FLASHPOINT

Flashpoint SDK

Table of Contents

Use Cases for Flashpoint SDK

- Easy to use tool kit that interacts with Flashpoint Indicators API platform
- Performs authentication to the Flashpoint API platform
- Expressive and intuitive syntax for communicating with the API
- Performs scrolling without need of creating POST requests to the scroll endpoint
- Contains data objects that has built-in JSON support

“Hello World” in Flashpoint SDK

1. Your initial step should be downloading and unzipping the SDK from fp.tools
2. Once that is complete, you want to change directory into the unzipped folder which is named fp_sdk-master
3. From there, you want to install the dependencies needed for the SDK to run on your machine. You will need to install the setup.py file

```
python3 setup.py install
```

4. Once that is complete, you can start writing code! So, create a new Python file, and import these lines.

```
from fp_sdk.apis.search import SearchClient
from fp_sdk import basetypes as bt
```

5. This will import the SearchClient which utilizes Flashpoint’s Search API, along with importing the several types of data that we have from basetypes. Note: We do not have data classes built for each basetype just yet, but we are working on it!
6. After you have accomplished that, and the imports work properly, it is time to initialize the Search Client properly!

```
client = SearchClient(jwt="ENTER YOUR JWT HERE"))
```

Note: As noted, you want to enter your JWT from fp.tools in place of where it says ENTER YOUR JWT HERE

7. In this scenario, we are going to watch to retrieve 5 card dumps. So, we are going to run a search through the SearchClient, and store it’s results.

```
results = client.search(basetypes=bt.CARD_DUMP, limit=5)
```

8. Just like that, we have a list of CardDumps parsed from our data! Utilize the rest of this documentation to see what methods/attributes each data class has. Enjoy!

Full “Hello World” Code Snippet

```
from fp_sdk.apis.search import SearchClient
from fp_sdk import basetypes as bt
client = SearchClient(jwt="ENTER YOUR JWT HERE"))
results = client.search(basetypes=bt.CARD_DUMP, limit=5)
```

Base FPClient Documentation

Main class for FP API interaction.

This class does the login/authentication of users, but most of the logic of interaction of different FP APIs is delegated to other classes.

This class is designed as an abstract base class, and should not be instantiated. Instead, use the subclasses.

Class Variables

– **jwt (str)** – A string containing the JWT.

– **base_url (str)** – A string containing the base url. This is used to specify an alternate API url.

****Deprecated in version 2.0.0****

– **user (str)** – A string containing a username.

– **password (str)** – A string containing a password.

Methods

login() → Union[bool, fp_sdk.client.TwoFactorToken]

Description: Performs a login when user and password were provided at instantiation.

****Removed in version 2.0.0****

two_factor_login(token: Union[fp_sdk.client.TwoFactorToken, str], otp: str) → bool

Description: Performs the second stage of a two factor login.

Parameters

Parameter Name	Type	Description
token	**(str, TwoFactorToken)**	A string or TwoFactorToken object
otp	**str**	A string containing the 2FA one time password

Indicators API Client Documentation

Subclass of the Base FP Client, which performs functionality in retrieving our IOC data.

Methods

get_attribute(attributeID: int, format: str, download: bool) → dict

Description: Method to gather an attribute from the indicators API

Parameters

Parameter Name	Type	Description
attributeID	**int**	The UUID or FPID that identifies a particular attribute.
download	**boolean**	For queries for individual attributes ,or events, download parameter downloads JSON of record. Note: The JSON is not wrapped in a list
format	**str**	The format that is supposed to be displayed. Accepted values are FP, MISP, or CSV. Note: Attribute endpoint currently only accepts FP format

get_attributes(.....) → Union[dict, Tuple[dict, fp_sdk.scroll_object.IndicatorScrollObject]]

Description: Method to gather attributes from the indicators API This method is used to retrieve a list of indicators of compromise (IOCs) that occur in the context of an event. The value of the IOC is inside the value object. Returns either a dict if there is no scrolling, or a tuple that contains a dict and a ScrollObject if there is scrolling.

