
Carbon Black Cloud Python API Documentation

Release 1.4.1

Carbon Black Developer Network

Mar 20, 2023

1	Major Features	3
2	Audience for the SDK	5
3	API Credentials	7
4	Getting Started	9
4.1	Installation	9
4.2	Authentication	11
4.3	Getting Started with the Carbon Black Cloud Python SDK - “Hello CBC”	20
4.4	Concepts	23
4.5	Guides and Resources	33
4.6	Porting Applications from CBAPI to Carbon Black Cloud SDK	34
4.7	Logging & Diagnostics	39
4.8	Testing	40
4.9	Changelog	41
5	Full SDK Documentation	49
5.1	Audit and Remediation	49
5.2	Credential Providers	69
5.3	Developing New Credential Providers	72
5.4	Endpoint Standard	75
5.5	Enterprise EDR	91
5.6	Platform	115
5.7	Workload	194
5.8	CBC SDK	213
5.9	Exceptions	338
6	Indices and tables	341
	Python Module Index	343
	Index	345

Release v1.4.1.

The Carbon Black Cloud Python SDK provides an easy interface to connect with Carbon Black Cloud products, including Endpoint Standard, Audit and Remediation, and Enterprise EDR. Use this SDK to more easily query and manage your endpoints, manipulate data as Python objects, and harness the full power of Carbon Black Cloud APIs.

CHAPTER 1

Major Features

- **Supports the following Carbon Black Cloud Products with extensions for new features and products planned**
 - Endpoint Standard
 - Audit and Remediation
 - Enterprise EDR
 - Platform
 - Workload
- **Reduced Complexity** The SDK manages the differences among Carbon Black Cloud APIs behind a single, consistent Python interface. Spend less time learning specific API calls, and more time controlling your environment.
- **More Efficient Performance** A built-in caching layer makes repeated access to the same resource more efficient. Instead of making identical API requests repeatedly, the SDK caches the results of the request the first time, and references the cache when you make future requests for the resource. This reduces the time required to access the resource later.

CHAPTER 2

Audience for the SDK

In general, the Carbon Black Cloud Python SDK is directed at those that:

- Have a working knowledge of Python.
- Have a basic understanding of what the Carbon Black Cloud does, and its basic terminology such as events, alerts, and watchlists.

CHAPTER 3

API Credentials

To use the SDK and access data in Carbon Black Cloud, you must set up API keys with the correct permissions if you are using the X-Auth-Token authentication method, or create an access token if you are using Bearer or Personal API Token. Different APIs have different permission requirements for use, which is explained in the [Developer Network Authentication Guide](#).

The SDK manages your API credentials for you. There are multiple ways to supply the SDK with your API credentials, which is explained in [Authentication](#).

CHAPTER 4

Getting Started

Get started with Carbon Black Cloud Python SDK [here](#). For detailed information on the objects and methods exposed by Carbon Black Cloud Python SDK, see the full SDK Documentation below.

4.1 Installation

If you already have Python installed, skip to [Use Pip](#).

4.1.1 Install Python

Carbon Black Cloud Python SDK is compatible with Python 3.7+. UNIX systems usually have Python installed by default; it will have to be installed on Windows systems separately.

If you believe you have Python installed already, run the following two commands at a command prompt:

```
$ python --version
Python 3.7.5

$ pip --version
pip 20.2.3 from /usr/local/lib/python3.7/site-packages (python 3.7)
```

If “python --version” reports back a version of 3.7.x or higher, you’re all set. If “pip” is not found, follow the instructions on this [guide](#).

If you’re on Windows, and Python is not installed yet, download the [latest Python installer](#) from python.org.



Ensure that the “Add Python to PATH” option is checked.

4.1.2 Use Pip

Once Python and Pip are installed, open a command prompt and type:

```
$ pip install carbon-black-cloud-sdk
```

This will download and install the latest version of the SDK from the Python PyPI packaging server.

4.1.3 Virtual Environments (optional)

If you are installing the SDK with the intent to contribute to it’s development, it is recommended that you use virtual environments to manage multiple installations.

A virtual environment is a Python environment such that the Python interpreter, libraries and scripts installed into it are isolated from those installed in other virtual environments, and (by default) any libraries installed in a “system” Python, i.e., one which is installed as part of your operating system¹.

See the [python.org virtual environment guide](https://docs.python.org/3/library/venv.html) for more information.

4.1.4 Get Source Code

Carbon Black Cloud Python SDK is actively developed on GitHub and the code is available from the [Carbon Black GitHub repository](#). The version of the SDK on GitHub reflects the latest development version.

To clone the latest version of the SDK repository from GitHub:

```
$ git clone git@github.com:carbonblack/carbon-black-cloud-sdk-python.git
```

Once you have a copy of the source, you can install it in “development” mode into your Python site-packages:

```
$ cd carbon-black-cloud-sdk-python
$ python setup.py develop
```

¹ <https://docs.python.org/3/library/venv.html>

This will link the version of carbon-black-cloud-sdk-python you cloned into your Python site-packages directory. Any changes you make to the cloned version of the SDK will be reflected in your local Python installation. This is a good choice if you are thinking of changing or further developing carbon-black-cloud-sdk-python.

4.2 Authentication

Carbon Black Cloud APIs require authentication to secure your data.

There are several methods for authentication listed below. Every method requires one of the following type of credentials X-Auth-Token, OAuth App with Bearer or Personal API Token. See the [Developer Network Authentication Guide](#) to learn how to generate the type of credentials your implementation uses.

The SDK only uses one Authentication method at a time. It is recommended to create Authentication Methods for specific actions, and use them as needed.

For example, if using the [Devices API](#) to search for mission critical devices, and the [Live Response API](#) to execute commands on those devices, generate one API credential with appropriate permissions and access level. Store the credential with a profile name, and reference the profile when creating CBCloudAPI objects.

Example contents of credentials.cbc file used for authentication with X-Auth-Token. Read more about the credentials.cbc below.

```
[platform]
url=https://defense-prod05.conferdeploy.net
token=ABCDEFGHJKLMNO123456789/ABCD123456
org_key=ABCD123456
ssl_verify=false
ssl_verify_hostname=no
```

Example code authentication with a profile named “platform”

```
# import relevant modules
>>> from cbc_sdk.platform import Device
>>> from cbc_sdk import CBCloudAPI

# create Platform API object
>>> platform_api = CBCloudAPI(profile='platform')

# search for specific devices with Platform Devices API
>>> important_devs = platform_api.select(Device).set_target_priorities(["MISSION_
↳CRITICAL"])

# execute commands with Live Response API
>>> for device in important_devs:
...     lr_session = platform_api.live_response.request_session(device.id)
...     lr_session.create_process(r'cmd.exe /c "ping.exe 192.168.1.1"')
...     lr_session.close()
```

For more examples on Live Response, check live-response

4.2.1 Authentication Methods

With a File:

Credentials may be stored in a `credentials.cbc` file. With support for multiple profiles, this method makes it easy to manage multiple API Keys for different products and permission levels.

```
>>> cbc_api = CBCloudAPI('~/.carbonblack/myfile.cbc', profile='default')
```

With Windows Registry:

Windows Registry is a secure option for storing API credentials on Windows systems.

```
>>> provider = RegistryCredentialProvider()
>>> cbc_api = CBCloudAPI(credential_provider=provider, profile='default')
```

With macOS's Keychain Access:

The Keychain Access which is built into macOS can also be used for authentication.

```
>>> provider = KeychainCredentialProvider('CBC API Credentials', 'default')
>>> cbc_api = CBCloudAPI(credential_provider=provider)
```

With Amazon Secrets Manger:

There is a support for the Amazon Secrets Manager, navigate to the section for further details of how to set it up.

```
>>> provider = AWSCredentialProvider(secret_arn='your-secret-arn-string')
>>> cbc_api = CBCloudAPI(credential_provider=provider)
```

With an External Credential Provider:

Credential Providers allow for custom methods of loading API credentials. This method requires you to write your own Credential Provider.

```
>>> provider = MyCredentialProvider()
>>> cbc_api = CBCloudAPI(credential_provider=provider, profile='default')
```

Not Recommended:

At Runtime:

Credentials may be passed into `CBCloudAPI()` via keyword parameters. This method should be used with caution, taking care to not share your API credentials when managing code with source control.

```
>>> cbc_api = CBCloudAPI(url='https://defense.conferdeploy.net', token='ABCD/
↳1234',
...     org_key='ABCDEFGH')
```

Not Recommended:

With Environmental Variables:

Environmental variables can be used for authentication, but pose a security risk. This method is not recommended unless absolutely necessary.

With a File

Credentials may be supplied in a file that resembles a Windows .INI file in structure, which allows for multiple “profiles” or sets of credentials to be supplied in a single file. The file format is backwards compatible with CBAPI, so older files can continue to be used.

Example of a credentials file containing two profiles


```
[default]
url=http://example.com
token=ABCDEFGHijklmnopqrstuvwx/12345678
org_key=A1B2C3D4
ssl_verify=false

[production]
url=http://example.com
token=QRSTUVWXYZABCDEFGHIJKLMN/76543210
org_key=A1B2C3D4
ssl_verify=false
ssl_verify_hostname=no
ssl_cert_file=foo.certs
ssl_force_tls_1_2=1
proxy=proxy.example
ignore_system_proxy=on
integration=MyApplication/1.3.1
```

Common fields between all types of credentials

Keyword	Default	Required
url		Yes
org_key		Yes
ssl_verify	1	No
ssl_verify_hostname	1	No
ignore_system_proxy	0	No
ssl_force_tls_1_2	0	No
ssl_cert_file		No
proxy		No
integration		No

X-AUTH-TOKEN specific fields

Keyword	Default	Required
token		Yes

OAuth App with Bearer specific fields

Keyword	Default	Required
csp_oauth_app_id		Yes
csp_oauth_app_secret		Yes

Personal API Token specific fields

Keyword	Default	Required
csp_api_token		Yes

Individual profiles or sections are delimited in the file by placing their name within square brackets: [profile_name]. Within each section, individual credential values are supplied in a keyword=value format.

Unrecognized keywords are ignored.

By default, the CBC SDK looks for credentials files in the following locations:

- The `.carbonblack` subdirectory of the current directory of the running process.
- The `.carbonblack` subdirectory of the user's home directory.
- The `/etc/carbonblack` subdirectory on Unix, or the `C:\Windows\carbonblack` subdirectory on Windows.

Within each of these directories, the SDK first looks for the `credentials.cbc` file, then the `credentials.psc` file (the older name for the credentials file under CBAPI).

You can override the file search logic and specify the full pathname of the credentials file in the keyword parameter `credential_file` when creating the `CBCloudAPI` object.

In all cases, you will have to specify the name of the profile to be retrieved from the credentials file in the keyword parameter `profile` when creating the `CBCloudAPI` object.

Example:

```
>>> cbc_api = CBCloudAPI(credential_file='~/.carbonblack/myfile.cbc', profile='default  
↪')
```

Note on File Security: It is recommended that the credentials file be secured properly on Unix. It should be owned by the user running the process, as should the directory containing it, and neither one should specify any file permissions for “group” or “other.” In numeric terms, that means the file should have 400 or 600 permissions, and its containing directory should have 500 or 700 permissions. This is similar to securing configuration or key files for `ssh`. If these permissions are incorrect, a warning message will be logged; a future version of the CBC SDK will disallow access to files altogether if they do not have the correct permissions.

Credential files *cannot* be properly secured in this manner under Windows; if they are used in that environment, a warning message will be logged.

With Windows Registry

CBC SDK also provides the ability to use the Windows Registry to supply credentials, a method which is more secure on Windows than other methods.

N.B.: Presently, to use the Windows Registry, you must supply its credential provider as an “external” credential provider. A future version of the CBC SDK will move to using this as a default provider when running on Windows.

By default, registry entries are stored under the key `HKEY_CURRENT_USER\Software\VMware Carbon Black\Cloud Credentials`. Under this key, there may be multiple subkeys, each of which specifies a “profile” (as with credential files). Within these subkeys, the following named values may be specified:

Common fields between all types of credentials

Keyword	Value Type	Default	Required
<code>url</code>	<code>REG_SZ</code>		Yes
<code>org_key</code>	<code>REG_SZ</code>		Yes
<code>ssl_verify</code>	<code>REG_DWORD</code>	1	No
<code>ssl_verify_hostname</code>	<code>REG_DWORD</code>	1	No
<code>ignore_system_proxy</code>	<code>REG_DWORD</code>	0	No
<code>ssl_force_tls_1_2</code>	<code>REG_DWORD</code>	0	No
<code>ssl_cert_file</code>	<code>REG_SZ</code>		No
<code>proxy</code>	<code>REG_SZ</code>		No
<code>integration</code>	<code>REG_SZ</code>		No

X-AUTH-TOKEN specific fields

Keyword	Value Type	Default	Required
token	REG_SZ		Yes

OAuth App with Bearer specific fields

Keyword	Value Type	Default	Required
csp_oauth_app_id	REG_SZ		Yes
csp_oauth_app_secret	REG_SZ		Yes

Personal API Token specific fields

Keyword	Value Type	Default	Required
csp_api_token	REG_SZ		Yes

Unrecognized named values are ignored.

To use the Registry credential provider, create an instance of it, then pass the reference to that instance in the `credential_provider` keyword parameter when creating `CBCloudAPI`. As with credential files, the name of the profile to be retrieved from the Registry should be specified in the keyword parameter `profile`.

Example:

```
>>> provider = RegistryCredentialProvider()
>>> cbc_api = CBCloudAPI(credential_provider=provider, profile='default')
```

Advanced Usage: The parameters `keypath` and `userkey` to `RegistryCredentialProvider` may be used to control the exact location of the “base” registry key where the sections of credentials are located. The `keypath` parameter allows specification of the path from `HKEY_CURRENT_USER` where the base registry key is located. If `userkey`, which is `True` by default, is `False`, the path will be interpreted as being rooted at `HKEY_LOCAL_MACHINE` rather than `HKEY_CURRENT_USER`.

Example:

```
>>> provider = RegistryCredentialProvider('Software\\Contoso\\My CBC Application')
>>> cbc_api = CBCloudAPI(credential_provider=provider, profile='default')
```

Note the use of doubled backslashes to properly escape them under Python.

With an External Credential Provider

Credentials may also be supplied by writing a class that conforms to the `CredentialProvider` interface protocol. When creating `CBCloudAPI`, pass a reference to a `CredentialProvider` object in the `credential_provider` keyword parameter. Then pass the name of the profile you want to retrieve from the provider object using the keyword parameter `profile`.

Example:

```
>>> provider = MyCredentialProvider()
>>> cbc_api = CBCloudAPI(credential_provider=provider, profile='default')
```

Details of writing a credential provider may be found in the [Developing a Custom Credential Provider](#) document.

At Runtime

The credentials may be passed into the `CBCloudAPI` object when it is created via the keyword parameters `url`, `token`, `org_key`, and (optionally) `ssl_verify` and `integration_name`.

Example:

```
>>> api = CBCloudAPI(url='https://example.com', token='ABCDEFGHJKLMNOPQRSTUVWXYZ/
↳12345678',
...                  org_key='A1B2C3D4', ssl_verify=False, integration_name='MyScript/
↳1.0')
```

The `integration_name` may be specified even if using another credential provider. If specified as a parameter, this overrides any integration name specified by means of the credential provider.

With Environmental Variables

The credentials may be supplied to CBC SDK via the environment variables `CBC_URL`, `CBC_TOKEN`, `CBC_ORG_KEY`, and `CBC_SSL_VERIFY`. For backwards compatibility with `CBAPI`, the environment variables `CBAPI_URL`, `CBAPI_TOKEN`, `CBAPI_ORG_KEY`, and `CBAPI_SSL_VERIFY` may also be used; if both are specified, the newer `CBC_XXX` environment variables override their corresponding `CBAPI_XXX` equivalents. To use the environment variables, they must be set before the application is run (at least `CBC_URL` or `CBAPI_URL`, and `CBC_TOKEN` or `CBAPI_TOKEN`), and the `credential_file` keyword parameter to `CBCloudAPI` must be either `None` or left unspecified. (The `profile` keyword parameter will be ignored.)

N.B.: Passing credentials via the environment can be insecure, and, if this method is used, a warning message to that effect will be generated in the log.

With macOS's Keychain Access

The SDK also supports the usage of macOS's Keychain Access. It works in a similar manner as our other authentication methods. Keychain Access is a key-value based password storage and since we have more than one key-value based entry we are going to use JSON to store our other entries, the JSON is going to be stored under the password value.

Note: You can start first by creating the JSON object, you can do that by using our CLI tool(<SDK_ROOT>/bin/set-macos-keychain.py) or by manually creating it. The tool can:

- Automatically import all of your profiles set in the `credentials.cbc` file. Or by setting a custom path to a file.
- Manually input the values of your credentials via prompt or by using system arguments.

Find out how to use the script in its docstring or by using `--help`.

You can remove the keys that you won't be using or leave them empty. Reference our [Explanation of API Credential Components](#).

```
{
  "url": "<URL>",
  "token" : "<TOKEN>",
  "org_key": "<ORG_KEY>",
  "ssl_verify": true,
  "ssl_verify_hostname": true,
```

(continues on next page)

(continued from previous page)

```

"ssl_cert_file": "<FILE_PATH>",
"ssl_force_tls_1_2": true,
"proxy": "<NAME_OF_THE_PROXY_HOST>",
"ignore_system_proxy": true,
"integration": "<INTEGRATION_NAME>"
}

```

Note: When you are storing a JSON object under the password's input in Keychain it is possible to see only the { in the input field, you can navigate with the arrows to check if the rest of the JSON is there.

Then we can move to storing that entry into the Keychain, create a new entry which looks like that:

Keychain Item Name:
CBC SDK API Credentials

Enter a name for this keychain item. If you are adding an Internet password item, enter its URL here (for example: https://www.apple.com).

Account Name:
default

Enter the account name associated with this keychain item.

Password:
{
 "url": "<URL>,"
 "token": "<TOKEN>"
}

Enter the password to be stored in the keychain.

Password Strength: Weak

☒ Show Password

Cancel Add

After we've set the entry in the Keychain Access we can now authenticate our SDK using the KeychainCredentialProvider.

```

>>> from cbc_sdk.credential_providers import KeychainCredentialProvider
>>> provider = KeychainCredentialProvider('CBC API Credentials', 'default')
>>> cbc_api = CBCCloudAPI(credential_provider=provider)

```

You will be prompted to type your password so that python can access the keychain in order to obtain the credentials.

4.2.2 With Amazon Secrets Manger

Configure the AWS credentials

A full and comprehensive guide configuring the files and credentials regarding AWS can be found in their [official documentation](#).

Adding a secret to the AWS Secrets Manager

There is an official [guide for creating a secret](#) by AWS.

Note: Add your secrets as a key/value pairs. In the Explanation of API Credential Components you can find full information on required fields and their purpose.

Using our credential provider for the SDK

After the configuration of the AWS Credentials and storing your secret in the AWS Secret Manager, we can start using the credential provider.

```
>>> from cbc_sdk.credential_providers import AWSCredentialProvider
>>> from cbc_sdk import CBCloudAPI
>>> provider = AWSCredentialProvider(secret_arn='your-secret-arn-string')
>>> cbc_api = CBCloudAPI(credential_provider=provider)
```

AWS Single Sign-On Provider (SSO)

If you wish to set the SSO provider follow this [tutorial](#) for setting the config.

Then you can use the `profile_name` attribute in the `AWSCredentialProvider` like so:

```
>>> from cbc_sdk.credential_providers import AWSCredentialProvider
>>> from cbc_sdk import CBCloudAPI
>>> provider = AWSCredentialProvider(secret_arn='your-secret-arn-string', profile_
↪ name="my-sso-profile")
>>> cbc_api = CBCloudAPI(credential_provider=provider)
```

4.2.3 Explanation of API Credential Components

When supplying API credentials to the SDK at runtime, with a file, or with Windows Registry, the credentials include these components:

Common fields between X-Auth-Token, OAuth App with Bearer and Personal API Token authentication methods

Keyword	Definition	De- fault	Re- quired
url	The URL used to access the Carbon Black Cloud.		Yes
org_key	The organization key specifying which organization to work with.		Yes
ssl_verify	A Boolean value (see below) indicating whether or not to validate the SSL connection.	True	No
ssl_verify_hostname	A Boolean value (see below) indicating whether or not to verify the host name of the server being connected to.	True	No
ignore_system_proxy	A Boolean value (see below). If this is True, any system proxy settings will be ignored in making the connection to the server.	False	No
ssl_force_tls12	A Boolean value (see below). If this is True, the connection will be forced to use TLS 1.2 rather than any later version.	False	No
ssl_certificate	The name of an optional certificate file used to validate the certificates of the SSL connection. If not specified, the standard system certificate verification will be used.		No
proxy	If specified, this is the name of a proxy host to be used in making the connection.		No
integration	The name of the integration to use these credentials. The string may optionally end with a slash character, followed by the integration's version number. Passed as part of the <code>User-Agent : HTTP</code> header on all requests made by the SDK.		No

X-AUTH-TOKEN specific fields

Key- word	Definition	Re- quired
token	The access token to authenticate with. Same structure as X-Auth-Token defined in the Developer Network Authentication Guide . Derived from an API Key's Secret Key and API ID.	Yes

OAuth App with Bearer specific fields

Keyword	Definition	Re- quired
csp_oauth_app_id	Client ID, enter the Client ID that you set in Create OAuth 2.0 Client.	Yes
csp_oauth_app_secret	Client Secret, enter the secret that was generated in Create OAuth 2.0 Client.	Yes

Personal API Token specific fields

Keyword	Definition	Re- quired
csp_api_token	API tokens are issued by users in an organization and are associated with the user's account and the organization from which they generated the API token.	Yes

When supplying API credentials to the SDK with environmental variables, the credentials include these components:

Keyword	Legacy	Default
CBC_URL	CBAPI_URL	
CBC_TOKEN	CBAPI_TOKEN	
CBC_ORG_KEY	CBAPI_ORG_KEY	
CBC_SSL_VERIFY	CBAPI_SSL_VERIFY	True

Alternative keywords are available to maintain backwards compatibility with CBAPI.

Boolean Values

Boolean values are specified by using the strings `true`, `yes`, `on`, or `1` to represent a `True` value, or the strings `false`, `no`, `off`, or `0` to represent a `False` value. All of these are case-insensitive. Any other string value specified will result in an error.

For example, to disable SSL connection validation, any of the following would work:

```
ssl_verify=False
ssl_verify=false
ssl_verify=No
ssl_verify=no
ssl_verify=Off
ssl_verify=off
ssl_verify=0
```

4.3 Getting Started with the Carbon Black Cloud Python SDK - “Hello CBC”

This document will help you get started with the Carbon Black Cloud Python SDK by installing it, configuring authentication for it, and executing a simple example program that makes one API call.

4.3.1 Installation

Make sure you are using Python 3. Use the command `pip install carbon-black-cloud-sdk` to install the SDK and all its dependencies. (In some environments, the correct command will be `pip3 install carbon-black-cloud-sdk` to use Python 3.)

You can also access the SDK in development mode by cloning the GitHub repository, and then executing `python setup.py develop` (in some environments, `python3 setup.py develop`) from the top-level directory. Setting your `PYTHONPATH` environment variable to the directory `[sdk]/src`, where `[sdk]` is the top-level directory of the SDK, will also work for these purposes. (On Windows, use `[sdk]\src`.)

See also the [Installation](#) section of this documentation for more information.

4.3.2 Authentication

To make use of APIs, you will need an *API token*, in case you are using Carbon Black Cloud to manage your identity and authentication, or if you are using VMware Cloud Services Platform, an *OAuth App with Bearer* or a *Personal API Token*. For our example, we will use a custom CBC-managed key with the ability to list devices. To learn more about the different authentication methods, click [here](#).

Log into the Carbon Black Cloud UI and go to `Settings > API Access`. Start by selecting `Access Levels` at the top of the screen and press `Add Access Level`. Fill in a name and description for your sample access level, keep `Copy permissions from set` to `None`, and, under the permission category `Device` and permission name `General information`, check the `Read` checkbox. Press `Save` to save and create the new access level.

Now select `API Keys` at the top of the screen and press `Add API Key`. Enter a name for the key, and, optionally, a description. For `Access Level type`, select `Custom`, and for `Custom Access Level`, select the access level you created above. Press `Save` to save and create the new API key. An `API Credentials` dialog will be displayed with the new API ID and secret key; this dialog may also be re-displayed at any time by finding the API key in the list, clicking the drop-down arrow under the `Actions` column, and selecting `API Credentials`.

We will use a credentials file to store the credential information by default. Create a directory named `.carbonblack` under your user home directory. (On Windows, this directory is generally `C:\Users\[username]`, where `[username]` is your user name.) Within this directory create a file `credentials.cbc` to store your credentials. Copy the following template to this new file:

```
[default]
url=
token=
org_key=
ssl_verify=True
```

Following the `url=` keyword, add the top-level URL you use to access the Carbon Black Cloud, including the `https://` prefix and the domain name, but without any of the path information following it.

Following the `token=` keyword, add the API Secret Key from the API Credentials dialog, followed by a forward slash (/) character, followed by the API ID from the API Credentials dialog. (The secret key is always 24 characters in length, and the API ID is always 10 characters in length.)

Following the `org_key=` keyword, add the organization key from your organization, which may be seen under the Org Key: heading at the top of the API Keys display under Settings > API Access. It is always 8 characters in length.

Save the completed `credentials.cbc` file, which should look like this (*example text only*):

```
[default]
url=https://example.net
token=ABCDEFGHGIJKLMNOPQRSTUWXYZ/ABCDEFGHIJ
org_key=A1B2C3D4
ssl_verify=True
```

On UNIX systems, you must make sure that the `credentials.cbc` file is properly secured. The simplest commands for doing so are:

```
$ chmod 600 ~/.carbonblack/credentials.cbc
$ chmod 700 ~/.carbonblack
```

For further information, please see the [Authentication](#) section of the documentation, as well as the [Authentication Guide](#) on the Carbon Black Cloud Developer Network.

4.3.3 Running the Example

The example we will be running is `list_devices.py`, located in the `examples/platform` subdirectory of the GitHub repository. If you cloned the repository, change directory to `[sdk]/examples/platform`, where `[sdk]` is the top-level directory of the SDK. (On Windows, use `[sdk]\examples\platform`.) Alternately, you may view the current version of that script in “raw” mode in GitHub, and use your browser’s Save As function to save the script locally. In that case, change directory to whichever directory you saved the script to.

Execute the script by using the command `python list_devices.py -q '1'` (in some environments, `python3 list_devices.py -q '1'`). If all is well, you will see a list of devices (endpoints) registered in your organization, showing their numeric ID, host name, IP address, and last checkin time.

You can change what devices are shown by modifying the query value supplied to the `-q` parameter, and also by using additional parameters to modify the search criteria. Execute the command `python list_devices.py --help` (in some environments, `python3 list_devices.py --help`) for a list of all possible command line parameters.

4.3.4 Inside the Example Script

Once the command-line arguments are parsed, we create a Carbon Black Cloud API object with a call to the helper function `get_cb_cloud_object()`. The standard `select()` method is used to create a query object that queries for devices; the query string is passed to that object via the `where()` method, and other criteria are added using specific setters.

The query is an iterable object, and calling upon its iterator methods invokes the query, which, in this case, is the [Search Devices](#) API. The example script turns those results into an in-memory list, then iterates on that list, printing only certain properties of each retrieved Device object.

4.3.5 Calling the SDK Directly

Now we'll repeat this example, but using the Python command line directly without a script.

Access your Python interpreter with the `python` command (or `python3` if required) and type:

```
>>> from cbc_sdk.rest_api import CBCloudAPI
>>> from cbc_sdk.platform import Device
>>> cb = CBCloudAPI(profile='default')
```

This imports the necessary classes and creates an instance of the base `CBCloudAPI` object. By default, the file credentials provider is used. We set it to use the default profile in your `credentials.cbc` file, which you set up earlier.

N.B.: On Windows, a security warning message will be generated about file access to CBC SDK credentials being inherently insecure.

```
>>> query = cb.select(Device).where('1')
```

This creates a query object that searches for all devices (the '1' causes all devices to be matched, as in SQL).

```
>>> devices = list(query)
```

For convenience, we load the entirety of the query results into an in-memory list.

```
>>> for device in devices:
...     print(device.id, device.name, device.last_internal_ip_address, device.last_
...           ↪contact_time)
...
```

Using a simple `for` loop, we print out the ID, host name, internal IP address, and last contact time from each returned device. Note that the contents of the list are `Device` objects, not dictionaries, so we access individual properties with the `object.property_name` syntax, rather than `object['property_name']`.

4.3.6 Setting the User-Agent

The SDK supports custom User-Agent's, which allow you to identify yourself when using the SDK to make API calls. The credential parameter `integration_name` is used for this. If you use a file to authenticate the SDK, this is how you could identify yourself:

```
[default]
url=http://example.com
token=ABCDEFGHijklmnopqrstuvwX/12345678
org_key=A1B2C3D4
integration_name=MyScript/0.9.0
```

See the [Authentication](#) documentation for more information about credentials.

