – Optional; Cursor for pagination.

Returns
Logbook data result.

Return type
dict

`get_user_workspaces`(*user_uuid: str*)

Retrieve user workspaces.

Parameters
user_uuid (*str*) – User UUID.

Returns
User workspaces result.

Return type
dict

class censys.asm.**CertificatesAssets**(**args*, ***kwargs*)

Bases: *Assets*

Certificates Assets API class.

class censys.asm.**Clouds**(*api_key: str | None = None*, ***kwargs*)

Bases: *CensysAsmAPI*

Clouds API class.

base_path = '/v1/clouds'

get_domain_counts(*since: str | date | datetime*) → dict

Retrieve domain counts by cloud.

Parameters
since (*Datetime*) – Date to include domains from.

Returns
Domain count result.

Return type
dict

get_host_counts(*since: str | date | datetime*) → dict

Retrieve host counts by cloud.

Hosts found after the date provided in the *since* parameter will be included in the new asset counts.

Parameters
since (*Datetime*) – Date to include hosts from.

Returns
Host count result.

Return type
dict

get_object_store_counts(*since: str | date | datetime*) → dict

Retrieve object store counts by cloud.

Parameters
since (*Datetime*) – Date to include object stores from.

Returns
Object store count result.

Return type
dict

get_subdomain_counts(*since: str | date | datetime*) → dict

Retrieve subdomain counts by cloud.

Parameters
since (*Datetime*) – Date to include subdomains from.

Returns
Subdomain count result.

Return type
dict

get_unknown_counts() → dict

Retrieve known and unknown counts for hosts by cloud.

Returns
Unknown count result.

Return type
dict

class censys.asm.**DomainsAssets**(**args*, ***kwargs*)

Bases: *Assets*

Domains Assets API class.

get_subdomains(*domain: str*, *page_number: int = 1*, *page_size: int | None = None*) → Iterator[dict]

List all subdomains of the parent domain.

Parameters

- **domain** – (str): Parent domain to query.
- **page_number** (*int*) – Optional; Page number to begin at when searching.
- **page_size** (*int*) – Optional; Page size for retrieving assets.

Yields
dict – The assets result returned.

censys.asm.**Events**

alias of *Logbook*

class censys.asm.**HostsAssets**(**args*, ***kwargs*)

Bases: *Assets*

Hosts Assets API class.

class censys.asm.**InventorySearch**(*api_key: str | None = None*, ***kwargs*)

Bases: *CensysAsmAPI*

Inventory Search API class.

aggregate(*workspaces: List[str]*, *query: str | None = None*, *aggregation: dict | None = None*) → dict

Aggregate inventory data.

Parameters

- **workspaces** (*List[str]*) – List of workspace IDs to search.
- **query** (*str, optional*) – Query string.
- **aggregation** (*dict, optional*) – Aggregation object.

Returns
Inventory aggregation results.

Return type
dict

base_path = '/inventory/v1'

fields(*fields: List[str] | None = None*) → dict

List inventory fields.

If no fields are specified, all fields will be returned.

Parameters
fields (*List[str], optional*) – List of fields to return.

Returns
Inventory field results.

Return type
dict

search(*workspaces: List[str] | None = None, query: str | None = None, page_size: int | None = None, cursor: str | None = None, sort: List[str] | None = None, fields: List[str] | None = None, pages: int | None = None*) → dict

Search inventory data.

Parameters

- **workspaces** (*List[str], optional*) – List of workspace IDs to search. Deprecated. The workspace associated with *CENSYS-API-KEY* will be used automatically.
- **query** (*str, optional*) – Query string.
- **page_size** (*int, optional*) – Number of results to return. Defaults to 50.
- **cursor** (*str, optional*) – Cursor to start search from.
- **sort** (*List[str], optional*) – List of fields to sort by.
- **fields** (*List[str], optional*) – List of fields to return.
- **pages** (*int, optional*) – Number of pages of results to return (when set to -1 returns all pages available).

Returns
Inventory search results.

Return type
dict

class censys.asm.**Logbook**(*api_key: str | None = None, **kwargs*)

Bases: *CensysAsmAPI*

Logbook API class.

base_path = '/v1/logbook'

get_cursor(*start: datetime | int | None = None, filters: List[str] | None = None*) → str

Requests a logbook cursor.

Parameters

- **start** (*[datetime.datetime, int]*) – Optional; Timestamp or event ID to begin searching.
- **filters** (*list*) – Optional; List of filters applied to logbook search results.

Returns
Cursor result.

Return type
str

get_events(*cursor: str | None = None*) → Iterator[dict]

Requests logbook events from inception or from the provided cursor.

Parameters
cursor (*str*) – Optional; Logbook cursor.

Yields
dict – Logbook event.

class censys.asm.**ObjectStoragesAssets**(**args*, ***kwargs*)

Bases: *Assets*

Object Storage Assets API class.

Please note that the Object Storage Assets API is currently in beta and is subject to change.

class censys.asm.**Risks**(*api_key: str | None = None*, ***kwargs*)

Bases: *CensysAsmAPI*

Risks API class.

base_path = '/v2/risk'

get_risk_events(*start: str | None = None*, *end: str | None = None*, *after_id: int | None = None*, *limit: int | None = None*, *cursor: str | None = None*, *accept: str | None = None*) → dict

Retrieve risk events.

Parameters

- **start** (*str*) – Optional; Starting event time, inclusive (in RFC3339 format).
- **end** (*str*) – Optional; Ending event time, inclusive (in RFC3339 format).
- **after_id** (*int*) – Optional; Risk event ID to query for events after.
- **limit** (*int*) – Optional; Max number of events to return.
- **cursor** (*str*) – Optional; Cursor value to continue collecting events started in a previous request.
- **accept** (*str*) – Optional; Accept header.

Returns
Risk events result.

Return type
dict

get_risk_instance(*risk_instance_id: int*, *include_events: bool | None = None*) → dict

Retrieve a risk instance.

Parameters

- **risk_instance_id** (*int*) – Risk instance ID.
- **include_events** (*bool*) – Optional; Whether to include events.

Returns
Risk instance result.

Return type
dict

get_risk_instances(*include_events: bool | None = None*, *accept: str | None = None*) → dict

Retrieve risk instances.

Parameters

- **include_events** (*bool*) – Optional; Whether to include events.
- **accept** (*str*) – Optional; Accept header.

Returns
Risk instances result.

Return type
dict

get_risk_type(*risk_type: str*, *include_events: bool | None = None*) → dict

Retrieve a risk type.

Parameters

- **risk_type** (*str*) – Risk type.
- **include_events** (*bool*) – Optional; Whether to include events.

Returns
Risk type result.

Return type
dict

get_risk_types(*limit: int | None = None*, *page: int | None = None*, *sort: List[str] | None = None*, *include_events: bool | None = None*, *accept: str | None = None*) → dict

Retrieve risk types.

Parameters

- **limit** (*int, optional*) – Maximum number of results to return. Defaults to 1000.
- **page** (*int, optional*) – Page number to begin at when searching. Defaults to 1.
- **sort** (*list*) – Optional; Sort by field(s).
- **include_events** (*bool*) – Optional; Whether to include events.
- **accept** (*str*) – Optional; Accept header.

Returns
Risk types result.

Return type
dict

patch_risk_instance(*risk_instance_id: int*, *data: dict*) → dict

Patch a risk instance.

Parameters

- **risk_instance_id** (*int*) – Risk instance ID.
- **data** (*dict*) – Risk instance data.

Returns
Risk instance result.

Return type
dict

patch_risk_instances(*data: dict*) → dict

Patch risk instances.

Parameters
data (*dict*) – Risk instances data.

Returns
Risk instances result.

Return type
dict

patch_risk_type(*risk_type: str*, *data: dict*) → dict

Patch a risk type.

Parameters

- **risk_type** (*str*) – Risk type.
- **data** (*dict*) – Risk type data.

Returns
Risk type result.

Return type
dict

risk_events_path = '/v2/risk-events'

risk_instances_path = '/v2/risk-instances'

risk_types_path = '/v2/risk-types'

search_risk_instances(*data: dict*, *accept: str | None = None*) → dict

Search risk instances.

Parameters

- **data** (*dict*) – Query data.
- **accept** (*str*) – Optional; Accept header.

Returns
Risk instances result.

Return type
dict

class censys.asm.**SavedQueries**(*api_key: str | None = None*, ***kwargs*)

Bases: *CensysAsmAPI*

Saved Queries API class.

add_saved_query(*query: str*, *query_name: str*) → dict

Add a new saved query to the ASM platform.

Parameters

- **query** (*str*) – Query string.
- **query_name** (*str*) – Saved query name.

Returns
Added saved query results.

Return type
dict

base_path = '/inventory/v1/saved-query'

delete_saved_query_by_id(*query_id: str*) → dict

Delete saved query by query ID.

Parameters
query_id (*str*) – The saved query's ID.

Returns
Delete results.

Return type
dict

edit_saved_query_by_id(*query_id: str*, *query: str*, *query_name: str*) → dict

Edit an existing saved query by query ID.

Parameters

- **query_id** (*str*) – The saved query's ID.
- **query** (*str*) – New query string.
- **query_name** (*str*) – New saved query name.

Returns
Edited saved query result.

Return type
dict

get_saved_queries(*query_name_prefix: str | None = None*, *page_size: int | None = None*, *page: int | None = None*, *filter_term: str | None = None*) → dict

Get saved queries.

Parameters

- **query_name_prefix** (*str, optional*) – Prefix for the saved query name.
- **page_size** (*int, optional*) – Number of results to return. Defaults to 50.
- **page** (*int, optional*) – Page number to begin at when searching. Defaults to 1.
- **filter_term** (*str, optional*) – Term used to filter the list of saved query names and the saved queries.

Returns
Saved queries results.

Return type
dict

get_saved_query_by_id(*query_id: str*) → dict

Get saved query by query ID.

Parameters
query_id (*str*) – The saved query's ID.

Returns
Saved query result.

Return type
dict

class censys.asm.Seeds(*api_key: str | None = None*, ***kwargs*)

Bases: *CensysAsmAPI*

Seeds API class.

add_seeds(*seeds: list*, *force: bool | None = None*) → dict

Add seeds to the ASM platform.

Parameters

- **seeds** (*list*) – List of seed objects to add.
- **force** (*bool, optional*) – Forces replace operation.

Returns
Added seeds results.

Return type
dict

base_path = '/v1/seeds'

delete_seed_by_id(*seed_id: int*) → dict

Delete a seed in the ASM platform by id.

Parameters
seed_id (*int*) – Seed ID to delete by.

Returns
Delete results.

Return type
dict

delete_seeds_by_label(*label: str*) → dict

Delete seeds in the ASM platform by label.

Parameters
label (*str*) – Label name to delete by.

Returns
Delete results.

Return type
dict

get_seed_by_id(*seed_id: int*) → dict

Requests seed data by ID.

Parameters
seed_id (*int*) – Seed ID to get.

Returns
Seed search result.

Return type
dict

get_seeds(*seed_type: str | None = None*, *label: str | None = None*) → List[dict]

Requests seed data.

Parameters

- **seed_type** (*str*) – Optional; Seed type [‘IP_ADDRESS’, ‘DOMAIN_NAME’, ‘CIDR’, ‘ASN’].
- **label** (*str*) – Optional; Seed label.

Returns
Seed search results.

Return type
List[dict]

replace_seeds_by_label(*label: str*, *seeds: list*, *force: bool | None = None*) → dict

Replace seeds in the ASM platform by label.

Parameters

- **label** (*str*) – Label name to replace by.
- **seeds** (*list*) – List of seed objects to add.
- **force** (*bool*) – Optional; Forces replace operation.

Returns
Added and removed seeds results.

Return type
dict

class censys.asm.**SubdomainsAssets**(**args*, ***kwargs*)

Bases: *Assets*

Subdomains Assets API class.

get_assets(*page_number: int = 1*, *page_size: int | None = None*, *tag: List[str] | None = None*, *tag_operator: str | None = None*, *source: List[str] | None = None*, *discovery_trail: bool | None = None*) → Iterator[dict]

Requests assets data.

Override for subdomains due to it return value in a different key.

Parameters

- **page_number** (*int*) – Optional; Page number to begin at when searching.
- **page_size** (*int*) – Optional; Page size for retrieving assets.
- **tag** (*list*) – Optional; List of tags to search for.
- **tag_operator** (*str*) – Optional; Operator to use when searching for tags.
- **source** (*list*) – Optional; List of sources to search for.
- **discovery_trail** (*bool*) – Optional; Bool indicating whether to return discovery trail.

Yields
dict – The assets result returned.

class censys.asm.**WebEntitiesAssets**(**args*, ***kwargs*)

Bases: *Assets*

Web Entities Assets API class.

get_assets(**args*, ***kwargs*)

Requests assets data.

This method is not implemented for web entities. Please see the inventory search and aggregation API.

Parameters

- ***args** – Variable length argument list.
- ****kwargs** – Arbitrary keyword arguments.

Raises

NotImplementedError – This method is not implemented.

get_instances(*name_and_port: str*, *page_size: int | None = None*, *cursor: str | None = None*) → Iterator[dict]

List all instances of the web entity.

Parameters

- **name_and_port** – (str): Web entity to query.
- **page_size** (*int*) – Optional; Page size for retrieving assets.
- **cursor** (*str*) – Optional; Cursor to use for pagination.

Yields

dict – The assets result returned.

censys.asm.api module

Base for interacting with the Censys ASM API.

class censys.asm.api.**CensysAsmAPI**(*api_key: str | None = None*, ***kwargs*)

Bases: *CensysAPIBase*

This is the base class for ASM's Seeds, Assets, and Events classes.

DEFAULT_URL: str = 'https://app.censys.io/api'

Default ASM API base URL.

get_workspace_id() → str

Get the workspace ID.

Returns

The workspace ID.

Return type

str

2.1.2 censys.common package

Common Code for the Censys Python SDK.

censys.common.base module

Base for interacting with the Censys APIs.

class censys.common.base.**CensysAPIBase**(*url: str | None = None*, *timeout: int | None = 30*, *max_retries: int | None = 5*, *user_agent: str | None = 'censys-python/2.2.19'*, *proxies: dict | None = None*, *cookies: dict | None = None*, ***kwargs*)

Bases: object

This is the base class for API queries.

DEFAULT_MAX_RETRIES: int = 5

Default max number of API retries.

DEFAULT_TIMEOUT: int = 30

Default API timeout.

DEFAULT_USER_AGENT: str = 'censys-python/2.2.19'

Default API user agent.

property **request_id: str | None**

The x-request-id header value for API requests.

The x-request-id header is not set when the value is None. Value is None by default

Returns
The value of the header.

Return type
Type[Optional[str]]

censys.common.config module

Interact with the config file.

censys.common.config.**get_config**() → ConfigParser

Reads and returns config.

Returns
Config for Censys.

Return type
configparser.ConfigParser

censys.common.config.**get_config_path**() → str

Returns the path to the config file.

Returns
Path to config file.

Return type
str

censys.common.config.**write_config**(*config: ConfigParser*) → None

Writes config to file.

Parameters
config (*configparser.ConfigParser*) – Configuration to write.

Raises
PermissionError – If the config file is not writable.

censys.common.types module

Common types for the Censys Python SDK.

censys.common.utils module

Common utilities for the Censys Python SDK.

censys.common.utils.**format_iso8601**(*time: str | date | datetime*) → str

Formats a datetime object into an ISO8601 string.

Parameters
time (*Datetime*) – Datetime object to format.

Returns
ISO8601 formatted string.

Return type
str

censys.common.utils.**format_rfc3339**(*time: str | date | datetime*) → str

Formats a datetime object into an RFC3339 string.

Parameters
time (*Datetime*) – Datetime object to format.

Returns
RFC3339 formatted string.