Parameters

Parameter Name	Type	Description
skip	**int**	Number of results objects to skip. The sum of the limit and skip parameters cannot exceed 10,000.
limit	**int**	The maximum number of results objects to return. The sum of the limit and skip parameters cannot exceed 10,000.
format	**str**	The format that is supposed to be displayed. Accepted values are FP, MISP, or CSV. Note: Attribute endpoint currently only accepts FP format
start_date	**str**	Only retrieve values created after the specified date.

		Note: Date format is in UTC and follows ISO_8601 or relative values (eg. 30s, 5m, 2h, 3d, 2w, 3M, 2y).
end_date	**str**	Only retrieve values created after the specified date. Note: Date format is in UTC and follows ISO_8601 or relative values (eg. 30s, 5m, 2h, 3d, 2w, 3M, 2y).
updated_since	**str**	Only retrieve values created after the specified date. Note: Date format is in UTC and follows ISO_8601 or relative values (eg. 30s, 5m, 2h, 3d, 2w, 3M, 2y).
updated_until	**str**	Only retrieve values updated or created until the specified date. Note: Date format is in UTC and follows ISO_8601 or relative values (eg. 30s, 5m, 2h, 3d, 2w, 3M, 2y).
scroll	**boolean**	Initiates the scrolling of results. To initiate scrolling set the scroll parameter to true on either the /event endpoint, or the /attribute endpoint. Scrolling can be enabled for both events and attributes. To continue scrolling for both events and attributes, do a POST request to the /scroll endpoint using the returned scroll_id endpoint. Note: Scrolling does not work with the CSV format.
search_fields	**str**	Search specific value types. This should be a string of the format a=something, b>10 for each comparison. Note: More information can be found here: https://www.circl.lu/doc/misp/ categories-and-types/#types
search_tags	**str**	Search for a keyword inside the Tags. Can have multiple keywords in a list, such as malware, ransomware.
report	**str**	Obtain items related to a specific report, identified by its FPID. For more information about reports, read the Reports API documentation.
query	**str**	A free text search, also accepts the Lucene query syntax

sort_timestamp	**str**	Sort by the timestamp, either ascending or descending
attack_ids	**str**	A comma-delimited list of MITRE ATTACK ids to filter events by.
types	**str**	Search by Attribute types. Can have multiple terms, e.g. api/v4/indicators/attribute?types=url,domain,ip-src.

get_csv_event(eventID: int, format: str, download: bool) → dict

Description: Method to gather an event in CSV format from the indicators API

Parameters

Parameter Name	Type	Description
eventID	**int**	The UUID or FPID that identifies a particular event.
download	**boolean**	For queries for individual attributes ,or events, download parameter downloads JSON of record. Note: The JSON is not wrapped in a list

get_csv_events(.....) →dict

Description: Method to gather events from the indicators API in CSV format. This method is used to gather a list of events. Events are groups of different indicators of compromise that contain metadata about the situation where these indicators have been observed. Scrolling does not work with CSV events.

Parameters

Parameter Name	Type	Description
skip	**int**	Number of results objects to skip. The sum of the limit and skip parameters cannot exceed 10,000.
limit	**int**	The maximum number of results objects to return. The sum of the limit and skip parameters cannot exceed 10,000.
start_date	**str**	Only retrieve values created after the specified date. Note: Date format is in UTC and follows ISO_8601 or relative values (eg. 30s, 5m, 2h, 3d, 2w, 3M, 2y).