4.4 Concepts

4.4.1 Live Response with Platform Devices

As of version 1.3.0 Live Response has been changed to support CUSTOM type API Keys which enables the platform Device model and Live Response session to be used with a single API key. Ensure your API key has the Device READ permission along with the desired live-response permissions

```
# Device information is accessible with Platform Devices
>>> from cbc_sdk import CBCloudAPI
>>> from cbc_sdk.platform import Device
>>> api = CBCloudAPI(profile='platform')
>>> platform_devices = api.select(Device).set_os(["WINDOWS", "LINUX"])
>>> for device in platform_devices:
...     print(
...         f'''
...         Device ID: {device.id}
...         Device Name: {device.name}
...
...     ''')
Device ID: 1234
Device Name: Win10x64

Device ID: 5678
Device Name: UbuntuDev

# Live Response is accessible with Platform Devices
>>> from cbc_sdk import CBCloudAPI
>>> from cbc_sdk.platform import Device
>>> api = CBCloudAPI(profile='platform')
>>> platform_device = api.select(Device, 1234)
>>> platform_device.lr_session()
url: /appservices/v6/orgs/{org_key}/liveresponse/sessions/428:1234 -> status: PENDING
[...]
```

For more examples on Live Response, check live-response

USB Devices

Note that USBDevice is distinct from either the Platform API Device or the Endpoint Standard Device. Access to USB devices is through the Endpoint Standard package from `cbc_sdk.endpoint_standard` import USBDevice.

```
# USB device information is accessible with Endpoint Standard
>>> from cbc_sdk import CBCloudAPI
>>> from cbc_sdk.endpoint_standard import USBDevice
>>> api = CBCloudAPI(profile='endpoint_standard')
>>> usb_devices = api.select(USBDevice).set_statuses(['APPROVED'])
>>> for usb in usb_devices:
...     print(f'''
...         USB Device ID: {usb.id}
...     ''')
```

(continues on next page)

(continued from previous page)

```

...         USB Device: {usb.vendor_name} {usb.product_name}
...         '''
USB Device ID: 774
USB Device: SanDisk Ultra

USB Device ID: 778
USB Device: SanDisk Cruzer Mini

```

4.4.2 Queries

Generally, to retrieve information from your Carbon Black Cloud instance you will:

1. *Create a Query*
2. *Refine the Query*
3. *Execute the Query*

Create Queries with `CBCloudAPI.select()`

Data is retrieved from the Carbon Black Cloud with `CBCloudAPI.select()` statements. A `select()` statement creates a query, which can be further *refined with parameters or criteria*, and then *executed*.

```

# Create a query for devices
>>> from cbc_sdk.platform import Device
>>> device_query = api.select(Device).where('avStatus:AV_ACTIVE')

# The query has not yet been executed
>>> type(device_query)
<class cbc_sdk.platform.devices.DeviceSearchQuery>

```

This query will search for Platform Devices with antivirus active.

Refine Queries with `where()`, `and_()`, and `or_()`

Queries can be refined during or after declaration with `where()`, `and_()`, and `or_()`.

```

# Create a query for events
>>> from cbc_sdk.endpoint_standard import Event
>>> event_query = api.select(Event).where(hostName='Win10').and_(ipAddress='10.0.0.1')

# Refine the query
>>> event_query.and_(applicationName='googleupdate.exe')
>>> event_query.and_(eventType='REGISTRY_ACCESS')
>>> event_query.and_(ownerNameExact='DevRel')

```

This query will search for Endpoint Standard Events created by the application `googleupdate.exe` accessing the registry on a device with a hostname containing `Win10`, an IP Address of `10.0.0.1`, and owned by `DevRel`.

Be Consistent When Refining Queries

All queries are of type `QueryBuilder()`, with support for either raw string-based queries, or keyword arguments.

```
# Equivalent queries
>>> from cbc_sdk.platform import Device
>>> string_query = api.select(Device).where("avStatus:AV_ACTIVE")
>>> keyword_query = api.select(Device).where(avStatus="AV_ACTIVE").
```

Queries must be consistent in their use of strings or keywords; do not mix strings and keywords.

```
# Not allowed
>>> from cbc_sdk.platform import Device
>>> mixed_query = api.select(Device).where(avStatus='Win7x').and_("virtualMachine:true
↪")
cbc_sdk.errors.ApiError: Cannot modify a structured query with a raw parameter
```

Execute a Query

A query is not executed on the server until it's accessed, either as an iterator (where it will generate results on demand as they're requested) or as a list (where it will retrieve the entire result set and save to a list).

```
# Create and Refine a query
>>> from cbc_sdk.platform import Device
>>> device_query = api.select(Device).where('avStatus:AV_ACTIVE').set_os(["WINDOWS"])

# Execute the query by accessing as a list
>>> matching_devices = [device for device in device_query]

>>> print(f"First matching device ID: {matching_devices[0].id}")
First matching device ID: 1234

# Or as an iterator
>>> for matching_device in device_query:
...     print(f"Matching device ID: {matching_device.id}")
Matching device ID: 1234
Matching device ID: 5678
```

You can also call the Python built-in `len()` on this object to retrieve the total number of items matching the query.

```
# Retrieve total number of matching devices
>>> len(device_query)
2
```

In this example, the matching device ID's are accessed with `device.id`. If using Endpoint Standard Devices, the device ID's are accessed with `device.deviceId`.

Query Parameters vs Criteria

For queries, some Carbon Black Cloud APIs use GET requests with parameters, and some use POST requests with criteria.

Parameters

Parameters modify a query. When modifying a query with `where()`, `and_()`, and `or_()`, those modifications become query parameters when sent to Carbon Black Cloud.

```
>>> device_query = api.select(endpoint_standard.Device).where(hostName='Win7').and_
↳ (ipAddress='10.0.0.1')
```

Executing this query results in an API call similar to GET /integrationServices/v3/device?hostName='Win7'&ipAddress='10.0.0.1'

Criteria

Criteria also modify a query, and can be used with or without parameters. When using CBC SDK, there are API-specific methods you can use to add criteria to queries.

```
# Create a query for alerts
>>> from cbc_sdk.platform import Alert
>>> alert_query = api.select(Alert)

# Refine the query with parameters
>>> alert_query.where(alert_severity=9).or_(alert_severity=10)

# Refine the query with criteria
>>> alert_query.set_device_os(["MAC"]).set_device_os_versions(["10.14.6"])
```

Executing this query results in an API call to POST /appservices/v6/orgs/{org_key}/alerts/_search with this JSON Request Body:

```
{
  "query": "alert_severity:9 OR alert_severity:10",
  "criteria": {
    "device_os": ["MAC"],
    "device_os_version": ["10.14.6"]
  }
}
```

The query parameters are sent in "query", and the criteria are sent in "criteria".

Modules with Support for Criteria

Run

- `cbc_sdk.audit_remediation.base.RunQuery.device_ids()`
- `cbc_sdk.audit_remediation.base.RunQuery.device_types()`
- `cbc_sdk.audit_remediation.base.RunQuery.policy_id()`

Result and Device Summary

- `cbc_sdk.audit_remediation.base.ResultQuery.set_device_ids()`
- `cbc_sdk.audit_remediation.base.ResultQuery.set_device_names()`
- `cbc_sdk.audit_remediation.base.ResultQuery.set_device_os()`
- `cbc_sdk.audit_remediation.base.ResultQuery.set_policy_ids()`
- `cbc_sdk.audit_remediation.base.ResultQuery.set_policy_names()`
- `cbc_sdk.audit_remediation.base.ResultQuery.set_status()`

ResultFacet and DeviceSummaryFacet

- `cbc_sdk.audit_remediation.base.FacetQuery.set_device_ids()`
- `cbc_sdk.audit_remediation.base.FacetQuery.set_device_names()`
- `cbc_sdk.audit_remediation.base.FacetQuery.set_device_os()`
- `cbc_sdk.audit_remediation.base.FacetQuery.set_policy_ids()`
- `cbc_sdk.audit_remediation.base.FacetQuery.set_policy_names()`
- `cbc_sdk.audit_remediation.base.FacetQuery.set_status()`

USBDeviceApprovalQuery <`cbc_sdk.endpoint_standard.usb_device_control.USBDeviceApprovalQuery`

- `cbc_sdk.endpoint_standard.usb_device_control.USBDeviceApprovalQuery.set_device_ids()`
- `cbc_sdk.endpoint_standard.usb_device_control.USBDeviceApprovalQuery.set_product_names()`
- `cbc_sdk.endpoint_standard.usb_device_control.USBDeviceApprovalQuery.set_vendor_names()`

USBDeviceQuery <`cbc_sdk.endpoint_standard.usb_device_control.USBDeviceQuery`

- `cbc_sdk.endpoint_standard.usb_device_control.USBDeviceQuery.set_endpoint_names()`
- `cbc_sdk.endpoint_standard.usb_device_control.USBDeviceQuery.set_product_names()`
- `cbc_sdk.endpoint_standard.usb_device_control.USBDeviceQuery.set_serial_numbers()`
- `cbc_sdk.endpoint_standard.usb_device_control.USBDeviceQuery.set_statuses()`
- `cbc_sdk.endpoint_standard.usb_device_control.USBDeviceQuery.set_vendor_names()`

Alert

- `cbc_sdk.platform.alerts.BaseAlertSearchQuery.set_categories()`
- `cbc_sdk.platform.alerts.BaseAlertSearchQuery.set_create_time()`
- `cbc_sdk.platform.alerts.BaseAlertSearchQuery.set_device_ids()`
- `cbc_sdk.platform.alerts.BaseAlertSearchQuery.set_device_names()`
- `cbc_sdk.platform.alerts.BaseAlertSearchQuery.set_device_os()`
- `cbc_sdk.platform.alerts.BaseAlertSearchQuery.set_device_os_versions()`
- `cbc_sdk.platform.alerts.BaseAlertSearchQuery.set_device_username()`
- `cbc_sdk.platform.alerts.BaseAlertSearchQuery.set_group_results()`
- `cbc_sdk.platform.alerts.BaseAlertSearchQuery.set_alert_ids()`
- `cbc_sdk.platform.alerts.BaseAlertSearchQuery.set_legacy_alert_ids()`
- `cbc_sdk.platform.alerts.BaseAlertSearchQuery.set_minimum_severity()`
- `cbc_sdk.platform.alerts.BaseAlertSearchQuery.set_policy_ids()`
- `cbc_sdk.platform.alerts.BaseAlertSearchQuery.set_policy_names()`
- `cbc_sdk.platform.alerts.BaseAlertSearchQuery.set_process_names()`

- `cbc_sdk.platform.alerts.BaseAlertSearchQuery.set_process_sha256()`
- `cbc_sdk.platform.alerts.BaseAlertSearchQuery.set_reputations()`
- `cbc_sdk.platform.alerts.BaseAlertSearchQuery.set_tags()`
- `cbc_sdk.platform.alerts.BaseAlertSearchQuery.set_target_priorities()`
- `cbc_sdk.platform.alerts.BaseAlertSearchQuery.set_threat_ids()`
- `cbc_sdk.platform.alerts.BaseAlertSearchQuery.set_types()`
- `cbc_sdk.platform.alerts.BaseAlertSearchQuery.set_workflows()`

WatchlistAlert

- `cbc_sdk.platform.alerts.WatchlistAlertSearchQuery.set_watchlist_ids()`
- `cbc_sdk.platform.alerts.WatchlistAlertSearchQuery.set_watchlist_names()`

CBAalyticsAlert

- `cbc_sdk.platform.alerts.CBAalyticsAlertSearchQuery.set_blocked_threat_categories()`
- `cbc_sdk.platform.alerts.CBAalyticsAlertSearchQuery.set_device_locations()`
- `cbc_sdk.platform.alerts.CBAalyticsAlertSearchQuery.set_kill_chain_statuses()`
- `cbc_sdk.platform.alerts.CBAalyticsAlertSearchQuery.set_not_blocked_threat_categories()`
- `cbc_sdk.platform.alerts.CBAalyticsAlertSearchQuery.set_policy_applied()`
- `cbc_sdk.platform.alerts.CBAalyticsAlertSearchQuery.set_reason_code()`
- `cbc_sdk.platform.alerts.CBAalyticsAlertSearchQuery.set_run_states()`
- `cbc_sdk.platform.alerts.CBAalyticsAlertSearchQuery.set_sensor_actions()`
- `cbc_sdk.platform.alerts.CBAalyticsAlertSearchQuery.set_threat_cause_vectors()`

Event

Process

Modules not yet Supported for Criteria

RunHistory

4.4.3 Asynchronous Queries

A number of queries allow for asynchronous mode of operation. Those utilize python futures and the request itself is performed in a separate worker thread. An internal thread pool is utilized to support multiple CBC queries executing in an asynchronous manner without blocking the main thread.

Execute an asynchronous query

Running asynchronous queries is done by invoking the `execute_async()` method, e.g:

```
>>> async_query = api.select(EnrichedEvent).where('process_name:chrome.exe').execute_
↳ async()
```

The `execute_async()` method returns a python future object that can be later on waited for results.

Fetching asynchronous queries' results

Results from asynchronous queries can be retrieved by using the `result()` method since they are actually futures:

```
>>> print(async_query.result())
```

This would block the main thread until the query completes.

Modules with support for asynchronous queries

`Process`

`ProcessFacet`

`EnrichedEvent`

`EnrichedEventFacet`

`USBDeviceApprovalQuery`

`USBDeviceBlockQuery`

`USBDeviceQuery`

4.4.4 Facets

Facet search queries return statistical information indicating the relative weighting of the requested values as per the specified criteria. There are two types of criteria that can be set, one is the `range` type which is used to specify discrete values (integers or timestamps - specified both as seconds since epoch and also as ISO 8601 strings). The results are then grouped by occurrence within the specified range. The other type is the `term` type which allow for one or more fields to use as a criteria on which to return weighted results.

Setting ranges

Ranges are configured via the `add_range()` method which accepts a dictionary of range settings or a list of range dictionaries:

```
>>> range = {
...     "bucket_size": "+1DAY",
...     "start": "2020-10-16T00:00:00Z",
...     "end": "2020-11-16T00:00:00Z",
...     "field": "device_timestamp"
... }
>>> query = api.select(EnrichedEventFacet).where(process_pid=1000).add_range(range)
```

The range settings are as follows:

- `field` - the field to return the range for, should be a discrete one (integer or ISO 8601 timestamp)
- `start` - the value to begin grouping at
- `end` - the value to end grouping at
- `bucket_size` - how large of a bucket to group results in. If grouping an ISO 8601 property, use a string like `'-3DAYS'`

Multiple ranges can be configured per query by passing a list of range dictionaries.

Setting terms

Terms are configured via the `add_facet_field()` method:

```
>>> query = api.select(EnrichedEventFacet).where(process_pid=1000).add_facet_field(
    ↪ "process_name")
```

The argument to `add_facet_field` method is the name of the field to be summarized.

Getting facet results

Facet results can be retrieved synchronously with the `.results` property, or asynchronously with the `.execute_async()` and `.result()` methods.

Create the query:

```
>>> event_facet_query = api.select(EventFacet).add_facet_field("event_type")
>>> event_facet_query.where(process_guid="WNEXFKQ7-00050603-0000066c-00000000-
    ↪ 1d6c9acb43e29bb")
>>> range = {
...     "bucket_size": "+1DAY",
...     "start": "2020-10-16T00:00:00Z",
...     "end": "2020-11-16T00:00:00Z",
...     "field": "device_timestamp"
... }
>>> event_facet_query.add_range(range)
```

1. With the `.results` property:

```
>>> synchronous_results = event_facet_query.results
>>> print(synchronous_results)
EventFacet object, bound to https://defense-eap01.conferdeploy.net.
-----
      num_found: 16
  processed_segments: 1
          ranges: [{'start': '2020-10-16T00:00:00Z', 'end': '2020...
          terms: [{'values': [{'total': 14, 'id': 'modload', 'na...
    total_segments: 1
```

2. With the `.execute_async()` and `.result()` methods:

```
>>> asynchronous_future = event_facet_query.execute_async()
>>> asynchronous_result = asynchronous_future.result()
>>> print(asynchronous_result)
EventFacet object, bound to https://defense-eap01.conferdeploy.net.
-----
      num_found: 16
  processed_segments: 1
          ranges: [{'start': '2020-10-16T00:00:00Z', 'end': '2020...
          terms: [{'values': [{'total': 14, 'id': 'modload', 'na...
    total_segments: 1
```

The result for facet queries is a single object with two properties: `terms` and `ranges` that contain the facet search result weighted as per the criteria provided.

```
>>> print(synchronous_result.terms)
[{'values': [{'total': 14, 'id': 'modload', 'name': 'modload'}, {'total': 2, 'id':
    ↪ 'crossproc', 'name': 'crossproc'}], 'field': 'event_type'}]
(continues on next page)
```

(continued from previous page)

```
>>> print(synchronous_result.ranges)
[{'start': '2020-10-16T00:00:00Z', 'end': '2020-11-16T00:00:00Z', 'bucket_size':
  ↳ '+1DAY', 'field': 'device_timestamp', 'values': None}]
```

Modules with support for facet searches

ProcessFacet

EventFacet

EnrichedEventFacet

4.4.5 Enriched Events

We can return the details for the enriched event for a specific event or we could return the details for all enriched events per alert.

Get details per event

```
>>> from cbc_sdk.endpoint_standard import EnrichedEvent
>>> query = cb.select(EnrichedEvent).where(alert_category='THREAT')
>>> # get the first event returned by the query
>>> item = query[0]
>>> details = item.get_details()
>>> print(
...     f'''
...     Category: {details.alert_category}
...     Type: {details.enriched_event_type}
...     Alert Id: {details.alert_id}
...     '''
... )
Category: ['THREAT']
Type: CREATE_PROCESS
Alert Id: ['3F0D00A6']
```

Get details for all events per alert

```
# Alert information is accessible with Platform CBAnalyticsAlert
>>> from cbc_sdk import CBCloudAPI
>>> from cbc_sdk.platform import CBAnalyticsAlert
>>> api = CBCloudAPI(profile='platform')
>>> query = cb.select(CBAnalyticsAlert).set_create_time(range="-4w")
>>> # get the first alert returned by the query
>>> alert = query[0]
>>> for event in alert.get_events():
...     print(
...         f'''
...         Category: {event.alert_category}
...         Type: {event.enriched_event_type}
...         Alert Id: {event.alert_id}
...         '''
...     )
Category: ['OBSERVED']
```

(continues on next page)

(continued from previous page)

```
Type: SYSTEM_API_CALL
Alert Id: ['BE084638']

Category: ['OBSERVED']
Type: NETWORK
Alert Id: ['BE084638']
```

4.4.6 Static Methods

In version 1.4.2 we introduced static methods on some classes. They handle API requests that are not tied to a specific resource id, thus they cannot be instance methods, instead static helper methods. Because those methods are static, they need a CBCloudAPI object to be passed as the first argument.

Search suggestions

```
# Search Suggestions for Observation
>>> from cbc_sdk import CBCloudAPI
>>> from cbc_sdk.platform import Observation
>>> api = CBCloudAPI(profile='platform')
>>> suggestions = Observation.search_suggestions(api, query="device_id", count=2)
>>> for suggestion in suggestions:
...     print(suggestion["term"], suggestion["required_skus_all"], suggestion[
↪ "required_skus_some"])
device_id [] ['threathunter', 'defense']
netconn_remote_device_id ['xdr'] []
```

```
# Search Suggestions for Alerts
>>> from cbc_sdk import CBCloudAPI
>>> from cbc_sdk.platform import BaseAlert
>>> api = CBCloudAPI(profile='platform')
>>> suggestions = BaseAlert.search_suggestions(api, query="device_id")
>>> for suggestion in suggestions:
...     print(suggestion["term"], suggestion["required_skus_some"])
device_id ['defense', 'threathunter', 'deviceControl']
device_os ['defense', 'threathunter', 'deviceControl']
...
workload_name ['kubernetesSecurityRuntimeProtection']
```

Bulk Get Details

```
# Observations get details per alert id
>>> from cbc_sdk import CBCloudAPI
>>> from cbc_sdk.platform import Observation
>>> api = CBCloudAPI(profile='platform')
>>> bulk_details = Observation.bulk_get_details(api, alert_id="4d49d171-0a11-0731-
↪ 5172-d0963b77d422")
>>> for obs in bulk_details:
...     print(
...         f'''
...         Category: {obs.alert_category}
...         Type: {obs.observation_type}
```

(continues on next page)

(continued from previous page)

```
...         Alert Id: {obs.alert_id}
...         '')
Category: ['THREAT']
Type: CB_ANALYTICS
Alert Id: ['4d49d171-0a11-0731-5172-d0963b77d422']
```

```
# Observations get details per observation_ids
>>> from cbc_sdk import CBCloudAPI
>>> from cbc_sdk.platform import Observation
>>> api = CBCloudAPI(profile='platform')
>>> bulk_details = Observation.bulk_get_details(api, observation_ids=["13A5F4E5-C4BD-
↳ 11ED-A7AB-005056A5B601:13a5f4e4-c4bd-11ed-a7ab-005056a5b611", "13A5F4E5-C4BD-11ED-
↳ A7AB-005056A5B601:13a5f4e4-c4bd-11ed-a7ab-005056a5b622"])
>>> for obs in bulk_details:
...     print(
...         f'''
...         Category: {obs.alert_category}
...         Type: {obs.observation_type}
...         Alert Id: {obs.alert_id}
...         ''')
Category: ['THREAT']
Type: CB_ANALYTICS
Alert Id: ['4d49d171-0a11-0731-5172-d0963b77d422']

Category: ['THREAT']
Type: CB_ANALYTICS
Alert Id: ['4d49d171-0a11-0731-5172-d0963b77d411']
```

4.5 Guides and Resources

Here we've listed a collection of tutorials, recorded demonstrations and other resources we think will be useful to get the most out of the Carbon Black Cloud Python SDK.

4.5.1 Audience for These Guides

In general, and unless otherwise indicated, these guides are directed at those that:

- Have a working knowledge of Python.
- Have a basic understanding of what the Carbon Black Cloud does, and its basic terminology such as events, alerts, and watchlists.

Certain guides may be more geared towards audiences with more experience with the Carbon Black Cloud, such as administrators.

4.5.2 Recordings

Demonstrations are found on our [YouTube channel](#).

A recent highlight shows how to schedule Audit and Remediation Tasks.

4.5.3 Guides

- alerts - Work and manage different types of alerts such as CB Analytics Alert, Watchlist Alerts and Device Control Alerts.
- device-control - Control the blocking of USB devices on endpoints.
- differential-analysis - Provides the ability to compare and understand the changes between two Live Query runs
- live-query - Live Query allows operators to ask questions of endpoints
- live-response - Live Response allows security operators to collect information and take action on remote endpoints in real time.
- recommendations - Work with Endpoint Standard recommendations for reputation override.
- reputation-override - Manage reputation overrides for known applications, IT tools or certs.
- unified-binary-store - The unified binary store (UBS) is responsible for storing all binaries and corresponding metadata for those binaries.
- users-grants - Work with users and access grants.
- vulnerabilities - View asset (Endpoint or Workload) vulnerabilities to increase security visibility.
- watchlists-feeds-reports - Work with Enterprise EDR watchlists, feeds, reports, and Indicators of Compromise (IOCs).
- workload - Advanced protection purpose-built for securing modern workloads to reduce the attack surface and strengthen security posture.

4.5.4 Examples

The [GitHub repository](#) also has some example scripts which will help you get started using the SDK.

4.6 Porting Applications from CBAPI to Carbon Black Cloud SDK

This guide will help you migrate from CBAPI to the Carbon Black Cloud Python SDK.

Note: CBAPI applications using Carbon Black EDR (Response) or Carbon Black App Control (Protection) cannot be ported, as support for on-premise products is not present in the CBC SDK. Continue to use CBAPI for these applications.

4.6.1 Overview

CBC SDK has changes to package names, folder structure, and functions. Import statements will need to change for the packages, modules, and functions listed in this guide.

4.6.2 Package Name Changes

A number of packages have new name equivalents in the CBC SDK. Endpoint Standard and Enterprise EDR have had parts replaced to use the most current API routes.

Top-level Package Name Change

The top-level package name has changed from CBAPI to CBC SDK.

CBAPI Name (old)	CBC SDK Name (new)
cbapi.psc	cbc_sdk

Product Name Changes

Carbon Black Cloud product names have been updated in the SDK.

CBAPI Name (old)	CBC SDK Name (new)
cbapi.psc.defense	cbc_sdk.endpoint_standard
cbapi.psc.livequery	cbc_sdk.audit_remediation
cbapi.psc.threathunter	cbc_sdk.enterprise_edr
cbapi.psc	cbc_sdk.platform

Import statements will need to change:

```
# Endpoint Standard (Defense)

# CBAPI
from cbapi.psc.defense import Device, Event, Policy

# CBC SDK
# note that the original "Event" has been decommissioned
from cbc_sdk.endpoint_standard import Device, EnrichedEvent, Policy
```

```
# Audit and Remediation (LiveQuery)

# CBAPI
from cbapi.psc.livequery import Run, RunHistory, Result, DeviceSummary

# CBC SDK
from cbc_sdk.audit_remediation import Run, RunHistory, Result, DeviceSummary
```

```
# Enterprise EDR (ThreatHunter)

# CBAPI
from cbapi.psc.threathunter import Feed, Report, Watchlist

# CBC SDK
from cbc_sdk.enterprise_edr import Feed, Report, Watchlist
```

Moved Packages and Models

Some modules have been moved to a more appropriate location.

CBAPI Name (old)	CBC SDK Name (new)
cbapi.example_helpers	cbc_sdk.helpers
cbapi.psc.alerts_query	cbc_sdk.platform
cbapi.psc.devices_query	cbc_sdk.platform

Import statements will need to change:

```
# Example Helpers

# CBAPI
from cbapi.example_helpers import build_cli_parser

# CBC SDK
from cbc_sdk.helpers import build_cli_parser
```

```
# Alerts

# CBAPI
from cbapi.psc.alerts_query import *

# CBC SDK
from cbc_sdk.platform import *
```

```
# Devices

# CBAPI
from cbapi.psc.devices_query import *

# CBC SDK
from cbc_sdk.platform import *
```

Replaced Modules

With the new Unified Platform Experience, Carbon Black Cloud APIs have been updated to provide a more consistent search experience. Platform search is replacing Endpoint Standard Event searching, and Enterprise EDR Process and Event searching.

For help beyond import statement changes, check out these resources:

- [Unified Platform Experience: What to Expect](#)
- [Migration Guide: Carbon Black Cloud Events API](#)
- [Advanced Search Tips for Carbon Black Cloud Platform Search](#)

Endpoint Standard

Endpoint Standard Events have been replaced with Enriched Events and the old event functionality has been decommissioned.

```
# Endpoint Standard Enriched Events

# CBAPI
from cbapi.psc.defense import Event

# CBC SDK (decommissioned--do not use)
from cbc_sdk.endpoint_standard import Event

# CBC SDK
from cbc_sdk.endpoint_standard import EnrichedEvent
```


Enterprise EDR

Enterprise EDR Processes and Events have been removed and replaced with Platform Processes and Events.

```
# Enterprise EDR Process and Event

# CBAPI
from cbapi.psc.threathunter import Process, Event

# CBC SDK
from cbc_sdk.platform import Process, Event
```

4.6.3 Folder Structure Changes

The directory structure for the SDK has been refined compared to CBAPI.

- Addition of the Platform folder
- Removal of Response and Protection folders
- Consolidation of model objects and query objects
- Product-specific `rest_api.py` files replaced with package level `rest_api.py`
 - from `cbapi.psc.threathunter import CbThreatHunterAPI` becomes `from cbc_sdk import CBCloudAPI, etc.`

Directory Tree Changes

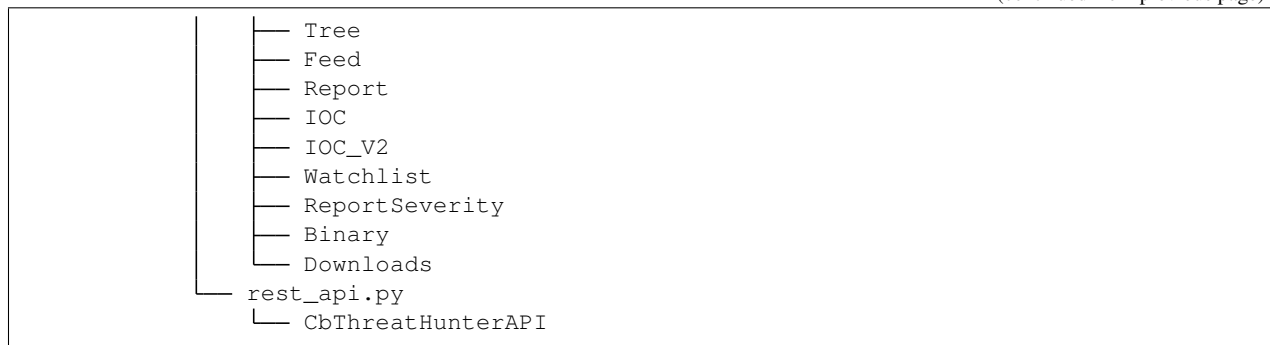
In general, each module's `models.py` and `query.py` files were combined into their respective `base.py` files.