Return type
str

censys.common.exceptions module

Exceptions for Censys.

exception censys.common.exceptions.**CensysAPIException**(*status_code: int*, *message: str*, *body: str | None = None*, *const: str | None = None*, *error_code: int | None = None*, *details: str | None = None*)

Bases: *CensysException*

Base Exception for Censys APIs.

exception censys.common.exceptions.**CensysAppDownForMaintenanceException**(*status_code: int*, *message: str*, *body: str | None = None*, *const: str | None = None*, *error_code: int | None = None*, *details: str | None = None*)

Bases: *CensysAsmException*

Exception raised when the ASM API is down for maintenance.

exception censys.common.exceptions.**CensysAsmException**(*status_code: int*, *message: str*, *body: str | None = None*, *const: str | None = None*, *error_code: int | None = None*, *details: str | None = None*)

Bases: *CensysAPIException*

Base Exception for the Censys ASM API.

exception censys.common.exceptions.**CensysAsmUnauthorizedException**(*status_code: int*, *message: str*, *body: str | None = None*, *const: str | None = None*, *error_code: int | None = None*, *details: str | None = None*)

Bases: *CensysAsmException*

Exception raised when the ASM API is unauthorized.

exception censys.common.exceptions.**CensysAssetExcludedException**(*status_code: int*, *message: str*, *body: str | None = None*, *const: str | None = None*, *error_code: int | None = None*, *details: str | None = None*)

Bases: *CensysAsmException*

Exception raised when the asset is excluded.

exception censys.common.exceptions.**CensysAssetNotFoundException**(*status_code: int*, *message: str*, *body: str | None = None*, *const: str | None = None*, *error_code: int | None = None*, *details: str | None = None*)

Bases: *CensysAsmException*

Exception raised when the asset is not found.

exception censys.common.exceptions.**CensysAssociatedAssetsThresholdWarningException**(*status_code: int*, *message: str*, *body: str | None = None*, *const: str | None = None*, *error_code: int | None = None*, *details: str | None = None*)

Bases: *CensysAsmException*

Exception raised when the asset count is within the warning threshold.

exception censys.common.exceptions.**CensysBadJSONBodyException**(*status_code: int*, *message: str*, *body: str | None = None*, *const: str | None = None*, *error_code: int | None = None*, *details: str | None = None*)

Bases: *CensysAsmException*

Exception raised when a bad JSON string is in the body.

exception censys.common.exceptions.**CensysCLIException**

Bases: *CensysException*

Exception raised when the CLI is passed invalid arguments.

exception censys.common.exceptions.**CensysCannotCreateTagWithNewColorException**(*status_code: int*, *message: str*, *body: str | None = None*, *const: str | None = None*, *error_code: int | None = None*, *details: str | None = None*)

Bases: *CensysAsmException*

Exception raised when the specified tag cannot be created with a new color.

exception censys.common.exceptions.**CensysCannotRemoveNonExistentSeedsException**(*status_code: int*, *message: str*, *body: str | None = None*, *const: str | None = None*, *error_code: int | None = None*, *details: str | None = None*)

Bases: *CensysAsmException*

Exception raised when trying to remove non existent seed nodes.

exception censys.common.exceptions.**CensysCannotRemoveNonSeedsException**(*status_code: int*, *message: str*, *body: str | None = None*, *const: str | None = None*, *error_code: int | None = None*, *details: str | None = None*)

Bases: *CensysAsmException*

Exception raised when trying to remove non seed nodes.

exception censys.common.exceptions.**CensysCertificateNotFoundException**(*status_code: int*, *message: str*, *body: str | None = None*, *const: str | None = None*, *error_code: int | None = None*, *details: str | None = None*)

Bases: *CensysAsmException*

Exception raised when the certificate is not found.

exception censys.common.exceptions.**CensysCommentNotFoundException**(*status_code: int*, *message: str*, *body: str | None = None*, *const: str | None = None*, *error_code: int | None = None*, *details: str | None = None*)

Bases: *CensysAsmException*

Exception raised when the requested comment is not found.

exception censys.common.exceptions.**CensysDomainNotFoundException**(*status_code: int*, *message: str*, *body: str | None = None*, *const: str | None = None*, *error_code: int | None = None*, *details: str | None = None*)

Bases: *CensysAsmException*

Exception raised when the domain is not found.

exception censys.common.exceptions.**CensysException**

Bases: Exception

Base Exception for Censys.

class censys.common.exceptions.**CensysExceptionMapper**

Bases: object

Map status code to Exception for the ASM and Search API.

```
ASM_EXCEPTIONS: Dict[int, Type[CensysAsmException]] = {10000:  <class
'censys.common.exceptions.CensysMissingApiKeyException'>, 10001:  <class
'censys.common.exceptions.CensysInvalidAPIKeyException'>, 10002:  <class
'censys.common.exceptions.CensysInvalidAuthTokenException'>, 10006:  <class
'censys.common.exceptions.CensysAsmUnauthorizedException'>, 10007:  <class
'censys.common.exceptions.CensysInvalidSeedDataException'>, 10008:  <class
'censys.common.exceptions.CensysInvalidRequestException'>, 10011:  <class
'censys.common.exceptions.CensysCannotRemoveNonSeedsException'>, 10012:  <class
'censys.common.exceptions.CensysCannotRemoveNonExistentSeedsException'>, 10013:
<class
'censys.common.exceptions.CensysNeedConfirmationToRemoveParentSeedsException'>,
10014:  <class 'censys.common.exceptions.CensysSeedNotFoundException'>, 10015:
<class 'censys.common.exceptions.CensysNotASeedException'>, 10016:  <class
'censys.common.exceptions.CensysTooManyInputNodesException'>, 10017:  <class
'censys.common.exceptions.CensysAssociatedAssetsThresholdWarningException'>, 10018:
<class 'censys.common.exceptions.CensysHostNotFoundException'>, 10019:  <class
'censys.common.exceptions.CensysDomainNotFoundException'>, 10020:  <class
'censys.common.exceptions.CensysCertificateNotFoundException'>, 10021:  <class
'censys.common.exceptions.CensysInvalidIPv4AddressException'>, 10022:  <class
'censys.common.exceptions.CensysAssetExcludedException'>, 10025:  <class
'censys.common.exceptions.CensysTagHasTrailingOrLeadingWhitespaceException'>, 10026:
<class 'censys.common.exceptions.CensysTagIsEmptyStringException'>, 10027:  <class
'censys.common.exceptions.CensysTagLabelsDifferOnlyInCasingException'>, 10028:
<class 'censys.common.exceptions.CensysTagLabelTooLongException'>, 10029:  <class
'censys.common.exceptions.CensysAppDownForMaintenanceException'>, 10034:  <class
'censys.common.exceptions.CensysTagColorTooLongException'>, 10035:  <class
'censys.common.exceptions.CensysCannotCreateTagWithNewColorException'>, 10036:
<class
'censys.common.exceptions.CensysTagColorHasTrailingOrLeadingWhitespaceException'>,
10037:  <class 'censys.common.exceptions.CensysInvalidColorException'>, 10038:
<class 'censys.common.exceptions.CensysInvalidSeedTypeException'>, 10039:  <class
'censys.common.exceptions.CensysTooManyRequestsException'>, 10040:  <class
'censys.common.exceptions.CensysInvalidLogbookCursorException'>, 10045:  <class
'censys.common.exceptions.CensysTeamNotFoundException'>, 10050:  <class
'censys.common.exceptions.CensysInvalidPageSizeException'>, 10051:  <class
'censys.common.exceptions.CensysPageNumberOutOfRangeException'>, 10054:  <class
'censys.common.exceptions.CensysInvalidCommentException'>, 10055:  <class
'censys.common.exceptions.CensysCommentNotFoundException'>, 10057:  <class
'censys.common.exceptions.CensysSubdomainNotFoundException'>, 10059:  <class
'censys.common.exceptions.CensysInvalidCloudAssetDataException'>, 10060:  <class
'censys.common.exceptions.CensysInvalidObjectStorageAssetIdentifierException'>,
10061:  <class
'censys.common.exceptions.CensysInvalidObjectStorageAssetNotFoundException'>, 10067:
<class 'censys.common.exceptions.CensysBadJSONBodyException'>, 10073:  <class
'censys.common.exceptions.CensysRiskNotFoundException'>, 10078:  <class
'censys.common.exceptions.CensysInvalidDateException'>, 10082:  <class
'censys.common.exceptions.CensysInvalidCloudException'>, 10086:  <class
'censys.common.exceptions.CensysAssetNotFoundException'>, 10091:  <class
'censys.common.exceptions.CensysInvalidKeywordsInBodyException'>, 10096:  <class
'censys.common.exceptions.CensysSearchAPITimeoutException'>, 10097:  <class
'censys.common.exceptions.CensysSearchAPIErrorException'>, 10098:  <class
'censys.common.exceptions.CensysInternalServerErrorException'>, 10099:  <class
'censys.common.exceptions.CensysInvalidCommentHTMLException'>, 10107:  <class
'censys.common.exceptions.CensysInvalidSearchAPIResponseException'>, 10120:  <class
'censys.common.exceptions.CensysTooSoonToResendInviteException'>}
```

Map of status code to ASM Exception.

SEARCH_EXCEPTIONS: Dict[int, Type[*CensysSearchException*]] = {401: <class 'censys.common.exceptions.CensysUnauthorizedException'>, 403: <class 'censys.common.exceptions.CensysUnauthorizedException'>, 404: <class 'censys.common.exceptions.CensysNotFoundException'>, 429: <class 'censys.common.exceptions.CensysRateLimitExceededException'>, 500: <class 'censys.common.exceptions.CensysInternalServerException'>}

Map of status code to Search Exception.

exception censys.common.exceptions.**CensysHostNotFoundException**(*status_code: int*, *message: str*, *body: str | None = None*, *const: str | None = None*, *error_code: int | None = None*, *details: str | None = None*)

Bases: *CensysAsmException*

Exception raised when the requested host is not found.

exception censys.common.exceptions.**CensysInternalServerErrorException**(*status_code: int*, *message: str*, *body: str | None = None*, *const: str | None = None*, *error_code: int | None = None*, *details: str | None = None*)

Bases: *CensysAsmException*

Exception raised when the search API returns an error.

exception censys.common.exceptions.**CensysInternalServerException**(*status_code: int*, *message: str*, *body: str | None = None*, *const: str | None = None*, *error_code: int | None = None*, *details: str | None = None*)

Bases: *CensysSearchException*

Exception raised when the server encountered an internal error.

exception censys.common.exceptions.**CensysInvalidAPIKeyException**(*status_code: int*, *message: str*, *body: str | None = None*, *const: str | None = None*, *error_code: int | None = None*, *details: str | None = None*)

Bases: *CensysAsmException*

Exception raised when the ASM API key is invalid.

exception censys.common.exceptions.**CensysInvalidAuthTokenException**(*status_code: int*, *message: str*, *body: str | None = None*, *const: str | None = None*, *error_code: int | None = None*, *details: str | None = None*)

Bases: *CensysAsmException*

Exception raised when the auth token is invalid.

exception censys.common.exceptions.**CensysInvalidCloudAssetDataException**(*status_code: int*, *message: str*, *body: str | None = None*, *const: str | None = None*, *error_code: int | None = None*, *details: str | None = None*)

Bases: *CensysAsmException*

Exception raised when invalid cloud asset data is submitted.

exception censys.common.exceptions.**CensysInvalidCloudException**(*status_code: int*, *message: str*, *body: str | None = None*, *const: str | None = None*, *error_code: int | None = None*, *details: str | None = None*)

Bases: *CensysAsmException*

Exception raised when an invalid cloud is submitted.

exception censys.common.exceptions.**CensysInvalidColorException**(*status_code: int*, *message: str*, *body: str | None = None*, *const: str | None = None*, *error_code: int | None = None*, *details: str | None = None*)

Bases: *CensysAsmException*

Exception raised when the specified color is invalid.

exception censys.common.exceptions.**CensysInvalidCommentException**(*status_code: int*, *message: str*, *body: str | None = None*, *const: str | None = None*, *error_code: int | None = None*, *details: str | None = None*)

Bases: *CensysAsmException*

Exception raised when the comment is invalid.

exception censys.common.exceptions.**CensysInvalidCommentHTMLException**(*status_code: int*, *message: str*, *body: str | None = None*, *const: str | None = None*, *error_code: int | None = None*, *details: str | None = None*)

Bases: *CensysAsmException*

Exception raised when the comment contains invalid HTML.

exception censys.common.exceptions.**CensysInvalidDateException**(*status_code: int*, *message: str*, *body: str | None = None*, *const: str | None = None*, *error_code: int | None = None*, *details: str | None = None*)

Bases: *CensysAsmException*

Exception raised when an invalid date is submitted.

exception censys.common.exceptions.**CensysInvalidIPv4AddressException**(*status_code: int*, *message: str*, *body: str | None = None*, *const: str | None = None*, *error_code: int | None = None*, *details: str | None = None*)

Bases: *CensysAsmException*

Exception raised when the IPv4 address is invalid.

exception censys.common.exceptions.**CensysInvalidKeywordsInBodyException**(*status_code: int*, *message: str*, *body: str | None = None*, *const: str | None = None*, *error_code: int | None = None*, *details: str | None = None*)

Bases: *CensysAsmException*

Exception raised when invalid keywords are in the body.

exception censys.common.exceptions.**CensysInvalidLogbookCursorException**(*status_code: int*, *message: str*, *body: str | None = None*, *const: str | None = None*, *error_code: int | None = None*, *details: str | None = None*)

Bases: *CensysAsmException*

Exception raised when the logbook cursor is invalid.

exception censys.common.exceptions.**CensysInvalidObjectStorageAssetIdentifierException**(*status_code: int*, *message: str*, *body: str | None = None*, *const: str | None = None*, *error_code: int | None = None*, *details: str | None = None*)

Bases: *CensysAsmException*

Exception raised when object storage name is not a valid asset URL.

exception censys.common.exceptions.**CensysInvalidObjectStorageAssetNotFoundException**(*status_code: int*, *message: str*, *body: str | None = None*, *const: str | None = None*, *error_code: int | None = None*, *details: str | None = None*)

Bases: *CensysAsmException*

Exception raised when no object storage assets with given URL were found.

exception censys.common.exceptions.**CensysInvalidPageSizeException**(*status_code: int*, *message: str*, *body: str | None = None*, *const: str | None = None*, *error_code: int | None = None*, *details: str | None = None*)

Bases: *CensysAsmException*

Exception raised when the page size is invalid.

exception censys.common.exceptions.**CensysInvalidRequestException**(*status_code: int*, *message: str*, *body: str | None = None*, *const: str | None = None*, *error_code: int | None = None*, *details: str | None = None*)

Bases: *CensysAsmException*

Exception raised when the HTTP request is invalid.

exception censys.common.exceptions.**CensysInvalidSearchAPIResponseException**(*status_code: int*, *message: str*, *body: str | None = None*, *const: str | None = None*, *error_code: int | None = None*, *details: str | None = None*)

Bases: *CensysAsmException*

Exception raised when the search API returns an error.

exception censys.common.exceptions.**CensysInvalidSeedDataException**(*status_code: int*, *message: str*, *body: str | None = None*, *const: str | None = None*, *error_code: int | None = None*, *details: str | None = None*)

Bases: *CensysAsmException*

Exception raised when the seed data is invalid.

exception censys.common.exceptions.**CensysInvalidSeedTypeException**(*status_code: int*, *message: str*, *body: str | None = None*, *const: str | None = None*, *error_code: int | None = None*, *details: str | None = None*)

Bases: *CensysAsmException*

Exception raised when the seed type is invalid.

exception censys.common.exceptions.**CensysJSONDecodeException**(*status_code: int*, *message: str*, *body: str | None = None*, *const: str | None = None*, *error_code: int | None = None*, *details: str | None = None*)

Bases: *CensysSearchException*

Exception raised when the resource requested is not valid JSON.

exception censys.common.exceptions.**CensysMissingApiKeyException**(*status_code: int*, *message: str*, *body: str | None = None*, *const: str | None = None*, *error_code: int | None = None*, *details: str | None = None*)

Bases: *CensysAsmException*

Exception raised when there is no provided ASM API key.

exception censys.common.exceptions.**CensysNeedConfirmationToRemoveParentSeedsException**(*status_code: int*, *message: str*, *body: str | None = None*, *const: str | None = None*, *error_code: int | None = None*, *details: str | None = None*)

Bases: *CensysAsmException*

Exception raised when confirmation is needed to remove seeds with children.