end_date	**str**	Only retrieve values created after the specified date. Note: Date format is in UTC and follows ISO_8601 or relative values (eg. 30s, 5m, 2h, 3d, 2w, 3M, 2y).
updated_since	**str**	Only retrieve values created after the specified date. Note: Date format is in UTC and follows ISO_8601 or relative values (eg. 30s, 5m, 2h, 3d, 2w, 3M, 2y).
updated_until	**str**	Only retrieve values updated or created until the specified date. Note: Date format is in UTC and follows ISO_8601 or relative values (eg. 30s, 5m, 2h, 3d, 2w, 3M, 2y).
search_fields	**str**	Search specific value types. This should be a string of the format a=something, b>10 for each comparison. Note: More information can be found here: https://www.circl.lu/doc/misp/ categories-and-types/#types
search_tags	**str**	Search for a keyword inside the Tags. Can have multiple keywords in a list, such as malware, ransomware.
report	**str**	Obtain items related to a specific report, identified by its FPID. For more information about reports, read the Reports API documentation.
query	**str**	A free text search, also accepts the Lucene query syntax
sort_timestamp	**str**	Sort by the timestamp, either ascending or descending
attack_ids	**str**	A comma-delimited list of MITRE ATTACK ids to filter events by.
types	**str**	Search by Attribute types. Can have multiple terms, e.g. api/v4/indicators/attribute?types=url,domain,ip-src.

get_event(eventID: int, format: str, download: bool) → dict

Description: Method to gather an event from the indicators API

Parameters

Parameter Name	Type	Description
eventID	**int**	The UUID or FPID that identifies a particular event.
download	**boolean**	For queries for individual attributes ,or events, download parameter downloads JSON of record. Note: The JSON is not wrapped in a list
format	**str**	The format that is supposed to be displayed. Accepted values are FP, MISP, or CSV. Note: Attribute endpoint currently only accepts FP format

get_events(.....) → Union[dict, Tuple[dict, fp_sdk.scroll_object.IndicatorScrollObject]]

Description: Method to gather events from the indicators API. This method is used to gather a list of events. Events are groupings of different indicators of compromise that contain metadata about the situation where these indicators have been observed. Returns either a dict if there is no scrolling, or a tuple that contains a dict and a ScrollObject is there is scrolling.

Parameters

Parameter Name	Type	Description
skip	**int**	Number of results objects to skip. The sum of the limit and skip parameters cannot exceed 10,000.
limit	**int**	The maximum number of results objects to return. The sum of the limit and skip parameters cannot exceed 10,000.
format	**str**	The format that is supposed to be displayed. Accepted values are FP, MISP, or CSV. Note: Attribute endpoint currently only accepts FP format
start_date	**str**	Only retrieve values created after the specified date.

		Note: Date format is in UTC and follows ISO_8601 or relative values (eg. 30s, 5m, 2h, 3d, 2w, 3M, 2y).
end_date	**str**	Only retrieve values created after the specified date. Note: Date format is in UTC and follows ISO_8601 or relative values (eg. 30s, 5m, 2h, 3d, 2w, 3M, 2y).
updated_since	**str**	Only retrieve values created after the specified date. Note: Date format is in UTC and follows ISO_8601 or relative values (eg. 30s, 5m, 2h, 3d, 2w, 3M, 2y).
updated_until	**str**	Only retrieve values updated or created until the specified date. Note: Date format is in UTC and follows ISO_8601 or relative values (eg. 30s, 5m, 2h, 3d, 2w, 3M, 2y).
scroll	**boolean**	Initiates the scrolling of results. To initiate scrolling set the scroll parameter to true on either the /event endpoint, or the /attribute endpoint. Scrolling can be enabled for both events and attributes. To continue scrolling for both events and attributes, do a POST request to the /scroll endpoint using the returned scroll_id endpoint. Note: Scrolling does not work with the CSV format.
search_fields	**str**	Search specific value types. This should be a string of the format a=something, b>10 for each comparison. Note: More information can be found here: https://www.circl.lu/doc/misp/ categories-and-types/#types
search_tags	**str**	Search for a keyword inside the Tags. Can have multiple keywords in a list, such as malware, ransomware.
report	**str**	Obtain items related to a specific report, identified by its FPID. For more information about reports, read the Reports API documentation.
query	**str**	A free text search, also accepts the Lucene query syntax

sort_timestamp	**str**	Sort by the timestamp, either ascending or descending
attack_ids	**str**	A comma-delimited list of MITRE ATTACK ids to filter events by.
types	**str**	Search by Attribute types. Can have multiple terms, e.g. api/v4/indicators/attribute?types=url,domain,ip-src.