CBAPI had the following abbreviated folder structure:

```
src
├── cbapi
│   └── psc
│       ├── defense
│       │   ├── models.py
│       │   │   ├── Device
│       │   │   ├── Event
│       │   │   └── Policy
│       │   └── rest_api.py
│       │       └── CbDefenseAPI
│       ├── livequery
│       │   ├── models.py
│       │   │   ├── Run
│       │   │   ├── RunHistory
│       │   │   ├── Result
│       │   │   ├── ResultFacet
│       │   │   ├── DeviceSummary
│       │   │   └── DeviceSummaryFacet
│       │   └── rest_api.py
│       │       └── CbLiveQueryAPI
│       └── threathunter
│           ├── models.py
│           │   ├── Process
│           │   └── Event
```

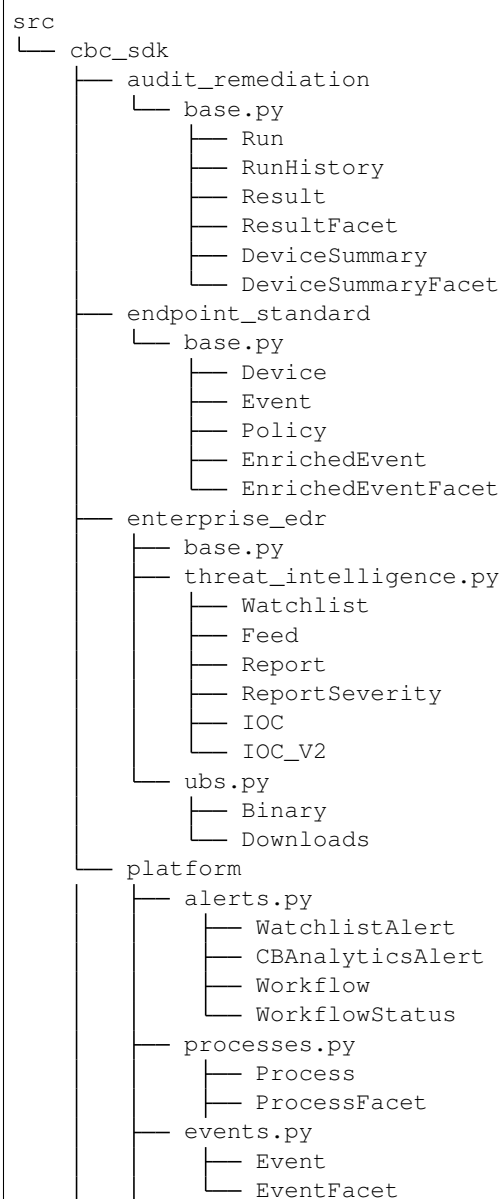
(continues on next page)

(continued from previous page)



Each product had a `models.py` and `rest_api.py` file.

CBC SDK has the following abbreviated folder structure:



(continues on next page)

(continued from previous page)

```

├── devices.py
│   └── Device
├── rest_api.py
│   └── CBCloudAPI.py

```

Now, each product has either a `base.py` file with all of its objects, or categorized files like `platform.alerts.py` and `platform.devices.py`. The package level `rest_api.py` replaced each product-specific `rest_api.py` file.

4.6.4 Function Changes

Helper Functions:

CBAPI Name (old)		CBC SDK Name (new)
<code>cbapi.example_helpers.get_cb_defense_object()</code>	<code>cbapi.</code>	<code>cbc_sdk.</code>
<code>example_helpers.get_cb_livequery_object()</code>	<code>cbapi.</code>	<code>helpers.</code>
<code>example_helpers.get_cb_threathunter_object()</code>	<code>cbapi.</code>	<code>get_cb_cloud_object()</code>
<code>example_helpers.get_cb_psc_object()</code>		

Audit and Remediation Queries:

CBAPI Name (old)	CBC SDK Name (new)
<code>cb.query(sql_query)</code>	<code>cb.select(Run).where(sql=sql_query)</code>
<code>cb.query_history(query_string)</code>	<code>cb.select(RunHistory).where(query_string)</code>
<code>cb.query(sql_query).policy_ids()</code>	<code>cb.select(Run).policy_id()</code>

API Objects:

CBAPI Name (old)	CBC SDK Name (new)
<code>cbapi.psc.defense.CbDefenseAPI</code>	<code>cbapi.psc.livequery.CbLiveQueryAPI</code>
<code>cbapi.psc.threathunter.CbThreatHunterAPI</code>	<code>cbapi.psc.CbPSCBaseAPI</code>
	<code>cbc_sdk.CBCloudAPI</code>

4.7 Logging & Diagnostics

The `cbc_sdk` provides extensive logging facilities to track down issues communicating with the REST API and understand potential performance bottlenecks.

4.7.1 Enabling Logging

The `cbc_sdk` uses Python's standard `logging` module for logging. To enable debug logging for the `cbc_sdk`, you can do the following:

```

>>> import logging
>>> logging.basicConfig(level=logging.DEBUG)

```

All REST API calls, including the API endpoint, any data sent via POST or PUT, and the time it took for the call to complete:

```
>>> devices = [ device for device in cb.select(Device) ]
DEBUG:cbc_sdk.connection:Sending HTTP POST /appservices/v6/orgs/ABCD1234/devices/_
↪search with {"criteria": {}, "exclusions": {}, "query": ""}
DEBUG:urllib3.connectionpool:Starting new HTTPS connection (1): defense-eap01.
↪conferdeploy.net:443
DEBUG:urllib3.connectionpool:https://defense-eap01.conferdeploy.net:443 "POST /
↪appservices/v6/orgs/ABCD1234/devices/_search HTTP/1.1" 200 None
DEBUG:cbc_sdk.connection:HTTP POST /appservices/v6/orgs/ABCD1234/devices/_search took
↪0.409s (response 200)
```

4.8 Testing

This document will provide information about how to run the functional tests for the CBC Python SDK in Linux and Windows platforms.

These instructions assume you already have the CBC SDK sources present locally. If not, they can be checked out from GitHub using the URL <https://github.com/carbonblack/carbon-black-cloud-sdk-python>; doing so will require you to either have Git installed or download the source tree packed as a zip archive from GitHub and then unarchive it.

4.8.1 Running the tests on Microsoft Windows

Install Python

From <http://python.org>, download the installer for the most recent Python 3.8 version (as of this writing, version 3.8.6 is the latest).

Fix the Execution PATH

Go to the Environment Variables dialog (System Control Panel or Properties page for My Computer/This PC, then select **Advanced system settings** and then the **Environment Variables** button). Ensure that the first two components of the user PATH environment variable are `%USERPROFILE%\AppData\Local\Programs\Python\Python38` and `%USERPROFILE%\AppData\Local\Programs\Python\Python38\Scripts`.

To test this, open a command window and use the command: `python --version`

It should run Python and show that you are running Python 3.8.

Install CBC Python SDK Requirements

From the top-level CBC SDK source directory, execute the following commands:

```
pip install -r requirements.txt
```

This will ensure that all required python modules are installed.

Execute the Functional Tests

From the top-level CBC SDK source directory, execute the following command:

```
pytest
```

The tests should return that they all completed successfully.

4.8.2 Running the tests on Linux

Carbon Black Cloud Python SDK provides a number of Dockerfiles inside the docker folder of the source root. Those contain the necessary instructions to build docker images containing a number of distributions with CBC Python SDK preinstalled in /app directory (relative to image root).

Build the docker image

Currently the following Dockerfiles are available:

- docker/amazon/Dockerfile - Amazon Linux (latest) image
- docker/ubuntu/Dockerfile - Ubuntu 18.04 image
- docker/rhel/Dockerfile - RHEL8 UBI image
- docker/suse/Dockerfile - OpenSUSE Leap (latest) image

Building the images should be done from the CBC SDK root directory by explicitly providing the path to the Dockerfile to be built, e.g for the RHEL one, the build command would be:

```
docker build -t cbc-sdk-python=rhel -f docker/rhel/Dockerfile .
```

By default, the docker Unix socket is owned by root user / docker group. In case you are running the build as a non-root user that isn't member of docker group, sudo should be used:

```
sudo docker build -t cbc-sdk-python=rhel -f docker/rhel/Dockerfile .
```

Run the container and execute the test

When the docker image builds, it should be started, e.g:

```
docker run -it cbc-sdk-python=rhel
```

This will run the container and spawn an interactive shell running in it. CBC Python SDK is installed in the /app directory, so pytest needs to be executed from there:

```
cd /app && pytest
```

4.9 Changelog

4.9.1 CBC SDK 1.4.1 - Released October 21, 2022

New Features:

- AWS workloads now supported in VM Workloads Search.
- Live Query Differential Analysis functionality.

Updates:

- VM Workloads Search updated to use new v2 APIs
- Added the `alertable` field to feeds.
- Devices API now supports faceting on three additional (public cloud related) fields.
- Added a user acceptance test script for the policy function updates.

Documentation:

- Added information on OAuth authentication to docs.

4.9.2 CBC SDK 1.4.0 - Released July 26,2022

Breaking Changes:

- `Policy` object has been moved from `cbc_sdk.endpoint_standard` to `cbc_sdk.platform`, as it now uses the new Policy Services API rather than the old APIs through Integration Services.
 - **N.B.:** This change means that you *must* use a custom API key with permissions under `org.policies` to manage policies, rather than an older “API key.”
 - To enable time to update integration logic, the `cbc_sdk.endpoint_standard` `Policy` object may still be imported from the old package, and supports operations that are backwards-compatible with the old one.
 - When developing a new integration, or updating an existing one `cbc_sdk.platform` should be used. There is a utility class `PolicyBuilder`, and as features are added to the Carbon Black Cloud, they will be added to this module.
- Official support for Python 3.6 has been dropped, since that version is now end-of-life. Added explicit testing support for Python versions 3.9 and 3.10. **N.B.:** End users should update their Python version to 3.7.x or greater.

New Features:

- Credentials handler now supports OAuth tokens.
- Added support for querying a single `Report` from a `Feed`.
- Added support for alert notes (create, delete, get, refresh).

Updates:

- Removed the (unused) `revoked` property from `Grant` objects.
- Increased the asynchronous query thread pool to 3 threads by default.
- Required version of `lxml` is now 4.9.1.
- Added a user acceptance test script for Alerts.

Bug Fixes:

- Added `max_rows` to USB device query, fixing pagination.
- Fixed an off-by-one error in Alerts Search resulting un duplicate alerts showing up in results.
- Fixed an error in alert faceting operations due to sending excess input to the server.

Documentation:

- Watchlists, Feeds, and Reports guide has been updated with additional clarification and examples.
- Updated description for some `Device` fields that are never populated.
- Additional sensor states added to `Device` documentation.

- Fixed the description of `BaseAlertSearchQuery.set_types` so that it mentions all valid alert types.
- Threat intelligence example has been deprecated.

4.9.3 CBC SDK 1.3.6 - Released April 19, 2022

New Features:

- Support for Device Facet API.
- Dynamic reference of query classes—now you can do `api.select("Device")` in addition to `api.select(Device)`.
- Support for Container Runtime Alerts.
- NSX Remediation functionality - set the NSX remediation state for workloads which support it.

Updates:

- Endpoint Standard specific `Event s` have been decommissioned and removed.
- SDK now uses Watchlist Manager apis `v3` instead of `v2`. `v2` APIs are being decommissioned.

Documentation:

- Added a `CONTRIBUTING` link to the `README.md` file.
- Change to Watchlist/Report documentation to properly reflect how to update a `Report` in a Watchlist.
- Cleaned up formatting.

4.9.4 CBC SDK 1.3.5 - Released January 26, 2022

New Features:

- Added asynchronous query support to Live Query.
- Added the ability to export query results from Live Query, either synchronously or asynchronously (via the `Job` object and the Jobs API). Synchronous exports include full-file export, line-by-line export, and ZIP file export. Asynchronous exports include full-file export and line-by-line export.
- Added a `CredentialProvider` that uses AWS Secrets Manager to store credential information.

Updates:

- Added `WatchlistAlert.get_process()` method to return the `Process` of a `WatchlistAlert`.
- Added several helpers to Live Query support to make it easier to get runs from a template, or results, device summaries, or facets from a run.
- Optimized API requests when performing query slicing.
- Updated pretty-printing of objects containing `dict` members.
- `lxml` dependency updated to version 4.6.5.

Bug Fixes:

- `User.delete()` now checks for an outstanding access grant on the user, and deletes it first if it exists.
- Fixed handling of URL when attaching a new IOC to a `Feed`.
- Getting and setting of `Report` ignore status is now supported even if that `Report` is part of a `Feed`.

Documentation:

- Information added about the target audience for the SDK.
- Improper reference to a credential property replaced in the Authentication guide.
- Broken example updated in Authentication guide.
- Added SDK guides for Vulnerabilities and Live Query APIs.
- Updated documentation for `ProcessFacet` model to better indicate support for full query string.

4.9.5 CBC SDK 1.3.4 - Released October 12, 2021

New Features:

- New `CredentialProvider` supporting Keychain storage of credentials (Mac OS only).
- Recommendations API - suggested reputation overrides for policy configuration.

Updates:

- Improved string representation of objects through `__str__()` mechanism.

Bug Fixes:

- Ensure proper `TimeoutError` is raised in several places where the wrong exception was being raised.
- Fix to allowed categories when performing alert queries.

Documentation Changes:

- Added guide page for alerts.
- Live Response documentation updated to note use of custom API keys.
- Clarified query examples in Concepts.
- Note that vulnerability assessment has been moved from `workload` to `platform`.
- Small typo fixes in watchlists, feeds, UBS, and reports guide.

4.9.6 CBC SDK 1.3.3 - Released August 10, 2021

Bug Fixes:

- Dependency fix on schema library.

4.9.7 CBC SDK 1.3.2 - Released August 10, 2021

New Features:

- Added asynchronous query options to Live Response APIs.
- Added functionality for Watchlists, Reports, and Feeds to simplify developer interaction.

Updates:

- Added documentation on the mapping between permissions and Live Response commands.

Bug Fixes:

- Fixed an error using the STIX/TAXII example with Cabby.
- Fixed a potential infinite loop in getting detailed search results for enriched events and processes.

- Comparison now case-insensitive on UBS download.

4.9.8 CBC SDK 1.3.1 - Released June 15, 2021

New Features:

- Allow the SDK to accept a pre-configured `Session` object to be used for access, to get around unusual configuration requirements.

Bug Fixes:

- Fix functions in `Grant` object for adding a new access profile to a user access grant.

4.9.9 CBC SDK 1.3.0 - Released June 8, 2021

New Features

- Add User Management, Grants, Access Profiles, Permitted Roles
- Move Vulnerability models to Platform package in preparation for supporting Endpoints and Workloads
- Refactor Vulnerability models
 - `VulnerabilitySummary.get_org_vulnerability_summary` static function changed to `Vulnerability.OrgSummary` model with query class
 - `VulnerabilitySummary` model moved inside `Vulnerability` to `Vulnerability.AssetView` sub model
 - `OrganizationalVulnerability` and `Vulnerability` consolidated into a single model to include Carbon Black Cloud context and CVE information together
 - `Vulnerability(cb, CVE_ID)` returns Carbon Black Cloud context and CVE information
 - `DeviceVulnerability.get_vulnerability_summary_per_device` static function moved to `get_vulnerability_summary` function on `Device` model
 - `affected_assets(os_product_id)` function changed to `get_affected_assets()` function and no longer requires `os_product_id`
- Add dashboard export examples
- Live Response migrated from v3 to v6 (migration guide)
 - Live Response uses API Keys of type Custom
- Add function to get Enriched Events for Alert

Bug Fixes

- Fix validate query from dropping `sort_by` for `Query` class
- Fix the ability to set expiration for binary download URL
- Fix bug in helpers `read_iocs` functionality
- Fix `install_sensor` and `bulk_install` on `ComputeResource` to use `id` instead of `uuid`
- Fix `DeviceSearchQuery` from duplicating `Device` due to base index of 1

4.9.10 CBC SDK 1.2.3 - Released April 19, 2021

Bug Fixes

- Prevent alert query from retrieving past 10k limit

4.9.11 CBC SDK 1.2.3 - Released April 19, 2021

Bug Fixes

- Prevent alert query from retrieving past 10k limit

4.9.12 CBC SDK 1.2.2 - Released April 5, 2021

Bug Fixes

- Add support for full credential property loading through BaseAPI constructor

4.9.13 CBC SDK 1.2.1 - Released March 31, 2021

New Features

- Add `__str__` functions for `Process.Tree` and `Process.Summary`
- Add `get_details` for `Process`
- Add `set_max_rows` to `DeviceQuery`

Bug Fixes

- Modify base class for `EnrichedEventQuery` to `Query` from `cbc_sdk.base` to support entire feature set for searching
- Document fixes for changelog and Workload
- Fix `_spawn_new_workers` to correctly find active devices for Carbon Black Cloud

4.9.14 CBC SDK 1.2.0 - Released March 9, 2021

New Features

- VMware Carbon Black Cloud Workload support for managing workloads:
 - Vulnerability Assessment
 - Sensor Lifecycle Management
 - VM Workloads Search
- Add tutorial for Reputation Override

Bug Fixes

- Fix to initialization of `ReputationOverride` objects

4.9.15 CBC SDK 1.1.1 - Released February 2, 2021

New Features

- Add easy way to add single approvals and blocks
- Add Device Control Alerts
- Add deployment_type support to the Device model

Bug Fixes

- Fix error when updating iocs in a Report model
- Set max_retries to None to use Connection init logic for retries

4.9.16 CBC SDK 1.1.0 - Released January 27, 2021

New Features

- Reputation Overrides for Endpoint Standard with Enterprise EDR support coming soon
- Device Control for Endpoint Standard
- Live Query Templates/Scheduled Runs and Template History
- Add set_time_range for Alert query

Bug Fixes

- Refactored code base to reduce query inheritance complexity
- Limit Live Query results to 10k cap to prevent 400 Bad Request
- Add missing criteria for Live Query RunHistory to search on template ids
- Add missing args.orgkey to get_cb_cloud_object to prevent exception from being thrown
- Refactor add and update criteria to use CriteriaBuilderSupportMixin

4.9.17 CBC SDK 1.0.1 - Released December 17, 2020

Bug Fixes

- Fix readme links
- Few ReadTheDocs fixes

4.9.18 CBC SDK 1.0.0 - Released December 16, 2020

New Features

- Enriched Event searches for Endpoint Standard
- Aggregation search added for Enriched Event Query
- Add support for fetching additional details for an Enriched Event
- Facet query support for Enriched Events, Processes, and Process Events
- Addition of Python Futures to support asynchronous calls for customers who want to leverage that feature , while continuing to also provide the simplified experience which hides the multiple calls required.

- Added translation support for MISP threat intel to cbc_sdk threat intel example

Updates

- Improved information and extra calls for Audit and Remediation (Live Query)
- Great test coverage – create extensions and submit PRs with confidence
- Process and Process Event searches updated to latest APIs and moved to platform package
- Flake8 formatting applied to all areas of the code
- Converted old docstrings to use google format docstrings
- Migrated STIX/TAXII Threat Intel module from cbapi to cbc_sdk examples

Bug Fixes

- Fixed off by one error for process event pagination
- Added support for default profile using CBCloudAPI()
- Retry limit to Process Event search to prevent infinite loop

See detailed information on the objects and methods exposed by the Carbon Black Cloud Python SDK [here](#).

5.1 Audit and Remediation

5.1.1 Submodules

5.1.2 `cbc_sdk.audit_remediation.base` module

Model and Query Classes for Audit and Remediation

class `DeviceSummary` (*cb*, *initial_data*)

Bases: `cbc_sdk.base.UnrefreshableModel`

Represents the summary of results from a single device during a single Audit and Remediation *Run*.

Parameters

- **id** – The result’s unique ID
- **total_results** – Number of results returned for this particular device
- **device** – Information associated with the device
- **time_received** – The time at which this result was received
- **status** – The result’s status
- **device_message** – Placeholder
- **metrics** – Metrics associated with the device

Initialize a `DeviceSummary` object with *initial_data*.

Parameters

- **cb** (`BaseAPI`) – Reference to API object used to communicate with the server.

- **initial_data** (*dict*) – Initial data used to populate the result.

class Metrics (*cb, initial_data*)

Bases: *cbc_sdk.base.UnrefreshableModel*

Represents the metrics for a result.

Initialize a DeviceSummary Metrics object with initial_data.

Parameters

- **cb** (*BaseAPI*) – Reference to API object used to communicate with the server.
- **initial_data** (*dict*) – Initial data used to populate the result.

device = {}

device_message = None

id = None

metrics = []

metrics_

Returns the reified *DeviceSummary.Metrics* for this result.

primary_key = 'device_id'

status = None

time_received = None

total_results = None

urlobject = '/livequery/v1/orgs/{}/runs/{}/results/device_summaries/_search'

class DeviceSummaryFacet (*cb, initial_data*)

Bases: *cbc_sdk.audit_remediation.base.ResultFacet*

Represents the summary of results for a single device summary in an Audit and Remediation *Run*.

Initialize a DeviceSummaryFacet object with initial_data.

Parameters

- **cb** (*BaseAPI*) – Reference to API object used to communicate with the server.
- **initial_data** (*dict*) – Initial data used to populate the result.

urlobject = '/livequery/v1/orgs/{}/runs/{}/results/device_summaries/_facet'

class FacetQuery (*doc_class, cb*)

Bases: *cbc_sdk.base.BaseQuery*, *cbc_sdk.base.QueryBuilderSupportMixin*, *cbc_sdk.base.IterableQueryMixin*, *cbc_sdk.base.CriteriaBuilderSupportMixin*, *cbc_sdk.base.AsyncQueryMixin*

Represents a query that receives facet information from a LiveQuery run.

Initialize the FacetQuery.

Parameters

- **doc_class** (*class*) – The model class that will be returned by this query.
- **cb** (*BaseAPI*) – Reference to API object used to communicate with the server.

facet_field (*field*)

Sets the facet fields to be received by this query.

Parameters `field(str or [str])` – Field(s) to be received.

Returns FacetQuery that will receive field(s) facet_field.

Return type *FacetQuery*

Example

```
>>> cb.select(ResultFacet).run_id(my_run).facet_field(["device.policy_name",  
↪ "device.os"])
```

run_id(run_id)

Sets the run ID to query results for.

Parameters `run_id(int)` – The run ID to retrieve results for.

Returns FacetQuery object with specified run_id.

Return type *FacetQuery*

Example

```
>>> cb.select(ResultFacet).run_id(my_run)
```

set_device_ids(device_ids)

Sets the device.id criteria filter.

Parameters `device_ids([int])` – Device IDs to filter on.

Returns The FacetQuery with specified device.id.

Return type *FacetQuery*

set_device_names(device_names)

Sets the device.name criteria filter.

Parameters `device_names([str])` – Device names to filter on.

Returns The FacetQuery with specified device.name.

Return type *FacetQuery*

set_device_os(device_os)

Sets the device.os criteria.

Parameters `device_os([str])` – Device OS's to filter on.

Returns The FacetQuery object with specified device_os.

Return type *FacetQuery*

Note: Device OS's can be one or more of ["WINDOWS", "MAC", "LINUX"].

set_policy_ids(policy_ids)

Sets the device.policy_id criteria.

Parameters `policy_ids([int])` – Device policy ID's to filter on.

Returns The FacetQuery object with specified policy_ids.

Return type *FacetQuery*

set_policy_names (*policy_names*)

Sets the device.policy_name criteria.

Parameters **policy_names** (*[str]*) – Device policy names to filter on.

Returns The FacetQuery object with specified policy_names.

Return type *FacetQuery*

set_statuses (*statuses*)

Sets the status criteria.

Parameters **statuses** (*[str]*) – Query statuses to filter on.

Returns The FacetQuery object with specified statuses.

Return type *FacetQuery*

MAX_RESULTS_LIMIT = 10000

Audit and Remediation Models

class Result (*cb, initial_data*)

Bases: *cbc_sdk.base.UnrefreshableModel*

Represents a single result from an Audit and Remediation *Run*.

Parameters

- **id** – The result's unique ID
- **device** – The device associated with the result
- **status** – The result's status
- **time_received** – The time at which this result was received
- **device_message** – Placeholder
- **fields** – The fields returned by the backing osquery query
- **metrics** – Metrics associated with the result's host

Initialize a Result object with initial_data.

Device, Fields, and Metrics objects are attached using initial_data.

Parameters

- **cb** (*BaseAPI*) – Reference to API object used to communicate with the server.
- **initial_data** (*dict*) – Initial data used to populate the result.

class Device (*cb, initial_data*)

Bases: *cbc_sdk.base.UnrefreshableModel*

Represents device information for a result.

Initialize a Device Result object with initial_data.

Parameters

- **cb** (*BaseAPI*) – Reference to API object used to communicate with the server.
- **initial_data** (*dict*) – Initial data used to populate the result.

primary_key = 'id'

class Fields (*cb, initial_data*)

Bases: *cbc_sdk.base.UnrefreshableModel*

Represents the fields of a result.

Initialize a Result Fields object with *initial_data*.

Parameters

- **cb** (*BaseAPI*) – Reference to API object used to communicate with the server.
- **initial_data** (*dict*) – Initial data used to populate the result.

class Metrics (*cb, initial_data*)

Bases: *cbc_sdk.base.UnrefreshableModel*

Represents the metrics of a result.

Initialize a Result Metrics object with *initial_data*.

Parameters

- **cb** (*BaseAPI*) – Reference to API object used to communicate with the server.
- **initial_data** (*dict*) – Initial data used to populate the result.

device = {}

device_

Returns the reified *Result.Device* for this result.

device_message = None

fields = {}

fields_

Returns the reified *Result.Fields* for this result.

id = None

metrics = {}

metrics_

Returns the reified *Result.Metrics* for this result.

primary_key = 'id'

query_device_summaries ()

Returns a ResultQuery for a DeviceSummary.

This represents the search for a summary of results from a single device of a *Run*. The query may be further augmented with additional criteria prior to enumerating its results.

Returns The query object returned by this operation.

Return type *ResultQuery*

query_device_summary_facets ()

Returns a ResultQuery for a DeviceSummaryFacet.

This represents the search for a summary of a single device summary of a *Run*. The query may be further augmented with additional criteria prior to enumerating its results.

Returns The query object returned by this operation.

Return type *ResultQuery*

query_result_facets()

Returns a ResultQuery for a ResultFacet.

This represents the search for a summary of results from a single field of a *Run*. The query may be further augmented with additional criteria prior to enumerating its results.

Returns The query object returned by this operation.

Return type *ResultQuery*

status = None

time_received = None

urlobject = '/livequery/v1/orgs/{}/runs/{}/results/_search'

class ResultFacet (*cb, initial_data*)

Bases: *cbc_sdk.base.UnrefreshableModel*

Represents the summary of results for a single field in an Audit and Remediation *Run*.

Parameters **field** – The name of the field being summarized

Initialize a ResultFacet object with initial_data.

Parameters

- **cb** (*BaseAPI*) – Reference to API object used to communicate with the server.
- **initial_data** (*dict*) – Initial data used to populate the result.

class Values (*cb, initial_data*)

Bases: *cbc_sdk.base.UnrefreshableModel*

Represents the values associated with a field.

Initialize a ResultFacet Values object with initial_data.

Parameters

- **cb** (*BaseAPI*) – Reference to API object used to communicate with the server.
- **initial_data** (*dict*) – Initial data used to populate the result.

field = None

primary_key = 'field'

urlobject = '/livequery/v1/orgs/{}/runs/{}/results/_facet'

values = []

values_

Returns the reified *ResultFacet.Values* for this result.

class ResultQuery (*doc_class, cb*)

Bases: *cbc_sdk.base.BaseQuery*, *cbc_sdk.base.QueryBuilderSupportMixin*, *cbc_sdk.base.IterableQueryMixin*, *cbc_sdk.base.CriteriaBuilderSupportMixin*, *cbc_sdk.base.AsyncQueryMixin*

Represents a query that retrieves results from a LiveQuery run.

Initialize the ResultQuery.

Parameters

- **doc_class** (*class*) – The model class that will be returned by this query.
- **cb** (*BaseAPI*) – Reference to API object used to communicate with the server.

async_export()

Create an asynchronous job that exports the results from the run.

This is recommended if you are expecting a very large result set. Once the Job is created, wait for it to be completed, then get the results from the Job using one of the `get_output` methods on the `cbc_sdk.platform.jobs()` object. To wait asynchronously for the results, use the Job object's `await_completion()` method.

Required Permissions: `livequery.manage(READ)`, `jobs.status(READ)`

Returns The Job object that represents the asynchronous job.

Return type `Job`

export_csv_as_file(filename)

Export the results from the run as CSV, writing the CSV to the named file.

Required Permissions: `livequery.manage(READ)`

Parameters `filename (str)` – Name of the file to write the results to.

export_csv_as_lines()

Export the results from the run as CSV, returning the CSV data as iterated lines.

Required Permissions: `livequery.manage(READ)`

Returns An iterable that can be used to get each line of CSV text in turn as a string.

Return type `iterable`

export_csv_as_stream(output, compressed=False)

Export the results from the run as CSV, writing the CSV to the given stream.

Required Permissions: `livequery.manage(READ)`

Parameters

- **output** (`RawIOBase`) – Stream to write the CSV data from the request to.
- **compressed** (`bool`) – True to download as a compressed ZIP file, False to download as CSV.

export_csv_as_string()

Export the results from the run as CSV, returning the CSV data as a string.

Required Permissions: `livequery.manage(READ)`

Returns The CSV data as one big string.

Return type `str`

export_zipped_csv(filename)

Export the results from the run as a zipped CSV, writing the zip data to the named file.

Required Permissions: `livequery.manage(READ)`

Parameters `filename (str)` – Name of the file to write the results to.

run_id(run_id)

Sets the run ID to query results for.

Parameters `run_id (int)` – The run ID to retrieve results for.

Returns ResultQuery object with specified run_id.

Return type *ResultQuery*

Example

```
>>> cb.select(Result).run_id(my_run)
```

set_device_ids (device_ids)

Sets the device.id criteria filter.

Parameters `device_ids ([int])` – Device IDs to filter on.

Returns The ResultQuery with specified device.id.

Return type *ResultQuery*

set_device_names (device_names)

Sets the device.name criteria filter.

Parameters `device_names ([str])` – Device names to filter on.

Returns The ResultQuery with specified device.name.

Return type *ResultQuery*

set_device_os (device_os)

Sets the device.os criteria.

Parameters `device_os ([str])` – Device OS's to filter on.

Returns The ResultQuery object with specified device_os.

Return type *ResultQuery*

Note: Device OS's can be one or more of ["WINDOWS", "MAC", "LINUX"].

set_policy_ids (policy_ids)

Sets the device.policy_id criteria.

Parameters `policy_ids ([int])` – Device policy ID's to filter on.

Returns The ResultQuery object with specified policy_ids.

Return type *ResultQuery*

set_policy_names (policy_names)

Sets the device.policy_name criteria.

Parameters `policy_names ([str])` – Device policy names to filter on.

Returns The ResultQuery object with specified policy_names.