exception censys.common.exceptions.**CensysNotASeedException**(*status_code: int*, *message: str*, *body: str | None = None*, *const: str | None = None*, *error_code: int | None = None*, *details: str | None = None*)

Bases: *CensysAsmException*

Exception raised when the requested resource is not a seed.

exception censys.common.exceptions.**CensysNotFoundException**(*status_code: int*, *message: str*, *body: str | None = None*, *const: str | None = None*, *error_code: int | None = None*, *details: str | None = None*)

Bases: *CensysSearchException*

Exception raised when the resource requested is not found.

exception censys.common.exceptions.**CensysPageNumberOutOfRangeException**(*status_code: int*, *message: str*, *body: str | None = None*, *const: str | None = None*, *error_code: int | None = None*, *details: str | None = None*)

Bases: *CensysAsmException*

Exception raised when the page number is out of range [1 - totalPages].

exception censys.common.exceptions.**CensysRateLimitExceededException**(*status_code: int*, *message: str*, *body: str | None = None*, *const: str | None = None*, *error_code: int | None = None*, *details: str | None = None*)

Bases: *CensysSearchException*

Exception raised when your Censys rate limit has been exceeded.

exception censys.common.exceptions.**CensysRiskNotFoundException**(*status_code: int*, *message: str*, *body: str | None = None*, *const: str | None = None*, *error_code: int | None = None*, *details: str | None = None*)

Bases: *CensysAsmException*

Exception raised when no risks are found with given risk_id.

exception censys.common.exceptions.**CensysSearchAPIErrorException**(*status_code: int*, *message: str*, *body: str | None = None*, *const: str | None = None*, *error_code: int | None = None*, *details: str | None = None*)

Bases: *CensysAsmException*

Exception raised when the search API returns an error.

exception censys.common.exceptions.**CensysSearchAPITimeoutException**(*status_code: int*, *message: str*, *body: str | None = None*, *const: str | None = None*, *error_code: int | None = None*, *details: str | None = None*)

Bases: *CensysAsmException*

Exception raised when the search API times out.

exception censys.common.exceptions.**CensysSearchException**(*status_code: int*, *message: str*, *body: str | None = None*, *const: str | None = None*, *error_code: int | None = None*, *details: str | None = None*)

Bases: *CensysAPIException*

Base Exception for the Censys search API.

exception censys.common.exceptions.**CensysSeedNotFoundException**(*status_code: int*, *message: str*, *body: str | None = None*, *const: str | None = None*, *error_code: int | None = None*, *details: str | None = None*)

Bases: *CensysAsmException*

Exception raised when the requested seed can not be found.

exception censys.common.exceptions.**CensysSubdomainNotFoundException**(*status_code: int*, *message: str*, *body: str | None = None*, *const: str | None = None*, *error_code: int | None = None*, *details: str | None = None*)

Bases: *CensysAsmException*

Exception raised when the requested subdomain is not found.

exception censys.common.exceptions.**CensysTagColorHasTrailingOrLeadingWhitespaceException**(*status_code: int*, *message: str*, *body: str | None = None*, *const: str | None = None*, *error_code: int | None = None*, *details: str | None = None*)

Bases: *CensysAsmException*

Exception raised when the tag color has trailing or leading whitespace.

exception censys.common.exceptions.**CensysTagColorTooLongException**(*status_code: int*, *message: str*, *body: str | None = None*, *const: str | None = None*, *error_code: int | None = None*, *details: str | None = None*)

Bases: *CensysAsmException*

Exception raised when the specified tag color is too long.

exception censys.common.exceptions.**CensysTagHasTrailingOrLeadingWhitespaceException**(*status_code: int*, *message: str*, *body: str | None = None*, *const: str | None = None*, *error_code: int | None = None*, *details: str | None = None*)

Bases: *CensysAsmException*

Exception raised when the specified tag has trailing or leading whitespace.

exception censys.common.exceptions.**CensysTagIsEmptyStringException**(*status_code: int*, *message: str*, *body: str | None = None*, *const: str | None = None*, *error_code: int | None = None*, *details: str | None = None*)

Bases: *CensysAsmException*

Exception raised when the specified tag is an empty string.

exception censys.common.exceptions.**CensysTagLabelTooLongException**(*status_code: int*, *message: str*, *body: str | None = None*, *const: str | None = None*, *error_code: int | None = None*, *details: str | None = None*)

Bases: *CensysAsmException*

Exception raised when the specified tag label is too long.

exception censys.common.exceptions.**CensysTagLabelsDifferOnlyInCasingException**(*status_code: int*, *message: str*, *body: str | None = None*, *const: str | None = None*, *error_code: int | None = None*, *details: str | None = None*)

Bases: *CensysAsmException*

Exception raised when the tag differs from an existing tag in only casing.

exception censys.common.exceptions.**CensysTeamNotFoundException**(*status_code: int*, *message: str*, *body: str | None = None*, *const: str | None = None*, *error_code: int | None = None*, *details: str | None = None*)

Bases: *CensysAsmException*

Exception raised when the team is not found.

exception censys.common.exceptions.**CensysTooManyInputNodesException**(*status_code: int*, *message: str*, *body: str | None = None*, *const: str | None = None*, *error_code: int | None = None*, *details: str | None = None*)

Bases: *CensysAsmException*

Exception raised when there are too many input nodes.

exception censys.common.exceptions.**CensysTooManyRequestsException**(*status_code: int*, *message: str*, *body: str | None = None*, *const: str | None = None*, *error_code: int | None = None*, *details: str | None = None*)

Bases: *CensysAsmException*

Exception raised when the allowed requests bandwidth is exceeded.

exception censys.common.exceptions.**CensysTooSoonToResendInviteException**(*status_code: int*, *message: str*, *body: str | None = None*, *const: str | None = None*, *error_code: int | None = None*, *details: str | None = None*)

Bases: *CensysAsmException*

Exception raised when it is too soon to resend the invite.

exception censys.common.exceptions.**CensysUnauthorizedException**(*status_code: int*, *message: str*, *body: str | None = None*, *const: str | None = None*, *error_code: int | None = None*, *details: str | None = None*)

Bases: *CensysSearchException*

Exception raised when you doesn't have access to the requested resource.

censys.common.deprecation module

Warns on deprecated class and functions.

class censys.common.deprecation.**DeprecationDecorator**(*message: str | None = None*)

Bases: object

Deprecation Decorator for classes and functions.

2.1.3 censys.search package

censys.search.v1 package

Interact with the Censys Search v1 APIs.

class censys.search.v1.**CensysData**(*api_id: str | None = None*, *api_secret: str | None = None*, ***kwargs*)

Bases: *CensysSearchAPIv1*

Interacts with the Data index.

For more details, see our documentation: https://search.censys.io/api

get_series() → dict

Get data on the types of scans we regularly perform (series).

Returns
The result set returned.

Return type
dict

view_result(*series_id: str*, *result_id: str*) → dict

View a specific result of a specific series.

Parameters

- **series_id** (*str*) – The ID of the series.
- **result_id** (*str*) – The ID of the result.

Returns
The result set returned.

Return type
dict

view_series(*series_id: str*) → dict

Get data on a specific series.

Parameters
series_id (*str*) – The ID of the series.

Returns
The result set returned.

Return type
dict

censys.search.v1.api module

Base for interacting with the Censys Search API.

class censys.search.v1.api.**CensysSearchAPIv1**(*api_id: str | None = None*, *api_secret: str | None = None*, ***kwargs*)

Bases: *CensysAPIBase*

This class is the base class for all v1 API indexes.

DEFAULT_URL: str = 'https://search.censys.io/api/v1'

Default Search API base URL.

INDEX_NAME: str | None = None

Name of Censys Index.

account() → dict

Gets the current account information.

This includes email and quota.

Returns
Account response.

Return type
dict

quota() → dict

Gets the current account's query quota.

Returns
Quota response.

Return type
dict

censys.search.v2 package

Interact with the Censys Search v2 APIs.

class censys.search.v2.**CensysCerts**(*api_id: str | None = None*, *api_secret: str | None = None*, ***kwargs*)

Bases: *CensysSearchAPIv2*

Interacts with the Certs index.

Please note that this class represents only the v2 API endpoints.

Examples

Inits Censys Certs.

```
>>> from censys.search import CensysCerts
>>> c = CensysCerts()
```

Search for hosts by sha256fp.

```
>>> c.get_hosts_by_cert(
↪"fb444eb8e68437bae06232b9f5091bccff62a768ca09e92eb5c9c2cf9d17c426")
(
    [
        {
            "ip": "string",
            "name": "string",
            "observed_at": "2021-08-02T14:56:38.711Z",
            "first_observed_at": "2021-08-02T14:56:38.711Z",
        }
    ],
    {
        "next": "nextCursorToken",
    },
)
```

INDEX_NAME: str = 'certificates'

Name of Censys Index.

aggregate(*query: str*, *field: str*, *num_buckets: int = 50*, ***kwargs*) → dict

Aggregates certificate records matching a specified query into buckets based on the given field.

Parameters

- **query** (*str*) – The query string to search for.
- **field** (*str*) – The field to aggregate on.
- **num_buckets** (*int*) – The number of buckets to return. Defaults to 50.
- ****kwargs** – Additional keyword arguments to pass to the underlying HTTP request.

Returns

Aggregation results.

Return type

dict

bulk(*fingerprints: List[str]*) → List[dict]

Fetches the certificate records for the specified SHA-256 fingerprints.

By default, this function uses the POST method, which allows for a larger number of fingerprints to be queried at once. If you wish to use the GET method, please use *CensysCerts.bulk_get* instead.

Parameters

fingerprints (*List[str]*) – List of certificate SHA256 fingerprints.

Returns

Certificate details.

Return type
dict

bulk_get(*fingerprints: List[str]*) → List[dict]

Fetches the certificate records for the specified SHA-256 fingerprints.

Using the GET method allows for a smaller number of fingerprints to be queried at once.

Parameters
fingerprints (*List[str]*) – List of certificate SHA256 fingerprints.

Returns
Certificate details.

Return type
dict

bulk_post(*fingerprints: List[str]*) → List[dict]

Fetches the certificate records for the specified SHA-256 fingerprints.

Using the POST method allows for a larger number of fingerprints to be queried at once.

Parameters
fingerprints (*List[str]*) – List of certificate SHA256 fingerprints.

Returns
Certificate details.

Return type
dict

bulk_view(*fingerprints: List[str]*) → List[dict]

Fetches the certificate records for the specified SHA-256 fingerprints.

By default, this function uses the POST method, which allows for a larger number of fingerprints to be queried at once. If you wish to use the GET method, please use *CensysCerts.bulk_get* instead.

Parameters
fingerprints (*List[str]*) – List of certificate SHA256 fingerprints.

Returns
Certificate details.

Return type
dict

get_hosts_by_cert(*fingerprint: str*, *cursor: str | None = None*) → dict

Returns a list of hosts which contain services presenting this certificate, including when the certificate was first observed.

Parameters

- **fingerprint** (*str*) – The SHA-256 fingerprint of the requested certificate.
- **cursor** (*str*) – Cursor token from the API response, which fetches the next page of hosts when added to the endpoint URL.

Returns
A list of hosts which contain services presenting this certificate.

Return type
dict

get_observations(*fingerprint: str*, *per_page: int = 50*, *start_time: str | date | datetime | None = None*, *end_time: str | date | datetime | None = None*, *cursor: str | None = None*) → dict

Returns a list of observations for the specified certificate.

Parameters

- **fingerprint** (*str*) – The SHA-256 fingerprint of the requested certificate.
- **per_page** (*int*) – The number of results to return per page. Defaults to 50.
- **start_time** (*str*) – The start time of the observations to return.
- **end_time** (*str*) – The end time of the observations to return.
- **cursor** (*str*) – Cursor token from the API response, which fetches the next page of observations when added to the endpoint URL.

Returns
A list of observations for the specified certificate.

Return type
dict

list_certs_with_tag(*tag_id: str*) → List[dict]

Returns a list of certs which are tagged with the specified tag.

Parameters
tag_id (*str*) – The ID of the tag.

Returns
A list of certs which are tagged with the specified tag.

Return type
List[dict]

raw_search(*query: str*, *per_page: int = 50*, *cursor: str | None = None*, *fields: List[str] | None = None*, *sort: str | List[str] | None = None*, ***kwargs*) → dict

Searches the Certs index.

Searches the Certs index for all records that match the given query. This method does no automatic pagination or post processing.

Parameters

- **query** (*str*) – The query string to search for.
- **per_page** (*int*) – The number of results to return per page. Defaults to 50.
- **cursor** (*str, optional*) – Cursor token from the API response, which fetches the next page of results when added to the endpoint URL.
- **fields** (*List[str], optional*) – Additional fields to return in the matched certificates outside of the default returned fields.
- **sort** (*List[str], optional*) – A list of fields to sort on. By default, fields will be sorted in ascending order.
- ****kwargs** – Additional keyword arguments to pass to the underlying HTTP request.

Returns
Search results.

Return type
dict

search(*query: str*, *per_page: int = 50*, *cursor: str | None = None*, *pages: int = 1*, *fields: List[str] | None = None*, *sort: str | List[str] | None = None*, ***kwargs*) → *Query*

Searches the Certs index.

By default, this function uses the POST method, which allows for a larger number of fingerprints to be queried at once. If you wish to use the GET method, please use *CensysCerts.search_get* instead.

Parameters

- **query** (*str*) – The query string to search for.
- **per_page** (*int*) – The number of results to return per page. Defaults to 50.
- **cursor** (*str, optional*) – Cursor token from the API response, which fetches the next page of results when added to the endpoint URL.
- **pages** (*int*) – The number of pages to return. Defaults to 1.
- **fields** (*List[str], optional*) – Additional fields to return in the matched certificates outside of the default returned fields.
- **sort** (*List[str], optional*) – A list of fields to sort on. By default, fields will be sorted in ascending order.
- ****kwargs** – Additional keyword arguments to pass to the underlying HTTP request.

Returns
A query object that can be used to iterate over the search results.

Return type
Query

search_get(*query: str*, *per_page: int = 50*, *cursor: str | None = None*, *fields: List[str] | None = None*, *sort: str | List[str] | None = None*, ***kwargs*) → dict

Searches the Certs index using the GET method.

Parameters

- **query** (*str*) – The query string to search for.
- **per_page** (*int*) – The number of results to return per page. Defaults to 50.
- **cursor** (*str, optional*) – Cursor token from the API response, which fetches the next page of results when added to the endpoint URL.
- **fields** (*List[str], optional*) – Additional fields to return in the matched certificates outside of the default returned fields.
- **sort** (*List[str], optional*) – A list of fields to sort on. By default, fields will be sorted in ascending order.
- ****kwargs** – Arbitrary keyword arguments.

Returns
Search results.

Return type
dict

search_post(*query: str*, *per_page: int = 50*, *cursor: str | None = None*, *fields: List[str] | None = None*, *sort: str | List[str] | None = None*, ***kwargs*) → dict

Searches the Certs index using the POST method.

This method returns the *result* field of the raw response. If you wish to access the raw response, please use *CensysCerts.search_post_raw* instead.

Parameters

- **query** (*str*) – The query string to search for.
- **per_page** (*int*) – The number of results to return per page. Defaults to 50.
- **cursor** (*str, optional*) – Cursor token from the API response, which fetches the next page of results when added to the endpoint URL.
- **fields** (*List[str], optional*) – Additional fields to return in the matched certificates outside of the default returned fields.
- **sort** (*List[str], optional*) – A list of fields to sort on. By default, fields will be sorted in ascending order.
- ****kwargs** – Arbitrary keyword arguments.

Returns
Search results.

Return type
dict

search_post_raw(*query: str*, *per_page: int = 50*, *cursor: str | None = None*, *fields: List[str] | None = None*, *sort: str | List[str] | None = None*, ***kwargs*) → dict

Searches the Certs index using the POST method. Returns the raw response.