get_misp_event(eventID: int, format: str, download: bool) → dict

Description: Method to gather an event from the indicators API in the MISP format

Parameters

Parameter Name	Type	Description
eventID	**int**	The UUID or FPID that identifies a particular event.
download	**boolean**	For queries for individual attributes ,or events, download parameter downloads JSON of record. Note: The JSON is not wrapped in a list

get_misp_events(.....) → Union[dict, Tuple[dict, fp_sdk.scroll_object.IndicatorScrollObject]]

Description: Method to gather events from the indicators API in the MISP format. This method is used to gather a list of events. Events are groupings of different indicators of compromise that contain metadata about the situation where these indicators have been observed. Returns either a dict if there is no scrolling, or a tuple that contains a dict and a ScrollObject if there is scrolling.

Parameters

Parameter Name	Type	Description
skip	**int**	Number of results objects to skip. The sum of the limit and skip parameters cannot exceed 10,000.
limit	**int**	The maximum number of results objects to return. The sum of the limit and skip parameters cannot exceed 10,000.
start_date	**str**	Only retrieve values created after the specified date. Note: Date format is in UTC and follows ISO_8601 or relative values (eg. 30s, 5m, 2h, 3d, 2w, 3M, 2y).

end_date	**str**	Only retrieve values created after the specified date. Note: Date format is in UTC and follows ISO_8601 or relative values (eg. 30s, 5m, 2h, 3d, 2w, 3M, 2y).
updated_since	**str**	Only retrieve values created after the specified date. Note: Date format is in UTC and follows ISO_8601 or relative values (eg. 30s, 5m, 2h, 3d, 2w, 3M, 2y).
updated_until	**str**	Only retrieve values updated or created until the specified date. Note: Date format is in UTC and follows ISO_8601 or relative values (eg. 30s, 5m, 2h, 3d, 2w, 3M, 2y).
scroll	**boolean**	Initiates the scrolling of results. To initiate scrolling set the scroll parameter to true on either the /event endpoint, or the /attribute endpoint. Scrolling can be enabled for both events and attributes. To continue scrolling for both events and attributes, do a POST request to the /scroll endpoint using the returned scroll_id endpoint. Note: Scrolling does not work with the CSV format.
search_fields	**str**	Search specific value types. This should be a string of the format a=something, b>10 for each comparison. Note: More information can be found here: https://www.circl.lu/doc/misp/ categories-and-types/#types
search_tags	**str**	Search for a keyword inside the Tags. Can have multiple keywords in a list, such as malware, ransomware.
report	**str**	Obtain items related to a specific report, identified by its FPID. For more information about reports, read the Reports API documentation.
query	**str**	A free text search, also accepts the Lucene query syntax
sort_timestamp	**str**	Sort by the timestamp, either ascending or descending
attack_ids	**str**	A comma-delimited list of MITRE ATTACK ids to filter events by.

types	**str**	Search by Attribute types. Can have multiple terms, e.g. api/v4/indicators/attribute?types=url,domain,ip-src.

get_simple_attribute(attributeID: int, format: str, download: bool) → dict

Description: Method to gather a simplified attribute from the indicators API

Parameters

Parameter Name	Type	Description
attributeID	**int**	The UUID or FPID that identifies a particular attribute.
download	**boolean**	For queries for individual attributes ,or events, download parameter downloads JSON of record. Note: The JSON is not wrapped in a list

get_simple_attributes(....) → Union[dict, Tuple[dict, fp_sdk.scroll_object.IndicatorScrollObject]]

Description: Method to gather simplified attributes from the indicators API This method is used to retrieve a list of indicators of compromise (IOCs) that occur in the context of an event. The value of the IOC is inside the value object. Returns either a dict if there is no scrolling, or a tuple that contains a dict and a ScrollObject is there is scrolling.