Return type *ResultQuery*

set_statuses (statuses)

Sets the status criteria.

Parameters `statuses ([str])` – Query statuses to filter on.

Returns The ResultQuery object with specified statuses.

Return type *ResultQuery*

sort_by (*key*, *direction*='ASC')

Sets the sorting behavior on a query's results.

Parameters

- **key** (*str*) – The key in the schema to sort by.
- **direction** (*str*) – The sort order, either “ASC” or “DESC”.

Returns ResultQuery object with specified sorting key and order.

Return type *ResultQuery*

Example

```
>>> cb.select(Result).run_id(my_run).where(username="foobar").sort_by("uid")
```

class Run (*cb*, *model_unique_id*=None, *initial_data*=None)

Bases: *cbc_sdk.base.NewBaseModel*

Represents an Audit and Remediation run.

Example:

```
>>> run = cb.select(Run, run_id)
>>> print(run.name, run.sql, run.create_time)
>>> print(run.status, run.match_count)
>>> run.refresh()
```

Parameters

- **org_key** – The organization key for this run
- **name** – The name of the Audit and Remediation run
- **id** – The run's unique ID
- **sql** – The Audit and Remediation query
- **created_by** – The user or API id that created the run
- **create_time** – When this run was created
- **status_update_time** – When the status of this run was last updated
- **timeout_time** – The time at which the query will stop requesting results from any devices who have not responded
- **cancellation_time** – The time at which a user or API id cancelled the run
- **cancelled_by** – The user or API id that cancelled the run
- **notify_on_finish** – Whether or not to send an email on query completion
- **active_org_devices** – The number of devices active in the organization
- **status** – The run status
- **device_filter** – Any device filter rules associated with the run
- **last_result_time** – When the most recent result for this run was reported
- **total_results** – The number of results received

- **match_count** – The number of devices which received a match to the query
- **no_match_count** – The number of devices which did not received a match to the query
- **error_count** – The number of devices which errored
- **not_supported_count** – The number of devices which do not support a portion of the osquery
- **cancelled_count** – The number of devices which were cancelled before they ran the query
- **not_started_count** – The number of devices which have not run the query
- **success_count** – The number of devices which succeeded in running the query
- **in_progress_count** – The number of devices which were currently executing the query
- **recommended_query_id** – The id of a query from the recommendedation route
- **template_id** – The template that created the run

Initialize a Run object with initial_data.

Required Permissions: livequery.manage(READ)

Parameters

- **cb** ([BaseAPI](#)) – Reference to API object used to communicate with the server.
- **model_unique_id** (*str*) – ID of the query run represented.
- **initial_data** (*dict*) – Initial data used to populate the query run.

```
active_org_devices = None
```

```
cancellation_time = None
```

```
cancelled_by = None
```

```
cancelled_count = None
```

```
create_time = None
```

```
created_by = None
```

```
delete()
```

Delete a query.

Required Permissions: livequery.manage(DELETE)

Returns True if the query was deleted successfully, False otherwise.

Return type bool

```
device_filter = {}
```

```
error_count = None
```

```
id = None
```

```
in_progress_count = None
```

```
last_result_time = None
```

```
match_count = None
```

`name = None`

`no_match_count = None`

`not_started_count = None`

`not_supported_count = None`

`notify_on_finish = None`

`org_key = None`

`primary_key = 'id'`

`query_device_summaries()`

Create a DeviceSummary query that searches for all device summaries on this run.

The query may be further augmented with additional criteria prior to enumerating its results.

Returns A query object which will search for all device summaries for this run.

Return type *ResultQuery*

Raises `ApiError` – If the query has been deleted.

`query_facets()`

Create a ResultFacet query that searches for all result facets on this run.

The query may be further augmented with additional criteria prior to enumerating its results.

Returns A query object which will search for all result facets for this run.

Return type *FacetQuery*

Raises `ApiError` – If the query has been deleted.

`query_results()`

Create a Result query that searches for all results on this run.

The query may be further augmented with additional criteria prior to enumerating its results.

Returns A query object which will search for all results for this run.

Return type *ResultQuery*

Raises `ApiError` – If the query has been deleted.

`recommended_query_id = None`

`schedule = {}`

`sql = None`

`status = None`

`status_update_time = None`

`stop()`

Stop a running query.

Required Permissions: `livequery.manage(UPDATE)`

Returns True if query was stopped successfully, False otherwise.

Return type `bool`

Raises `ServerError` – If the server response cannot be parsed as JSON.

`success_count = None`

```
template_id = None
timeout_time = None
total_results = None
urlobject = '/livequery/v1/orgs/{}/runs'
urlobject_single = '/livequery/v1/orgs/{}/runs/{}'
```

```
class RunHistory(cb, initial_data=None)
    Bases: cbc_sdk.audit_remediation.base.Run
```

Represents a historical Audit and Remediation *Run*.

Initialize a RunHistory object with initial_data.

Parameters

- **cb** (*BaseAPI*) – Reference to API object used to communicate with the server.
- **initial_data** (*dict*) – Initial data used to populate the history object.

```
urlobject_history = '/livequery/v1/orgs/{}/runs/_search'
```

```
class RunHistoryQuery(doc_class, cb)
    Bases: cbc_sdk.base.BaseQuery, cbc_sdk.base.QueryBuilderSupportMixin, cbc_sdk.
    base.IterableQueryMixin, cbc_sdk.base.CriteriaBuilderSupportMixin, cbc_sdk.
    base.AsyncQueryMixin
```

Represents a query that retrieves historic LiveQuery runs.

Initialize the RunHistoryQuery.

Parameters

- **doc_class** (*class*) – The model class that will be returned by this query.
- **cb** (*BaseAPI*) – Reference to API object used to communicate with the server.

```
set_template_ids(template_ids)
```

Sets the template_id criteria filter.

Parameters **template_ids** (*[str]*) – Template IDs to filter on.

Returns The RunHistoryQuery with specified template_id.

Return type *RunHistoryQuery*

```
sort_by(key, direction='ASC')
```

Sets the sorting behavior on a query's results.

Parameters

- **key** (*str*) – The key in the schema to sort by.
- **direction** (*str*) – The sort order, either “ASC” or “DESC”.

Returns RunHistoryQuery object with specified sorting key and order.

Return type *RunHistoryQuery*

Example:

```
>>> cb.select(Result).run_id(my_run).where(username="foobar").sort_by("uid")
```


class RunQuery (*doc_class, cb*)

Bases: *cbc_sdk.base.BaseQuery, cbc_sdk.base.AsyncQueryMixin*

Represents a query that either creates or retrieves the status of a LiveQuery run.

Initialize the RunQuery.

Parameters

- **doc_class** (*class*) – The model class that will be returned by this query.
- **cb** (*BaseAPI*) – Reference to API object used to communicate with the server.

device_ids (*device_ids*)

Restricts the devices that this Audit and Remediation run is performed on to the given IDs.

Parameters **device_ids** (*[int]*) – Device IDs to perform the Run on.

Returns The RunQuery with specified device_ids.

Return type *RunQuery*

device_types (*device_types*)

Restricts the devices that this Audit and Remediation run is performed on to the given OS.

Parameters **device_types** (*[str]*) – Device types to perform the Run on.

Returns The RunQuery object with specified device_types.

Return type *RunQuery*

Note: Device type can be one of [“WINDOWS”, “MAC”, “LINUX”].

name (*name*)

Sets this Audit and Remediation run’s name.

If no name is explicitly set, the run is named after its SQL.

Parameters **name** (*str*) – The name for this Run.

Returns The RunQuery object with specified name.

Return type *RunQuery*

notify_on_finish ()

Sets the notify-on-finish flag on this Audit and Remediation run.

Returns The RunQuery object with *notify_on_finish* set to True.

Return type *RunQuery*

policy_id (*policy_id*)

Restricts this Audit and Remediation run to the given policy ID.

Parameters **policy_id** (*int*) or (*list[int]*) – Policy ID to perform the Run on.

Returns The RunQuery object with specified policy_id.

Return type *RunQuery*

schedule (*rrule, timezone*)

Sets a schedule for the SQL Query to recur

A schedule requires an rrule and a timezone to determine the time to rerun the SQL query. rrule is defined in RFC 2445 however only a subset of the functionality is supported here. If a Run is created with a

schedule then the Run will contain a `template_id` to the corresponding template and a new Run will be created each time the schedule is met.

Example RRule, Daily

Field	Values
BYSECOND	0
BYMINUTE	0 or 30
BYHOUR	0 to 23

Daily at 1:30PM

RRULE:FREQ=DAILY;BYHOUR=13;BYMINUTE=30;BYSECOND=0

Example RRule, Weekly

Field	Values
BYSECOND	0
BYMINUTE	0
BYHOUR	0 to 23
BYDAY	One or more: SU, MO, TU, WE, TH, FR, SA

Monday and Friday of the week at 2:30 AM

RRULE:FREQ=WEEKLY;BYDAY=MO,FR;BYHOUR=13;BYMINUTE=30;BYSECOND=0

Example RRule, Monthly

Note: Either (BYDAY and BYSETPOS) or BYMONTHDAY is required.

Field	Values
BYSECOND	0
BYMINUTE	0 or 30
BYHOUR	0 to 23
BYDAY	One or more: SU, MO, TU, WE, TH, FR, SA
BYSETPOS	-1, 1, 2, 3, 4
BYMONTHDAY	One or more: 1 to 28

Last Monday of the Month at 2:30 AM

RRULE:FREQ=MONTHLY;BYDAY=MO;BYSETPOS=-1;BYHOUR=2;BYMINUTE=30;BYSECOND=0

1st and 15th of the Month at 2:30 AM

RRULE:FREQ=DAILY;BYMONTHDAY=1,15;BYHOUR=2;BYMINUTE=30;BYSECOND=0

Parameters

- **rrule** (*string*) – A recurrence rule (RFC 2445) specifying the frequency and time at which the query will recur
- **timezone** (*string*) – The timezone database name to use as a base for the rule

Returns The RunQuery with a recurrence schedule.

Return type *RunQuery*

submit()

Submits this Audit and Remediation run.

Returns A new *Run* instance containing the run's status.

Return type *Run*

Raises *ApiError* – If the Run does not have SQL set, or if the Run has already been submitted.

where(sql)

Sets this Audit and Remediation run's underlying SQL.

Parameters *sql* (*str*) – The SQL to execute for the Run.

Returns The RunQuery object with specified sql.

Return type *RunQuery*

class Template (*cb*, *model_unique_id=None*, *initial_data=None*)

Bases: *cbc_sdk.audit_remediation.base.Run*

Represents an Audit and Remediation Live Query Template.

Example:

```
>>> template = cb.select(Template, template_id)
>>> print(template.name, template.sql, template.create_time)
>>> print(template.status, template.match_count, template.schedule)
>>> template.refresh()
```

Parameters

- **org_key** – The organization key for this run
- **name** – The name of the Audit and Remediation run
- **id** – The run's unique ID
- **sql** – The Audit and Remediation query
- **created_by** – The user or API id that created the run
- **create_time** – When this run was created
- **status_update_time** – When the status of this run was last updated
- **timeout_time** – The time at which the query will stop requesting results from any devices who have not responded
- **cancellation_time** – The time at which a user or API id cancelled the run
- **cancelled_by** – The user or API id that cancelled the run
- **archive_time** – The time at which a user or API id cancelled the run
- **archived_by** – The user or API id that archived the run
- **notify_on_finish** – Whether or not to send an email on query completion
- **active_org_devices** – The number of devices active in the organization
- **status** – The run status
- **device_filter** – Any device filter rules associated with the run
- **last_result_time** – When the most recent result for this run was reported
- **total_results** – The number of results received

- **match_count** – The number of devices which received a match to the query
- **no_match_count** – The number of devices which did not received a match to the query
- **error_count** – The number of devices which errored
- **not_supported_count** – The number of devices which do not support a portion of the osquery
- **cancelled_count** – The number of devices which were cancelled before they ran the query
- **not_started_count** – The number of devices which have not run the query
- **success_count** – The number of devices which succeeded in running the query
- **in_progress_count** – The number of devices which were currently executing the query
- **recommended_query_id** – The id of a query from the recommendedation route
- **template_id** – The template that created the run

Initialize a Template object with `initial_data`.

Required Permissions: `livequery.manage(READ)`

Parameters

- **cb** (`BaseAPI`) – Reference to API object used to communicate with the server.
- **model_unique_id** (`str`) – ID of the query run represented.
- **initial_data** (`dict`) – Initial data used to populate the query run.

```
active_org_devices = None
archive_time = None
archived_by = None
cancellation_time = None
cancelled_by = None
cancelled_count = None
create_time = None
created_by = None
device_filter = {}
error_count = None
id = None
in_progress_count = None
last_result_time = None
match_count = None
name = None
no_match_count = None
not_started_count = None
```

`not_supported_count = None`

`notify_on_finish = None`

`org_key = None`

`primary_key = 'id'`

`query_runs()`

Create a RunHistory query that searches for all runs created by this template ID.

The query may be further augmented with additional criteria prior to enumerating its results.

Returns A query object which will search for all runs based on this template.

Return type *RunHistoryQuery*

`recommended_query_id = None`

`schedule = {}`

`sql = None`

`status = None`

`status_update_time = None`

`stop()`

Stop a template.

Required Permissions: `livequery.manage(UPDATE)`

Returns True if query was stopped successfully, False otherwise.

Return type `bool`

Raises `ServerError` – If the server response cannot be parsed as JSON.

`success_count = None`

`template_id = None`

`timeout_time = None`

`total_results = None`

`urlobject = '/livequery/v1/orgs/{}/templates'`

`urlobject_single = '/livequery/v1/orgs/{}/templates/{}'`

`class TemplateHistory(cb, initial_data=None)`

Bases: *cbc_sdk.audit_remediation.base.Template*

Represents a historical Audit and Remediation *Template*.

Initialize a Template object with `initial_data`.

Required Permissions: `livequery.manage(READ)`

Parameters

- **cb** (*BaseAPI*) – Reference to API object used to communicate with the server.
- **initial_data** (*dict*) – Initial data used to populate the query run.

`urlobject_history = '/livequery/v1/orgs/{}/templates/_search'`

class TemplateHistoryQuery (*doc_class, cb*)

Bases: *cbc_sdk.base.BaseQuery, cbc_sdk.base.QueryBuilderSupportMixin, cbc_sdk.base.IterableQueryMixin, cbc_sdk.base.CriteriaBuilderSupportMixin, cbc_sdk.base.AsyncQueryMixin*

Represents a query that retrieves historic LiveQuery templates.

Initialize the TemplateHistoryQuery.

Parameters

- **doc_class** (*class*) – The model class that will be returned by this query.
- **cb** (*BaseAPI*) – Reference to API object used to communicate with the server.

sort_by (*key, direction='ASC'*)

Sets the sorting behavior on a query's results.

Parameters

- **key** (*str*) – The key in the schema to sort by.
- **direction** (*str*) – The sort order, either “ASC” or “DESC”.

Returns object with specified sorting key and order.

Return type *TemplateHistoryQuery*

Example:

```
>>> cb.select(Result).run_id(my_run).where(username="foobar").sort_by("uid")
```

5.1.3 cbc_sdk.audit_remediation.differential module

Model and Query Classes for Differential Analysis

ASYNC_RATE_LIMIT = 100

Differential Analysis Models

class Differential (*cb, initial_data=None*)

Bases: *cbc_sdk.base.NewBaseModel*

Represents a Differential Analysis run.

Example:

```
>>> query = cb.select(Differential).newer_run_id(newer_run_id)
>>> run = query.submit()
>>> print(run)
>>> print(run.diff_results)
```

Parameters

- **newer_run_id** – id against which the older run id results will be compared
- **newer_run_create_time** – Timestamp of the primary run in ISO 8601 UTC format
- **older_run_id** – This can be optional. If not specified, the previous run as compared to the primary will be chosen. This can be optional if you are comparing recurring runs only.
- **older_run_create_time** – Timestamp of the older run in ISO 8601 UTC format

- **diff_processed_time** – The time it took to process the results in seconds and milliseconds
- **newer_run_not_responded_devices** – Array of device IDs that have not responded
- **older_run_not_responded_devices** – Array of device IDs that have not responded
- **diff_results** – An object containing either count of changes only or count and actual diff results

Initialize a Differential object with initial_data.

Required Permissions for CBC: livequery.manage(READ)

Required Permissions for CSP: _API.Live.Query:livequery.Manage.read

Parameters

- **cb** ([BaseAPI](#)) – Reference to API object used to communicate with the server.
- **initial_data** (*dict*) – Initial data used to populate the query run.

```
diff_processed_time = None
diff_results = {}
newer_run_create_time = None
newer_run_id = None
newer_run_not_responded_devices = []
older_run_create_time = None
older_run_id = None
older_run_not_responded_devices = []
urlobject = '/livequery/v1/orgs/{}/differential/runs/_search'
```

```
class DifferentialQuery(doc_class, cb)
    Bases: cbc\_sdk.base.BaseQuery, cbc\_sdk.base.IterableQueryMixin, cbc\_sdk.base.CriteriaBuilderSupportMixin
```

Query used to compare two Live Query runs.

Initialize the DifferentialQuery.

Parameters

- **doc_class** (*class*) – The model class that will be returned by this query.
- **cb** ([BaseAPI](#)) – Reference to API object used to communicate with the server.

async_export ()

Create an asynchronous job that exports the results from the run.

This is recommended if you are expecting a very large result set. Once the Job is created, wait for it to be completed, then get the results from the Job using one of the `get_output` methods on the `cbc_sdk.platform.jobs` object. To wait for the results, use the Job object's `await_completion()` method.

Example

```
>>> # Get the differential
>>> query = cb.select(Differential).newer_run_id(newer_run_id)
>>> export = query.async_export()
>>> # wait for the export to finish
>>> export.await_completion()
>>> # write the results to a file
>>> export.get_output_as_file("example_data.json")
```

Required CBC Permissions: livequery.manage(READ), jobs.status(READ)

Required CSP Permissions: _API.Live.Query:livequery.Manage.read, _API.Background_Tasks.jobs.status.read

Returns The Job object that represents the asynchronous job.

Return type *Job*

count_only (*count_only*)

Return only count of diff results per device or complete diff metadata result.

The default value is true, which means only the count will be returned.

Example

```
>>> query = cb.select(Differential).newer_run_id(newer_run_id).count_
↳ only(True)
>>> run = query.submit()
```

Parameters **count_only** (*string*) – Boolean that indicates whether to return actual metadata or return just the count of differences

Returns This instance.

Return type *DifferentialQuery*

Raises *ApiError* – If invalid values are passed in the list.

newer_run_id (*newer_run_id*)

Set the id against which the older_run_id results will be compared.

Example

```
>>> query = cb.select(Differential).newer_run_id(newer_run_id)
>>> run = query.submit()
```

Parameters **newer_run_id** (*string*) – id against which the older_run_id results will be compared.

Returns This instance.

Return type *DifferentialQuery*

Raises *ApiError* – If invalid values are passed.

older_run_id (*older_run_id*)

This can be optional.

If not specified, the previous run as compared to the primary will be chosen if it is a recurring one. If comparing two individual runs, this is required.

Example

```
>>> query = cb.select(Differential).newer_run_id(newer_run_id).older_run_
↳id(older_run_id)
>>> run = query.submit()
```

Parameters **older_run_id** (*string*) – id against which the newer_run_id results will be compared.

Returns This instance.

Return type *DifferentialQuery*

Raises `ApiError` – If invalid values are passed.

set_device_ids (*device_ids*)

Restricts the query on to the specified devices only.

Example

```
>>> query = cb.select(Differential).newer_run_id(newer_run_id).set_device_
↳ids([12345, 56789])
>>> run = query.submit()
```

Parameters **device_ids** (*list*) – List of device id(s)

Returns This instance.

Return type *DifferentialQuery*

Raises `ApiError` – If invalid values are passed in the list.

submit ()

Submits this Differential Analysis run.

Returns A new *Differential* instance containing the run's content.

Return type *Run*

5.1.4 Module contents

5.2 Credential Providers

5.2.1 Submodules

5.2.2 cbc_sdk.credential_providers.default module

Function which gives us the default credentials handler for use by CBCloudAPI.

class DefaultProvider

Bases: `object`

Intermediate class defined to allow insertion of a “test point” into `default_credential_provider()`.

get_default_provider (*credential_file*)

Return the default credential provider that CBCloudAPI should use.

Parameters **credential_file** (*str*) – Credential file as specified to the initialization of the API.

Returns The default credential provider that CBCloudAPI should use.

Return type *CredentialProvider*

default_credential_provider (*credential_file*)

Return the default credential provider that CBCloudAPI should use.

Parameters **credential_file** (*str*) – Credential file as specified to the initialization of the API.

Returns The default credential provider that CBCloudAPI should use.

Return type *CredentialProvider*

5.2.3 `cbc_sdk.credential_providers.envron_credential_provider` module

Credentials provider that reads the credentials from the environment.

class EnvironCredentialProvider

Bases: `cbc_sdk.credentials.CredentialProvider`

The object which provides credentials based on variables in the environment.

Initializes the EnvironCredentialProvider.

get_credentials (*section=None*)

Return a Credentials object containing the configured credentials.

Parameters **section** (*str*) – The credential section to retrieve (not used in this provider).

Returns The credentials retrieved from that source.

Return type *Credentials*

Raises `CredentialError` – If there is any error retrieving the credentials.

5.2.4 `cbc_sdk.credential_providers.file_credential_provider` module

Credentials provider that reads the credentials from a file.

class FileCredentialProvider (*credential_file=None*)

Bases: `cbc_sdk.credentials.CredentialProvider`

The object which provides credentials based on a credential file.

Initialize the FileCredentialProvider.

Parameters **credential_file** (*object*) – A string or path-like object representing the credentials file, or a list of strings or path-like objects representing the search path for the credentials file.

get_credentials (*section=None*)

Return a Credentials object containing the configured credentials.

Parameters **section** (*str*) – The credential section to retrieve.

Returns The credentials retrieved from that source.

Return type *Credentials*

Raises *CredentialError* – If there is any error retrieving the credentials.

5.2.5 cbc_sdk.credential_providers.keychain_credential_provider module

Credentials provider that reads the credentials from the macOS's keychain.

class **KeychainCredentialProvider** (*keychain_name, keychain_username*)

Bases: *cbc_sdk.credentials.CredentialProvider*

This credential provider reads from the macOS's Keychain.

Initialize the KeychainCredentialProvider.

Parameters

- **keychain_name** (*str*) – The name of the entry in the Keychain.
- **keychain_username** (*str*) – The username which you've set in the Keychain.

Raises *CredentialError* – If we attempt to instantiate this provider on a non-macOS system.

get_credentials (*section=None*)

Return a Credentials object containing the configured credentials.

Parameters

- **section** (*None*) – Since Keychain doesn't support sections it is left
- **satisfy the Signature of CredentialProvider** (*to*) –

Returns The credentials retrieved from that source.

Return type *Credentials*

Raises *CredentialError* – If there is any error retrieving the credentials.

5.2.6 cbc_sdk.credential_providers.registry_credential_provider module

Credentials provider that reads the credentials from the environment.

OpenKey (*base, path*)

Stub to maintain source compatibility

QueryValueEx (*key, name*)

Stub to maintain source compatibility

class **RegistryCredentialProvider** (*keypath=None, userkey=True*)

Bases: *cbc_sdk.credentials.CredentialProvider*

The credentials provider that reads from the Windows Registry.

Initialize the RegistryCredentialProvider.

Parameters

- **keypath** (*str*) – Path from the selected base key to the key that will contain individual sections.
- **userkey** (*bool*) – True if the keypath starts at HKEY_CURRENT_USER, False if at HKEY_LOCAL_MACHINE.

Raises `CredentialError` – If we attempt to instantiate this provider on a non-Windows system.

get_credentials (*section=None*)

Return a `Credentials` object containing the configured credentials.

Parameters **section** (*str*) – The credential section to retrieve.

Returns The credentials retrieved from that source.

Return type *Credentials*

Raises `CredentialError` – If there is any error retrieving the credentials.

5.2.7 Module contents

5.3 Developing New Credential Providers

The credentials management framework for the CBC SDK is designed to allow different handlers to be implemented, which may supply credentials to the `CBCloudAPI` in ways not implemented by existing credential handlers.

5.3.1 Writing the Credential Provider

Find all classes required to implement a new credential provider in the `cbc_sdk.credentials` package. See below for descriptions of the classes. It is recommended, but not required, that your new credential provider inherit from the `CredentialProvider` abstract class, and that you implement the methods from that abstract class as detailed.

The arguments to the standard `__init__()` method are not defined by the interface specification; those may be used to initialize your credential provider in any desired fashion.

5.3.2 Using the Credential Provider

Create an instance of your credential provider object and pass it as the keyword parameter `credential_provider` when creating your `CBCloudAPI` object.

Example:

```
>>> provider = MyCredentialProvider()
>>> cbc_api = CBCloudAPI(credential_provider=provider, profile='default')
```

Your credential provider's `get_credentials()` method will be called, passing in any profile specified in the `profile` keyword parameter used when creating `CBCloudAPI`.

5.3.3 Credential Provider Reference

These are the classes from the `cbc_sdk.credentials` package that are used in making a credential provider.

CredentialValue class

This class is of an enumerated type, and represents the various credential items loaded by the credential provider and fed to the rest of the SDK code. The possible values are:

- URL - The URL used to access the Carbon Black Cloud. This value *must* be specified.
- TOKEN - The access token to be used to authenticate to the server. It is the same structure as the X-Auth-Token: defined for direct API access in [the developer documentation](#). This value *must* be specified.
- ORG_KEY - The organization key specifying which organization to work with. This value *must* be specified.
- SSL_VERIFY - A Boolean value indicating whether or not to validate the SSL connection. The default is True.
- SSL_VERIFY_HOSTNAME - A Boolean value indicating whether or not to verify the host name of the server being connected to. The default is True.
- SSL_CERT_FILE - The name of an optional certificate file used to validate the certificates of the SSL connection. If not specified, the standard system certificate verification will be used.
- SSL_FORCE_TLS_1_2 - A Boolean value. If this is True, the connection will be forced to use TLS 1.2 rather than any later version. The default is False.
- PROXY - If specified, this is the name of a proxy host to be used in making the connection.
- IGNORE_SYSTEM_PROXY - A Boolean value. If this is True, any system proxy settings will be ignored in making the connection to the server. The default is False.
- INTEGRATION - The name of the integration to use these credentials. The string may optionally end with a slash character, followed by the integration's version number. Passed as part of the User-Agent: HTTP header on all requests made by the SDK.

Values of this type have one method:

requires_boolean_value

```
def requires_boolean_value(self):
```

Returns whether or not this particular credential item takes a Boolean value.

Returns: True if the credential item takes a Boolean value, False if the credential item takes a string value.

Credentials class

The class that holds credentials retrieved from the credential provider, and is used by the rest of the SDK. It is effectively immutable after creation.

__init__

```
def __init__(self, values=None):
```

Initializes a new Credentials object.

Parameters:

- values (type dict): A dictionary containing the values to initialize the Credentials object with. The keys of this dictionary may be either CredentialValue objects or their lowercase string equivalents, e.g. CredentialValue.URL or "url". The values in the dict are strings for those credential items with string values. For credential items with Boolean values, the values may be either bool values, numeric values (with 0 being treated as False and non-zero values treated as True), or string values. In the case of string values, the value must be "0", "false", "off", or "no" to be treated as a False value, or "1", "true", "on", or "yes" to

be treated as a `True` value (all values case-insensitive). If an unrecognized string is used for a Boolean value, `CredentialError` will be raised. Unrecognized keys in the dict are ignored. Any missing items will be replaced by the default for that item.

Raises:

- `CredentialError` - If there is an error parsing a Boolean value string.

get_value

```
def get_value(self, key):
```

Retrieves a specific credential value from this object.

Parameters:

- `key` (type `CredentialValue`): Indicates which item to retrieve.

Returns: The value of that credential item (`str` or `bool` type).

__getattr__

```
def __getattr__(self, name):
```

Retrieves a specific credential value from this object. This is a bit of “syntactic sugar” allowing other code to access credential values, for instance, as `cred_object.url` instead of `cred_object.get_value(CredentialValue.URL)`.

Parameters:

- `name` (type `str`): Indicates which item to retrieve.

Returns: The value of that credential item (`str` or `bool` type).

Raises:

- `AttributeError` - If the credential item name was unrecognized.

CredentialProvider class

All credential providers *should* extend this abstract class, but, in any event, *must* implement the protocol it defines.

get_credentials

```
def get_credentials(self, section=None):
```

Return a `Credentials` object containing the configured credentials.

Parameters:

- `section` (type `str`): Indicates the credential section to retrieve. May be interpreted by the credential provider in any manner it likes; may also be ignored.

Returns: A `Credentials` object containing the retrieved credentials.

Raises:

- `CredentialError` - If there is an error retrieving the credentials.

5.4 Endpoint Standard

5.4.1 Decommissioned Functionality

The Endpoint Standard events (`cbc_sdk.endpoint_standard.Event`) have been decommissioned and should no longer be used. Any attempt to use them will raise a `FunctionalityDecommissioned` exception. Please use `cbc_sdk.endpoint_standard.EnrichedEvent` instead. Refer to [this migration guide](#) on the Carbon Black Developer Network Community for more information.

5.4.2 Submodules

5.4.3 `cbc_sdk.endpoint_standard.base` module

Model and Query Classes for Endpoint Standard

```
class EnrichedEvent (cb, model_unique_id=None, initial_data=None, force_init=False,
                    full_doc=True)
```

Bases: `cbc_sdk.base.UnrefreshableModel`

Represents an enriched event retrieved by one of the Enterprise EDR endpoints.

Initialize the `EnrichedEvent` object.