Parameters

- **query** (*str*) – The query string to search for.
- **per_page** (*int*) – The number of results to return per page. Defaults to 50.
- **cursor** (*str, optional*) – Cursor token from the API response, which fetches the next page of results when added to the endpoint URL.
- **fields** (*List[str], optional*) – Additional fields to return in the matched certificates outside of the default returned fields.
- **sort** (*List[str], optional*) – A list of fields to sort on. By default, fields will be sorted in ascending order.
- ****kwargs** – Arbitrary keyword arguments.

Returns
Search results.

Return type
dict

view(*document_id: str*, ***kwargs*) → dict

Fetches the certificate record for the specified SHA-256 fingerprint.

Parameters

- **document_id** (*str*) – The SHA-256 fingerprint of the requested certificate.
- ****kwargs** – Arbitrary keyword arguments.

Returns
Certificate details.

Return type
dict

class censys.search.v2.**CensysHosts**(*api_id: str | None = None*, *api_secret: str | None = None*, ***kwargs*)

Bases: *CensysSearchAPIv2*

Interacts with the Hosts index.

Examples

Inits Censys Hosts.

```
>>> from censys.search import CensysHosts
>>> h = CensysHosts()
```

Simple host search.

```
>>> for page in h.search("services.service_name: HTTP"):
>>>     print(page)
[
    {
    'services':
        [
            {'service_name': 'HTTP', 'port': 80},
            {'service_name': 'HTTP', 'port': 443}
        ],
    'ip': '1.0.0.0'
    },
    ...
]
```

Fetch a specific host and its services

```
>>> h.view("1.0.0.0")
{
    'ip': '8.8.8.8',
    'services': [{}],
    ...
}
```

Simple host aggregate.

```
>>> h.aggregate("services.service_name: HTTP", "services.port", num_buckets=5)
{
    'total_omitted': 591527370,
    'buckets': [
        {'count': 56104072, 'key': '80'},
        {'count': 43527894, 'key': '443'},
        {'count': 23070429, 'key': '7547'},
        {'count': 12970769, 'key': '30005'},
        {'count': 12825150, 'key': '22'}
    ],
    'potential_deviation': 3985101,
    'field': 'services.port',
    'query': 'services.service_name: HTTP',
    'total': 172588754
}
```

Fetch a list of host names for the specified IP address.

```
>>> h.view_host_names("1.1.1.1")
['one.one.one.one']
```

Fetch a list of events for the specified IP address.

```
>>> h.view_host_events("1.1.1.1")
[{'timestamp': '2019-01-01T00:00:00.000Z'}]
```

INDEX_NAME: str = 'hosts'

Name of Censys Index.

aggregate(*query: str*, *field: str*, *num_buckets: int = 50*, *virtual_hosts: str | None = None*, ***kwargs: Any*) → dict

Aggregate host index.

Creates a report on the breakdown of the values of a field in a result set. For more details, see our documentation: https://search.censys.io/api

Parameters

- **query** (*str*) – The query to be executed.
- **field** (*str*) – The field you are running a breakdown on.
- **num_buckets** (*int*) – Optional; The maximum number of values. Defaults to 50.
- **virtual_hosts** (*str*) – Optional; Whether to include virtual hosts in the results. Valid values are "EXCLUDE", "INCLUDE", and "ONLY".
- ****kwargs** (*Any*) – Optional; Additional arguments to be passed to the query.

Returns
The result set returned.

Return type
dict

bulk_view(*document_ids: List[str]*, *max_workers: int = 20*, *at_time: str | date | datetime | None = None*, ***kwargs: Any*) → Dict[str, dict]

Bulk view documents from current index.

View the current structured data we have on a list of documents.

Parameters

- **document_ids** (*List[str]*) – The IDs of the documents you are requesting.
- **max_workers** (*int*) – Optional; The number of workers to use. Defaults to 20.
- **at_time** (*[str, datetime.date, datetime.datetime]*) – Optional; Fetches a document at a given point in time.
- ****kwargs** (*Any*) – Optional; Additional arguments to be passed to the query.

Returns
The result set returned.

Return type
Dict[str, dict]

list_hosts_with_tag(*tag_id: str*) → List[str]

Returns a list of hosts which are tagged with the specified tag.

Parameters
tag_id (*str*) – The ID of the tag.

Returns
A list of host IP addresses.

Return type
List[str]

metadata() → dict

Get metadata for the host index.

Returns
The result set returned.

Return type
dict

search(*query: str*, *per_page: int = 100*, *cursor: str | None = None*, *pages: int = 1*, *fields: List[str] | None = None*, *sort: str | List[str] | None = None*, *virtual_hosts: str | None = None*, ***kwargs: Any*) → *Query*

Search host index.

Searches the given index for all records that match the given query. For more details, see our documentation: https://search.censys.io/api

Parameters

- **query** (*str*) – The query to be executed.
- **per_page** (*int*) – Optional; The number of results to be returned for each page. Defaults to 100.
- **cursor** (*int*) – Optional; The cursor of the desired result set.
- **pages** (*int*) – Optional; The number of pages returned. Defaults to 1.
- **fields** (*List[str]*) – Optional; The fields to return. Defaults to all fields.
- **sort** (*str*) – Optional; The method used to sort results. Valid values are "RELEVANCE", "DESCENDING", and "ASCENDING".
- **virtual_hosts** (*str*) – Optional; Whether to include virtual hosts in the results. Valid values are "EXCLUDE", "INCLUDE", and "ONLY".
- ****kwargs** (*Any*) – Optional; Additional arguments to be passed to the query.

Returns
Query object that can be a callable or an iterable.

Return type
Query

view(*document_id: str*, *at_time: str | date | datetime | None = None*, ***kwargs: Any*) → dict

View document from current index.

View the current structured data we have on a specific document. For more details, see our documentation: https://search.censys.io/api

Parameters

- **document_id** (*str*) – The ID of the document you are requesting.

- **at_time** ([str, datetime.date, datetime.datetime]) – Optional; Fetches a document at a given point in time.
- ****kwargs** (Any) – Optional; Additional arguments to be passed to the query.

Returns
The result set returned.

Return type
dict

view_host_certificates(*ip: str*, *per_page: int = 100*, *start_time: str | date | datetime | None = None*, *cursor: str | None = None*) → dict

Returns a list of certificates for the specified host.

Parameters

- **ip** (str) – The IP address of the requested host.
- **per_page** (int) – Optional; The number of results to be returned for each page. Defaults to 100.
- **start_time** (Datetime) – Optional; An RFC3339 timestamp which represents the beginning chronological point-in-time (inclusive) from which events are returned.
- **cursor** (str) – Optional; Cursor token from the API response.

Returns
A list of certificates.

Return type
dict

view_host_diff(*ip: str*, *ip_b: str | None = None*, *at_time: str | date | datetime | None = None*, *at_time_b: str | date | datetime | None = None*)

Fetches a diff of the specified IP address.

Parameters

- **ip** (str) – The IP address of the requested host.
- **ip_b** (str) – Optional; The IP address of the second host.
- **at_time** (Datetime) – Optional; An RFC3339 timestamp which represents the point-in-time used as the basis for Host A.
- **at_time_b** (Datetime) – Optional; An RFC3339 timestamp which represents the point-in-time used as the basis for Host B.

Returns
A diff of the hosts.

Return type
dict

view_host_events(*ip: str*, *start_time: str | date | datetime | None = None*, *end_time: str | date | datetime | None = None*, *per_page: int | None = None*, *cursor: str | None = None*, *reversed: bool | None = None*) → dict

Fetches a list of events for the specified IP address.

Parameters

- **ip** (str) – The IP address of the requested host.

- **start_time** (*Datetime*) – Optional; An RFC3339 timestamp which represents the beginning chronological point-in-time (inclusive) from which events are returned.
- **end_time** (*Datetime*) – Optional; An RFC3339 timestamp which represents the ending chronological point-in-time (exclusive) from which events are returned.
- **per_page** (*int*) – Optional; The maximum number of hits to return in each response (minimum of 1, maximum of 50).
- **cursor** (*str*) – Optional; Cursor token from the API response.
- **reversed** (*bool*) – Optional; Reverse the order of the return events, that is, return events in reversed chronological order.

Returns
A list of events.

Return type
dict

view_host_names(*ip: str*, *per_page: int | None = None*, *cursor: str | None = None*) → List[str]

Fetches a list of host names for the specified IP address.

Parameters

- **ip** (*str*) – The IP address of the requested host.
- **per_page** (*int*) – Optional; The number of results to be returned for each page. Defaults to 100.
- **cursor** (*int*) – Optional; The cursor of the desired result set.

Returns
A list of host names.

Return type
List[str]

censys.search.v2.api module

Base for interacting with the Censys Search API.

class censys.search.v2.api.**CensysSearchAPIv2**(*api_id: str | None = None*, *api_secret: str | None = None*, ***kwargs*)

Bases: *CensysAPIBase*

This class is the base class for the Hosts index.

Examples

```
>>> c = CensysSearchAPIv2()
```

DEFAULT_URL: str = 'https://search.censys.io/api'

Default Search API base URL.

INDEX_NAME: str = ''

Name of Censys Index.

class Query(*api: CensysSearchAPIv2*, *query: str*, *per_page: int | None = None*, *cursor: str | None = None*, *pages: int = 1*, *fields: List[str] | None = None*, *sort: str | List[str] | None = None*, ***kwargs: Any*)

Bases: Iterable

Query class that is callable and iterable.

Object Searches the given index for all records that match the given query. For more details, see our documentation: https://search.censys.io/api

total: int | None = None

view_all(*max_workers: int = 20*) → Dict[str, dict]

View each document returned from query.

Please note that each result returned by the query will be looked up using the view method.

Parameters
max_workers (*int*) – The number of workers to use. Defaults to 20.

Returns
Dictionary mapping documents to that document's result set.

Return type
Dict[str, dict]

account() → dict

Gets the current account's query quota.

Returns
Quota response.

Return type
dict

add_comment(*document_id: str*, *contents: str*) → dict

Add comment to a document.

Parameters

- **document_id** (*str*) – The ID of the document you are requesting.
- **contents** (*str*) – The contents of the comment.

Returns
The result set returned.

Return type
dict

add_tag_to_document(*document_id: str*, *tag_id: str*)

Add a tag to a document.

Parameters

- **document_id** (*str*) – The ID of the document.
- **tag_id** (*str*) – The ID of the tag.

aggregate(*query: str*, *field: str*, *num_buckets: int = 50*, ***kwargs: Any*) → dict

Aggregate current index.

Creates a report on the breakdown of the values of a field in a result set. For more details, see our documentation: https://search.censys.io/api

Parameters

- **query** (*str*) – The query to be executed.
- **field** (*str*) – The field you are running a breakdown on.
- **num_buckets** (*int*) – Optional; The maximum number of values. Defaults to 50.
- ****kwargs** (*Any*) – Optional; Additional arguments to be passed to the query.

Returns
The result set returned.

Return type
dict

bulk_view(*document_ids: List[str]*, *max_workers: int = 20*, ***kwargs: Any*) → Dict[str, dict]

Bulk view documents from current index.

View the current structured data we have on a list of documents. For more details, see our documentation: https://search.censys.io/api

Parameters

- **document_ids** (*List[str]*) – The IDs of the documents you are requesting.
- **max_workers** (*int*) – The number of workers to use. Defaults to 20.
- ****kwargs** (*Any*) – Optional; Additional arguments to be passed to the query.

Returns
Dictionary mapping document IDs to that document's result set.

Return type
Dict[str, dict]

create_tag(*name: str*, *color: str | None = None*) → dict

Create a tag.

Parameters

- **name** (*str*) – The name of the tag.
- **color** (*str*) – Optional; The color of the tag.

Returns
The result set returned.

Return type
dict

delete_comment(*document_id: str*, *comment_id: str*) → dict

Delete comment from a document.

Parameters

- **document_id** (*str*) – The ID of the document you are requesting.
- **comment_id** (*str*) – The ID of the comment you are requesting.

Returns
The result set returned.

Return type
dict

delete_tag(*tag_id: str*)

Delete a tag.

Parameters
tag_id (*str*) – The ID of the tag.

get_comment(*document_id: str*, *comment_id: str*) → dict

Get comment for a document.

Parameters

- **document_id** (*str*) – The ID of the document you are requesting.
- **comment_id** (*str*) – The ID of the comment you are requesting.

Returns
The result set returned.

Return type
dict

get_comments(*document_id: str*) → List[dict]

Get comments for a document.

Parameters
document_id (*str*) – The ID of the document you are requesting.

Returns
The list of comments.

Return type
List[dict]

get_tag(*tag_id: str*) → dict

Get a tag.

Parameters
tag_id (*str*) – The ID of the tag.

Returns
The result set returned.

Return type
dict

list_all_tags() → List[dict]

List all tags.

Returns
The list of tags.

Return type
List[dict]

list_tags_on_document(*document_id: str*) → List[dict]

List tags on a document.

Parameters
document_id (*str*) – The ID of the document.

Returns
The list of tags.

Return type
List[dict]

quota() → dict

Returns metadata of a given search query.

Returns
The metadata of the result set returned.

Return type
dict

raw_search(*query: str*, *per_page: int = 100*, *cursor: str | None = None*, *fields: List[str] | None = None*, *sort: str | List[str] | None = None*, ***kwargs: Any*) → dict

Search current index.

Searches the given index for all records that match the given query. This method does no automatic pagination or post processing.

Parameters

- **query** (*str*) – The query to be executed.
- **per_page** (*int*) – Optional; The number of results to be returned for each page. Defaults to 100.
- **cursor** (*int*) – Optional; The cursor of the desired result set.
- **fields** (*List[str]*) – Optional; The fields to be returned. Defaults to base fields.
- **sort** (*Union[str, List[str]]*) – Optional; The fields to sort by. Defaults to None.
- ****kwargs** (*Any*) – Optional; Additional arguments to be passed to the query.

Returns
The raw result set.

Return type
dict

remove_tag_from_document(*document_id: str*, *tag_id: str*)

Remove a tag from a document.

Parameters

- **document_id** (*str*) – The ID of the document.
- **tag_id** (*str*) – The ID of the tag.

search(*query: str*, *per_page: int = 100*, *cursor: str | None = None*, *pages: int = 1*, *fields: List[str] | None = None*, *sort: str | List[str] | None = None*, ***kwargs: Any*) → *Query*

Search current index.

Searches the given index for all records that match the given query. For more details, see our documentation: https://search.censys.io/api

Parameters

- **query** (*str*) – The query to be executed.
- **per_page** (*int*) – Optional; The number of results to be returned for each page. Defaults to 100.
- **cursor** (*int*) – Optional; The cursor of the desired result set.

- **pages** (*int*) – Optional; The number of pages returned. Defaults to 1.
- **fields** (*List[str]*) – Optional; The fields to be returned. Defaults to base fields.
- **sort** (*Union[str, List[str]]*) – Optional; The fields to sort by. Defaults to None.
- ****kwargs** (*Any*) – Optional; Additional arguments to be passed to the query.

Returns
Query object that can be a callable or an iterable.

Return type
Query

search_get(*query: str*, *per_page: int = 100*, *cursor: str | None = None*, *fields: List[str] | None = None*, *sort: str | List[str] | None = None*, ***kwargs: Any*) → dict

Search current index using GET method.

Parameters

- **query** (*str*) – The query to be executed.
- **per_page** (*int*) – Optional; The number of results to be returned for each page. Defaults to 100.
- **cursor** (*int*) – Optional; The cursor of the desired result set.
- **fields** (*List[str]*) – Optional; The fields to be returned. Defaults to base fields.
- **sort** (*Union[str, List[str]]*) – Optional; The fields to sort by. Defaults to None.
- ****kwargs** (*Any*) – Optional; Additional arguments to be passed to the query.

Returns
The raw result set.

Return type
dict

search_get_raw(*query: str*, *per_page: int = 100*, *cursor: str | None = None*, *fields: List[str] | None = None*, *sort: str | List[str] | None = None*, ***kwargs: Any*) → dict

Search current index using GET method.

Parameters

- **query** (*str*) – The query to be executed.
- **per_page** (*int*) – Optional; The number of results to be returned for each page. Defaults to 100.
- **cursor** (*int*) – Optional; The cursor of the desired result set.
- **fields** (*List[str]*) – Optional; The fields to be returned. Defaults to base fields.
- **sort** (*Union[str, List[str]]*) – Optional; The fields to sort by. Defaults to None.
- ****kwargs** (*Any*) – Optional; Additional arguments to be passed to the query.