Parameters

Parameter Name	Type	Description
skip	**int**	Number of results objects to skip. The sum of the limit and skip parameters cannot exceed 10,000.
limit	**int**	The maximum number of results objects to return. The sum of the limit and skip parameters cannot exceed 10,000.
start_date	**str**	Only retrieve values created after the specified date. Note: Date format is in UTC and follows ISO_8601 or relative values (eg. 30s, 5m, 2h, 3d, 2w, 3M, 2y).
end_date	**str**	Only retrieve values created after the specified date.

		Note: Date format is in UTC and follows ISO_8601 or relative values (eg. 30s, 5m, 2h, 3d, 2w, 3M, 2y).
updated_since	**str**	Only retrieve values created after the specified date. Note: Date format is in UTC and follows ISO_8601 or relative values (eg. 30s, 5m, 2h, 3d, 2w, 3M, 2y).
updated_until	**str**	Only retrieve values updated or created until the specified date. Note: Date format is in UTC and follows ISO_8601 or relative values (eg. 30s, 5m, 2h, 3d, 2w, 3M, 2y).
scroll	**boolean**	Initiates the scrolling of results. To initiate scrolling set the scroll parameter to true on either the /event endpoint, or the /attribute endpoint. Scrolling can be enabled for both events and attributes. To continue scrolling for both events and attributes, do a POST request to the /scroll endpoint using the returned scroll_id endpoint. Note: Scrolling does not work with the CSV format.
search_fields	**str**	Search specific value types. This should be a string of the format a=something, b>10 for each comparison. Note: More information can be found here: https://www.circl.lu/doc/misp/ categories-and-types/#types
search_tags	**str**	Search for a keyword inside the Tags. Can have multiple keywords in a list, such as malware, ransomware.
report	**str**	Obtain items related to a specific report, identified by its FPID. For more information about reports, read the Reports API documentation.
query	**str**	A free text search, also accepts the Lucene query syntax
sort_timestamp	**str**	Sort by the timestamp, either ascending or descending
attack_ids	**str**	A comma-delimited list of MITRE ATTACK ids to filter events by.

types	**str**	Search by Attribute types. Can have multiple terms, e.g. api/v4/indicators/attribute?types=url,domain,ip-src.

get_stix_event(eventID: int, format: str, download: bool) → dict

Description: Method to gather an event from the indicators API in the STIX format

Parameters

Parameter Name	Type	Description
eventID	**int**	The UUID or FPID that identifies a particular event.
download	**boolean**	For queries for individual attributes ,or events, download parameter downloads JSON of record. Note: The JSON is not wrapped in a list

get_stix_events(.....) → Union[dict, Tuple[dict, fp_sdk.scroll_object.IndicatorScrollObject]]

Description: Method to gather events from the indicators API in the STIX format. This method is used to gather a list of events. Events are groupings of different indicators of compromise that contain metadata about the situation where these indicators have been observed. Returns either a dict if there is no scrolling, or a tuple that contains a dict and a ScrollObject if there is scrolling.