Parameters

- **cb** (`CBCloudAPI`) – A reference to the `CBCloudAPI` object.
- **model_unique_id** (*Any*) – The unique ID for this particular instance of the model object.
- **initial_data** (*dict*) – The data to use when initializing the model object.
- **force_init** (*bool*) – True to force object initialization.
- **full_doc** (*bool*) – True to mark the object as fully initialized.

```
approve_process_sha256 (description="")
```

Approves the application by adding the `process_sha256` to the `WHITE_LIST`

Parameters **description** – The justification for why the application was added to the `WHITE_LIST`

Returns

ReputationOverride object created in the Carbon Black Cloud

Return type `ReputationOverride` (`cbc_sdk.platform.ReputationOverride`)

```
ban_process_sha256 (description="")
```

Bans the application by adding the `process_sha256` to the `BLACK_LIST`

Parameters **description** – The justification for why the application was added to the `BLACK_LIST`

Returns

ReputationOverride object created in the Carbon Black Cloud

Return type `ReputationOverride` (`cbc_sdk.platform.ReputationOverride`)

```
default_sort = 'device_timestamp'
```

get_details (*timeout=0, async_mode=False*)

Requests detailed results.

Parameters

- **timeout** (*int*) – Event details request timeout in milliseconds.
- **async_mode** (*bool*) – True to request details in an asynchronous manner.

Note:

- When using asynchronous mode, this method returns a python future. You can call result() on the future object to wait for completion and get the results.
-

primary_key = 'event_id'

process_sha256

Returns a string representation of the SHA256 hash for this process.

Returns SHA256 hash of the process.

Return type hash (str)

class EnrichedEventFacet (*cb, model_unique_id, initial_data*)

Bases: *cbc_sdk.base.UnrefreshableModel*

Represents an enriched event retrieved by one of the Enterprise EDR endpoints.

Parameters

- **job_id** – The Job ID assigned to this query
- **terms** – Contains the Enriched Event Facet search results
- **ranges** – Groupings for search result properties that are ISO 8601 timestamps or numbers
- **contacted** – The number of searchers contacted for this query
- **completed** – The number of searchers that have reported their results

Initialize the Terms object with initial data.

class Ranges (*cb, initial_data*)

Bases: *cbc_sdk.base.UnrefreshableModel*

Represents the range (bucketed) facet fields and values associated with an Enriched Event Facet query.

Initialize an EnrichedEventFacet Ranges object with initial_data.

facets

Returns the reified *EnrichedEventFacet.Terms._facets* for this result.

fields

Returns the ranges fields for this result.

class Terms (*cb, initial_data*)

Bases: *cbc_sdk.base.UnrefreshableModel*

Represents the facet fields and values associated with an Enriched Event Facet query.

Initialize an EnrichedEventFacet Terms object with initial_data.

facets

Returns the terms' facets for this result.


```

    fields
        Returns the terms facets' fields for this result.

    completed = None
    contacted = None
    job_id = None
    num_found = None
    primary_key = 'job_id'
    ranges = []
    ranges_
        Returns the reified EnrichedEventFacet.Ranges for this result.

    result_url = '/api/investigate/v2/orgs/{}/enriched_events/facet_jobs/{}/results'
    submit_url = '/api/investigate/v2/orgs/{}/enriched_events/facet_jobs'
    terms = {}
    terms_
        Returns the reified EnrichedEventFacet.Terms for this result.

```

class EnrichedEventQuery (*doc_class*, *cb*)

Bases: *cbc_sdk.base.Query*

Represents the query logic for an Enriched Event query.

This class specializes *Query* to handle the particulars of enriched events querying.

Initialize the EnrichedEventQuery object.

Parameters

- **doc_class** (*class*) – The class of the model this query returns.
- **cb** (*CBCCloudAPI*) – A reference to the CBCCloudAPI object.

aggregation (*field*)

Performs an aggregation search where results are grouped by an aggregation field

Parameters **field** (*str*) – The aggregation field, either 'process_sha256' or 'device_id'

or_ (***kwargs*)

or_() criteria are explicitly provided to EnrichedEvent queries.

This method overrides the base class in order to provide *or_()* functionality rather than raising an exception.

set_rows (*rows*)

Sets the 'rows' query body parameter to the 'start search' API call, determining how many rows to request.

Parameters **rows** (*int*) – How many rows to request.

timeout (*msecs*)

Sets the timeout on a event query.

Parameters **msecs** (*int*) – Timeout duration, in milliseconds.

Returns

The Query object with new **milliseconds** parameter.

Return type *Query* (*EnrichedEventQuery*)

Example

```
>>> cb.select(EnrichedEvent).where(process_name="foo.exe").timeout(5000)
```

class Event (*cb, model_unique_id, initial_data=None*)

Bases: object

Represents an Endpoint Standard Event.

This functionality has been decommissioned. Please use EnrichedEvent instead. More information may be found here: <https://community.carbonblack.com/t5/Developer-Relations/Migration-Guide-Carbon-Black-Cloud-Events-API/m-p/95915/thread-id/2519>

This functionality has been decommissioned. Do not use.

Parameters

- **cb** (*BaseAPI*) – Unused.
- **model_unique_id** (*int*) – Unused.
- **initial_data** (*dict*) – Unused.

Raises *FunctionalityDecommissioned* – Always.

info_key = 'eventInfo'

primary_key = 'eventId'

urlobject = '/integrationServices/v3/event'

log = <Logger *cbc_sdk.endpoint_standard.base* (WARNING)>
Endpoint Standard Models

5.4.4 cbc_sdk.endpoint_standard.recommendation module

Model and query APIs for Recommendations

class Recommendation (*cb, model_unique_id, initial_data=None*)

Bases: *cbc_sdk.base.NewBaseModel*

Represents a recommended proposed policy change for the organization.

Parameters

- **changed_by** – Who made the last update to the workflow
- **create_time** – The time the recommendation was created
- **ref_id** – Reference id for an accepted Recommendation which is the id of the created Reputation Override
- **status** – Status of the recommendation
- **update_time** – The last time the recommendation was updated
- **comment** – A comment added when the recommendation was updated

Initialize the Recommendation object.

Parameters

- **cb** (*BaseAPI*) – Reference to API object used to communicate with the server.
- **model_unique_id** (*str*) – ID of the recommendation represented.

- **initial_data** (*dict*) – Initial data used to populate the recommendation.

class RecommendationApplication (*cb, model_unique_id, initial_data=None*)

Bases: *cbc_sdk.base.UnrefreshableModel*

Represents the rule application of a proposed change to an organization's policies.

Parameters

- **type** – Application type
- **value** – Application value

Initialize the RecommendationApplication object.

Parameters

- **cb** (*BaseAPI*) – Reference to API object used to communicate with the server.
- **model_unique_id** (*str*) – Should be None.
- **initial_data** (*dict*) – Initial data used to populate the object.

type = None

value = None

class RecommendationImpact (*cb, model_unique_id, initial_data=None*)

Bases: *cbc_sdk.base.UnrefreshableModel*

Represents metadata about a recommendation to be used in the decision to accept or reject it.

Parameters

- **event_count** – Number of alerts encountered for recommendation
- **impact_score** – Impact score
- **impacted_devices** – Number of devices impacted by the recommendation
- **org_adoption** – Priority for adoption of this recommendation
- **update_time** – The last time this impact was updated

Initialize the RecommendationImpact object.

Parameters

- **cb** (*BaseAPI*) – Reference to API object used to communicate with the server.
- **model_unique_id** (*str*) – Should be None.
- **initial_data** (*dict*) – Initial data used to populate the object.

event_count = None

impact_score = None

impacted_devices = None

org_adoption = None

update_time = None

class RecommendationNewRule (*cb, model_unique_id, initial_data=None*)

Bases: *cbc_sdk.base.UnrefreshableModel*

Represents the proposed change to an organization's policies from a recommendation.

Parameters

- **action** – Rule action
- **application** – Rule application
- **certificate_authority** – Certificate authority
- **filename** – File name
- **include_child_processes** – Include child processes
- **operation** – Operation
- **override_list** – Override list
- **override_type** – Override type
- **path** – File path
- **sha256_hash** – SHA256 hash
- **signed_by** – Signed by

Initialize the RecommendationNewRule object.

Parameters

- **cb** ([BaseAPI](#)) – Reference to API object used to communicate with the server.
- **model_unique_id** (*str*) – Should be None.
- **initial_data** (*dict*) – Initial data used to populate the object.

action = None

application = {}

application_

Return the object representing the rule application of a proposed change to an organization's policies.

Returns The object representing the rule application of a proposed change.

Return type [RecommendationApplication](#)

certificate_authority = None

filename = None

include_child_processes = None

operation = None

override_list = None

override_type = None

path = None

sha256_hash = None

signed_by = None

class RecommendationWorkflow (*cb, model_unique_id, initial_data=None*)

Bases: [cbc_sdk.base.UnrefreshableModel](#)

Represents the lifecycle state of a recommendation.

Parameters

- **changed_by** – Who made the last update to the workflow
- **create_time** – The time the recommendation was created

- **ref_id** – Reference id for an accepted Recommendation which is the id of the created Reputation Override
- **status** – Status of the recommendation
- **update_time** – The last time the recommendation was updated
- **comment** – A comment added when the recommendation was updated

Initialize the RecommendationWorkflow object.

Parameters

- **cb** ([BaseAPI](#)) – Reference to API object used to communicate with the server.
- **model_unique_id** (*str*) – Should be None.
- **initial_data** (*dict*) – Initial data used to populate the object.

changed_by = None

comment = None

create_time = None

ref_id = None

status = None

update_time = None

accept (*comment=None*)

Accept this recommendation, converting it into a reputation override.

Parameters **comment** (*str*) – Optional comment associated with the action.

Returns True if we successfully refreshed this Recommendation's state, False if not.

Return type bool

changed_by = None

comment = None

create_time = None

impact_

Return the object representing metadata about the recommendation.

Returns The object representing metadata about the recommendation.

Return type [RecommendationImpact](#)

new_rule_

Return the object representing the proposed change to an organization's policies from the recommendation.

Returns The object representing the proposed change to an organization's policies.

Return type [RecommendationNewRule](#)

primary_key = 'recommendation_id'

ref_id = None

reject (*comment=None*)

Reject this recommendation.

Parameters **comment** (*str*) – Optional comment associated with the action.

Returns True if we successfully refreshed this Recommendation's state, False if not.

Return type bool

reputation_override()

Returns the reputation override associated with the recommendation (if the recommendation was accepted).

Returns The associated reputation override, or None if there is none.

Return type *ReputationOverride*

reset (*comment=None*)

Reset the recommendation, undoing any created reputation override and setting it back to NEW state.

Parameters **comment** (*str*) – Optional comment associated with the action.

Returns True if we successfully refreshed this Recommendation's state, False if not.

Return type bool

status = None

update_time = None

urlobject = '/recommendation-service/v1/orgs/{0}/recommendation'

urlobject_single = '/recommendation-service/v1/orgs/{0}/recommendation/{1}'

workflow_

Returns the object representing the lifecycle state of the recommendation.

Returns The object representing the lifecycle state of the recommendation.

Return type *RecommendationWorkflow*

class RecommendationQuery (*doc_class, cb*)

Bases: *cbc_sdk.base.BaseQuery*, *cbc_sdk.base.CriteriaBuilderSupportMixin*, *cbc_sdk.base.IterableQueryMixin*, *cbc_sdk.base.AsyncQueryMixin*

Query used to locate Recommendation objects.

Initialize the RecommendationQuery.

Parameters

- **doc_class** (*class*) – The model class that will be returned by this query.
- **cb** (*BaseAPI*) – Reference to API object used to communicate with the server.

VALID_POLICY_TYPES = ['reputation_override', 'sensor_policy']

VALID_STATUSES = ['NEW', 'REJECTED', 'ACCEPTED']

set_hashes (*hashes*)

Restricts the recommendations that this query is performed on to the specified hashes.

Parameters **hashes** (*list*) – List of hashes to restrict the search to.

Returns This instance.

Return type *RecommendationQuery*

Raises *ApiError* – If invalid values are passed in the list.

set_policy_types (*policy_types*)

Restricts the recommendations that this query is performed on to the specified policy types.

Parameters **policy_types** (*list*) – List of policy types to restrict the search to.

Returns This instance.

Return type *RecommendationQuery*

Raises `ApiError` – If invalid values are passed in the list.

set_statuses (*statuses*)

Restricts the recommendations that this query is performed on to the specified status values.

Parameters **statuses** (*list*) – List of status values to restrict the search to. If no statuses are specified, the search defaults to NEW only.

Returns This instance.

Return type *RecommendationQuery*

Raises `ApiError` – If invalid values are passed in the list.

sort_by (*key*, *direction*='ASC')

Sets the sorting behavior on a query's results.

Example

```
>>> cb.select(USBDevice).sort_by("product_name")
```

Parameters

- **key** (*str*) – The key in the schema to sort by.
- **direction** (*str*) – The sort order, either “ASC” or “DESC”.

Returns This instance.

Return type *USBDeviceQuery*

log = <Logger `cbc_sdk.endpoint_standard.recommendation` (WARNING)>
Recommendation models

5.4.5 cbc_sdk.endpoint_standard.usb_device_control module

Model and Query Classes for USB Device Control

class `USBDevice` (*cb*, *model_unique_id*, *initial_data*=None)

Bases: `cbc_sdk.base.NewBaseModel`

Represents a USB device.

Parameters

- **created_at** – the UTC date the external USB device configuration was created in ISO 8601 format
- **device_friendly_name** – human readable name for the external USB device
- **device_name** – name of the external USB device
- **device_type** – type of external USB device
- **endpoint_count** – number of endpoints that the external USB device has connected to
- **first_seen** – first timestamp that the external USB device was seen
- **id** – the id for this external USB device
- **interface_type** – type of interface used by external USB device

- **last_endpoint_id** – ID of the last endpoint the device accessed
- **last_endpoint_name** – name of the last endpoint the device accessed
- **last_policy_id** – ID of the last policy associated with the device
- **last_seen** – last timestamp that the external USB device was seen
- **org_key** – unique org key of the organization that the external USB device was connected to
- **product_id** – product ID of the external USB device in decimal form
- **product_name** – product name of the external USB device
- **serial_number** – serial number of external device
- **status** – Calculated status of device
- **updated_at** – the UTC date the external USB device configuration was updated in ISO 8601 format
- **vendor_id** – ID of the Vendor for the external USB device in decimal form
- **vendor_name** – vendor name of the external USB device

Initialize the USBDevice object.

Parameters

- **cb** ([BaseAPI](#)) – Reference to API object used to communicate with the server.
- **model_unique_id** (*str*) – ID of the alert represented.
- **initial_data** (*dict*) – Initial data used to populate the alert.

approve (*approval_name, notes*)

Creates and saves an approval for this USB device, allowing it to be treated as approved from now on.

Parameters

- **approval_name** (*str*) – The name for this new approval.
- **notes** (*str*) – Notes to be added to this approval.

Returns The new approval.

Return type [USBDeviceApproval](#)

created_at = None

device_friendly_name = None

device_name = None

device_type = None

endpoint_count = None

first_seen = None

get_endpoints ()

Returns the information about endpoints associated with this USB device.

Returns List of information about USB endpoints, each item specified as a dict.

Return type list

classmethod get_vendors_and_products_seen (*cb*)

Returns all vendors and products that have been seen for the organization.

Parameters `cb` ([BaseAPI](#)) – Reference to API object used to communicate with the server.

Returns A list of vendors and products seen for the organization, each vendor being represented by a dict.

Return type list

```
id = None
interface_type = None
last_endpoint_id = None
last_endpoint_name = None
last_policy_id = None
last_seen = None
org_key = None
primary_key = 'id'
product_id = None
product_name = None
serial_number = None
status = None
updated_at = None
urlobject = '/device_control/v3/orgs/{0}/devices'
urlobject_single = '/device_control/v3/orgs/{0}/devices/{1}'
vendor_id = None
vendor_name = None
```

class `USBDeviceApproval` (*cb, model_unique_id, initial_data=None*)

Bases: [cbc_sdk.base.MutableBaseModel](#)

Represents a USB device approval.

Parameters

- **approval_name** – the name of the approval
- **created_at** – the UTC date the approval was created in ISO 8601 format
- **id** – the id for this approval
- **notes** – the notes for the approval
- **product_id** – product ID of the approval’s external USB device in hex form
- **product_name** – product name of the approval’s external USB device
- **serial_number** – serial number of the approval’s external device
- **updated_at** – the UTC date the approval was updated in ISO 8601 format
- **updated_by** – the user who updated the record last
- **vendor_id** – ID of the Vendor for the approval’s external USB device in hex form
- **vendor_name** – vendor name of the approval’s external USB device

Initialize the `USBDeviceApproval` object.

Parameters

- **cb** ([BaseAPI](#)) – Reference to API object used to communicate with the server.
- **model_unique_id** (*str*) – ID of the alert represented.
- **initial_data** (*dict*) – Initial data used to populate the alert.

approval_name = None

classmethod bulk_create (*cb, approvals*)

Creates multiple approvals and returns the USBDeviceApproval objects. Data is supplied as a list of dicts.

Parameters

- **cb** ([BaseAPI](#)) – Reference to API object used to communicate with the server.
- **approvals** (*list*) – List of dicts containing approval data to be created, formatted as shown below.

Example

```
>>> [
    {
        "approval_name": "string",
        "notes": "string",
        "product_id": "string",
        "serial_number": "string",
        "vendor_id": "string"
    }
]
```

Returns A list of USBDeviceApproval objects representing the approvals that were created.

Return type list

classmethod bulk_create_csv (*cb, approval_data*)

Creates multiple approvals and returns the USBDeviceApproval objects. Data is supplied as text in CSV format.

Parameters

- **cb** ([BaseAPI](#)) – Reference to API object used to communicate with the server.
- **approval_data** (*str*) – CSV data for the approvals to be created. Header line **MUST** be included as shown below.

Example

vendor_id,product_id,serial_number,approval_name,notes

string,string,string,string,string

Returns A list of USBDeviceApproval objects representing the approvals that were created.

Return type list

classmethod create_from_usb_device (*usb_device*)

Creates a new, unsaved approval object from a USBDeviceObject, filling in its basic fields.

Parameters **usb_device** ([USBDevice](#)) – The USB device to create the approval from.

Returns The new approval object.

Return type *USBDeviceApproval*

```

created_at = None
id = None
notes = None
primary_key = 'id'
product_id = None
product_name = None
serial_number = None
updated_at = None
updated_by = None
urlobject = '/device_control/v3/orgs/{0}/approvals'
urlobject_single = '/device_control/v3/orgs/{0}/approvals/{1}'
vendor_id = None
vendor_name = None

```

class *USBDeviceApprovalQuery* (*doc_class*, *cb*)

Bases: *cbc_sdk.base.BaseQuery*, *cbc_sdk.base.QueryBuilderSupportMixin*, *cbc_sdk.base.CriteriaBuilderSupportMixin*, *cbc_sdk.base.IterableQueryMixin*, *cbc_sdk.base.AsyncQueryMixin*

Represents a query that is used to locate USBDeviceApproval objects.

Initialize the USBDeviceApprovalQuery.

Parameters

- **doc_class** (*class*) – The model class that will be returned by this query.
- **cb** (*BaseAPI*) – Reference to API object used to communicate with the server.

set_device_ids (*device_ids*)

Restricts the device approvals that this query is performed on to the specified device IDs.

Parameters **device_ids** (*list*) – List of string device IDs.

Returns This instance.

Return type *USBDeviceApprovalQuery*

set_product_names (*product_names*)

Restricts the device approvals that this query is performed on to the specified product names.

Parameters **product_names** (*list*) – List of string product names.

Returns This instance.

Return type *USBDeviceApprovalQuery*

set_vendor_names (*vendor_names*)

Restricts the device approvals that this query is performed on to the specified vendor names.

Parameters **vendor_names** (*list*) – List of string vendor names.

Returns This instance.

Return type *USBDeviceApprovalQuery*

class `USBDeviceBlock` (*cb*, *model_unique_id*, *initial_data=None*)

Bases: *cbc_sdk.base.NewBaseModel*

Represents a USB device block.

Parameters

- **created_at** – the UTC date the block was created in ISO 8601 format
- **id** – the id for this block
- **policy_id** – policy id which is blocked
- **updated_at** – the UTC date the block was updated in ISO 8601 format

Initialize the USBDeviceBlock object.

Parameters

- **cb** (*BaseAPI*) – Reference to API object used to communicate with the server.
- **model_unique_id** (*str*) – ID of the alert represented.
- **initial_data** (*dict*) – Initial data used to populate the alert.

classmethod `bulk_create` (*cb*, *policy_ids*)

Creates multiple blocks and returns the USBDeviceBlocks that were created.

Parameters

- **cb** (*BaseAPI*) – Reference to API object used to communicate with the server.
- **policy_ids** (*list*) – List of policy IDs to have blocks created for.

Returns A list of USBDeviceBlock objects representing the approvals that were created.

Return type *list*

classmethod `create` (*cb*, *policy_id*)

Creates a USBDeviceBlock for a given policy ID.

Parameters

- **cb** (*BaseAPI*) – Reference to API object used to communicate with the server.
- **policy_id** (*str/int*) – Policy ID to create a USBDeviceBlock for.

Returns New USBDeviceBlock object representing the block.

Return type *USBDeviceBlock*

`created_at = None`

`delete()`

Delete this object.

`id = None`

`policy_id = None`

`primary_key = 'id'`

`updated_at = None`

`urlobject = '/device_control/v3/orgs/{0}/blocks'`

`urlobject_single = '/device_control/v3/orgs/{0}/blocks/{1}'`

class USBDeviceBlockQuery (*doc_class, cb*)

Bases: *cbc_sdk.base.BaseQuery, cbc_sdk.base.IterableQueryMixin, cbc_sdk.base.AsyncQueryMixin*

Represents a query that is used to locate USBDeviceBlock objects.

Initialize the USBDeviceBlockQuery.

Parameters

- **doc_class** (*class*) – The model class that will be returned by this query.
- **cb** (*BaseAPI*) – Reference to API object used to communicate with the server.

class USBDeviceQuery (*doc_class, cb*)

Bases: *cbc_sdk.base.BaseQuery, cbc_sdk.base.QueryBuilderSupportMixin, cbc_sdk.base.CriteriaBuilderSupportMixin, cbc_sdk.base.IterableQueryMixin, cbc_sdk.base.AsyncQueryMixin*

Represents a query that is used to locate USBDevice objects.

Initialize the USBDeviceQuery.

Parameters

- **doc_class** (*class*) – The model class that will be returned by this query.
- **cb** (*BaseAPI*) – Reference to API object used to communicate with the server.

VALID_FACET_FIELDS = ['vendor_name', 'product_name', 'endpoint.endpoint_name', 'status']

VALID_STATUSES = ['APPROVED', 'UNAPPROVED']

facets (*fieldlist, max_rows=0*)

Return information about the facets for all known USB devices, using the defined criteria.

Parameters

- **fieldlist** (*list*) – List of facet field names. Valid names are “vendor_name”, “product_name”, “endpoint.endpoint_name”, and “status”.
- **max_rows** (*int*) – The maximum number of rows to return. 0 means return all rows.

Returns A list of facet information specified as dicts.

Return type list

set_endpoint_names (*endpoint_names*)

Restricts the devices that this query is performed on to the specified endpoint names.

Parameters **endpoint_names** (*list*) – List of string endpoint names.

Returns This instance.

Return type *USBDeviceQuery*

set_max_rows (*max_rows*)

Sets the max number of usb devices to fetch in a singular query

Parameters **max_rows** (*integer*) – Max number of usb devices

Returns This instance.

Return type *USBDeviceQuery*

Raises *ApiError* – If rows is negative or greater than 10000

set_product_names (*product_names*)

Restricts the devices that this query is performed on to the specified product names.

Parameters **product_names** (*list*) – List of string product names.

Returns This instance.

Return type *USBDeviceQuery*

set_serial_numbers (*serial_numbers*)

Restricts the devices that this query is performed on to the specified serial numbers.

Parameters **serial_numbers** (*list*) – List of string serial numbers.

Returns This instance.

Return type *USBDeviceQuery*

set_statuses (*statuses*)

Restricts the devices that this query is performed on to the specified status values.

Parameters **statuses** (*list*) – List of string status values. Valid values are APPROVED and UNAPPROVED.

Returns This instance.

Return type *USBDeviceQuery*

set_vendor_names (*vendor_names*)

Restricts the devices that this query is performed on to the specified vendor names.

Parameters **vendor_names** (*list*) – List of string vendor names.

Returns This instance.

Return type *USBDeviceQuery*

sort_by (*key, direction='ASC'*)

Sets the sorting behavior on a query's results.

Example

```
>>> cb.select(USBDevice).sort_by("product_name")
```

Parameters

- **key** (*str*) – The key in the schema to sort by.
- **direction** (*str*) – The sort order, either “ASC” or “DESC”.

Returns This instance.

Return type *USBDeviceQuery*

log = <Logger cbc_sdk.endpoint_standard.usb_device_control (WARNING)>
USB Device Control models

5.4.6 Module contents

5.5 Enterprise EDR

5.5.1 Submodules

5.5.2 cbc_sdk.enterprise_edr.auth_events module

Model and Query Classes for Auth Events

class AuthEvent (*cb*, *model_unique_id=None*, *initial_data=None*, *force_init=False*, *full_doc=False*)

Bases: *cbc_sdk.base.NewBaseModel*

Represents an AuthEvent

Initialize the AuthEvent object.

Required RBAC Permissions: org.search.events (CREATE, READ)

Parameters

- **cb** (*CBCloudAPI*) – A reference to the CBCloudAPI object.
- **model_unique_id** (*Any*) – The unique ID for this particular instance of the model object.
- **initial_data** (*dict*) – The data to use when initializing the model object.
- **force_init** (*bool*) – True to force object initialization.
- **full_doc** (*bool*) – False to mark the object as not fully initialized.

Example

```
>>> cb = CBCloudAPI(profile="example_profile")
>>> events = cb.select(AuthEvent).where("auth_username:SYSTEM")
>>> print(*events)
```

auth_domain_name = None

auth_event_action = None

auth_remote_device = None

auth_remote_port = None

auth_username = None

backend_timestamp = None

static bulk_get_details (*cb*, *alert_id=None*, *event_ids=None*, *timeout=0*)

Bulk get details

Parameters

- **cb** (*CBCloudAPI*) – A reference to the CBCloudAPI object.
- **alert_id** (*str*) – An alert id to fetch associated events
- **event_ids** (*list*) – A list of event ids to fetch
- **timeout** (*int*) – AuthEvent details request timeout in milliseconds.

Returns list of Auth Events

Return type list

Example

```
>>> cb = CBCloudAPI(profile="example_profile")
>>> bulk_details = AuthEvent.bulk_get_details(cb, event_ids=['example-value'])
>>> print(bulk_details)
```

Raises `ApiError` – if cb is not instance of `CBCloudAPI`

`childproc_count = None`

`crossproc_count = None`

`device_group_id = None`

`device_id = None`

`device_name = None`

`device_policy_id = None`

`device_timestamp = None`

`event_id = None`

`filemod_count = None`

static `get_auth_events_descriptions(cb)`

Returns descriptions and status messages of Auth Events.

Parameters `cb` (`CBCloudAPI`) – A reference to the `CBCloudAPI` object.

Returns Descriptions and status messages of Auth Events as dict objects.

Return type dict

Raises `ApiError` – if cb is not instance of `CBCloudAPI`

Example

```
>>> cb = CBCloudAPI(profile="example_profile")
>>> descriptions = AuthEvent.get_auth_events_descriptions(cb)
>>> print(descriptions)
```

get_details (`timeout=0, async_mode=False`)

Requests detailed results.

Parameters

- **timeout** (`int`) – AuthEvent details request timeout in milliseconds.
- **async_mode** (`bool`) – True to request details in an asynchronous manner.

Returns Auth Events object enriched with the details fields

Return type `AuthEvent`

Note:

- When using asynchronous mode, this method returns a python future. You can call result() on the future object to wait for completion and get the results.

Examples

```
>>> cb = CBCloudAPI(profile="example_profile")
```

```
>>> events = cb.select(AuthEvent).where(process_pid=2000)
>>> print(events[0].get_details())
```

```
ingress_time = None
modload_count = None
netconn_count = None
org_id = None
parent_guid = None
parent_pid = None
primary_key = 'event_id'
process_guid = None
process_hash = []
process_name = None
process_pid = []
process_username = []
regmod_count = None
scriptload_count = None
```

static search_suggestions (*cb, query, count=None*)

Returns suggestions for keys and field values that can be used in a search.

Parameters

- **cb** (*CBCloudAPI*) – A reference to the CBCloudAPI object.
- **query** (*str*) – A search query to use.
- **count** (*int*) – (optional) Number of suggestions to be returned

Returns A list of search suggestions expressed as dict objects.

Return type list

Raises *ApiError* – if cb is not instance of CBCloudAPI

Example

```
>>> cb = CBCloudAPI(profile="example_profile")
>>> suggestions = AuthEvent.search_suggestions(cb, 'auth')
>>> print(suggestions)
```

```
validation_url = '/api/investigate/v2/orgs/{}/auth_events/search_validation'
```

```
windows_event_id = None
```

```
class AuthEventFacet (cb, model_unique_id, initial_data)
```

Bases: `cbc_sdk.base.UnrefreshableModel`

Represents an AuthEvent facet retrieved.

Example:

```
>>> cb = CBCloudAPI (profile="example_profile")
>>> events_facet = cb.select (AuthEventFacet) .where ("auth_
↳username:SYSTEM") .add_facet_field ("process_name")
>>> print (events_facet.results)
```

Parameters

- **terms** – Contains the Auth Event Facet search results
- **ranges** – Groupings for search result properties that are ISO 8601 timestamps or numbers
- **contacted** – The number of searchers contacted for this query
- **completed** – The number of searchers that have reported their results

Initialize the Terms object with initial data.

```
class Ranges (cb, initial_data)
```

Bases: `cbc_sdk.base.UnrefreshableModel`

Represents the range (bucketed) facet fields and values associated with an AuthEvent Facet query.