Returns
The raw result set.

Return type
dict

search_post(*query: str*, *per_page: int = 100*, *cursor: str | None = None*, *fields: List[str] | None = None*, *sort: str | List[str] | None = None*, ***kwargs*) → dict

Searches the Certs index using the POST method.

This method returns the *result* field of the raw response. If you wish to access the raw response, please use *search_post_raw* instead.

Parameters

- **query** (*str*) – The query string to search for.
- **per_page** (*int*) – The number of results to return per page. Defaults to 50.
- **cursor** (*str, optional*) – Cursor token from the API response, which fetches the next page of results when added to the endpoint URL.
- **fields** (*List[str], optional*) – Additional fields to return in the matched documents. Defaults to base fields.
- **sort** (*Union[str, List[str]], optional*) – The fields to sort by. Defaults to None.
- ****kwargs** – Arbitrary keyword arguments.

Returns
Search results.

Return type
dict

search_post_raw(*query: str*, *per_page: int = 100*, *cursor: str | None = None*, *fields: List[str] | None = None*, *sort: str | List[str] | None = None*, ***kwargs*) → dict

Searches the given index for all records that match the given query.

Parameters

- **query** (*str*) – The query string to search for.
- **per_page** (*int*) – The number of results to return per page. Defaults to 50.
- **cursor** (*str, optional*) – Cursor token from the API response, which fetches the next page of results when added to the endpoint URL.
- **fields** (*List[str], optional*) – The fields to be returned. Defaults to base fields.
- **sort** (*Union[str, List[str]], optional*) – The fields to sort by. Defaults to None.
- ****kwargs** – Arbitrary keyword arguments.

Returns
Search results.

Return type
dict

update_comment(*document_id: str*, *comment_id: str*, *contents: str*) → dict

Update comment from a document.

Parameters

- **document_id** (*str*) – The ID of the document you are requesting.
- **comment_id** (*str*) – The ID of the comment you are requesting.
- **contents** (*str*) – The contents of the comment.

Returns
The result set returned.

Return type
dict

update_tag(*tag_id: str*, *name: str*, *color: str | None = None*) → dict

Update a tag.

Parameters

- **tag_id** (*str*) – The ID of the tag.
- **name** (*str*) – The name of the tag.
- **color** (*str*) – The color of the tag.

Returns
The result set returned.

Return type
dict

view(*document_id: str*, ***kwargs: Any*) → dict

View document from current index.

View the current structured data we have on a specific document. For more details, see our documentation: https://search.censys.io/api

Parameters

- **document_id** (*str*) – The ID of the document you are requesting.
- ****kwargs** (*Any*) – Optional; Additional arguments to be passed to the query.

Returns
The result set returned.

Return type
dict

An easy-to-use and lightweight API wrapper for Censys Search API (search.censys.io).

class censys.search.**CensysCerts**(*api_id: str | None = None*, *api_secret: str | None = None*, ***kwargs*)

Bases: *CensysSearchAPIv2*

Interacts with the Certs index.

Please note that this class represents only the v2 API endpoints.

Examples

Inits Censys Certs.

```
>>> from censys.search import CensysCerts
>>> c = CensysCerts()
```

Search for hosts by sha256fp.

```
>>> c.get_hosts_by_cert(
↪"fb444eb8e68437bae06232b9f5091bccff62a768ca09e92eb5c9c2cf9d17c426")
(
    [
        {
            "ip": "string",
            "name": "string",
            "observed_at": "2021-08-02T14:56:38.711Z",
            "first_observed_at": "2021-08-02T14:56:38.711Z",
        }
    ],
    {
        "next": "nextCursorToken",
    },
)
```

INDEX_NAME: str = 'certificates'

Name of Censys Index.

aggregate(*query: str*, *field: str*, *num_buckets: int = 50*, ***kwargs*) → dict

Aggregates certificate records matching a specified query into buckets based on the given field.

Parameters

- **query** (*str*) – The query string to search for.
- **field** (*str*) – The field to aggregate on.
- **num_buckets** (*int*) – The number of buckets to return. Defaults to 50.
- ****kwargs** – Additional keyword arguments to pass to the underlying HTTP request.

Returns

Aggregation results.

Return type

dict

bulk(*fingerprints: List[str]*) → List[dict]

Fetches the certificate records for the specified SHA-256 fingerprints.

By default, this function uses the POST method, which allows for a larger number of fingerprints to be queried at once. If you wish to use the GET method, please use *CensysCerts.bulk_get* instead.

Parameters

fingerprints (*List[str]*) – List of certificate SHA256 fingerprints.

Returns

Certificate details.

Return type

dict

bulk_get(*fingerprints: List[str]*) → List[dict]

Fetches the certificate records for the specified SHA-256 fingerprints.

Using the GET method allows for a smaller number of fingerprints to be queried at once.

Parameters

fingerprints (*List[str]*) – List of certificate SHA256 fingerprints.

Returns
Certificate details.

Return type
dict

bulk_post(*fingerprints: List[str]*) → List[dict]

Fetches the certificate records for the specified SHA-256 fingerprints.

Using the POST method allows for a larger number of fingerprints to be queried at once.

Parameters
fingerprints (*List[str]*) – List of certificate SHA256 fingerprints.

Returns
Certificate details.

Return type
dict

bulk_view(*fingerprints: List[str]*) → List[dict]

Fetches the certificate records for the specified SHA-256 fingerprints.

By default, this function uses the POST method, which allows for a larger number of fingerprints to be queried at once. If you wish to use the GET method, please use *CensysCerts.bulk_get* instead.

Parameters
fingerprints (*List[str]*) – List of certificate SHA256 fingerprints.

Returns
Certificate details.

Return type
dict

get_hosts_by_cert(*fingerprint: str*, *cursor: str | None = None*) → dict

Returns a list of hosts which contain services presenting this certificate, including when the certificate was first observed.

Parameters

- **fingerprint** (*str*) – The SHA-256 fingerprint of the requested certificate.
- **cursor** (*str*) – Cursor token from the API response, which fetches the next page of hosts when added to the endpoint URL.

Returns
A list of hosts which contain services presenting this certificate.

Return type
dict

get_observations(*fingerprint: str*, *per_page: int = 50*, *start_time: str | date | datetime | None = None*, *end_time: str | date | datetime | None = None*, *cursor: str | None = None*) → dict

Returns a list of observations for the specified certificate.

Parameters

- **fingerprint** (*str*) – The SHA-256 fingerprint of the requested certificate.
- **per_page** (*int*) – The number of results to return per page. Defaults to 50.
- **start_time** (*str*) – The start time of the observations to return.

- **end_time** (*str*) – The end time of the observations to return.
- **cursor** (*str*) – Cursor token from the API response, which fetches the next page of observations when added to the endpoint URL.

Returns
A list of observations for the specified certificate.

Return type
dict

list_certs_with_tag(*tag_id: str*) → List[dict]

Returns a list of certs which are tagged with the specified tag.

Parameters
tag_id (*str*) – The ID of the tag.

Returns
A list of certs which are tagged with the specified tag.

Return type
List[dict]

raw_search(*query: str*, *per_page: int = 50*, *cursor: str | None = None*, *fields: List[str] | None = None*, *sort: str | List[str] | None = None*, ***kwargs*) → dict

Searches the Certs index.

Searches the Certs index for all records that match the given query. This method does no automatic pagination or post processing.

Parameters

- **query** (*str*) – The query string to search for.
- **per_page** (*int*) – The number of results to return per page. Defaults to 50.
- **cursor** (*str, optional*) – Cursor token from the API response, which fetches the next page of results when added to the endpoint URL.
- **fields** (*List[str], optional*) – Additional fields to return in the matched certificates outside of the default returned fields.
- **sort** (*List[str], optional*) – A list of fields to sort on. By default, fields will be sorted in ascending order.
- ****kwargs** – Additional keyword arguments to pass to the underlying HTTP request.

Returns
Search results.

Return type
dict

search(*query: str*, *per_page: int = 50*, *cursor: str | None = None*, *pages: int = 1*, *fields: List[str] | None = None*, *sort: str | List[str] | None = None*, ***kwargs*) → *Query*

Searches the Certs index.

By default, this function uses the POST method, which allows for a larger number of fingerprints to be queried at once. If you wish to use the GET method, please use *CensysCerts.search_get* instead.

Parameters

- **query** (*str*) – The query string to search for.
- **per_page** (*int*) – The number of results to return per page. Defaults to 50.

- **cursor** (*str, optional*) – Cursor token from the API response, which fetches the next page of results when added to the endpoint URL.
- **pages** (*int*) – The number of pages to return. Defaults to 1.
- **fields** (*List[str], optional*) – Additional fields to return in the matched certificates outside of the default returned fields.
- **sort** (*List[str], optional*) – A list of fields to sort on. By default, fields will be sorted in ascending order.
- ****kwargs** – Additional keyword arguments to pass to the underlying HTTP request.

Returns
A query object that can be used to iterate over the search results.

Return type
Query

search_get(*query: str*, *per_page: int = 50*, *cursor: str | None = None*, *fields: List[str] | None = None*, *sort: str | List[str] | None = None*, ***kwargs*) → dict

Searches the Certs index using the GET method.

Parameters

- **query** (*str*) – The query string to search for.
- **per_page** (*int*) – The number of results to return per page. Defaults to 50.
- **cursor** (*str, optional*) – Cursor token from the API response, which fetches the next page of results when added to the endpoint URL.
- **fields** (*List[str], optional*) – Additional fields to return in the matched certificates outside of the default returned fields.
- **sort** (*List[str], optional*) – A list of fields to sort on. By default, fields will be sorted in ascending order.
- ****kwargs** – Arbitrary keyword arguments.

Returns
Search results.

Return type
dict

search_post(*query: str*, *per_page: int = 50*, *cursor: str | None = None*, *fields: List[str] | None = None*, *sort: str | List[str] | None = None*, ***kwargs*) → dict

Searches the Certs index using the POST method.

This method returns the *result* field of the raw response. If you wish to access the raw response, please use *CensysCerts.search_post_raw* instead.

Parameters

- **query** (*str*) – The query string to search for.
- **per_page** (*int*) – The number of results to return per page. Defaults to 50.
- **cursor** (*str, optional*) – Cursor token from the API response, which fetches the next page of results when added to the endpoint URL.
- **fields** (*List[str], optional*) – Additional fields to return in the matched certificates outside of the default returned fields.

- **sort** (*List[str], optional*) – A list of fields to sort on. By default, fields will be sorted in ascending order.
- ****kwargs** – Arbitrary keyword arguments.

Returns
Search results.

Return type
dict

search_post_raw(*query: str*, *per_page: int = 50*, *cursor: str | None = None*, *fields: List[str] | None = None*, *sort: str | List[str] | None = None*, ***kwargs*) → dict

Searches the Certs index using the POST method. Returns the raw response.

Parameters

- **query** (*str*) – The query string to search for.
- **per_page** (*int*) – The number of results to return per page. Defaults to 50.
- **cursor** (*str, optional*) – Cursor token from the API response, which fetches the next page of results when added to the endpoint URL.
- **fields** (*List[str], optional*) – Additional fields to return in the matched certificates outside of the default returned fields.
- **sort** (*List[str], optional*) – A list of fields to sort on. By default, fields will be sorted in ascending order.
- ****kwargs** – Arbitrary keyword arguments.

Returns
Search results.

Return type
dict

view(*document_id: str*, ***kwargs*) → dict

Fetches the certificate record for the specified SHA-256 fingerprint.

Parameters

- **document_id** (*str*) – The SHA-256 fingerprint of the requested certificate.
- ****kwargs** – Arbitrary keyword arguments.

Returns
Certificate details.

Return type
dict

class censys.search.**CensysData**(*api_id: str | None = None*, *api_secret: str | None = None*, ***kwargs*)

Bases: *CensysSearchAPIv1*

Interacts with the Data index.

For more details, see our documentation: https://search.censys.io/api

get_series() → dict

Get data on the types of scans we regularly perform (series).

Returns
The result set returned.

Return type
dict

view_result(*series_id: str*, *result_id: str*) → dict

View a specific result of a specific series.

Parameters

- **series_id** (*str*) – The ID of the series.
- **result_id** (*str*) – The ID of the result.

Returns
The result set returned.

Return type
dict

view_series(*series_id: str*) → dict

Get data on a specific series.

Parameters
series_id (*str*) – The ID of the series.

Returns
The result set returned.

Return type
dict

class censys.search.**CensysHosts**(*api_id: str | None = None*, *api_secret: str | None = None*, ***kwargs*)

Bases: *CensysSearchAPIv2*

Interacts with the Hosts index.

Examples

Inits Censys Hosts.

```
>>> from censys.search import CensysHosts
>>> h = CensysHosts()
```

Simple host search.

```
>>> for page in h.search("services.service_name: HTTP"):
>>>     print(page)
[
    {
    'services':
        [
            {'service_name': 'HTTP', 'port': 80},
            {'service_name': 'HTTP', 'port': 443}
        ],
    'ip': '1.0.0.0'
    },
    ...
]
```

Fetch a specific host and its services

```
>>> h.view("1.0.0.0")
{
    'ip': '8.8.8.8',
    'services': [{}],
    ...
}
```

Simple host aggregate.

```
>>> h.aggregate("services.service_name: HTTP", "services.port", num_buckets=5)
{
    'total_omitted': 591527370,
    'buckets': [
        {'count': 56104072, 'key': '80'},
        {'count': 43527894, 'key': '443'},
        {'count': 23070429, 'key': '7547'},
        {'count': 12970769, 'key': '30005'},
        {'count': 12825150, 'key': '22'}
    ],
    'potential_deviation': 3985101,
    'field': 'services.port',
    'query': 'services.service_name: HTTP',
    'total': 172588754
}
```

Fetch a list of host names for the specified IP address.

```
>>> h.view_host_names("1.1.1.1")
['one.one.one.one']
```

Fetch a list of events for the specified IP address.

```
>>> h.view_host_events("1.1.1.1")
[{'timestamp': '2019-01-01T00:00:00.000Z'}]
```

INDEX_NAME: str = 'hosts'

Name of Censys Index.

aggregate(*query: str*, *field: str*, *num_buckets: int = 50*, *virtual_hosts: str | None = None*, ***kwargs: Any*) → dict

Aggregate host index.

Creates a report on the breakdown of the values of a field in a result set. For more details, see our documentation: https://search.censys.io/api

Parameters

- **query** (*str*) – The query to be executed.
- **field** (*str*) – The field you are running a breakdown on.
- **num_buckets** (*int*) – Optional; The maximum number of values. Defaults to 50.
- **virtual_hosts** (*str*) – Optional; Whether to include virtual hosts in the results. Valid values are "EXCLUDE", "INCLUDE", and "ONLY".
- ****kwargs** (*Any*) – Optional; Additional arguments to be passed to the query.

Returns
The result set returned.

Return type
dict

bulk_view(*document_ids: List[str]*, *max_workers: int = 20*, *at_time: str | date | datetime | None = None*, ***kwargs: Any*) → Dict[str, dict]

Bulk view documents from current index.

View the current structured data we have on a list of documents.

Parameters

- **document_ids** (*List[str]*) – The IDs of the documents you are requesting.
- **max_workers** (*int*) – Optional; The number of workers to use. Defaults to 20.
- **at_time** (*[str, datetime.date, datetime.datetime]*) – Optional; Fetches a document at a given point in time.
- ****kwargs** (*Any*) – Optional; Additional arguments to be passed to the query.

Returns
The result set returned.

Return type
Dict[str, dict]

list_hosts_with_tag(*tag_id: str*) → List[str]

Returns a list of hosts which are tagged with the specified tag.

Parameters
tag_id (*str*) – The ID of the tag.

Returns
A list of host IP addresses.

Return type
List[str]

metadata() → dict

Get metadata for the host index.

Returns
The result set returned.