Parameters

Parameter Name	Type	Description
skip	**int**	Number of results objects to skip. The sum of the limit and skip parameters cannot exceed 10,000.
limit	**int**	The maximum number of results objects to return. The sum of the limit and skip parameters cannot exceed 10,000.
start_date	**str**	Only retrieve values created after the specified date. Note: Date format is in UTC and follows ISO_8601 or relative values (eg. 30s, 5m, 2h, 3d, 2w, 3M, 2y).

end_date	**str**	Only retrieve values created after the specified date. Note: Date format is in UTC and follows ISO_8601 or relative values (eg. 30s, 5m, 2h, 3d, 2w, 3M, 2y).
updated_since	**str**	Only retrieve values created after the specified date. Note: Date format is in UTC and follows ISO_8601 or relative values (eg. 30s, 5m, 2h, 3d, 2w, 3M, 2y).
updated_until	**str**	Only retrieve values updated or created until the specified date. Note: Date format is in UTC and follows ISO_8601 or relative values (eg. 30s, 5m, 2h, 3d, 2w, 3M, 2y).
scroll	**boolean**	Initiates the scrolling of results. To initiate scrolling set the scroll parameter to true on either the /event endpoint, or the /attribute endpoint. Scrolling can be enabled for both events and attributes. To continue scrolling for both events and attributes, do a POST request to the /scroll endpoint using the returned scroll_id endpoint. Note: Scrolling does not work with the CSV format.
search_fields	**str**	Search specific value types. This should be a string of the format a=something, b>10 for each comparison. Note: More information can be found here: https://www.circl.lu/doc/misp/ categories-and-types/#types
search_tags	**str**	Search for a keyword inside the Tags. Can have multiple keywords in a list, such as malware, ransomware.
report	**str**	Obtain items related to a specific report, identified by its FPID. For more information about reports, read the Reports API documentation.
query	**str**	A free text search, also accepts the Lucene query syntax
sort_timestamp	**str**	Sort by the timestamp, either ascending or descending
attack_ids	**str**	A comma-delimited list of MITRE ATTACK ids to filter events by.

types	**str**	Search by Attribute types. Can have multiple terms, e.g. api/v4/indicators/attribute?types=url,domain,ip-src.

Search API Client Documentation

Subclass of the Base FP Client, which performs functionality in running searches against Flashpoint's datasets.

Methods

search(.....) → dict

Description: General search method. This method is used to execute queries against the Flashpoint Search API. It accepts a variety of keyword arguments corresponding to the Search API parameters. It also accepts a basetypes parameter to easily run broad queries on specific basetypes. The SDK also includes a fp_sdk. basetypes module with sets of basetypes predefined. Returns either a dict if there is no scrolling, or a tuple that contains a dict and a ScrollObject is there is scrolling.

Parameters

Parameter Name	Type	Description
q	**str**	Free text search query using ES URI search described here Either query or q parameter is required. If both are provided the query takes precedence and q is ignored. Note: * or ? wildcard characters are not allowed as first characters in query terms.
query	**str**	Free text search query using ES URI search described here Either query or q parameter is required. If both are provided the query takes precedence and q is ignored. Note: * or ? wildcard characters are not allowed as first characters in query terms.
basetypes	**list**	A list of basetype strings. The module fp_sdk. basetypes exists to assist with using basetypes.
from_	**int**	Defines the offset from the first result object. The sum of from and size parameters cannot exceed 10,000.

size	**int**	The maximum number of result objects to return. The sum of from and size parameters cannot exceed 10,000
skip	**int**	Alias for *from*
limit	**int**	Alias for *size*
sort	**list**	A list of field:order pairs to sort the results by **Example:** [“created_at:desc”, “author:asc”]
search_after	**str**	The identified of the item to search after
aggregations	**str**	Fields to get aggregations on
highlight	**bool**	Enable highlighting on matched terms in the results.
highlight_query	**str**	Free form text query to highlight within results (but not filter).
highlight_size	**int**	The approximate number of characters to return in the highlighted element. Set to 0 to return the full document. The highlight parameter must be true for this to take effect.
source	**bool**	Return _source object from returned hits
source_includes	**list**	A list of fields to include in the _source field. **Example:** source_includes=[‘fpid’, ‘created_at’]
source_excludes	**list**	A list of fields to exclude in the _source field. **Example:** source_excludes=[‘body’]