Initialize an AuthEventFacet Ranges object with initial_data.

facets

Returns the reified `AuthEventFacet.Terms._facets` for this result.

fields

Returns the ranges fields for this result.

```
class Terms (cb, initial_data)
```

Bases: `cbc_sdk.base.UnrefreshableModel`

Represents the facet fields and values associated with an AuthEvent Facet query.

Initialize an AuthEventFacet Terms object with initial_data.

facets

Returns the terms' facets for this result.

fields

Returns the terms facets' fields for this result.

```
completed = None
```

```
contacted = None
```

```
num_found = None
```

```
primary_key = 'job_id'
```

```
ranges = []
```

```
ranges_
```

Returns the reified `AuthEventFacet.Ranges` for this result.

```
result_url = '/api/investigate/v2/orgs/{}/auth_events/facet_jobs/{}/results'
```

```
submit_url = '/api/investigate/v2/orgs/{}/auth_events/facet_jobs'
```

```
terms = []
```

```
terms_
```

Returns the reified *AuthEventFacet.Terms* for this result.

```
class AuthEventGroup (cb, initial_data=None)
```

Bases: `object`

Represents AuthEventGroup

Initialize AuthEventGroup object

Parameters

- **cb** (`CBCloudAPI`) – A reference to the CBCloudAPI object.
- **initial_data** (`dict`) – The data to use when initializing the model object.

Notes

The constructed object will have the following data: - `group_start_timestamp` - `group_end_timestamp` - `group_key` - `group_value`

Example

```
>>> cb = CBCloudAPI(profile="example_profile")
>>> groups = set(cb.select(AuthEvent).where(process_pid=2000).group_results(
    ↪ "device_name"))
>>> for group in groups:
>>>     print(group._info)
```

```
class AuthEventQuery (doc_class, cb)
```

Bases: `cbc_sdk.base.Query`

Represents the query logic for an AuthEvent query.

This class specializes *Query* to handle the particulars of Auth Events querying.

Initialize the AuthEventQuery object.

Parameters

- **doc_class** (`class`) – The class of the model this query returns.
- **cb** (`CBCloudAPI`) – A reference to the CBCloudAPI object.

Example

```
>>> cb = CBCloudAPI(profile="example_profile")
>>> events = cb.select(AuthEvent).where("auth_username:SYSTEM")
>>> print(*events)
```

```
VALID_GROUP_FIELDS = ('auth_domain_name', 'auth_event_action', 'auth_remote_port', 'au
```

```
group_results (fields, max_events_per_group=None, rows=500, start=None,
               range_duration=None, range_field=None, range_method=None)
```

Get group results grouped by provided fields.

Parameters

- **fields** (*str* / *list*) – field or fields by which to perform the grouping
- **max_events_per_group** (*int*) – Maximum number of events in a group, if not provided all events will be returned
- **rows** (*int*) – Number of rows to request, can be paginated
- **start** (*int*) – First row to use for pagination
- **ranges** (*dict*) – dict with information about duration, field, method

Returns grouped results

Return type dict

Examples

```
>>> cb = CBCloudAPI(profile="example_profile")
>>> groups = set(cb.select(AuthEvent).where(process_pid=2000).group_results(
↳ "device_name"))
>>> for group in groups:
>>>     print(group._info)
```

or_ (***kwargs*)

or_() criteria are explicitly provided to AuthEvent queries.

This method overrides the base class in order to provide *or_()* functionality rather than raising an exception.

Example

```
>>> cb = CBCloudAPI(profile="example_profile")
>>> events = cb.select(AuthEvent).where(process_name="chrome.exe").or_
↳ (process_name="firefox.exe")
>>> print(*events)
```

set_rows (*rows*)

Sets the ‘rows’ query body parameter to the ‘start search’ API call, determining how many rows to request.

Parameters **rows** (*int*) – How many rows to request.

Returns AuthEventQuery object

Return type *Query*

Example

```
>>> cb = CBCloudAPI(profile="example_profile")
>>> events = cb.select(AuthEvent).where(process_name="chrome.exe").set_rows(5)
>>> print(*events)
```

timeout (*msecs*)

Sets the timeout on a Auth Event query.

Parameters **msecs** (*int*) – Timeout duration, in milliseconds.

Returns

The Query object with new `milliseconds` parameter.

Return type *Query* (*AuthEventQuery*)

Example

```
>>> cb = CBCloudAPI(profile="example_profile")
>>> events = cb.select(AuthEvent).where(process_name="chrome.exe").
↳ timeout(5000)
>>> print(*events)
```

5.5.3 cbc_sdk.enterprise_edr.threat_intelligence module

Model Classes for Enterprise Endpoint Detection and Response

class Feed (*cb*, *model_unique_id=None*, *initial_data=None*)

Bases: *cbc_sdk.enterprise_edr.threat_intelligence.FeedModel*

Represents an Enterprise EDR feed's metadata.

Parameters

- **name** – A human-friendly name for this feed
- **owner** – The feed owner's connector ID
- **provider_url** – A URL supplied by the feed's provider
- **summary** – A human-friendly summary for the feed
- **category** – The feed's category
- **source_label** – The feed's source label
- **access** – The feed's access (public or private)
- **id** – The feed's unique ID

Initialize the Feed object.

Parameters

- **cb** (*CBCloudAPI*) – A reference to the CBCloudAPI object.
- **model_unique_id** (*str*) – The unique ID of the feed.
- **initial_data** (*dict*) – The initial data for the object.

class FeedBuilder (*cb*, *info*)

Bases: *object*

Helper class allowing Feeds to be assembled.

Creates a new FeedBuilder object.

Parameters

- **cb** (*CBCloudAPI*) – A reference to the CBCloudAPI object.
- **info** (*dict*) – The initial information for the new feed.

add_reports (*reports*)

Adds new reports to the new feed.

Parameters **reports** (*list [Report]*) – New reports to be added to the feed.

Returns This object.
Return type *FeedBuilder*

build()

Builds the new Feed.

Returns The new Feed.
Return type *Feed*

set_alertable (*alertable*)

Sets the alertable for the new feed. Defaults to true if not specified.

Parameters **alertable** (*bool*) – Indicator whether the feed supports alerting.
Returns This object.
Return type *FeedBuilder*

set_category (*category*)

Sets the category for the new feed.

Parameters **category** (*str*) – New category for the feed.
Returns This object.
Return type *FeedBuilder*

set_name (*name*)

Sets the name for the new feed.

Parameters **name** (*str*) – New name for the feed.
Returns This object.
Return type *FeedBuilder*

set_provider_url (*provider_url*)

Sets the provider URL for the new feed.

Parameters **provider_url** (*str*) – New provider URL for the feed.
Returns This object.
Return type *FeedBuilder*

set_source_label (*source_label*)

Sets the source label for the new feed.

Parameters **source_label** (*str*) – New source label for the feed.
Returns This object.
Return type *FeedBuilder*

set_summary (*summary*)

Sets the summary for the new feed.

Parameters **summary** (*str*) – New summary for the feed.
Returns This object.
Return type *FeedBuilder*

access = None

append_reports (*reports*)

Append the given Reports to this Feed's current Reports.

Parameters **reports** (*[Report]*) – List of Reports to append to Feed.
Raises *InvalidObjectError* – If *id* is missing.

append_reports_rawdata (*report_data*)

Append the given report data, formatted as per the API documentation for reports, to this Feed's Reports.

Parameters **report_data** (*list[dict]*) –
Raises *InvalidObjectError* – If *id* is missing or validation of the data fails.

category = None

classmethod create (*cb, name, provider_url, summary, category, alertable=True*)

Begins creating a new feed by making a FeedBuilder to hold the new feed data.

Parameters

- **cb** (*CBCloudAPI*) – A reference to the CBCloudAPI object.
- **name** (*str*) – Name for the new feed.
- **provider_url** (*str*) – Provider URL for the new feed.
- **summary** (*str*) – Summary for the new feed.
- **category** (*str*) – Category for the new feed.

Returns The new FeedBuilder object to be used to create the feed.

Return type *FeedBuilder*

delete ()

Deletes this feed from the Enterprise EDR server.

Raises *InvalidObjectError* – If *id* is missing.

id = None

name = None

owner = None

primary_key = 'id'

provider_url = None

replace_reports (*reports*)

Replace this Feed's Reports with the given Reports.

Parameters **reports** (*[Report]*) – List of Reports to replace existing Reports with.

Raises *InvalidObjectError* – If *id* is missing.

replace_reports_rawdata (*report_data*)

Replace this Feed's Reports with the given reports, specified as raw data.

Parameters **report_data** (*list[dict]*) –

Raises *InvalidObjectError* – If *id* is missing or validation of the data fails.

reports

Returns a list of Reports associated with this feed.

Returns List of Reports in this Feed.

Return type *Reports ([Report])*

save (*public=False*)

Saves this feed on the Enterprise EDR server.

Parameters **public** (*bool*) – Whether to make the feed publicly available.

Returns The saved Feed.

Return type *Feed (Feed)*

source_label = None

summary = None

update (**kwargs)

Update this feed's metadata with the given arguments.

Parameters ****kwargs** (*dict* (*str*, *str*)) – The fields to update.

Raises

- `InvalidObjectError` – If *id* is missing or `Feed.validate()` fails.
- `ApiError` – If an invalid field is specified.

Example

```
>>> feed.update(access="private")
```

urlobject = `'/threathunter/feedmgr/v2/orgs/{}/feeds'`

urlobject_single = `'/threathunter/feedmgr/v2/orgs/{}/feeds/{}'`

validate ()

Checks to ensure this feed contains valid data.

Raises `InvalidObjectError` – If the feed contains invalid data.

class FeedModel (*cb*, *model_unique_id=None*, *initial_data=None*, *force_init=False*, *full_doc=False*)

Bases: `cbc_sdk.base.UnrefreshableModel`, `cbc_sdk.base.CreatableModelMixin`,
`cbc_sdk.base.MutableBaseModel`

A common base class for models used by the Feed and Watchlist APIs.

Initialize the `NewBaseModel` object.

Parameters

- **cb** (`CBCloudAPI`) – A reference to the `CBCloudAPI` object.
- **model_unique_id** (*Any*) – The unique ID for this particular instance of the model object.
- **initial_data** (*dict*) – The data to use when initializing the model object.
- **force_init** (*bool*) – True to force object initialization.
- **full_doc** (*bool*) – True to mark the object as fully initialized.

SCHEMA_IOCv2 = `Schema({'id': And(And(<class 'str'>), <built-in function len>), 'match`

SCHEMA_REPORT = `Schema({'id': And(And(<class 'str'>), <built-in function len>), 'time`

class FeedQuery (*doc_class*, *cb*)

Bases: `cbc_sdk.base.SimpleQuery`

Represents the logic for a Feed query.

```
>>> cb.select(Feed)
>>> cb.select(Feed, id)
>>> cb.select(Feed).where(include_public=True)
```

Initialize the `FeedQuery` object.

Parameters

- **doc_class** (*class*) – The class of the model this query returns.
- **cb** (`CBCloudAPI`) – A reference to the `CBCloudAPI` object.

results

Return a list of Feed objects matching self._args parameters.

where (***kwargs*)

Add kwargs to self._args dictionary.

class `IOC` (*cb, model_unique_id=None, initial_data=None, report_id=None*)

Bases: `cbc_sdk.enterprise_edr.threat_intelligence.FeedModel`

Represents a collection of categorized IOCs. These objects are officially deprecated and replaced by IOC_V2.

Parameters

- **md5** – A list of MD5 checksums
- **ipv4** – A list of IPv4 addresses
- **ipv6** – A list of IPv6 addresses
- **dns** – A list of domain names
- **query** – A list of dicts, each containing an IOC query

Creates a new IOC instance.

Parameters

- **cb** (`BaseAPI`) – Reference to API object used to communicate with the server.
- **model_unique_id** (*str*) – Unique ID of this IOC.
- **initial_data** (*dict*) – Initial data used to populate the IOC.
- **report_id** (*str*) – ID of the report this IOC belongs to (if this is a watchlist IOC).

Raises `ApiError` – If *initial_data* is None.

dns = []

ipv4 = []

ipv6 = []

md5 = []

query = []

validate ()

Checks to ensure this IOC contains valid data.

Raises `InvalidObjectError` – If the IOC contains invalid data.

class `IOC_V2` (*cb, model_unique_id=None, initial_data=None, report_id=None*)

Bases: `cbc_sdk.enterprise_edr.threat_intelligence.FeedModel`

Represents a collection of IOCs of a particular type, plus matching criteria and metadata.

Parameters

- **id** – The IOC_V2's unique ID
- **match_type** – How IOCs in this IOC_V2 are matched
- **values** – A list of IOCs
- **field** – The kind of IOCs contained in this IOC_V2
- **link** – A URL for some reference for this IOC_V2

Creates a new IOC_V2 instance.

Parameters

- **cb** ([BaseAPI](#)) – Reference to API object used to communicate with the server.
- **model_unique_id** (*Any*) – Unused.
- **initial_data** (*dict*) – Initial data used to populate the IOC.
- **report_id** (*str*) – ID of the report this IOC belongs to (if this is a watchlist IOC).

Raises [ApiError](#) – If *initial_data* is None.

classmethod **create_equality** (*cb, iocid, field, *values*)

Creates a new “equality” IOC.

Parameters

- **cb** ([BaseAPI](#)) – Reference to API object used to communicate with the server.
- **iocid** (*str*) – ID for the new IOC. If this is None, a UUID will be generated for the IOC.
- **field** (*str*) – Name of the field to be matched by this IOC.
- ***values** (*list (str)*) – String values to match against the value of the specified field.

Returns New IOC data structure.

Return type [IOC_V2](#)

Raises [ApiError](#) – If there is not at least one value to match against.

classmethod **create_query** (*cb, iocid, query*)

Creates a new “query” IOC.

Parameters

- **cb** ([BaseAPI](#)) – Reference to API object used to communicate with the server.
- **iocid** (*str*) – ID for the new IOC. If this is None, a UUID will be generated for the IOC.
- **query** (*str*) – Query to be incorporated in this IOC.

Returns New IOC data structure.

Return type [IOC_V2](#)

Raises [ApiError](#) – If the query string is not present.

classmethod **create_regex** (*cb, iocid, field, *values*)

Creates a new “regex” IOC.

Parameters

- **cb** ([BaseAPI](#)) – Reference to API object used to communicate with the server.
- **iocid** (*str*) – ID for the new IOC. If this is None, a UUID will be generated for the IOC.
- **field** (*str*) – Name of the field to be matched by this IOC.
- ***values** (*list (str)*) – Regular expression values to match against the value of the specified field.

Returns New IOC data structure.

Return type [IOC_V2](#)

Raises `ApiError` – If there is not at least one regular expression to match against.

field = `None`

id = `None`

ignore()

Sets the ignore status on this IOC.

Only watchlist IOCs have an ignore status.

Raises `InvalidObjectError` – If *id* is missing or this IOC is not from a Watchlist.

ignored

Returns whether or not this IOC is ignored.

Only watchlist IOCs have an ignore status.

Returns True if the IOC is ignored, False otherwise.

Return type `bool`

Raises `InvalidObjectError` – If this IOC is missing an *id* or is not a Watchlist IOC.

Example

```
>>> if ioc.ignored:
...     ioc.unignore()
```

classmethod `ipv6_equality_format` (*input*)

Turns a canonically-formatted IPv6 address into a string suitable for use in an equality IOC.

Parameters *input* (*str*) – The IPv6 address to be translated.

Returns The translated form of IPv6 address.

Return type `str`

Raises `ApiError` – If the string is not in valid format.

link = `None`

match_type = `None`

primary_key = `'id'`

unignore()

Removes the ignore status on this IOC.

Only watchlist IOCs have an ignore status.

Raises `InvalidObjectError` – If *id* is missing or this IOC is not from a Watchlist.

validate()

Checks to ensure this IOC contains valid FQDN.

Raises `InvalidObjectError` – If the IOC contains invalid data.

values = `[]`

class `Report` (*cb*, *model_unique_id=None*, *initial_data=None*, *feed_id=None*, *from_watchlist=False*)

Bases: `cbc_sdk.enterprise_edr.threat_intelligence.FeedModel`

Represents reports retrieved from an Enterprise EDR feed.

Parameters

- **id** – The report’s unique ID
- **timestamp** – When this report was created
- **title** – A human-friendly title for this report
- **description** – A human-friendly description for this report
- **severity** – The severity of the IOCs within this report
- **link** – A URL for some reference for this report
- **tags** – A list of tags for this report
- **iocs_v2** – A list of IOC_V2 dicts associated with this report
- **visibility** – The visibility of this report

Initialize the ReportSeverity object.

Parameters

- **cb** ([CBCloudAPI](#)) – A reference to the CBCloudAPI object.
- **model_unique_id** (*str*) – The ID of the Report (only works for Reports in Watchlists).
- **initial_data** (*dict*) – The initial data for the object.
- **feed_id** (*str*) – The ID of the feed this report is for.
- **from_watchlist** (*bool*) – If the report is in a watchlist

class ReportBuilder (*cb, report_body*)

Bases: [object](#)

Helper class allowing Reports to be assembled.

Initialize a new ReportBuilder.

Parameters

- **cb** ([CBCloudAPI](#)) – A reference to the CBCloudAPI object.
- **report_body** (*dict*) – Partial report body which should be filled in with all “required” fields.

add_ioc (*ioc*)

Adds an IOC to the new report.

Parameters **ioc** ([IOC_V2](#)) – The IOC to be added to the report.

Returns This object.

Return type [ReportBuilder](#)

add_tag (*tag*)

Adds a tag value to the new report.

Parameters **tag** (*str*) – The new tag for the object.

Returns This object.

Return type [ReportBuilder](#)

build ()

Builds the actual Report from the internal data of the ReportBuilder.

Returns The new Report.

Return type [Report](#)

set_description (*description*)

Set the description for the new report.

Parameters **description** (*str*) – New description for the report.

Returns This object.

Return type *ReportBuilder*

set_link (*link*)

Set the link for the new report.

Parameters **link** (*str*) – New link for the report.

Returns This object.

Return type *ReportBuilder*

set_severity (*severity*)

Set the severity for the new report.

Parameters **severity** (*int*) – New severity for the report.

Returns This object.

Return type *ReportBuilder*

set_timestamp (*timestamp*)

Set the timestamp for the new report.

Parameters **timestamp** (*int*) – New timestamp for the report.

Returns This object.

Return type *ReportBuilder*

set_title (*title*)

Set the title for the new report.

Parameters **title** (*str*) – New title for the report.

Returns This object.

Return type *ReportBuilder*

set_visibility (*visibility*)

Set the visibility for the new report.

Parameters **visibility** (*str*) – New visibility for the report.

Returns This object.

Return type *ReportBuilder*

append_iocs (*iocs*)

Append a list of IOCs to this Report.

Parameters **iocs** (*list[IOC_V2]*) – List of IOCs to be added.

classmethod create (*cb, title, description, severity, timestamp=None, tags=None*)

Begin creating a new Report by returning a ReportBuilder.

Parameters

- **cb** (*CBCloudAPI*) – A reference to the CBCloudAPI object.
- **title** (*str*) – Title for the new report.
- **description** (*str*) – Description for the new report.
- **severity** (*int*) – Severity value for the new report.
- **timestamp** (*int*) – UNIX-epoch timestamp for the new report. If omitted, current time will be used.
- **tags** (*list[str]*) – Tags to be added to the report. If omitted, there will be none.

Returns Reference to the ReportBuilder object.

Return type *ReportBuilder*

custom_severity

Returns the custom severity for this report.

Returns

The custom severity for this Report, if it exists.

Return type *ReportSeverity* (*ReportSeverity*)

Raises *InvalidObjectError* – If *id* is missing or this Report is from a Watchlist.

delete()

Deletes this report from the Enterprise EDR server.

Raises *InvalidObjectError* – If *id* is missing, or *feed_id* is missing and this report is a Feed Report.

Example

```
>>> report.delete()
```

description = None

id = None

ignore()

Sets the ignore status on this report.

Raises *InvalidObjectError* – If *id* is missing or feed ID is missing.

ignored

Returns the ignore status for this report.

Returns True if this Report is ignored, False otherwise.

Return type (bool)

Raises *InvalidObjectError* – If *id* is missing or feed ID is missing.

Example

```
>>> if report.ignored:
...     report.unignore()
```

iocs = {}

iocs_

Returns a list of IOC_V2's associated with this report.

Returns List of IOC_V2's for associated with the Report.

Return type *IOC_V2* (*IOC_V2*)

Example

```
>>> for ioc in report.iocs_:
...     print(ioc.values)
```

iocs_v2 = []

link = None

primary_key = 'id'

remove_iocs (*iocs*)

Remove a list of IOCs from this Report.

Parameters **iocs** (*list* [*IOC_V2*]) – List of IOCs to be removed.

remove_iocs_by_id (*ids_list*)

Remove IOCs from this report by specifying their IDs.

Parameters **ids_list** (*list* [*str*]) – List of IDs of the IOCs to be removed.

save_watchlist ()

Saves this report *as a watchlist report*.

Note: This method **cannot** be used to save a feed report. To save feed reports, create them with *cb.create* and use *Feed.replace*.

This method **cannot** be used to save a report that is *already* part of a watchlist. Use the *update()* method instead.

Raises *InvalidObjectError* – If *Report.validate()* fails.

severity = *None*

tags = []

timestamp = *None*

title = *None*

unignore ()

Removes the ignore status on this report.

Raises *InvalidObjectError* – If *id* is missing or feed ID is missing.

update (***kwargs*)

Update this Report with the given arguments.

Parameters ****kwargs** (*dict* (*str*, *str*)) – The Report fields to update.

Returns The updated Report.

Return type *Report* (*Report*)

Raises *InvalidObjectError* – If *id* is missing, or *feed_id* is missing and this report is a Feed Report, or *Report.validate()* fails.

Note: The report's timestamp is always updated, regardless of whether passed explicitly.

```
>>> report.update(title="My new report title")
```

urlobject = '/threathunter/feedmgr/v2/orgs/{}/feeds/{}/reports'

urlobject_single = '/threathunter/watchlistmgr/v3/orgs/{}/reports/{}'

validate ()

Checks to ensure this report contains valid data.

Raises *InvalidObjectError* – If the report contains invalid data.

visibility = *None*

class ReportQuery (*doc_class, cb*)
Bases: *cbc_sdk.base.SimpleQuery*
Represents the logic for a Report query.

Example

```
>>> cb.select(Report).where(feed_id=id)
>>> cb.select(Report, id)
>>> cb.select(Report, id, from_watchlist=True)
```

Initialize the ReportQuery object.

Parameters

- **doc_class** (*class*) – The class of the model this query returns.
- **cb** (*CBCloudAPI*) – A reference to the CBCloudAPI object.

results

Return a list of Report objects

where (***kwargs*)

Add kwargs to self._args dictionary.

class ReportSeverity (*cb, initial_data=None*)
Bases: *cbc_sdk.enterprise_edr.threat_intelligence.FeedModel*
Represents severity information for a Watchlist Report.

Parameters

- **report_id** – The unique ID for the corresponding report
- **severity** – The severity level

Initialize the ReportSeverity object.

Parameters

- **cb** (*CBCloudAPI*) – A reference to the CBCloudAPI object.
- **initial_data** (*dict*) – The initial data for the object.

primary_key = 'report_id'

report_id = None

severity = None

class Watchlist (*cb, model_unique_id=None, initial_data=None*)
Bases: *cbc_sdk.enterprise_edr.threat_intelligence.FeedModel*
Represents an Enterprise EDR watchlist.

Parameters

- **name** – A human-friendly name for the watchlist
- **description** – A short description of the watchlist
- **id** – The watchlist's unique id
- **tags_enabled** – Whether tags are currently enabled
- **alerts_enabled** – Whether alerts are currently enabled

- **create_timestamp** – When this watchlist was created
- **last_update_timestamp** – Report IDs associated with this watchlist
- **report_ids** – Report IDs associated with this watchlist
- **classifier** – A key, value pair specifying an associated feed

Initialize the Watchlist object.

Parameters

- **cb** ([CBCloudAPI](#)) – A reference to the CBCloudAPI object.
- **model_unique_id** (*str*) – The unique ID of the watch list.
- **initial_data** (*dict*) – The initial data for the object.

class WatchlistBuilder (*cb, name*)

Bases: [object](#)

Helper class allowing Watchlists to be assembled.

Creates a new WatchlistBuilder object.

Parameters

- **cb** ([CBCloudAPI](#)) – A reference to the CBCloudAPI object.
- **name** (*str*) – Name for the new watchlist.

add_report_ids (*report_ids*)

Adds report IDs to the watchlist.

Parameters **report_ids** (*list[str]*) – List of report IDs to add to the watchlist.

Returns This object.

Return type [WatchlistBuilder](#)

add_reports (*reports*)

Adds reports to the watchlist.

Parameters **reports** (*list[Report]*) – List of reports to be added to the watchlist.

Returns This object.

Return type [WatchlistBuilder](#)

build ()

Builds the new Watchlist using information in the builder. The new watchlist must still be saved.

Returns The new Watchlist.

Return type [Watchlist](#)

set_alerts_enabled (*flag*)

Sets whether alerts will be enabled on the new watchlist.

Parameters **flag** (*bool*) – True to enable alerts, False to disable them. Default is False.

Returns This object.

Return type [WatchlistBuilder](#)

set_description (*description*)

Sets the description for the new watchlist.

Parameters **description** (*str*) – New description for the watchlist.

Returns This object.

Return type [WatchlistBuilder](#)

set_name (*name*)

Sets the name for the new watchlist.

Parameters **name** (*str*) – New name for the watchlist.

Returns This object.

Return type *WatchlistBuilder*

set_tags_enabled (*flag*)

Sets whether tags will be enabled on the new watchlist.

Parameters **flag** (*bool*) – True to enable tags, False to disable them. Default is True.

Returns This object.

Return type *WatchlistBuilder*

add_report_ids (*report_ids*)

Adds new report IDs to the watchlist.

Parameters **report_ids** (*list[str]*) – List of report IDs to be added to the watchlist.

add_reports (*reports*)

Adds new reports to the watchlist.

Parameters **reports** (*list[Report]*) – List of reports to be added to the watchlist.

alerts_enabled = None

classifier = {}

classifier_

Returns the classifier key and value, if any, for this watchlist.

Returns Watchlist's classifier key and value. None: If there is no classifier key and value.

Return type tuple(str, str)

classmethod create (*cb, name*)

Starts creating a new Watchlist by returning a WatchlistBuilder that can be used to set attributes.

Parameters

- **cb** (*CBCloudAPI*) – A reference to the CBCloudAPI object.
- **name** (*str*) – Name for the new watchlist.

Returns The builder for the new watchlist. Call build() to create the actual Watchlist.

Return type *WatchlistBuilder*

classmethod create_from_feed (*feed, name=None, description=None, enable_alerts=False, enable_tags=True*)

Creates a new Watchlist that encapsulates a Feed.

Parameters

- **feed** (*Feed*) – The feed to be encapsulated by this Watchlist.
- **name** (*str*) – Name for the new watchlist. The default is to use the Feed name.
- **description** (*str*) – Description for the new watchlist. The default is to use the Feed summary.
- **enable_alerts** (*bool*) –
- **enable_tags** (*bool*) –

Returns A new Watchlist object, which must be saved to the server.

Return type *Watchlist*

create_timestamp = None

delete ()

Deletes this watchlist from the Enterprise EDR server.

Raises `InvalidObjectError` – If *id* is missing.

description = `None`

disable_alerts()
Disable alerts for this watchlist.

Raises `InvalidObjectError` – If *id* is missing.

disable_tags()
Disable tagging for this watchlist.

Raises `InvalidObjectError` – if *id* is missing.

enable_alerts()
Enable alerts for this watchlist. Alerts are not retroactive.

Raises `InvalidObjectError` – If *id* is missing.

enable_tags()
Enable tagging for this watchlist.

Raises `InvalidObjectError` – If *id* is missing.

feed
Returns the Feed linked to this Watchlist, if there is one.

id = `None`

last_update_timestamp = `None`

name = `None`

report_ids = `[]`

reports
Returns a list of Report objects associated with this watchlist.

Returns List of Reports associated with the watchlist.

Return type Reports ([\[Report\]](#))

Note: If this Watchlist is a classifier (i.e. feed-linked) Watchlist, *reports* will be empty. To get the reports associated with the linked Feed, use feed like:

```
>>> for report in watchlist.feed.reports:
...     print(report.title)
```

save()
Saves this watchlist on the Enterprise EDR server.

Returns The saved Watchlist.

Return type [Watchlist](#) ([Watchlist](#))

Raises `InvalidObjectError` – If `Watchlist.validate()` fails.

tags_enabled = `None`

update(kwargs)**
Updates this watchlist with the given arguments.

Parameters ****kwargs** (`dict(str, str)`) – The fields to update.

Raises

- `InvalidObjectError` – If `id` is missing or `Watchlist.validate()` fails.
- `ApiError` – If `report_ids` is given and is empty.

Example

```
>>> watchlist.update(name="New Name")
```

`urlobject = '/threathunter/watchlistmgr/v3/orgs/{}/watchlists'`

`urlobject_single = '/threathunter/watchlistmgr/v3/orgs/{}/watchlists/{}'`

`validate()`

Checks to ensure this watchlist contains valid data.

Raises `InvalidObjectError` – If the watchlist contains invalid data.

class WatchlistQuery (*doc_class, cb*)

Bases: `cbc_sdk.base.SimpleQuery`

Represents the logic for a Watchlist query.

```
>>> cb.select(Watchlist)
```

Initialize the WatchlistQuery object.