Return type
dict

search(*query: str*, *per_page: int = 100*, *cursor: str | None = None*, *pages: int = 1*, *fields: List[str] | None = None*, *sort: str | List[str] | None = None*, *virtual_hosts: str | None = None*, ***kwargs: Any*) → *Query*

Search host index.

Searches the given index for all records that match the given query. For more details, see our documentation: https://search.censys.io/api

Parameters

- **query** (*str*) – The query to be executed.
- **per_page** (*int*) – Optional; The number of results to be returned for each page. Defaults to 100.

- **cursor** (*int*) – Optional; The cursor of the desired result set.
- **pages** (*int*) – Optional; The number of pages returned. Defaults to 1.
- **fields** (*List[str]*) – Optional; The fields to return. Defaults to all fields.
- **sort** (*str*) – Optional; The method used to sort results. Valid values are "RELEVANCE", "DESCENDING", and "ASCENDING".
- **virtual_hosts** (*str*) – Optional; Whether to include virtual hosts in the results. Valid values are "EXCLUDE", "INCLUDE", and "ONLY".
- ****kwargs** (*Any*) – Optional; Additional arguments to be passed to the query.

Returns
Query object that can be a callable or an iterable.

Return type
Query

view(*document_id: str*, *at_time: str | date | datetime | None = None*, ***kwargs: Any*) → dict

View document from current index.

View the current structured data we have on a specific document. For more details, see our documentation: https://search.censys.io/api

Parameters

- **document_id** (*str*) – The ID of the document you are requesting.
- **at_time** (*[str, datetime.date, datetime.datetime]*) – Optional; Fetches a document at a given point in time.
- ****kwargs** (*Any*) – Optional; Additional arguments to be passed to the query.

Returns
The result set returned.

Return type
dict

view_host_certificates(*ip: str*, *per_page: int = 100*, *start_time: str | date | datetime | None = None*, *cursor: str | None = None*) → dict

Returns a list of certificates for the specified host.

Parameters

- **ip** (*str*) – The IP address of the requested host.
- **per_page** (*int*) – Optional; The number of results to be returned for each page. Defaults to 100.
- **start_time** (*Datetime*) – Optional; An RFC3339 timestamp which represents the beginning chronological point-in-time (inclusive) from which events are returned.
- **cursor** (*str*) – Optional; Cursor token from the API response.

Returns
A list of certificates.

Return type
dict

view_host_diff(*ip: str*, *ip_b: str | None = None*, *at_time: str | date | datetime | None = None*, *at_time_b: str | date | datetime | None = None*)

Fetches a diff of the specified IP address.

Parameters

- **ip** (*str*) – The IP address of the requested host.
- **ip_b** (*str*) – Optional; The IP address of the second host.
- **at_time** (*Datetime*) – Optional; An RFC3339 timestamp which represents the point-in-time used as the basis for Host A.
- **at_time_b** (*Datetime*) – Optional; An RFC3339 timestamp which represents the point-in-time used as the basis for Host B.

Returns
A diff of the hosts.

Return type
dict

view_host_events(*ip: str*, *start_time: str | date | datetime | None = None*, *end_time: str | date | datetime | None = None*, *per_page: int | None = None*, *cursor: str | None = None*, *reversed: bool | None = None*) → dict

Fetches a list of events for the specified IP address.

Parameters

- **ip** (*str*) – The IP address of the requested host.
- **start_time** (*Datetime*) – Optional; An RFC3339 timestamp which represents the beginning chronological point-in-time (inclusive) from which events are returned.
- **end_time** (*Datetime*) – Optional; An RFC3339 timestamp which represents the ending chronological point-in-time (exclusive) from which events are returned.
- **per_page** (*int*) – Optional; The maximum number of hits to return in each response (minimum of 1, maximum of 50).
- **cursor** (*str*) – Optional; Cursor token from the API response.
- **reversed** (*bool*) – Optional; Reverse the order of the return events, that is, return events in reversed chronological order.

Returns
A list of events.

Return type
dict

view_host_names(*ip: str*, *per_page: int | None = None*, *cursor: str | None = None*) → List[str]

Fetches a list of host names for the specified IP address.

Parameters

- **ip** (*str*) – The IP address of the requested host.
- **per_page** (*int*) – Optional; The number of results to be returned for each page. Defaults to 100.
- **cursor** (*int*) – Optional; The cursor of the desired result set.

Returns
A list of host names.

Return type
List[str]

class censys.search.**SearchClient**(**args*, ***kwargs*)

Bases: object

Client for interacting with all Search APIs.

All indexes are passed the args and kwargs that are provided.

Examples

Inits SearchClient.

```
>>> from censys.search import SearchClient
>>> c = SearchClient()
```

Access both v1 and v2 indexes.

```
>>> data = c.v1.data # CensysData()
>>> hosts = c.v2.hosts # CensysHosts()
>>> certs = c.v2.certs # CensysCerts()
```

2.2 CLI Reference

2.2.1 censys

```
usage: censys [-h] [-v] {account,asm,config,hnri,search,subdomains,view} ...
```

-h, --help
show this help message and exit

-v, --version
display version

censys account

Check Censys account details and quota

```
usage: censys account [-h] [--api-id API_ID] [--api-secret API_SECRET] [-j]
```

-h, --help
show this help message and exit

--api-id <api_id>
a Censys API ID (alternatively you can use the env variable CENSYS_API_ID)

--api-secret <api_secret>
a Censys API SECRET (alternatively you can use the env variable CENSYS_API_SECRET)

-j, --json
Output in JSON format

censys asm

Interact with the Censys ASM API

```
usage: censys asm [-h]
                  {config,add-seeds,delete-seeds,delete-all-seeds,delete-labeled-seeds,
↪replace-labeled-seeds,list-seeds,list-saved-queries,add-saved-query,get-saved-query-by-
↪id,edit-saved-query-by-id,delete-saved-query-by-id,execute-saved-query-by-name,execute-
↪saved-query-by-id,search}
                  ...
```

-h, --help
show this help message and exit

censys asm add-saved-query

Add a saved query to ASM

```
usage: censys asm add-saved-query [-h] [--api-key API_KEY] --query-name
                                  QUERY_NAME --query QUERY [-v]
```

-h, --help
show this help message and exit

--api-key <api_key>
a Censys ASM API Key (alternatively you can use the env variable CENSYS_ASM_API_KEY)

--query-name <query_name>
Name of the saved query

--query <query>
Query string

-v, --verbose
verbose output

censys asm add-seeds

Add seeds to ASM

```
usage: censys asm add-seeds [-h] [--api-key API_KEY] [-v]
                            [--default-type {IP_ADDRESS,DOMAIN_NAME,CIDR,ASN}]
                            [--csv]
                            (--input-file INPUT_FILE | --json JSON | --nmap-xml NMAP_XML)
                            [-l LABEL]
```

-h, --help
show this help message and exit

--api-key <api_key>
a Censys ASM API Key (alternatively you can use the env variable CENSYS_ASM_API_KEY)

-v, --verbose
verbose output

--default-type {IP_ADDRESS,DOMAIN_NAME,CIDR,ASN}
: type of the seed(s) if type is not already provided (default: IP_ADDRESS)

--csv
: process input in CSV format

--input-file <input_file>, **-i** <input_file>
: input file name containing valid seeds in JSON format, unless –csv is specified (use - for stdin)

--json <json>, **-j** <json>
: input string containing valid json seeds

--nmap-xml <nmap_xml>
: input file name containing valid xml nmap output

-l <label>, **--label** <label>
: label to apply to seeds without label (default: “”)

censys asm config

Configure Censys ASM API Settings

```
usage: censys asm config [-h]
```

-h, **--help**
: show this help message and exit

censys asm delete-all-seeds

Delete all ASM seeds

```
usage: censys asm delete-all-seeds [-h] [--api-key API_KEY] [-f] [-v]
```

-h, **--help**
: show this help message and exit

--api-key <api_key>
: a Censys ASM API Key (alternatively you can use the env variable CENSYS_ASM_API_KEY)

-f, **--force**
: force delete all (no confirmation prompt)

-v, **--verbose**
: verbose output

censys asm delete-labeled-seeds

Delete all ASM seeds with specified label

```
usage: censys asm delete-labeled-seeds [-h] [--api-key API_KEY] -l LABEL [-v]
```

-h, --help
show this help message and exit

--api-key <api_key>
a Censys ASM API Key (alternatively you can use the env variable CENSYS_ASM_API_KEY)

-l <label>, **--label** <label>
label for which to delete all seeds

-v, --verbose
verbose output

censys asm delete-saved-query-by-id

Delete a saved query by ID

```
usage: censys asm delete-saved-query-by-id [-h] [--api-key API_KEY] --query-id
                                           QUERY_ID [-v]
```

-h, --help
show this help message and exit

--api-key <api_key>
a Censys ASM API Key (alternatively you can use the env variable CENSYS_ASM_API_KEY)

--query-id <query_id>
ID of the saved query

-v, --verbose
verbose output

censys asm delete-seeds

Delete ASM seeds

```
usage: censys asm delete-seeds [-h] [--api-key API_KEY] [-v] [--csv]
                               (--input-file INPUT_FILE | --json JSON | --nmap-xml NMAP_
↪XML)
```

-h, --help
show this help message and exit

--api-key <api_key>
a Censys ASM API Key (alternatively you can use the env variable CENSYS_ASM_API_KEY)

-v, --verbose
verbose output

--csv

process input in CSV format

--input-file <input_file>, **-i** <input_file>

input file name containing valid seeds in JSON format, unless –csv is specified (use - for stdin)

--json <json>, **-j** <json>

input string containing valid json seeds

--nmap-xml <nmap_xml>

input file name containing valid xml nmap output

censys asm edit-saved-query-by-id

Edit a saved query by ID

```
usage: censys asm edit-saved-query-by-id [-h] [--api-key API_KEY] --query-id
                                         QUERY_ID --query-name QUERY_NAME
                                         --query QUERY [-v]
```

-h, **--help**

show this help message and exit

--api-key <api_key>

a Censys ASM API Key (alternatively you can use the env variable CENSYS_ASM_API_KEY)

--query-id <query_id>

ID of the saved query

--query-name <query_name>

Name of the saved query

--query <query>

Query string

-v, **--verbose**

verbose output

censys asm execute-saved-query-by-id

Execute a saved query by id in inventory search

```
usage: censys asm execute-saved-query-by-id [-h] [--api-key API_KEY]
                                            --query-id QUERY_ID
                                            [--page-size PAGE_SIZE]
                                            [--sort SORT [SORT ...]]
                                            [--fields FIELDS [FIELDS ...]]
                                            [--pages PAGES] [-v]
```

-h, **--help**

show this help message and exit

--api-key <api_key>

a Censys ASM API Key (alternatively you can use the env variable CENSYS_ASM_API_KEY)

--query-id <query_id>
: Query ID

--page-size <page_size>
: Number of results to return. Defaults to 50.

--sort <sort>
: Sort order for results

--fields <fields>
: Fields to include in results

--pages <pages>
: Number of pages to return. Defaults to 1.

-v, --verbose
: verbose output

censys asm execute-saved-query-by-name

Execute a saved query by name in inventory search

```
usage: censys asm execute-saved-query-by-name [-h] [--api-key API_KEY]
                                              --query-name QUERY_NAME
                                              [--page-size PAGE_SIZE]
                                              [--sort SORT [SORT ...]]
                                              [--fields FIELDS [FIELDS ...]]
                                              [--pages PAGES] [-v]
```

-h, --help
: show this help message and exit

--api-key <api_key>
: a Censys ASM API Key (alternatively you can use the env variable CENSYS_ASM_API_KEY)

--query-name <query_name>
: Query name

--page-size <page_size>
: Number of results to return. Defaults to 50.

--sort <sort>
: Sort order for results

--fields <fields>
: Fields to include in results

--pages <pages>
: Number of pages to return. Defaults to 1.

-v, --verbose
: verbose output

censys asm get-saved-query-by-id

Get a saved query by ID

```
usage: censys asm get-saved-query-by-id [-h] [--api-key API_KEY] --query-id
                                        QUERY_ID [-v]
```

-h, --help
show this help message and exit

--api-key <api_key>
a Censys ASM API Key (alternatively you can use the env variable CENSYS_ASM_API_KEY)

--query-id <query_id>
ID of the saved query

-v, --verbose
verbose output

censys asm list-saved-queries

List all ASM saved queries, optionally filtered by query name prefix and filter term

```
usage: censys asm list-saved-queries [-h] [--api-key API_KEY]
                                     [--query-name-prefix QUERY_NAME_PREFIX]
                                     [--filter-term FILTER_TERM]
                                     [--page-size PAGE_SIZE] [--page PAGE]
                                     [--csv] [-v]
```

-h, --help
show this help message and exit

--api-key <api_key>
a Censys ASM API Key (alternatively you can use the env variable CENSYS_ASM_API_KEY)

--query-name-prefix <query_name_prefix>
Prefix for the saved query name to filter by

--filter-term <filter_term>
Term used to filter the list of saved query names and the saved queries

--page-size <page_size>
Number of results to return. Defaults to 50.

--page <page>
Page number to begin at when searching. Defaults to 1.

--csv
output in CSV format (otherwise JSON)

-v, --verbose
verbose output

censys asm list-seeds

List all ASM seeds, optionally filtered by label and type

```
usage: censys asm list-seeds [-h] [--api-key API_KEY]
                             [-t {IP_ADDRESS,DOMAIN_NAME,CIDR,ASN}] [-l LABEL]
                             [--csv] [-v]
```

-h, --help
show this help message and exit

--api-key <api_key>
a Censys ASM API Key (alternatively you can use the env variable CENSYS_ASM_API_KEY)

-t {IP_ADDRESS,DOMAIN_NAME,CIDR,ASN}, **--type** {IP_ADDRESS,DOMAIN_NAME,CIDR,ASN}
type of the seed to list, if not specified, all types returned

-l <label>, **--label** <label>
label of seeds to list, if not specified, all labels returned

--csv
output in CSV format (otherwise JSON)

-v, --verbose
verbose output

censys asm replace-labeled-seeds

Replace all ASM seeds with specified label with new seeds

```
usage: censys asm replace-labeled-seeds [-h] [--api-key API_KEY] -l LABEL [-v]
                                        [--default-type {IP_ADDRESS,DOMAIN_NAME,CIDR,ASN}
↪]
                                        [--csv]
                                        (--input-file INPUT_FILE | --json JSON | --nmap-
↪xml NMAP_XML)
```

-h, --help
show this help message and exit

--api-key <api_key>
a Censys ASM API Key (alternatively you can use the env variable CENSYS_ASM_API_KEY)

-l <label>, **--label** <label>
label for which to replace all seeds

-v, --verbose
verbose output

--default-type {IP_ADDRESS,DOMAIN_NAME,CIDR,ASN}
type of the seed(s) if type is not already provided (default: IP_ADDRESS)

--csv
process input in CSV format

--input-file <input_file>, **-i** <input_file>

input file name containing valid seeds in JSON format, unless –csv is specified (use - for stdin)

--json <json>, **-j** <json>

input string containing valid json seeds

--nmap-xml <nmap_xml>

input file name containing valid xml nmap output

censys asm search

Execute a query in inventory search

```
usage: censys asm search [-h] [--api-key API_KEY] --query QUERY
                         [--page-size PAGE_SIZE] [--cursor CURSOR]
                         [--sort SORT [SORT ...]]
                         [--fields FIELDS [FIELDS ...]]
                         [--workspaces WORKSPACES] [--pages PAGES] [-v]
```

-h, --help

show this help message and exit

--api-key <api_key>

a Censys ASM API Key (alternatively you can use the env variable CENSYS_ASM_API_KEY)

--query <query>

Query string

--page-size <page_size>

Number of results to return. Defaults to 50.

--cursor <cursor>

Cursor to use for pagination

--sort <sort>

Sort order for results

--fields <fields>

Fields to include in results

--workspaces <workspaces>

Workspace IDs to search. Deprecated. The workspace associated with *CENSYS-API-KEY* will be used automatically.

--pages <pages>

Number of pages to return. Defaults to 1.