tradition_query	**bool**	Apply traditional logic for interpreting boolean operators in the query string. Only affects behavior of the query parameter; is ignored if q is used. The input query string is parsed into boolean query tree that contains standard query_string queries as its building blocks.
fields	**list**	A list of fields to search against. Applies to query terms that don't have fields names explicitly set in the query string.
scroll	**str**	Initiate a scroll session and sets the time-to-live of the session. The scroll API can be used to paginate results of large queries. The value should be an Elasticsearch time unit, eg. 1m. This value cannot exceed 10 minutes.
default_operator	**str**	The default operator for the query_string URI search query. Applies only when q parameter is used. Can take either 'AND' or 'OR' values.
timeout	**str**	A search timeout, bounding the request to be executed within the specified time value and bail with the hits accumulated up to that point when expired. Cannot exceed 30 seconds. Accepts an Elasticsearch time unit, eg. 1m.

Base Data Object Documentation

Description: Base data class. Other data classes should inherit this class. This class should not be instantiated directly. To make a data object, use a specific subclass or use the from_response or from_basetypes class methods.

Properties

Property	Description
fpid	Unique FP identifier
last_observed_at	Integer representation of the last observed at time
last_obserted_at_str	String representation of the last observed at time

---

Indicators Common Data Object Documentation

Description: Class to hold properties for both IOC Attribute and IOC Event objects

Properties

Property	Description
sources	Sources listed under the tags which classify the Event or Attribute

Attribute Data Object Documentation

Description: Class to represent IOC Attribute data in an OO manner

Properties

Property	Description
event	The event that is parenting this specific Attribute
report	The flashpoint report linked to the specific attribute that is being used, performs HTTP requests to the /report endpoint
value	Represents the payload of an attribute, and is dependent on the type of attribue

Event Data Object Documentation

Description: Class to represent IOC Event data in an OO manner

Properties

Property	Description
attributes	All of the attributes linked to this specific Event

Methods

get_all_attributes()

Description: Function that performs HTTP calls to retrieve extended information of all Attributes that are children to this Event

---

Forum Post Data Object Documentation

Description: Class to represent Forum Post data in an OO manner

Properties

Property	Description
body	The body content of the post
created_at	The date it was created at
site	Returns a site object for the relevant site
site_actor	Returns a ForumUser with the relevant site-actor information
text	The extracted text from the post
thread_id	Returns the thread ID of this post

Methods

get_posts_from_thread(limit: int = 100)

Description: Queries the search API for other posts from the same thread.

Forum User Data Object Documentation

Description: Class to represent Form User data in an OO manner

Properties

Property	Description
handle	The handle that the site actor uses in the forum
native_id	The id of the user in the forum

Site Data Object Documentation

Description: Class to represent Site data in an OO manner

Properties

Property	Description
description	The site description

tags	Tags associated with the site
title	The site title

Common Chat Data Object Documentation

Description: Class to hold properties for Discord, Telegram, and QQ chat data objects.

Properties

Property	Description
body	The body content of the post
aliases	The different aliases of the threat actor
handle	The handle used for the threat actor in this instance
site	Details about the site the threat actor is currently on

Discord Chat Data Object Documentation

Description: Class to represent Discord chat data in an Object-Oriented (OO) manner.

Telegram Chat Data Object Documentation

Description: Class to represent Telegram chat data in an OO manner.

QQ Chat Data Object Documentation

Description: Class to represent QQ chat data in an OO manner.