Parameters

- **doc_class** (*class*) – The class of the model this query returns.
- **cb** (`CBCCloudAPI`) – A reference to the CBCCloudAPI object.

results

Return a list of all Watchlist objects.

`log = <Logger cbc_sdk.enterprise_edr.threat_intelligence (WARNING)>`
Models

5.5.4 cbc_sdk.enterprise_edr.ubs module

Model Classes for Enterprise Endpoint Detection and Response

class Binary (*cb, model_unique_id*)

Bases: `cbc_sdk.base.UnrefreshableModel`

Represents a retrievable binary.

Parameters

- **sha256** – The SHA-256 hash of the file
- **md5** – The MD5 hash of the file
- **file_available** – If true, the file is available for download
- **available_file_size** – The size of the file available for download
- **file_size** – The size of the actual file (represented by the hash)
- **os_type** – The OS that this file is designed for
- **architecture** – The set of architectures that this file was compiled for

- **lang_id** – The Language ID value for the Windows VERSIONINFO resource
- **charset_id** – The Character set ID value for the Windows VERSIONINFO resource
- **internal_name** – The internal name from FileVersionInformation
- **product_name** – The product name from FileVersionInformation
- **company_name** – The company name from FileVersionInformation
- **trademark** – The trademark from FileVersionInformation
- **file_description** – The file description from FileVersionInformation
- **file_version** – The file version from FileVersionInformation
- **comments** – Comments from FileVersionInformation
- **original_filename** – The original filename from FileVersionInformation
- **product_description** – The product description from FileVersionInformation
- **product_version** – The product version from FileVersionInformation
- **private_build** – The private build from FileVersionInformation
- **special_build** – The special build from FileVersionInformation

Initialize the Binary object.

Parameters

- **cb** (`CBCloudAPI`) – A reference to the CBCloudAPI object.
- **model_unique_id** (`str`) – The SHA-256 of the binary being retrieved.

class Summary (`cb`, `model_unique_id`)

Bases: `cbc_sdk.base.UnrefreshableModel`

Represents a summary of organization-specific information for a retrievable binary.

Initialize the Summary object.

Parameters

- **cb** (`CBCloudAPI`) – A reference to the CBCloudAPI object.
- **model_unique_id** (`str`) – The SHA-256 of the binary being retrieved.

primary_key = 'sha256'

urlobject_single = '/ubs/v1/orgs/{}/sha256/{}/summary/device'

architecture = []

available_file_size = None

charset_id = None

comments = None

company_name = None

download_url (`expiration_seconds=3600`)

Returns a URL that can be used to download the file for this binary. Returns None if no download found.

Parameters **expiration_seconds** (`int`) – How long the download should be valid for.

Returns A pre-signed AWS download URL. None: If no download is found.

Return type URL (`str`)

Raises `InvalidObjectError` – If the URL retrieval should be retried.

```
file_available = None
file_description = None
file_size = None
file_version = None
internal_name = None
lang_id = None
md5 = None
original_filename = None
os_type = None
primary_key = 'sha256'
private_build = None
product_description = None
product_name = None
product_version = None
sha256 = None
special_build = None
summary
    Returns organization-specific information about this binary.
trademark = None
urlobject_single = '/ubs/v1/orgs/{}/sha256/{}/metadata'
```

class Downloads (*cb, shas, expiration_seconds=3600*)

Bases: `cbc_sdk.base.UnrefreshableModel`

Represents download information for a list of process hashes.

Initialize the Downloads object.

Parameters

- **cb** (`CBCloudAPI`) – A reference to the `CBCloudAPI` object.
- **shas** (*list*) – A list of SHA hash values for binaries.
- **expiration_seconds** (*int*) – Number of seconds until this request expires.

class FoundItem (*cb, item*)

Bases: `cbc_sdk.base.UnrefreshableModel`

Represents the download URL and process hash for a successfully located binary.

Initialize the FoundItem object.

Parameters

- **cb** (`CBCloudAPI`) – A reference to the `CBCloudAPI` object.
- **item** (*dict*) – The values for a successfully-retrieved item.

```
primary_key = 'sha256'
```

found

Returns a list of Downloads.FoundItem, one for each binary found in the binary store.

urlobject = '/ubs/v1/orgs/{}/file/_download'

5.5.5 Module contents

5.6 Platform

5.6.1 Submodules

5.6.2 cbc_sdk.platform.alerts module

Model and Query Classes for Platform Alerts and Workflows

class BaseAlert (*cb, model_unique_id, initial_data=None*)

Bases: *cbc_sdk.platform.base.PlatformModel*

Represents a basic alert.

Parameters

- **category** – Alert category - Monitored vs Threat
- **create_time** – Time the alert was created
- **device_id** – ID of the device (empty for Container Runtime alerts)
- **device_name** – Device name (empty for Container Runtime alerts)
- **device_os** – Device OS (empty for Container Runtime alerts)
- **device_os_version** – Device OS Version (empty for Container Runtime alerts)
- **device_username** – Logged on user during the alert. This is filled on a best-effort approach. If the user is not available it may be populated with the device owner (empty for Container Runtime alerts)
- **first_event_time** – Time of the first event in an alert
- **group_details** – Group details for when alert grouping is on
- **id** – Unique ID for this alert
- **last_event_time** – Time of the last event in an alert
- **last_update_time** – Time the alert was last updated
- **legacy_alert_id** – Unique short ID for this alert. This is deprecated and only available on alerts stored in the old schema.
- **notes_present** – Are notes present for this threatId
- **org_key** – Unique identifier for the organization to which the alert belongs
- **policy_id** – ID of the policy the device was in at the time of the alert
- **policy_name** – Name of the policy the device was in at the time of the alert
- **severity** – Threat ranking
- **tags** – Tags for the alert
- **target_value** – Device priority as assigned via the policy

- **threat_id** – ID of the threat to which this alert belongs. Threats are comprised of a combination of factors that can be repeated across devices.
- **type** – Type of the alert
- **workflow** – User-updatable status of the alert

Initialize the BaseAlert object.

Parameters

- **cb** ([BaseAPI](#)) – Reference to API object used to communicate with the server.
- **model_unique_id** (*str*) – ID of the alert represented.
- **initial_data** (*dict*) – Initial data used to populate the alert.

class Note (*cb, alert, model_unique_id, initial_data=None*)

Bases: [cbc_sdk.platform.base.PlatformModel](#)

Represents a note within an alert.

Parameters

- **author** – User who created the note
- **create_time** – Time the note was created
- **id** – Unique ID for this note
- **note** – Note contents

Initialize the Note object.

Parameters

- **cb** ([BaseAPI](#)) – Reference to API object used to communicate with the server.
- **alert** ([BaseAlert](#)) – The alert where the note is saved.
- **model_unique_id** (*str*) – ID of the note represented.
- **initial_data** (*dict*) – Initial data used to populate the note.

author = None

create_time = None

delete()

Deletes a note from an alert.

id = None

note = None

primary_key = 'id'

urlobject = '/appservices/v6/orgs/{0}/alerts/{1}/notes'

urlobject_single = '/appservices/v6/orgs/{0}/alerts/{1}/notes/{2}'

category = None

create_note (*note*)

Creates a new note.

create_time = None

device_id = None

device_name = None

device_os = None

device_os_version = None

device_username = None

dismiss (*remediation=None, comment=None*)

Dismisses this alert.

Parameters

- **remediation** (*str*) – The remediation status to set for the alert.
- **comment** (*str*) – The comment to set for the alert.

dismiss_threat (*remediation=None, comment=None*)

Dismisses all alerts with the same threat ID, past or future.

Parameters

- **remediation** (*str*) – The remediation status to set for the alert.
- **comment** (*str*) – The comment to set for the alert.

first_event_time = None

group_details = {}

id = None

last_event_time = None

last_update_time = None

legacy_alert_id = None

notes_ ()

Retrieves all notes for an alert.

notes_present = None

org_key = None

policy_id = None

policy_name = None

primary_key = 'id'

static search_suggestions (*cb, query*)

Returns suggestions for keys and field values that can be used in a search.

Parameters

- **cb** (*CBCloudAPI*) – A reference to the CBCloudAPI object.
- **query** (*str*) – A search query to use.

Returns A list of search suggestions expressed as dict objects.

Return type list

Raises *ApiError* – if cb is not instance of CBCloudAPI

severity = None

tags = []

target_value = None

threat_id = None

type = None

update (*remediation=None, comment=None*)

Updates this alert while leaving it open.

Parameters

- **remediation** (*str*) – The remediation status to set for the alert.
- **comment** (*str*) – The comment to set for the alert.

update_threat (*remediation=None, comment=None*)

Updates the status of all alerts with the same threat ID, past or future, while leaving them in OPEN state.

Parameters

- **remediation** (*str*) – The remediation status to set for the alert.
- **comment** (*str*) – The comment to set for the alert.

urlobject = '/appservices/v6/orgs/{0}/alerts'

urlobject_single = '/appservices/v6/orgs/{0}/alerts/{1}'

workflow = {}

workflow_

Returns the workflow associated with this alert.

Returns The workflow associated with this alert.

Return type *Workflow*

class BaseAlertSearchQuery (*doc_class, cb*)

Bases: *cbc_sdk.base.BaseQuery, cbc_sdk.base.QueryBuilderSupportMixin, cbc_sdk.base.IterableQueryMixin, cbc_sdk.base.CriteriaBuilderSupportMixin*

Represents a query that is used to locate BaseAlert objects.

Initialize the BaseAlertSearchQuery.

Parameters

- **doc_class** (*class*) – The model class that will be returned by this query.
- **cb** (*BaseAPI*) – Reference to API object used to communicate with the server.

VALID_ALERT_TYPES = ['CB_ANALYTICS', 'DEVICE_CONTROL', 'WATCHLIST', 'CONTAINER_RUNTIME']

VALID_CATEGORIES = ['THREAT', 'MONITORED']

VALID_FACET_FIELDS = ['ALERT_TYPE', 'CATEGORY', 'REPUTATION', 'WORKFLOW', 'TAG', 'POLICY']

VALID_REPUTATIONS = ['KNOWN_MALWARE', 'SUSPECT_MALWARE', 'PUP', 'NOT_LISTED', 'ADAPTIVE']

VALID_WORKFLOW_VALS = ['OPEN', 'DISMISSED']

dismiss (*remediation=None, comment=None*)

Dismiss all alerts matching the given query. The alerts will be left in a DISMISSED state after this request.

Parameters

- **remediation** (*str*) – The remediation state to set for all alerts.
- **comment** (*str*) – The comment to set for all alerts.

Returns The request ID, which may be used to select a WorkflowStatus object.

Return type str

facets (*fieldlist*, *max_rows=0*)

Return information about the facets for this alert by search, using the defined criteria.

Parameters

- **fieldlist** (*list*) – List of facet field names. Valid names are “ALERT_TYPE”, “CATEGORY”, “REPUTATION”, “WORKFLOW”, “TAG”, “POLICY_ID”, “POLICY_NAME”, “DEVICE_ID”, “DEVICE_NAME”, “APPLICATION_HASH”, “APPLICATION_NAME”, “STATUS”, “RUN_STATE”, “POLICY_APPLIED_STATE”, “POLICY_APPLIED”, and “SENSOR_ACTION”.
- **max_rows** (*int*) – The maximum number of rows to return. 0 means return all rows.

Returns A list of facet information specified as dicts.

Return type list

set_alert_ids (*alert_ids*)

Restricts the alerts that this query is performed on to the specified alert IDs.

Parameters **alert_ids** (*list*) – List of string alert IDs.

Returns This instance.

Return type *BaseAlertSearchQuery*

set_categories (*categories*)

Restricts the alerts that this query is performed on to the specified categories.

Parameters **categories** (*list*) – List of categories to be restricted to. Valid categories are “THREAT”, “MONITORED”, “INFO”, “MINOR”, “SERIOUS”, and “CRITICAL.”

Returns This instance.

Return type *BaseAlertSearchQuery*

set_create_time (**args*, ***kwargs*)

Restricts the alerts that this query is performed on to the specified creation time.

The time may either be specified as a start and end point or as a range.

Parameters

- ***args** (*list*) – Not used.
- ****kwargs** (*dict*) – Used to specify start= for start time, end= for end time, and range= for range.

Returns This instance.

Return type *BaseAlertSearchQuery*

set_device_ids (*device_ids*)

Restricts the alerts that this query is performed on to the specified device IDs.

Parameters **device_ids** (*list*) – List of integer device IDs.

Returns This instance.

Return type *BaseAlertSearchQuery*

set_device_names (*device_names*)

Restricts the alerts that this query is performed on to the specified device names.

Parameters `device_names` (*list*) – List of string device names.

Returns This instance.

Return type *BaseAlertSearchQuery*

set_device_os (*device_os*)

Restricts the alerts that this query is performed on to the specified device operating systems.

Parameters `device_os` (*list*) – List of string operating systems. Valid values are “WINDOWS”, “ANDROID”, “MAC”, “IOS”, “LINUX”, and “OTHER.”

Returns This instance.

Return type *BaseAlertSearchQuery*

set_device_os_versions (*device_os_versions*)

Restricts the alerts that this query is performed on to the specified device operating system versions.

Parameters `device_os_versions` (*list*) – List of string operating system versions.

Returns This instance.

Return type *BaseAlertSearchQuery*

set_device_username (*users*)

Restricts the alerts that this query is performed on to the specified user names.

Parameters `users` (*list*) – List of string user names.

Returns This instance.

Return type *BaseAlertSearchQuery*

set_group_results (*do_group*)

Specifies whether or not to group the results of the query.

Parameters `do_group` (*bool*) – True to group the results, False to not do so.

Returns This instance.

Return type *BaseAlertSearchQuery*

set_legacy_alert_ids (*alert_ids*)

Restricts the alerts that this query is performed on to the specified legacy alert IDs.

Parameters `alert_ids` (*list*) – List of string legacy alert IDs.

Returns This instance.

Return type *BaseAlertSearchQuery*

set_minimum_severity (*severity*)

Restricts the alerts that this query is performed on to the specified minimum severity level.

Parameters `severity` (*int*) – The minimum severity level for alerts.

Returns This instance.

Return type *BaseAlertSearchQuery*

set_policy_ids (*policy_ids*)

Restricts the alerts that this query is performed on to the specified policy IDs.

Parameters `policy_ids` (*list*) – List of integer policy IDs.

Returns This instance.

Return type *BaseAlertSearchQuery*

set_policy_names (*policy_names*)

Restricts the alerts that this query is performed on to the specified policy names.

Parameters **policy_names** (*list*) – List of string policy names.

Returns This instance.

Return type *BaseAlertSearchQuery*

set_process_names (*process_names*)

Restricts the alerts that this query is performed on to the specified process names.

Parameters **process_names** (*list*) – List of string process names.

Returns This instance.

Return type *BaseAlertSearchQuery*

set_process_sha256 (*shas*)

Restricts the alerts that this query is performed on to the specified process SHA-256 hash values.

Parameters **shas** (*list*) – List of string process SHA-256 hash values.

Returns This instance.

Return type *BaseAlertSearchQuery*

set_reputations (*reps*)

Restricts the alerts that this query is performed on to the specified reputation values.

Parameters **reps** (*list*) – List of string reputation values. Valid values are “KNOWN_MALWARE”, “SUSPECT_MALWARE”, “PUP”, “NOT_LISTED”, “ADAPTIVE_WHITE_LIST”, “COMMON_WHITE_LIST”, “TRUSTED_WHITE_LIST”, and “COMPANY_BLACK_LIST”.

Returns This instance.

Return type *BaseAlertSearchQuery*

set_tags (*tags*)

Restricts the alerts that this query is performed on to the specified tag values.

Parameters **tags** (*list*) – List of string tag values.

Returns This instance.

Return type *BaseAlertSearchQuery*

set_target_priorities (*priorities*)

Restricts the alerts that this query is performed on to the specified target priority values.

Parameters **priorities** (*list*) – List of string target priority values. Valid values are “LOW”, “MEDIUM”, “HIGH”, and “MISSION_CRITICAL”.

Returns This instance.

Return type *BaseAlertSearchQuery*

set_threat_ids (*threats*)

Restricts the alerts that this query is performed on to the specified threat ID values.

Parameters **threats** (*list*) – List of string threat ID values.

Returns This instance.

Return type *BaseAlertSearchQuery*

set_time_range (*key*, ***kwargs*)

Restricts the alerts that this query is performed on to the specified time range.

The time may either be specified as a start and end point or as a range.

Parameters

- **key** (*str*) – The key to use for criteria one of `create_time`, `first_event_time`, `last_event_time`, or `last_update_time`
- ****kwargs** (*dict*) – Used to specify `start=` for start time, `end=` for end time, and `range=` for range.

Returns This instance.

Return type *BaseAlertSearchQuery*

set_types (*alerttypes*)

Restricts the alerts that this query is performed on to the specified alert type values.

Parameters **alerttypes** (*list*) – List of string alert type values. Valid values are “CB_ANALYTICS”, “WATCHLIST”, “DEVICE_CONTROL”, and “CONTAINER_RUNTIME”.

Returns This instance.

Return type *BaseAlertSearchQuery*

Note: - When filtering by fields that take a list parameter, an empty list will be treated as a wildcard and match everything.

set_workflows (*workflow_vals*)

Restricts the alerts that this query is performed on to the specified workflow status values.

Parameters **workflow_vals** (*list*) – List of string alert type values. Valid values are “OPEN” and “DISMISSED”.

Returns This instance.

Return type *BaseAlertSearchQuery*

sort_by (*key*, *direction='ASC'*)

Sets the sorting behavior on a query’s results.

Example

```
>>> cb.select(BaseAlert).sort_by("name")
```

Parameters

- **key** (*str*) – The key in the schema to sort by.
- **direction** (*str*) – The sort order, either “ASC” or “DESC”.

Returns This instance.

Return type *BaseAlertSearchQuery*

update (*remediation=None*, *comment=None*)

Update all alerts matching the given query. The alerts will be left in an OPEN state after this request.

Parameters

- **remediation** (*str*) – The remediation state to set for all alerts.
- **comment** (*str*) – The comment to set for all alerts.

Returns The request ID, which may be used to select a WorkflowStatus object.

Return type *str*

class **CBAntivirusAlert** (*cb, model_unique_id, initial_data=None*)

Bases: *cbc_sdk.platform.alerts.BaseAlert*

Represents CB Analytics alerts.

Initialize the BaseAlert object.

Parameters

- **cb** (*BaseAPI*) – Reference to API object used to communicate with the server.
- **model_unique_id** (*str*) – ID of the alert represented.
- **initial_data** (*dict*) – Initial data used to populate the alert.

get_events (*timeout=0, async_mode=False*)

Requests enriched events detailed results.

Parameters

- **timeout** (*int*) – Event details request timeout in milliseconds.
- **async_mode** (*bool*) – True to request details in an asynchronous manner.

Returns EnrichedEvents matching the legacy_alert_id

Return type *list*

Note:

- When using asynchronous mode, this method returns a python future. You can call result() on the future object to wait for completion and get the results.
-

urlobject = `'/appservices/v6/orgs/{0}/alerts/cbanalytics'`

class **CBAntivirusAlertSearchQuery** (*doc_class, cb*)

Bases: *cbc_sdk.platform.alerts.BaseAlertSearchQuery*

Represents a query that is used to locate CBAntivirusAlert objects.

Initialize the CBAntivirusAlertSearchQuery.

Parameters

- **doc_class** (*class*) – The model class that will be returned by this query.
- **cb** (*BaseAPI*) – Reference to API object used to communicate with the server.

VALID_KILL_CHAIN_STATUSES = ['RECONNAISSANCE', 'WEAPONIZE', 'DELIVER_EXPLOIT', 'INSTALL']

VALID_LOCATIONS = ['ONSITE', 'OFFSITE', 'UNKNOWN']

VALID_POLICY_APPLIED = ['APPLIED', 'NOT_APPLIED']

VALID_RUN_STATES = ['DID_NOT_RUN', 'RAN', 'UNKNOWN']

VALID_SENSOR_ACTIONS = ['POLICY_NOT_APPLIED', 'ALLOW', 'ALLOW_AND_LOG', 'TERMINATE', '']

VALID_THREAT_CATEGORIES = ['UNKNOWN', 'NON_MALWARE', 'NEW_MALWARE', 'KNOWN_MALWARE', '']

```
VALID_THREAT_CAUSE_VECTORS = ['EMAIL', 'WEB', 'GENERIC_SERVER', 'GENERIC_CLIENT', 'REM
```

set_blocked_threat_categories (*categories*)

Restricts the alerts that this query is performed on to the specified threat categories that were blocked.

Parameters **categories** (*list*) – List of threat categories to look for. Valid values are “UNKNOWN”, “NON_MALWARE”, “NEW_MALWARE”, “KNOWN_MALWARE”, and “RISKY_PROGRAM”.

Returns This instance.

Return type *CBAnalyticsAlertSearchQuery*

set_device_locations (*locations*)

Restricts the alerts that this query is performed on to the specified device locations.

Parameters **locations** (*list*) – List of device locations to look for. Valid values are “ON-SITE”, “OFFSITE”, and “UNKNOWN”.

Returns This instance.

Return type *CBAnalyticsAlertSearchQuery*

set_kill_chain_statuses (*statuses*)

Restricts the alerts that this query is performed on to the specified kill chain statuses.

Parameters **statuses** (*list*) – List of kill chain statuses to look for. Valid values are “RECONNAISSANCE”, “WEAPONIZE”, “DELIVER_EXPLOIT”, “INSTALL_RUN”, “COMMAND_AND_CONTROL”, “EXECUTE_GOAL”, and “BREACH”.

Returns This instance.

Return type *CBAnalyticsAlertSearchQuery*

set_not_blocked_threat_categories (*categories*)

Restricts the alerts that this query is performed on to the specified threat categories that were NOT blocked.

Parameters **categories** (*list*) – List of threat categories to look for. Valid values are “UNKNOWN”, “NON_MALWARE”, “NEW_MALWARE”, “KNOWN_MALWARE”, and “RISKY_PROGRAM”.

Returns This instance.

Return type *CBAnalyticsAlertSearchQuery*

set_policy_applied (*applied_statuses*)

Restricts the alerts that this query is performed on to the specified policy status values.

Parameters **applied_statuses** (*list*) – List of status values to look for. Valid values are “APPLIED” and “NOT_APPLIED”.

Returns This instance.

Return type *CBAnalyticsAlertSearchQuery*

set_reason_code (*reason*)

Restricts the alerts that this query is performed on to the specified reason codes (enum values).

Parameters **reason** (*list*) – List of string reason codes to look for.

Returns This instance.

Return type *CBAnalyticsAlertSearchQuery*

set_run_states (*states*)

Restricts the alerts that this query is performed on to the specified run states.

Parameters **states** (*list*) – List of run states to look for. Valid values are “DID_NOT_RUN”, “RAN”, and “UNKNOWN”.

Returns This instance.

Return type *CBAnalyticsAlertSearchQuery*

set_sensor_actions (*actions*)

Restricts the alerts that this query is performed on to the specified sensor actions.

Parameters **actions** (*list*) – List of sensor actions to look for. Valid values are “POLICY_NOT_APPLIED”, “ALLOW”, “ALLOW_AND_LOG”, “TERMINATE”, and “DENY”.

Returns This instance.

Return type *CBAnalyticsAlertSearchQuery*

set_threat_cause_vectors (*vectors*)

Restricts the alerts that this query is performed on to the specified threat cause vectors.

Parameters **vectors** (*list*) – List of threat cause vectors to look for. Valid values are “EMAIL”, “WEB”, “GENERIC_SERVER”, “GENERIC_CLIENT”, “REMOTE_DRIVE”, “REMOVABLE_MEDIA”, “UNKNOWN”, “APP_STORE”, and “THIRD_PARTY”.

Returns This instance.

Return type *CBAnalyticsAlertSearchQuery*

class ContainerRuntimeAlert (*cb, model_unique_id, initial_data=None*)

Bases: *cbc_sdk.platform.alerts.BaseAlert*

Represents Container Runtime alerts.

Initialize the BaseAlert object.

Parameters

- **cb** (*BaseAPI*) – Reference to API object used to communicate with the server.
- **model_unique_id** (*str*) – ID of the alert represented.
- **initial_data** (*dict*) – Initial data used to populate the alert.

urlobject = '/appservices/v6/orgs/{0}/alerts/containerruntime'

class ContainerRuntimeAlertSearchQuery (*doc_class, cb*)

Bases: *cbc_sdk.platform.alerts.BaseAlertSearchQuery*

Represents a query that is used to locate ContainerRuntimeAlert objects.

Initialize the ContainerRuntimeAlertSearchQuery.

Parameters

- **doc_class** (*class*) – The model class that will be returned by this query.
- **cb** (*BaseAPI*) – Reference to API object used to communicate with the server.

set_cluster_names (*names*)

Restricts the alerts that this query is performed on to the specified Kubernetes cluster names.

Parameters **names** (*list*) – List of Kubernetes cluster names to look for.

Returns This instance.

Return type *ContainerRuntimeAlertSearchQuery*

set_egress_group_ids (*ids*)

Restricts the alerts that this query is performed on to the specified egress group IDs.

Parameters **ids** (*list*) – List of egress group IDs to look for.

Returns This instance.

Return type *ContainerRuntimeAlertSearchQuery*

set_egress_group_names (*names*)

Restricts the alerts that this query is performed on to the specified egress group names.

Parameters **names** (*list*) – List of egress group names to look for.

Returns This instance.

Return type *ContainerRuntimeAlertSearchQuery*

set_ip_reputations (*reputations*)

Restricts the alerts that this query is performed on to the specified IP reputation values.

Parameters **reputations** (*list*) – List of IP reputation values to look for.

Returns This instance.

Return type *ContainerRuntimeAlertSearchQuery*

set_namespaces (*namespaces*)

Restricts the alerts that this query is performed on to the specified Kubernetes namespaces.

Parameters **namespaces** (*list*) – List of Kubernetes namespaces to look for.

Returns This instance.

Return type *ContainerRuntimeAlertSearchQuery*

set_ports (*ports*)

Restricts the alerts that this query is performed on to the specified listening ports.

Parameters **ports** (*list*) – List of listening ports to look for.

Returns This instance.

Return type *ContainerRuntimeAlertSearchQuery*

set_protocols (*protocols*)

Restricts the alerts that this query is performed on to the specified protocols.

Parameters **protocols** (*list*) – List of protocols to look for.

Returns This instance.

Return type *ContainerRuntimeAlertSearchQuery*

set_remote_domains (*domains*)

Restricts the alerts that this query is performed on to the specified remote domains.

Parameters **domains** (*list*) – List of remote domains to look for.

Returns This instance.

Return type *ContainerRuntimeAlertSearchQuery*

set_remote_ips (*addrs*)

Restricts the alerts that this query is performed on to the specified remote IP addresses.

Parameters **addrs** (*list*) – List of remote IP addresses to look for.

Returns This instance.

Return type *ContainerRuntimeAlertSearchQuery*

set_replica_ids (*ids*)

Restricts the alerts that this query is performed on to the specified pod names.

Parameters *ids* (*list*) – List of pod names to look for.

Returns This instance.

Return type *ContainerRuntimeAlertSearchQuery*

set_rule_ids (*ids*)

Restricts the alerts that this query is performed on to the specified Kubernetes policy rule IDs.

Parameters *ids* (*list*) – List of Kubernetes policy rule IDs to look for.

Returns This instance.

Return type *ContainerRuntimeAlertSearchQuery*

set_rule_names (*names*)

Restricts the alerts that this query is performed on to the specified Kubernetes policy rule names.

Parameters *names* (*list*) – List of Kubernetes policy rule names to look for.

Returns This instance.

Return type *ContainerRuntimeAlertSearchQuery*

set_workload_ids (*ids*)

Restricts the alerts that this query is performed on to the specified workload IDs.

Parameters *ids* (*list*) – List of workload IDs to look for.

Returns This instance.

Return type *ContainerRuntimeAlertSearchQuery*

set_workload_kinds (*kinds*)

Restricts the alerts that this query is performed on to the specified workload types.

Parameters *kinds* (*list*) – List of workload types to look for.

Returns This instance.

Return type *ContainerRuntimeAlertSearchQuery*

set_workload_names (*names*)

Restricts the alerts that this query is performed on to the specified workload names.

Parameters *names* (*list*) – List of workload names to look for.

Returns This instance.

Return type *ContainerRuntimeAlertSearchQuery*

class DeviceControlAlert (*cb, model_unique_id, initial_data=None*)

Bases: *cbc_sdk.platform.alerts.BaseAlert*

Represents Device Control alerts.

Initialize the BaseAlert object.

Parameters

- **cb** (*BaseAPI*) – Reference to API object used to communicate with the server.
- **model_unique_id** (*str*) – ID of the alert represented.

- **initial_data** (*dict*) – Initial data used to populate the alert.

urlobject = '/appservices/v6/orgs/{0}/alerts/devicecontrol'

class DeviceControlAlertSearchQuery (*doc_class, cb*)

Bases: *cbc_sdk.platform.alerts.BaseAlertSearchQuery*

Represents a query that is used to locate DeviceControlAlert objects.

Initialize the DeviceControlAlertSearchQuery.

Parameters

- **doc_class** (*class*) – The model class that will be returned by this query.
- **cb** (*BaseAPI*) – Reference to API object used to communicate with the server.

set_external_device_friendly_names (*names*)

Restricts the alerts that this query is performed on to the specified external device friendly names.

Parameters **names** (*list*) – List of external device friendly names to look for.

Returns This instance.

Return type *DeviceControlAlertSearchQuery*

set_external_device_ids (*ids*)

Restricts the alerts that this query is performed on to the specified external device IDs.

Parameters **ids** (*list*) – List of external device IDs to look for.

Returns This instance.

Return type *DeviceControlAlertSearchQuery*

set_product_ids (*ids*)

Restricts the alerts that this query is performed on to the specified product IDs.