-v, --verbose

verbose output

censys config

Configure Censys Search API Settings

```
usage: censys config [-h]
```

-h, --help
show this help message and exit

censys hnri

Home Network Risk Identifier (H.N.R.I.)

```
usage: censys hnri [-h] [--api-id API_ID] [--api-secret API_SECRET] [-O]
```

-h, --help
show this help message and exit

--api-id <api_id>
a Censys API ID (alternatively you can use the env variable CENSYS_API_ID)

--api-secret <api_secret>
a Censys API SECRET (alternatively you can use the env variable CENSYS_API_SECRET)

-O, --open
open your IP in browser

censys search

Query Censys Search for resource data by providing a query string, the resource index, and the fields to be returned

```
usage: censys search [-h] [--api-id API_ID] [--api-secret API_SECRET]
                     [--index-type hosts|certificates] [-o OUTPUT] [-O]
                     [--pages PAGES] [--per-page PER_PAGE] [--timeout TIMEOUT]
                     [--fields FIELDS [FIELDS ...]]
                     [--sort-order {RELEVANCE,ASCENDING,DESCENDING,RANDOM}]
                     [--virtual-hosts INCLUDE|EXCLUDE|ONLY]
                     [--sort SORT [SORT ...]]
                     query
```

query
a string written in Censys Search syntax

-h, --help
show this help message and exit

--api-id <api_id>
a Censys API ID (alternatively you can use the env variable CENSYS_API_ID)

--api-secret <api_secret>
a Censys API SECRET (alternatively you can use the env variable CENSYS_API_SECRET)

--index-type {hosts,certificates}
which resource index to query

-o <output>, **--output** <output>
: output file path

-O, **--open**
: open query in browser

--pages <pages>
: number of pages of results to return (when set to -1 returns all pages available)

--per-page <per_page>
: number of results to return per page

--timeout <timeout>
: number of seconds to wait for a response

--fields <fields>
: additional fields to return in the matching results

--sort-order {RELEVANCE,ASCENDING,DESCENDING,RANDOM}
: sort order of results

--virtual-hosts {INCLUDE,EXCLUDE,ONLY}
: whether to include virtual hosts in the results

--sort <sort>
: fields to sort by

censys subdomains

Enumerates subdomains using the Censys Search Certificates index

```
usage: censys subdomains [-h] [--api-id API_ID] [--api-secret API_SECRET]
                         [--pages PAGES] [-j]
                         domain
```

domain
: The base domain to search for

-h, **--help**
: show this help message and exit

--api-id <api_id>
: a Censys API ID (alternatively you can use the env variable CENSYS_API_ID)

--api-secret <api_secret>
: a Censys API SECRET (alternatively you can use the env variable CENSYS_API_SECRET)

--pages <pages>
: Max records to query

-j, **--json**
: Output in JSON format

censys view

View a document in Censys Search by providing a document id and the resource index

```
usage: censys view [-h] [--api-id API_ID] [--api-secret API_SECRET]
                   [--index-type hosts|certificates] [-o OUTPUT] [-O]
                   [--at-time YYYY-MM-DD (HH:mm)]
                   document_id
```

document_id
: a document id (IP address or SHA-256 certificate fingerprint) to view

-h, --help
: show this help message and exit

--api-id <api_id>
: a Censys API ID (alternatively you can use the env variable CENSYS_API_ID)

--api-secret <api_secret>
: a Censys API SECRET (alternatively you can use the env variable CENSYS_API_SECRET)

--index-type {hosts,certificates}
: which resource index to query

-o <output>, **--output** <output>
: json output file path

-O, --open
: open document in browser

--at-time <yyyy-mm-dd (hh:mm)>
: Fetches a document at a given point in time

CHAPTER

THREE

THE CONTRIBUTOR GUIDE

3.1 Contributing

All contributions (no matter how small) are always welcome.

3.1.1 Working on your first Pull Request?

You can learn how from this *free* series How to Contribute to an Open Source Project on GitHub

3.2 Development

Clone the repository:

SSH

HTTPS

GitHub CLI

```
$ git clone git@github.com:censys/censys-python.git
```

```
$ git clone https://github.com/censys/censys-python.git
```

```
$ gh repo clone censys/censys-python
```

Install dependencies via `pip`:

```
$ cd censys-python/
$ poetry install
```

Run the test suite with `pytest`. More information about testing is available at *Testing*.

3.3 Testing

Testing is done using pytest.

To run the full test suite against your changes, install *dev dependencies* and simply run `pytest`. Which should return without any errors.

```
$ pytest
```

Note: Tests currently require credentials to be setup. More information about credentials is available at *Quick Start*.

PYTHON MODULE INDEX

C

censys.asm, 29
censys.asm.api, 42
censys.common, 42
censys.common.base, 43
censys.common.config, 43
censys.common.deprecation, 60
censys.common.exceptions, 44
censys.common.types, 44
censys.common.utils, 44
censys.search.v1, 60
censys.search.v1.api, 61
censys.search.v2, 61
censys.search.v2.api, 71

INDEX

Symbols

-O
    censys-hnri command line option, 98
    censys-search command line option, 99
    censys-view command line option, 100
--api-id
    censys-account command line option, 89
    censys-hnri command line option, 98
    censys-search command line option, 98
    censys-subdomains command line option, 99
    censys-view command line option, 100
--api-key
    censys-asm-add-saved-query command line option, 90
    censys-asm-add-seeds command line option, 90
    censys-asm-delete-all-seeds command line option, 91
    censys-asm-delete-labeled-seeds command line option, 92
    censys-asm-delete-saved-query-by-id command line option, 92
    censys-asm-delete-seeds command line option, 92
    censys-asm-edit-saved-query-by-id command line option, 93
    censys-asm-execute-saved-query-by-id command line option, 93
    censys-asm-execute-saved-query-by-name command line option, 94
    censys-asm-get-saved-query-by-id command line option, 95
    censys-asm-list-saved-queries command line option, 95
    censys-asm-list-seeds command line option, 96
    censys-asm-replace-labeled-seeds command line option, 96
    censys-asm-search command line option, 97
--api-secret
    censys-account command line option, 89
    censys-hnri command line option, 98
    censys-search command line option, 98
    censys-subdomains command line option, 99
    censys-view command line option, 100
--at-time
    censys-view command line option, 100
--csv
    censys-asm-add-seeds command line option, 91
    censys-asm-delete-seeds command line option, 92
    censys-asm-list-saved-queries command line option, 95
    censys-asm-list-seeds command line option, 96
    censys-asm-replace-labeled-seeds command line option, 96
--cursor
    censys-asm-search command line option, 97
--default-type
    censys-asm-add-seeds command line option, 90
    censys-asm-replace-labeled-seeds command line option, 96
--fields
    censys-asm-execute-saved-query-by-id command line option, 94
    censys-asm-execute-saved-query-by-name command line option, 94
    censys-asm-search command line option, 97
    censys-search command line option, 99
--filter-term
    censys-asm-list-saved-queries command line option, 95
--force
    censys-asm-delete-all-seeds command line option, 91
--help
    censys command line option, 89
    censys-account command line option, 89
    censys-asm command line option, 90
    censys-asm-add-saved-query command line option, 90

censys-asm-add-seeds command line option, 90
censys-asm-config command line option, 91
censys-asm-delete-all-seeds command line option, 91
censys-asm-delete-labeled-seeds command line option, 92
censys-asm-delete-saved-query-by-id command line option, 92
censys-asm-delete-seeds command line option, 92
censys-asm-edit-saved-query-by-id command line option, 93
censys-asm-execute-saved-query-by-id command line option, 93
censys-asm-execute-saved-query-by-name command line option, 94
censys-asm-get-saved-query-by-id command line option, 95
censys-asm-list-saved-queries command line option, 95
censys-asm-list-seeds command line option, 96
censys-asm-replace-labeled-seeds command line option, 96
censys-asm-search command line option, 97
censys-config command line option, 98
censys-hnri command line option, 98
censys-search command line option, 98
censys-subdomains command line option, 99
censys-view command line option, 100
--index-type
censys-search command line option, 98
censys-view command line option, 100
--input-file
censys-asm-add-seeds command line option, 91
censys-asm-delete-seeds command line option, 93
censys-asm-replace-labeled-seeds command line option, 96
--json
censys-account command line option, 89
censys-asm-add-seeds command line option, 91
censys-asm-delete-seeds command line option, 93
censys-asm-replace-labeled-seeds command line option, 97
censys-subdomains command line option, 99
--label
censys-asm-add-seeds command line option, 91
censys-asm-delete-labeled-seeds command line option, 92
censys-asm-list-seeds command line option, 96
censys-asm-replace-labeled-seeds command line option, 96
--nmap-xml
censys-asm-add-seeds command line option, 91
censys-asm-delete-seeds command line option, 93
censys-asm-replace-labeled-seeds command line option, 97
--open
censys-hnri command line option, 98
censys-search command line option, 99
censys-view command line option, 100
--output
censys-search command line option, 98
censys-view command line option, 100
--page
censys-asm-list-saved-queries command line option, 95
--page-size
censys-asm-execute-saved-query-by-id command line option, 94
censys-asm-execute-saved-query-by-name command line option, 94
censys-asm-list-saved-queries command line option, 95
censys-asm-search command line option, 97
--pages
censys-asm-execute-saved-query-by-id command line option, 94
censys-asm-execute-saved-query-by-name command line option, 94
censys-asm-search command line option, 97
censys-search command line option, 99
censys-subdomains command line option, 99
--per-page
censys-search command line option, 99
--query
censys-asm-add-saved-query command line option, 90
censys-asm-edit-saved-query-by-id command line option, 93
censys-asm-search command line option, 97
--query-id
censys-asm-delete-saved-query-by-id command line option, 92
censys-asm-edit-saved-query-by-id command line option, 93
censys-asm-execute-saved-query-by-id command line option, 93
censys-asm-get-saved-query-by-id

command line option, 95
--query-name
censys-asm-add-saved-query command line option, 90
censys-asm-edit-saved-query-by-id command line option, 93
censys-asm-execute-saved-query-by-name command line option, 94
--query-name-prefix
censys-asm-list-saved-queries command line option, 95
--sort
censys-asm-execute-saved-query-by-id command line option, 94
censys-asm-execute-saved-query-by-name command line option, 94
censys-asm-search command line option, 97
censys-search command line option, 99
--sort-order
censys-search command line option, 99
--timeout
censys-search command line option, 99
--type
censys-asm-list-seeds command line option, 96
--verbose
censys-asm-add-saved-query command line option, 90
censys-asm-add-seeds command line option, 90
censys-asm-delete-all-seeds command line option, 91
censys-asm-delete-labeled-seeds command line option, 92
censys-asm-delete-saved-query-by-id command line option, 92
censys-asm-delete-seeds command line option, 92
censys-asm-edit-saved-query-by-id command line option, 93
censys-asm-execute-saved-query-by-id command line option, 94
censys-asm-execute-saved-query-by-name command line option, 94
censys-asm-get-saved-query-by-id command line option, 95
censys-asm-list-saved-queries command line option, 95
censys-asm-list-seeds command line option, 96
censys-asm-replace-labeled-seeds command line option, 96
censys-asm-search command line option, 97
--version
censys command line option, 89
--virtual-hosts
censys-search command line option, 99
--workspaces
censys-asm-search command line option, 97
-f
censys-asm-delete-all-seeds command line option, 91
-h
censys command line option, 89
censys-account command line option, 89
censys-asm command line option, 90
censys-asm-add-saved-query command line option, 90
censys-asm-add-seeds command line option, 90
censys-asm-config command line option, 91
censys-asm-delete-all-seeds command line option, 91
censys-asm-delete-labeled-seeds command line option, 92
censys-asm-delete-saved-query-by-id command line option, 92
censys-asm-delete-seeds command line option, 92
censys-asm-edit-saved-query-by-id command line option, 93
censys-asm-execute-saved-query-by-id command line option, 93
censys-asm-execute-saved-query-by-name command line option, 94
censys-asm-get-saved-query-by-id command line option, 95
censys-asm-list-saved-queries command line option, 95
censys-asm-list-seeds command line option, 96
censys-asm-replace-labeled-seeds command line option, 96
censys-asm-search command line option, 97
censys-config command line option, 98
censys-hnri command line option, 98
censys-search command line option, 98
censys-subdomains command line option, 99
censys-view command line option, 100
-i
censys-asm-add-seeds command line option, 91
censys-asm-delete-seeds command line option, 93
censys-asm-replace-labeled-seeds command line option, 96
-j
censys-account command line option, 89

censys-asm-add-seeds command line option, 91
censys-asm-delete-seeds command line option, 93
censys-asm-replace-labeled-seeds command line option, 97
censys-subdomains command line option, 99
-l
censys-asm-add-seeds command line option, 91
censys-asm-delete-labeled-seeds command line option, 92
censys-asm-list-seeds command line option, 96
censys-asm-replace-labeled-seeds command line option, 96
-o
censys-search command line option, 98
censys-view command line option, 100
-t
censys-asm-list-seeds command line option, 96
-v
censys command line option, 89
censys-asm-add-saved-query command line option, 90
censys-asm-add-seeds command line option, 90
censys-asm-delete-all-seeds command line option, 91
censys-asm-delete-labeled-seeds command line option, 92
censys-asm-delete-saved-query-by-id command line option, 92
censys-asm-delete-seeds command line option, 92
censys-asm-edit-saved-query-by-id command line option, 93
censys-asm-execute-saved-query-by-id command line option, 94
censys-asm-execute-saved-query-by-name command line option, 94
censys-asm-get-saved-query-by-id command line option, 95
censys-asm-list-saved-queries command line option, 95
censys-asm-list-seeds command line option, 96
censys-asm-replace-labeled-seeds command line option, 96
censys-asm-search command line option, 97

A

account() (*censys.search.v1.api.CensysSearchAPIv1 method*), 61
account() (*censys.search.v2.api.CensysSearchAPIv2 method*), 72
add_cloud_assets() (*censys.asm.Beta method*), 31
add_comment() (*censys.asm.Assets method*), 29
add_comment() (*censys.search.v2.api.CensysSearchAPIv2 method*), 72
add_saved_query() (*censys.asm.SavedQueries method*), 38
add_seeds() (*censys.asm.Seeds method*), 40
add_tag() (*censys.asm.Assets method*), 29
add_tag_to_document() (*censys.search.v2.api.CensysSearchAPIv2 method*), 72
aggregate() (*censys.asm.InventorySearch method*), 34
aggregate() (*censys.search.v2.api.CensysSearchAPIv2 method*), 72
aggregate() (*censys.search.v2.CensysCerts method*), 62
aggregate() (*censys.search.v2.CensysHosts method*), 68
ASM_EXCEPTIONS (*censys.common.exceptions.CensysExceptionMapper attribute*), 47
AsmClient (*class in censys.asm*), 29
asset_type (*censys.asm.Assets attribute*), 29
Assets (*class in censys.asm*), 29

B

base_path (*censys.asm.Beta attribute*), 31
base_path (*censys.asm.Clouds attribute*), 33
base_path (*censys.asm.InventorySearch attribute*), 34
base_path (*censys.asm.Logbook attribute*), 35
base_path (*censys.asm.Risks attribute*), 36
base_path (*censys.asm.SavedQueries attribute*), 38
base_path (*censys.asm.Seeds attribute*), 40
Beta (*class in censys.asm*), 31
bulk() (*censys.search.v2.CensysCerts method*), 62
bulk_get() (*censys.search.v2.CensysCerts method*), 63
bulk_post() (*censys.search.v2.CensysCerts method*), 63
bulk_view() (*censys.search.v2.api.CensysSearchAPIv2 method*), 73
bulk_view() (*censys.search.v2.CensysCerts method*), 63
bulk_view() (*censys.search.v2.CensysHosts method*), 68