Examples

Indicators Client Example

```
from fp_sdk.apis.indicators import IndicatorsClient
import os

client = IndicatorsClient(jwt=os.getenv("JWT"))  # Make sure you have your JWT exported
indicator_result = client.get_event(eventID="Hu2SoTWJWteLrH9mR94JbQ")


# Gather the results, along with the scroll object to continue scrolling
event_results, event_pages = client.get_events(scroll=True, limit=2, updated_since='2d')
for n, page in enumerate(event_pages):
   event_results.extend(page)
```

```
    # If we want to break scrolling early, can use clear_scroll() to make sure
    # we don't get "Too many active scroll session errors" in the future
    if n >= 2:
        event_pages.clear_scroll()
        break

# This iterates through all of the events, however there won't be much as the limit is 10000
print(event_results)
event_count = 1

for event in event_results:
    print(f"Event Number: {event_count}")
    print(f"Sources: {event.sources}")

    attribute_count = 1
    # Performs an API call to retrieve JSON of all Attributes, along with transforming them
into FPAttribute Data Objects
    event_attributes = event.get_all_attributes()
    for attribute in event_attributes:
        print("Event Number: {} Attribute Number: {}".format(event_count, attribute_count))
        print(f"Sources: {attribute.sources}")
        print(f"Value: {attribute.value}")
        attribute_count += 1

    event_count += 1
    print()
```

Cards Data Object Example

```
from fp_sdk.apis.search import SearchClient
from fp_sdk import basetypes as bt
import os


def print_results(results):
    for r in results:
        # using properties for common data points
        print(f"BIN: {r.bin}")
        print(f"Exp: {r.expiration}")
        print(f"Card Type: {r.card_type}")
        print(f"Bank: {r.bank_name}")

        # data objects can also be accessed like dictionaries
        print(f"Cardholder: {r.get('cardholder_information', {}).get('first')}")
        print()
```

```
client = SearchClient(jwt=os.getenv("JWT"))
results = client.search(basetypes=bt.CARD_DUMP, limit=5)
print_results(results)

results = client.search(basetypes=bt.CARD_CVV, limit=5)
print_results(results)
```

Chat Data Object Example

```
from fp_sdk.apis.search import SearchClient
from fp_sdk import basetypes as bt
import os


def print_results(results):
   for r in results:
       # Using properties for common data points
       print(f"Site Title: {r.site.title}")
       print(f"Aliases: {r.aliases}")
       print(f"Handle: {r.handle}")
       print(f"Body: {r.body}")
       print()


client = SearchClient(jwt=os.getenv("JWT"))
chat_telegram_results = client.search(basetypes=bt.TELEGRAM_MESSAGE, limit=10)
print_results(chat_telegram_results)

chat_qq_results = client.search(basetypes=bt.QQ_MESSAGE, limit=10)
print_results(chat_qq_results)

chat_discord_results = client.search(basetypes=bt.DISCORD_MESSAGE, limit=10)
print_results(chat_discord_results)
```

Forums Data Object Example

```
from fp_sdk.apis.search import SearchClient
from fp_sdk import basetypes as bt
import os


def print_results(results):
   for res in results:
       print(f"Post from {res.site.title} - {res.site.description}")
       for p in res.get_posts_from_thread(limit=5):
           print(f"{p} \n")
           print(f"Site Actor Handle: {p.site_actor.handle}")
           print(f"Post text: '{p.text}'")
```

```
            print(f"Created at: {p.created_at.get('date-time')} \n")


client = SearchClient(jwt=os.getenv("JWT"))
results = client.search(basetypes=bt.FORUM_POST, limit=2)
print_results(results)
```

Change Log

Release 2.0.0

- Removed 2fa login method
- Removed unused dependencies
- Deprecated FPClient class variables ‘user’ and ‘password’

ABOUT FLASHPOINT

Flashpoint delivers Business Risk Intelligence (BRI) to empower business units and functions across organizations with a decision advantage over potential threats and adversaries. The company’s sophisticated technology and human-powered analysis enable enterprises and public sector organizations globally to bolster cybersecurity, confront fraud, detect insider threats, enhance physical security, assess M&A opportunities, and address vendor risk and supply chain integrity.

For more information, visit www.flashpoint-intel.com or follow us on Twitter at @FlashpointIntel.