C

censys command line option
--help, 89
--version, 89
-h, 89
-v, 89

censys.asm
    module, 29
censys.asm.api
    module, 42
censys.common
    module, 42
censys.common.base
    module, 43
censys.common.config
    module, 43
censys.common.deprecation
    module, 60
censys.common.exceptions
    module, 44
censys.common.types
    module, 44
censys.common.utils
    module, 44
censys.search.v1
    module, 60
censys.search.v1.api
    module, 61
censys.search.v2
    module, 61
censys.search.v2.api
    module, 71
censys-account command line option
    --api-id, 89
    --api-secret, 89
    --help, 89
    --json, 89
    -h, 89
    -j, 89
censys-asm command line option
    --help, 90
    -h, 90
censys-asm-add-saved-query command line option
    --api-key, 90
    --help, 90
    --query, 90
    --query-name, 90
    --verbose, 90
    -h, 90
    -v, 90
censys-asm-add-seeds command line option
    --api-key, 90
    --csv, 91
    --default-type, 90
    --help, 90
    --input-file, 91
    --json, 91
    --label, 91
    --nmap-xml, 91
    --verbose, 90
    -h, 90
    -i, 91
    -j, 91
    -l, 91
    -v, 90
censys-asm-config command line option
    --help, 91
    -h, 91
censys-asm-delete-all-seeds command line option
    --api-key, 91
    --force, 91
    --help, 91
    --verbose, 91
    -f, 91
    -h, 91
    -v, 91
censys-asm-delete-labeled-seeds command line option
    --api-key, 92
    --help, 92
    --label, 92
    --verbose, 92
    -h, 92
    -l, 92
    -v, 92
censys-asm-delete-saved-query-by-id command line option
    --api-key, 92
    --help, 92
    --query-id, 92
    --verbose, 92
    -h, 92
    -v, 92
censys-asm-delete-seeds command line option
    --api-key, 92
    --csv, 92
    --help, 92
    --input-file, 93
    --json, 93
    --nmap-xml, 93
    --verbose, 92
    -h, 92
    -i, 93
    -j, 93
    -v, 92
censys-asm-edit-saved-query-by-id command line option
    --api-key, 93
    --help, 93
    --query, 93
    --query-id, 93
    --query-name, 93

--verbose, 93
-h, 93
-v, 93
censys-asm-execute-saved-query-by-id command line option
--api-key, 93
--fields, 94
--help, 93
--page-size, 94
--pages, 94
--query-id, 93
--sort, 94
--verbose, 94
-h, 93
-v, 94
censys-asm-execute-saved-query-by-name command line option
--api-key, 94
--fields, 94
--help, 94
--page-size, 94
--pages, 94
--query-name, 94
--sort, 94
--verbose, 94
-h, 94
-v, 94
censys-asm-get-saved-query-by-id command line option
--api-key, 95
--help, 95
--query-id, 95
--verbose, 95
-h, 95
-v, 95
censys-asm-list-saved-queries command line option
--api-key, 95
--csv, 95
--filter-term, 95
--help, 95
--page, 95
--page-size, 95
--query-name-prefix, 95
--verbose, 95
-h, 95
-v, 95
censys-asm-list-seeds command line option
--api-key, 96
--csv, 96
--help, 96
--label, 96
--type, 96
--verbose, 96
-h, 96
-l, 96
-t, 96
-v, 96
censys-asm-replace-labeled-seeds command line option
--api-key, 96
--csv, 96
--default-type, 96
--help, 96
--input-file, 96
--json, 97
--label, 96
--nmap-xml, 97
--verbose, 96
-h, 96
-i, 96
-j, 97
-l, 96
-v, 96
censys-asm-search command line option
--api-key, 97
--cursor, 97
--fields, 97
--help, 97
--page-size, 97
--pages, 97
--query, 97
--sort, 97
--verbose, 97
--workspaces, 97
-h, 97
-v, 97
censys-config command line option
--help, 98
-h, 98
censys-hnri command line option
-0, 98
--api-id, 98
--api-secret, 98
--help, 98
--open, 98
-h, 98
censys-search command line option
-0, 99
--api-id, 98
--api-secret, 98
--fields, 99
--help, 98
--index-type, 98
--open, 99
--output, 98
--pages, 99
--per-page, 99

--sort, 99
--sort-order, 99
--timeout, 99
--virtual-hosts, 99
-h, 98
-o, 98
query, 98
censys-subdomains command line option
--api-id, 99
--api-secret, 99
--help, 99
--json, 99
--pages, 99
-h, 99
-j, 99
domain, 99
censys-view command line option
-O, 100
--api-id, 100
--api-secret, 100
--at-time, 100
--help, 100
--index-type, 100
--open, 100
--output, 100
-h, 100
-o, 100
document_id, 100
CensysAPIBase (*class in censys.common.base*), 43
CensysAPIException, 44
CensysAppDownForMaintenanceException, 44
CensysAsmAPI (*class in censys.asm.api*), 42
CensysAsmException, 44
CensysAsmUnauthorizedException, 45
CensysAssetExcludedException, 45
CensysAssetNotFoundException, 45
CensysAssociatedAssetsThresholdWarningException, 45
CensysBadJSONBodyException, 46
CensysCannotCreateTagWithNewColorException, 46
CensysCannotRemoveNonExistentSeedsException, 46
CensysCannotRemoveNonSeedsException, 46
CensysCertificateNotFoundException, 47
CensysCerts (*class in censys.search.v2*), 61
CensysCLIException, 46
CensysCommentNotFoundException, 47
CensysData (*class in censys.search.v1*), 60
CensysDomainNotFoundException, 47
CensysException, 47
CensysExceptionMapper (*class in censys.common.exceptions*), 47
CensysHostNotFoundException, 49
CensysHosts (*class in censys.search.v2*), 66
CensysInternalServerErrorException, 49
CensysInternalServerException, 49
CensysInvalidAPIKeyException, 49
CensysInvalidAuthTokenException, 49
CensysInvalidCloudAssetDataException, 50
CensysInvalidCloudException, 50
CensysInvalidColorException, 50
CensysInvalidCommentException, 50
CensysInvalidCommentHTMLException, 50
CensysInvalidDateException, 50
CensysInvalidIPv4AddressException, 50
CensysInvalidKeywordsInBodyException, 51
CensysInvalidLogbookCursorException, 51
CensysInvalidObjectStorageAssetIdentifierException, 51
CensysInvalidObjectStorageAssetNotFoundException, 52
CensysInvalidPageSizeException, 53
CensysInvalidRequestException, 53
CensysInvalidSearchAPIResponseException, 53
CensysInvalidSeedDataException, 54
CensysInvalidSeedTypeException, 54
CensysJSONDecodeException, 54
CensysMissingApiKeyException, 54
CensysNeedConfirmationToRemoveParentSeedsException, 54
CensysNotASeedException, 55
CensysNotFoundException, 55
CensysPageNumberOutOfRangeException, 55
CensysRateLimitExceededException, 56
CensysRiskNotFoundException, 56
CensysSearchAPIErrorException, 56
CensysSearchAPITimeoutException, 56
CensysSearchAPIv1 (*class in censys.search.v1.api*), 61
CensysSearchAPIv2 (*class in censys.search.v2.api*), 71
CensysSearchAPIv2.Query (*class in censys.search.v2.api*), 71
CensysSearchException, 56
CensysSeedNotFoundException, 56
CensysSubdomainNotFoundException, 56
CensysTagColorHasTrailingOrLeadingWhitespaceException, 57
CensysTagColorTooLongException, 57
CensysTagHasTrailingOrLeadingWhitespaceException, 57
CensysTagIsEmptyStringException, 58
CensysTagLabelsDifferOnlyInCasingException, 58
CensysTagLabelTooLongException, 58
CensysTeamNotFoundException, 59
CensysTooManyInputNodesException, 59
CensysTooManyRequestsException, 59
CensysTooSoonToResendInviteException, 59

CensysUnauthorizedException, 59
CertificatesAssets (*class in censys.asm*), 32
Clouds (*class in censys.asm*), 33
create_tag() (*censys.search.v2.api.CensysSearchAPIv2 method*), 73

D

DEFAULT_MAX_RETRIES (*censys.common.base.CensysAPIBase attribute*), 43
DEFAULT_TIMEOUT (*censys.common.base.CensysAPIBase attribute*), 43
DEFAULT_URL (*censys.asm.api.CensysAsmAPI attribute*), 42
DEFAULT_URL (*censys.search.v1.api.CensysSearchAPIv1 attribute*), 61
DEFAULT_URL (*censys.search.v2.api.CensysSearchAPIv2 attribute*), 71
DEFAULT_USER_AGENT (*censys.common.base.CensysAPIBase attribute*), 43
delete_comment() (*censys.asm.Assets method*), 30
delete_comment() (*censys.search.v2.api.CensysSearchAPIv2 method*), 73
delete_saved_query_by_id() (*censys.asm.SavedQueries method*), 39
delete_seed_by_id() (*censys.asm.Seeds method*), 40
delete_seeds_by_label() (*censys.asm.Seeds method*), 40
delete_tag() (*censys.asm.Assets method*), 30
delete_tag() (*censys.search.v2.api.CensysSearchAPIv2 method*), 73
DeprecationDecorator (*class in censys.common.deprecation*), 60
document_id
    censys-view command line option, 100
domain
    censys-subdomains command line option, 99
DomainsAssets (*class in censys.asm*), 34

E

edit_saved_query_by_id() (*censys.asm.SavedQueries method*), 39
Events (*in module censys.asm*), 34

F

fields() (*censys.asm.InventorySearch method*), 34
format_iso8601() (*in module censys.common.utils*), 44
format_rfc3339() (*in module censys.common.utils*), 44

G

get_asset_by_id() (*censys.asm.Assets method*), 30
get_asset_counts() (*censys.asm.Beta method*), 31
get_assets() (*censys.asm.Assets method*), 30
get_assets() (*censys.asm.SubdomainsAssets method*), 41
get_assets() (*censys.asm.WebEntitiesAssets method*), 41
get_comment() (*censys.search.v2.api.CensysSearchAPIv2 method*), 74
get_comment_by_id() (*censys.asm.Assets method*), 30
get_comments() (*censys.asm.Assets method*), 31
get_comments() (*censys.search.v2.api.CensysSearchAPIv2 method*), 74
get_config() (*in module censys.common.config*), 43
get_config_path() (*in module censys.common.config*), 43
get_cursor() (*censys.asm.Logbook method*), 35
get_domain_counts() (*censys.asm.Clouds method*), 33
get_events() (*censys.asm.Logbook method*), 35
get_host_counts() (*censys.asm.Clouds method*), 33
get_host_counts_by_country() (*censys.asm.Beta method*), 32
get_hosts_by_cert() (*censys.search.v2.CensysCerts method*), 63
get_input_assets() (*censys.asm.Beta method*), 32
get_instances() (*censys.asm.WebEntitiesAssets method*), 42
get_logbook_data() (*censys.asm.Beta method*), 32
get_object_store_counts() (*censys.asm.Clouds method*), 33
get_observations() (*censys.search.v2.CensysCerts method*), 63
get_risk_events() (*censys.asm.Risks method*), 36
get_risk_instance() (*censys.asm.Risks method*), 36
get_risk_instances() (*censys.asm.Risks method*), 36
get_risk_type() (*censys.asm.Risks method*), 37
get_risk_types() (*censys.asm.Risks method*), 37
get_saved_queries() (*censys.asm.SavedQueries method*), 39
get_saved_query_by_id() (*censys.asm.SavedQueries method*), 39
get_seed_by_id() (*censys.asm.Seeds method*), 40
get_seeds() (*censys.asm.Seeds method*), 40
get_series() (*censys.search.v1.CensysData method*), 60
get_subdomain_counts() (*censys.asm.Clouds method*), 33
get_subdomains() (*censys.asm.DomainsAssets method*), 34
get_tag() (*censys.search.v2.api.CensysSearchAPIv2 method*), 74
get_unknown_counts() (*censys.asm.Clouds method*), 33
get_user_workspaces() (*censys.asm.Beta method*), 32

get_workspace_id() (*censys.asm.api.CensysAsmAPI method*), 42

H

HostsAssets (*class in censys.asm*), 34

I

INDEX_NAME (*censys.search.v1.api.CensysSearchAPIv1 attribute*), 61
INDEX_NAME (*censys.search.v2.api.CensysSearchAPIv2 attribute*), 71
INDEX_NAME (*censys.search.v2.CensysCerts attribute*), 62
INDEX_NAME (*censys.search.v2.CensysHosts attribute*), 68
InventorySearch (*class in censys.asm*), 34

L

list_all_tags() (*censys.search.v2.api.CensysSearchAPIv2 method*), 74
list_certs_with_tag() (*censys.search.v2.CensysCerts method*), 64
list_hosts_with_tag() (*censys.search.v2.CensysHosts method*), 68
list_tags_on_document() (*censys.search.v2.api.CensysSearchAPIv2 method*), 74
Logbook (*class in censys.asm*), 35

M

metadata() (*censys.search.v2.CensysHosts method*), 69
module
    censys.asm, 29
    censys.asm.api, 42
    censys.common, 42
    censys.common.base, 43
    censys.common.config, 43
    censys.common.deprecation, 60
    censys.common.exceptions, 44
    censys.common.types, 44
    censys.common.utils, 44
    censys.search.v1, 60
    censys.search.v1.api, 61
    censys.search.v2, 61
    censys.search.v2.api, 71

O

ObjectStoragesAssets (*class in censys.asm*), 36

P

patch_risk_instance() (*censys.asm.Risks method*), 37
patch_risk_instances() (*censys.asm.Risks method*), 37
patch_risk_type() (*censys.asm.Risks method*), 38

Q

query
    censys-search command line option, 98
quota() (*censys.search.v1.api.CensysSearchAPIv1 method*), 61
quota() (*censys.search.v2.api.CensysSearchAPIv2 method*), 75

R

raw_search() (*censys.search.v2.api.CensysSearchAPIv2 method*), 75
raw_search() (*censys.search.v2.CensysCerts method*), 64
remove_tag_from_document() (*censys.search.v2.api.CensysSearchAPIv2 method*), 75
replace_seeds_by_label() (*censys.asm.Seeds method*), 41
request_id (*censys.common.base.CensysAPIBase property*), 43
risk_events_path (*censys.asm.Risks attribute*), 38
risk_instances_path (*censys.asm.Risks attribute*), 38
risk_types_path (*censys.asm.Risks attribute*), 38
Risks (*class in censys.asm*), 36

S

SavedQueries (*class in censys.asm*), 38
search() (*censys.asm.InventorySearch method*), 35
search() (*censys.search.v2.api.CensysSearchAPIv2 method*), 75
search() (*censys.search.v2.CensysCerts method*), 64
search() (*censys.search.v2.CensysHosts method*), 69
SEARCH_EXCEPTIONS (*censys.common.exceptions.CensysExceptionMapper attribute*), 49
search_get() (*censys.search.v2.api.CensysSearchAPIv2 method*), 76
search_get() (*censys.search.v2.CensysCerts method*), 65
search_get_raw() (*censys.search.v2.api.CensysSearchAPIv2 method*), 76
search_post() (*censys.search.v2.api.CensysSearchAPIv2 method*), 76
search_post() (*censys.search.v2.CensysCerts method*), 65
search_post_raw() (*censys.search.v2.api.CensysSearchAPIv2 method*), 77

search_post_raw() (*censys.search.v2.CensysCerts method*), 66
search_risk_instances() (*censys.asm.Risks method*), 38
Seeds (*class in censys.asm*), 39
SubdomainsAssets (*class in censys.asm*), 41

T

total (*censys.search.v2.api.CensysSearchAPIv2.Query attribute*), 72

U

update_comment() (*censys.search.v2.api.CensysSearchAPIv2 method*), 77
update_tag() (*censys.search.v2.api.CensysSearchAPIv2 method*), 78

V

view() (*censys.search.v2.api.CensysSearchAPIv2 method*), 78
view() (*censys.search.v2.CensysCerts method*), 66
view() (*censys.search.v2.CensysHosts method*), 69
view_all() (*censys.search.v2.api.CensysSearchAPIv2.Query method*), 72
view_host_certificates() (*censys.search.v2.CensysHosts method*), 70
view_host_diff() (*censys.search.v2.CensysHosts method*), 70
view_host_events() (*censys.search.v2.CensysHosts method*), 70
view_host_names() (*censys.search.v2.CensysHosts method*), 71
view_result() (*censys.search.v1.CensysData method*), 60
view_series() (*censys.search.v1.CensysData method*), 60

W

WebEntitiesAssets (*class in censys.asm*), 41
write_config() (*in module censys.common.config*), 43