Parameters **ids** (*list*) – List of product IDs to look for.

Returns This instance.

Return type *DeviceControlAlertSearchQuery*

set_product_names (*names*)

Restricts the alerts that this query is performed on to the specified product names.

Parameters **names** (*list*) – List of product names to look for.

Returns This instance.

Return type *DeviceControlAlertSearchQuery*

set_serial_numbers (*serial_numbers*)

Restricts the alerts that this query is performed on to the specified serial numbers.

Parameters **serial_numbers** (*list*) – List of serial numbers to look for.

Returns This instance.

Return type *DeviceControlAlertSearchQuery*

set_vendor_ids (*ids*)

Restricts the alerts that this query is performed on to the specified vendor IDs.

Parameters **ids** (*list*) – List of vendor IDs to look for.

Returns This instance.

Return type *DeviceControlAlertSearchQuery*

set_vendor_names (*names*)

Restricts the alerts that this query is performed on to the specified vendor names.

Parameters *names* (*list*) – List of vendor names to look for.

Returns This instance.

Return type *DeviceControlAlertSearchQuery*

class WatchlistAlert (*cb, model_unique_id, initial_data=None*)

Bases: *cbc_sdk.platform.alerts.BaseAlert*

Represents watch list alerts.

Initialize the BaseAlert object.

Parameters

- **cb** (*BaseAPI*) – Reference to API object used to communicate with the server.
- **model_unique_id** (*str*) – ID of the alert represented.
- **initial_data** (*dict*) – Initial data used to populate the alert.

get_process (*async_mode=False*)

Gets the process corresponding with the alert.

Parameters *async_mode* – True to request process in an asynchronous manner.

Returns The process corresponding to the alert.

Return type *Process*

urlobject = '/appservices/v6/orgs/{0}/alerts/watchlist'

class WatchlistAlertSearchQuery (*doc_class, cb*)

Bases: *cbc_sdk.platform.alerts.BaseAlertSearchQuery*

Represents a query that is used to locate WatchlistAlert objects.

Initialize the WatchlistAlertSearchQuery.

Parameters

- **doc_class** (*class*) – The model class that will be returned by this query.
- **cb** (*BaseAPI*) – Reference to API object used to communicate with the server.

set_watchlist_ids (*ids*)

Restricts the alerts that this query is performed on to the specified watchlist ID values.

Parameters *ids* (*list*) – List of string watchlist ID values.

Returns This instance.

Return type *WatchlistAlertSearchQuery*

set_watchlist_names (*names*)

Restricts the alerts that this query is performed on to the specified watchlist name values.

Parameters *names* (*list*) – List of string watchlist name values.

Returns This instance.

Return type *WatchlistAlertSearchQuery*

```
class Workflow(cb, initial_data=None)
```

Bases: `cbc_sdk.base.UnrefreshableModel`

Represents the workflow associated with alerts.

Parameters

- **changed_by** – Username of the user who changed the workflow
- **comment** – Comment when updating the workflow
- **last_update_time** – When the workflow was last updated
- **remediation** – Alert remediation code. Indicates the result of the investigation into the alert
- **state** – State of the workflow

Initialize the Workflow object.

Parameters

- **cb** (`BaseAPI`) – Reference to API object used to communicate with the server.
- **initial_data** (`dict`) – Initial data used to populate the workflow.

```
changed_by = None
```

```
comment = None
```

```
last_update_time = None
```

```
remediation = None
```

```
state = None
```

```
class WorkflowStatus(cb, model_unique_id, initial_data=None)
```

Bases: `cbc_sdk.platform.base.PlatformModel`

Represents the current workflow status of a request.

Parameters

- **errors** – Errors for dismiss alerts or threats, if no errors it won't be included in response
- **failed_ids** – Failed ids
- **id** – Time based id for async job, it's not unique across the orgs
- **num_hits** – Total number of alerts to be operated on
- **num_success** – Successfully operated number of alerts
- **status** – Status for the async progress
- **workflow** – Requested workflow change

Initialize the BaseAlert object.

Parameters

- **cb** (`BaseAPI`) – Reference to API object used to communicate with the server.
- **model_unique_id** (`str`) – ID of the request being processed.
- **initial_data** (`dict`) – Initial data used to populate the status.

```
errors = []
```

```
failed_ids = []
```

finished

Returns whether this request has been completed.

Returns True if the request is in “finished” state, False if not.

Return type bool

id = None**id_**

Returns the request ID of the associated request.

Returns The request ID of the associated request.

Return type str

in_progress

Returns whether this request is currently in progress.

Returns True if the request is in “in progress” state, False if not.

Return type bool

num_hits = None**num_success = None****primary_key = 'id'****queued**

Returns whether this request has been queued.

Returns True if the request is in “queued” state, False if not.

Return type bool

status = None**urlobject_single = '/appservices/v6/orgs/{0}/workflow/status/{1}'****workflow = {}****workflow_**

Returns the current workflow associated with this request.

Returns The current workflow associated with this request.

Return type *Workflow*

5.6.3 cbc_sdk.platform.base module

Model and Query Classes for Platform

```
class PlatformModel (cb,      model_unique_id=None,      initial_data=None,      force_init=False,
                      full_doc=False)
```

Bases: *cbc_sdk.base.NewBaseModel*

Represents the base of all Platform API model classes.

Initialize the PlatformModel object.

Parameters

- **cb** (*CBCloudAPI*) – A reference to the CBCloudAPI object.
- **model_unique_id** (*Any*) – The unique ID for this particular instance of the model object.

- **initial_data** (*dict*) – The data to use when initializing the model object.
- **force_init** (*bool*) – True to force object initialization.
- **full_doc** (*bool*) – True to mark the object as fully initialized.

```
log = <Logger cbc_sdk.platform.base (WARNING)>  
Platform Models
```

5.6.4 cbc_sdk.platform.devices module

Model and Query Classes for Platform Devices

```
class Device (cb, model_unique_id, initial_data=None)  
Bases: cbc_sdk.platform.base.PlatformModel
```

Represents a device (endpoint).

Parameters

- **activation_code** – Device activation code
- **activation_code_expiry_time** – When the expiration code expires and cannot be used to register a device
- **ad_group_id** – Device's AD group
- **av_ave_version** – AVE version (part of AV Version)
- **av_engine** – Current AV version
- **av_last_scan_time** – Last AV scan time
- **av_master** – Whether the device is an AV Master (?)
- **av_pack_version** – Pack version (part of AV Version)
- **av_product_version** – AV Product version (part of AV Version)
- **av_status** – AV Statuses
- **av_update_servers** – Device's AV servers
- **av_vdf_version** – VDF version (part of AV Version)
- **current_sensor_policy_name** – Current MSM policy name
- **deregistered_time** – When the device was deregistered with the PSC backend
- **device_id** – ID of the device
- **device_meta_data_item_list** – MSM Device metadata
- **device_owner_id** – ID of the user who owns the device
- **email** – Email of the user who owns the device
- **encoded_activation_code** – Encoded device activation code
- **first_name** – First name of the user who owns the device
- **id** – ID of the device
- **last_contact_time** – Time the device last checked into the PSC backend
- **last_device_policy_changed_time** – Last time the device's policy was changed

- **last_device_policy_requested_time** – Last time the device requested policy updates
- **last_external_ip_address** – Device’s external IP
- **last_internal_ip_address** – Device’s internal IP
- **last_location** – Location of the device (on-/off-premises)
- **last_name** – Last name of the user who owns the device
- **last_policy_updated_time** – Last time the device was MSM processed
- **last_reported_time** – Time when device last reported an event to PSC backend
- **last_reset_time** – When the sensor was last reset
- **last_shutdown_time** – When the device last shut down
- **linux_kernel_version** – Linux kernel version
- **login_user_name** – Last active logged in username
- **mac_address** – Device’s hardware MAC address
- **middle_name** – Middle name of the user who owns the device
- **name** – Device Hostname
- **organization_id** – Org ID to which the device belongs
- **organization_name** – Name of the org that owns this device
- **os** – Device type
- **os_version** – Version of the OS
- **passive_mode** – Whether the device is in passive mode (bypass?)
- **policy_id** – ID of the policy this device is using
- **policy_name** – Name of the policy this device is using
- **policy_override** – Manually assigned policy (overrides mass sensor management)
- **quarantined** – Whether the device is quarantined
- **registered_time** – When the device was registered with the PSC backend
- **scan_last_action_time** – Not used. Intended for when the background scan was last active
- **scan_last_complete_time** – Not Used. Intended for when the background scan was last completed
- **scan_status** – Not Used. Intended for Background scan status
- **sensor_out_of_date** – Whether the device is out of date
- **sensor_states** – Active sensor states
- **sensor_version** – Version of the PSC sensor
- **status** – Device status
- **target_priority_type** – Priority of the device
- **uninstall_code** – Code to enter to uninstall this device
- **vdi_base_device** – VDI Base device

- **virtual_machine** – Whether this device is a Virtual Machine (VMware AppDefense integration)
- **virtualization_provider** – VM Virtualization Provider
- **windows_platform** – Type of windows platform (client/server, x86/x64)
- **deployment_type** – Classification determined by the device lifecycle management policy

Initialize the Device object.

Parameters

- **cb** ([BaseAPI](#)) – Reference to API object used to communicate with the server.
- **model_unique_id** (*str*) – ID of the alert represented.
- **initial_data** (*dict*) – Initial data used to populate the alert.

activation_code = None

activation_code_expiry_time = None

ad_group_id = None

av_ave_version = None

av_engine = None

av_last_scan_time = None

av_master = None

av_pack_version = None

av_product_version = None

av_status = []

av_update_servers = []

av_vdf_version = None

background_scan (*flag*)

Set the background scan option for this device.

Required Permissions: device.bg-scan (EXECUTE)

Parameters **flag** (*bool*) – True to turn background scan on, False to turn it off.

Returns The JSON output from the request.

Return type str

bypass (*flag*)

Set the bypass option for this device.

Required Permissions: device.bypass (EXECUTE)

Parameters **flag** (*bool*) – True to enable bypass, False to disable it.

Returns The JSON output from the request.

Return type str

current_sensor_policy_name = None

delete_sensor()

Delete this sensor device.

Required Permissions: device.deregistered (DELETE)

Returns The JSON output from the request.

Return type str

deployment_type = None

deregistered_time = None

deviceId

Warn user that Platform Devices use 'id', not 'device_id'.

Platform Device API's return 'id' in API responses, where Endpoint Standard API's return 'deviceId'.

device_id = None

device_meta_data_item_list = []

device_owner_id = None

email = None

encoded_activation_code = None

first_name = None

get_vulnerability_summary(*category=None*)

Get the vulnerabilities associated with this device

Required Permissions: vulnerabilityAssessment.data (READ)

Parameters **category** (*string*) – (optional) vulnerabilty category (OS, APP)

Returns summary for the vulnerabilities for this device

Return type dict

get_vulnerabilities()

Get an Operating System or Application Vulnerability List for a specific device.

Returns vulnerabilities for this device

Return type dict

id = None

last_contact_time = None

last_device_policy_changeded_time = None

last_device_policy_requested_time = None

last_external_ip_address = None

last_internal_ip_address = None

last_location = None

last_name = None

last_policy_updated_time = None

last_reported_time = None

`last_reset_time = None`

`last_shutdown_time = None`

`linux_kernel_version = None`

`login_user_name = None`

`lr_session (async_mode=False)`

Retrieve a Live Response session object for this Device.

Required Permissions: org.liveresponse.session (CREATE)

Returns Live Response session for the Device.

Return type *LiveResponseSession*

Raises *ApiError* – If there is an error establishing a Live Response session for this Device.

`mac_address = None`

`middle_name = None`

`name = None`

`nsx_available`

Returns whether NSX actions are available on this device.

Returns True if NSX actions are available, False if not.

Return type bool

`nsx_remediation (tag, set_tag=True)`

Start an NSX Remediation job on this device to change the tag.

Required Permissions: appliances.nsx.remediation(EXECUTE)

Parameters

- **tag** (*str*) – The NSX tag to apply to this device. Valid values are “CB-NSX-Quarantine”, “CB-NSX-Isolate”, and “CB-NSX-Custom”.
- **set_tag** (*bool*) – True to toggle the specified tag on, False to toggle it off. Default True.

Returns The object representing all running jobs. None if the operation is a no-op.

Return type *NSXRemediationJob*

`organization_id = None`

`organization_name = None`

`os = None`

`os_version = None`

`passive_mode = None`

`policy_id = None`

`policy_name = None`

`policy_override = None`

`primary_key = 'id'`

quarantine (*flag*)

Set the quarantine option for this device.

Required Permissions: device.quarantine (EXECUTE)

Parameters **flag** (*bool*) – True to enable quarantine, False to disable it.

Returns The JSON output from the request.

Return type str

quarantined = None

registered_time = None

scan_last_action_time = None

scan_last_complete_time = None

scan_status = None

sensor_out_of_date = None

sensor_states = []

sensor_version = None

status = None

target_priority_type = None

uninstall_code = None

uninstall_sensor ()

Uninstall this sensor device.

Required Permissions: device.uninstall (EXECUTE)

Returns The JSON output from the request.

Return type str

update_policy (*policy_id*)

Set the current policy for this device.

Required Permissions: device.policy (UPDATE)

Parameters **policy_id** (*int*) – ID of the policy to set for the devices.

Returns The JSON output from the request.

Return type str

update_sensor_version (*sensor_version*)

Update the sensor version for this device.

Required Permissions: org.kits (EXECUTE)

Parameters **sensor_version** (*dict*) – New version properties for the sensor.

Returns The JSON output from the request.

Return type str

urlobject = '/appservices/v6/orgs/{0}/devices'

```
urlobject_single = '/appservices/v6/orgs/{0}/devices/{1}'
```

```
vdi_base_device = None
```

```
virtual_machine = None
```

```
virtualization_provider = None
```

```
vulnerability_refresh()
```

Perform an action on a specific device. Only REFRESH is supported.

Required Permissions: vulnerabilityAssessment.data (EXECUTE)

```
windows_platform = None
```

```
class DeviceFacet (cb, model_unique_id, initial_data=None)
```

Bases: `cbc_sdk.base.UnrefreshableModel`

Represents a device field in a facet search.

Parameters

- **field** – Name of the field being faceted
- **values** – The values of the faceted field.

Initialize the DeviceFacet object.

Parameters

- **cb** (`BaseAPI`) – Reference to API object used to communicate with the server.
- **model_unique_id** (`str`) – Not used.
- **initial_data** (`dict`) – Initial data used to populate the facet.

```
class DeviceFacetValue (cb, outer, model_unique_id, initial_data)
```

Bases: `cbc_sdk.base.UnrefreshableModel`

Represents a value of a particular field.

Initialize the DeviceFacetValue object.

Parameters

- **cb** (`BaseAPI`) – Reference to API object used to communicate with the server.
- **outer** (`DeviceFacet`) – Reference to outer facet object.
- **model_unique_id** (`str`) – Value ID.
- **initial_data** (`dict`) – Initial data used to populate the facet value.

```
query_devices()
```

Set up a device query to find all devices that match this facet value.

Example

```
>>> facets = api.select(Device).where('').facets(['policy_id'])
>>> for value in facets[0].values_:
...     print(f"Policy ID = {value.id}:")
...     for dev in value.query_devices():
...         print(f"        {dev.name} ({dev.last_external_ip_address})")
```

Returns A new DeviceQuery set with the criteria, which may have additional criteria added to it.

Return type DeviceQuery

field = None

primary_key = 'id'

urlobject = '/appservices/v6/orgs/{0}/devices/_facet'

values = []

values_

Return the list of facet values for this facet.

Returns The list of values for this facet.

Return type list[DeviceFacetValue]

class DeviceSearchQuery (*doc_class, cb*)

Bases: *cbc_sdk.base.BaseQuery, cbc_sdk.base.QueryBuilderSupportMixin, cbc_sdk.base.CriteriaBuilderSupportMixin, cbc_sdk.base.IterableQueryMixin, cbc_sdk.base.AsyncQueryMixin*

Represents a query that is used to locate Device objects.

Initialize the DeviceSearchQuery.

Parameters

- **doc_class** (*class*) – The model class that will be returned by this query.
- **cb** (*BaseAPI*) – Reference to API object used to communicate with the server.

VALID_DEPLOYMENT_TYPES = ['ENDPOINT', 'WORKLOAD']

VALID DIRECTIONS = ['ASC', 'DESC']

VALID_FACET_FIELDS = ['policy_id', 'status', 'os', 'ad_group_id', 'cloud_provider_account_id']

VALID_OS = ['WINDOWS', 'ANDROID', 'MAC', 'IOS', 'LINUX', 'OTHER']

VALID_PRIORITIES = ['LOW', 'MEDIUM', 'HIGH', 'MISSION_CRITICAL']

VALID_STATUSES = ['PENDING', 'REGISTERED', 'UNINSTALLED', 'DEREGISTERED', 'ACTIVE', 'INACTIVE']

background_scan (*scan*)

Set the background scan option for the specified devices.

Required Permissions: device.bg-scan (EXECUTE)

Parameters **scan** (*bool*) – True to turn background scan on, False to turn it off.

Returns The JSON output from the request.

Return type str

bypass (*enable*)

Set the bypass option for the specified devices.

Required Permissions: device.bypass (EXECUTE)

Parameters **enable** (*bool*) – True to enable bypass, False to disable it.

Returns The JSON output from the request.

Return type str

delete_sensor()

Delete the specified sensor devices.

Required Permissions: device.deregistered (DELETE)

Returns The JSON output from the request.

Return type str

download()

Uses the query parameters that have been set to download all device listings in CSV format.

Example

```
>>> cb.select(Device).set_status(["ALL"]).download()
```

Required Permissions: device (READ)

Returns The CSV raw data as returned from the server.

Return type str

Raises `ApiError` – If status values have not been set before calling this function.

facets (fieldlist, max_rows=0)

Return information about the facets for all known evices, using the defined criteria.

Example

```
>>> query = api.select(Device).where('')
>>> facets = query.facets(['policy_id', 'status', 'os', 'ad_group_id'])
>>> for f in facets:
...     print(f"Field {f.field} - {len(f.values_)} distinct values")
```

Required Permissions: device (READ)

Parameters

- **fieldlist** (*list[str]*) – List of facet field names. Valid names are “policy_id”, “status”, “os”, “ad_group_id”, “cloud_provider_account_id”, “auto_scaling_group_name”, and “virtual_private_cloud_id”.
- **max_rows** (*int*) – The maximum number of rows to return. 0 means return all rows.

Returns A list of facet information.

Return type list[*DeviceFacet*]

quarantine (enable)

Set the quarantine option for the specified devices.

Required Permissions: device.quarantine (EXECUTE)

Parameters **enable** (*bool*) – True to enable quarantine, False to disable it.

Returns The JSON output from the request.

Return type str

set_ad_group_ids (*ad_group_ids*)

Restricts the devices that this query is performed on to the specified AD group IDs.

Parameters **ad_group_ids** (*list*) – List of AD group IDs to restrict the search to.

Returns This instance.

Return type *DeviceSearchQuery*

Raises *ApiError* – If invalid (non-int) values are passed in the list.

set_auto_scaling_group_name (*group_names*)

Restricts the devices that this query is performed on to the specified auto scaling group names.

Parameters **group_names** (*list*) – List of group names to restrict search to.

Returns This instance.

Return type *DeviceSearchQuery*

set_cloud_provider_account_id (*account_ids*)

Restricts the devices that this query is performed on to the specified cloud provider account IDs.

Parameters **account_ids** (*list*) – List of account IDs to restrict search to.

Returns This instance.

Return type *DeviceSearchQuery*

set_deployment_type (*deployment_type*)

Restricts the devices that this query is performed on to the specified deployment types.

Parameters **deployment_type** (*list*) – List of deployment types to restrict search to.

Returns This instance.

Return type *DeviceSearchQuery*

Raises *ApiError* – If invalid deployment type values are passed in the list.

set_device_ids (*device_ids*)

Restricts the devices that this query is performed on to the specified device IDs.

Parameters **device_ids** (*list*) – List of device IDs to restrict the search to.

Returns This instance.

Return type *DeviceSearchQuery*

Raises *ApiError* – If invalid (non-int) values are passed in the list.

set_exclude_sensor_versions (*sensor_versions*)

Restricts the devices that this query is performed on to exclude specified sensor versions.

Parameters **sensor_versions** (*list*) – List of sensor versions to be excluded.

Returns This instance.

Return type *DeviceSearchQuery*

Raises *ApiError* – If invalid (non-string) values are passed in the list.

set_last_contact_time (**args, **kwargs*)

Restricts the devices that this query is performed on to the specified last contact time.

Parameters

- ***args** (*list*) – Not used, retained for compatibility.
- ****kwargs** (*dict*) – Keyword arguments to this function. The critical ones are “start” (the start time), “end” (the end time), and “range” (the range value).

Returns This instance.

Return type *DeviceSearchQuery*

Raises *ApiError* – If an invalid combination of keyword parameters are specified.

set_max_rows (*max_rows*)

Sets the max number of devices to fetch in a singular query

Parameters **max_rows** (*integer*) – Max number of devices

Returns This instance.

Return type *DeviceSearchQuery*

Raises *ApiError* – If rows is negative or greater than 10000

set_os (*operating_systems*)

Restricts the devices that this query is performed on to the specified operating systems.

Parameters **operating_systems** (*list*) – List of operating systems to restrict search to. Valid values in this list are “WINDOWS”, “ANDROID”, “MAC”, “IOS”, “LINUX”, and “OTHER”.

Returns This instance.

Return type *DeviceSearchQuery*

Raises *ApiError* – If invalid operating system values are passed in the list.

set_policy_ids (*policy_ids*)

Restricts the devices that this query is performed on to the specified policy IDs.

Parameters **policy_ids** (*list*) – List of policy IDs to restrict the search to.

Returns This instance.

Return type *DeviceSearchQuery*

Raises *ApiError* – If invalid (non-int) values are passed in the list.

set_status (*statuses*)

Restricts the devices that this query is performed on to the specified status values.

Parameters **statuses** (*list*) – List of statuses to restrict search to. Valid values in this list are “PENDING”, “REGISTERED”, “UNINSTALLED”, “DEREGISTERED”, “ACTIVE”, “INACTIVE”, “ERROR”, “ALL”, “BYPASS_ON”, “BYPASS”, “QUARANTINE”, “SENSOR_OUTOFDATE”, “DELETED”, and “LIVE”.

Returns This instance.

Return type *DeviceSearchQuery*

Raises *ApiError* – If invalid status values are passed in the list.

set_target_priorities (*target_priorities*)

Restricts the devices that this query is performed on to the specified target priority values.

Parameters **target_priorities** (*list*) – List of priorities to restrict search to. Valid values in this list are “LOW”, “MEDIUM”, “HIGH”, and “MISSION_CRITICAL”.

Returns This instance.

Return type *DeviceSearchQuery*

Raises `ApiError` – If invalid priority values are passed in the list.

set_virtual_private_cloud_id (*cloud_ids*)

Restricts the devices that this query is performed on to the specified virtual private cloud IDs.

Parameters `cloud_ids` (*list*) – List of cloud IDs to restrict search to.

Returns This instance.

Return type *DeviceSearchQuery*

sort_by (*key*, *direction*=`'ASC'`)

Sets the sorting behavior on a query's results.

Example

```
>>> cb.select(Device).sort_by("status")
```

Parameters

- **key** (*str*) – The key in the schema to sort by.
- **direction** (*str*) – The sort order, either “ASC” or “DESC”.

Returns This instance.

Return type *DeviceSearchQuery*

Raises `ApiError` – If an invalid direction value is passed.

uninstall_sensor ()

Uninstall the specified sensor devices.

Required Permissions: `device.uninstall` (EXECUTE)

Returns The JSON output from the request.

Return type `str`

update_policy (*policy_id*)

Set the current policy for the specified devices.

Required Permissions: `device.policy` (UPDATE)

Parameters `policy_id` (*int*) – ID of the policy to set for the devices.

Returns The JSON output from the request.

Return type `str`

update_sensor_version (*sensor_version*)

Update the sensor version for the specified devices.

Required Permissions: `org.kits` (EXECUTE)

Parameters `sensor_version` (*dict*) – New version properties for the sensor.

Returns The JSON output from the request.

Return type `str`

5.6.5 cbc_sdk.platform.events module

Model and Query Classes for Events

class Event (*cb*, *model_unique_id=None*, *initial_data=None*, *force_init=False*, *full_doc=True*)

Bases: *cbc_sdk.base.UnrefreshableModel*

Events can be queried for via *CBCloudAPI.select* or an already selected process with *Process.events()*.

Examples

```
>>> events_query = (api.select(Event).where(process_guid=
    "WNEXFKQ7-00050603-0000066c-00000000-1d6c9acb43e29bb"))
# retrieve results synchronously
>>> events = [event for event in events_query]
# retrieve results asynchronously
>>> future = events_query.execute_async()
>>> events = future.result()
# use an already selected process
>>> process = api.select(Process, "WNEXFKQ7-00050603-0000066c-00000000-
    ↪1d6c9acb43e29bb")
>>> events_query = process.events()
>>> events = [event for event in events_query]
```

Initialize the Event object.

Parameters

- **cb** (*CBCloudAPI*) – A reference to the *CBCloudAPI* object.
- **model_unique_id** (*str*) – The unique ID for this particular instance of the model object.
- **initial_data** (*dict*) – The data to use when initializing the model object.
- **force_init** (*bool*) – True to force object initialization.
- **full_doc** (*bool*) – True to mark the object as fully initialized.

default_sort = 'last_update desc'

primary_key = 'process_guid'

urlobject = '/api/investigate/v2/orgs/{}/events/{}/_search'

validation_url = '/api/investigate/v1/orgs/{}/events/search_validation'

class EventFacet (*cb*, *model_unique_id*, *initial_data*)

Bases: *cbc_sdk.base.UnrefreshableModel*

Represents the results of an *EventFacetQuery*.

EventFacet objects contain both Terms and Ranges. Each of those contain facet fields and values.

Access all of the Terms facet data with *EventFacet.Terms.facets()* or see just the field names with *EventFacet.Terms.fields()*.

Access all of the Ranges facet data with *EventFacet.Ranges.facets()* or see just the field names with *EventFacet.Ranges.fields()*.

Event Facets can be queried for via *CBCloudAPI.select(EventFacet)*. Specify a Process GUID with *‘.where(process_guid=“example_guid”)*, and facet field(s) with *.add_facet_field(“my_facet_field”)*.

Examples

```
>>> event_facet_query = (api.select(EventFacet).where(process_guid=
"WNEXFKQ7-00050603-0000066c-00000000-1d6c9acb43e29bb"))
>>> event_facet_query.add_facet_field("event_type")
# retrieve results synchronously
>>> facet = event_facet_query.results
# retrieve results asynchronously
>>> future = event_facet_query.execute_async()
>>> result = future.result()
# result is a list with one item, so access the first item
>>> facet = result[0]
```

Initialize an EventFacet object with initial_data.

class Ranges (*cb, initial_data*)

Bases: *cbc_sdk.base.UnrefreshableModel*

Represents the range (bucketed) facet fields and values associated with an Event Facet query.

Initialize a ProcessFacet Ranges object with initial_data.

facets

Returns the reified *EventFacet.Terms._facets* for this result.

fields

Returns the ranges fields for this result.

class Terms (*cb, initial_data*)

Bases: *cbc_sdk.base.UnrefreshableModel*

Represents the facet fields and values associated with an Event Facet query.

Initialize a ProcessFacet Terms object with initial_data.

facets

Returns the terms' facets for this result.

fields

Returns the terms facets' fields for this result.

primary_key = 'process_guid'

ranges_

Returns the reified *EventFacet.Ranges* for this result.

terms_

Returns the reified *EventFacet.Terms* for this result.

urlobject = '/api/investigate/v2/orgs/{}/events/{}/_facet'

class EventFacetQuery (*cls, cb, query=None*)

Bases: *cbc_sdk.base.FacetQuery*

Represents the logic for an Event Facet query.

Initialize the FacetQuery object.

class EventQuery (*doc_class, cb*)

Bases: *cbc_sdk.base.Query*

Represents the logic for an Event query.

Initialize the Query object.

Parameters

- **doc_class** (*class*) – The class of the model this query returns.
- **cb** ([CBCloudAPI](#)) – A reference to the CBCloudAPI object.

5.6.6 cbc_sdk.platform.grants module

Model and Query Classes for Administrative Grants and Profiles

class Grant (*cb, model_unique_id, initial_data=None*)

Bases: [cbc_sdk.base.MutableBaseModel](#)

Represents a grant of access to the Carbon Black Cloud.

Parameters

- **principal** – URN of principal
- **expires** – Date and time the grant expires
- **roles** – URNs of roles assigned to grant (obsolete)
- **profiles** – Profiles assigned to this grant
- **org_ref** – URN of org that this grant references
- **principal_name** – Name of principal
- **created_by** – URN of user that created this grant
- **updated_by** – URN of user that last updated this grant
- **create_time** – Date and time the grant was created
- **update_time** – Date and time the grant was last updated
- **can_manage** – True if can manage (TBD)

Initialize the Grant object.

Parameters

- **cb** ([BaseAPI](#)) – Reference to API object used to communicate with the server.
- **model_unique_id** (*str*) – URN of the principal associated with this grant.
- **initial_data** (*dict*) – Initial data used to populate the grant.

class GrantBuilder (*cb, principal*)

Bases: `object`

Auxiliary object used to construct a new grant.

Creates the empty GrantBuilder object.

Parameters

- **cb** ([CBCloudAPI](#)) – The reference to the API object that accesses the server.
- **principal** (*str*) – The URN for the principal.

add_role (*role*)

Adds a role to be associated with the new grant.

Parameters **role** (*str*) – URN of the role to be added.

Returns This object.

Return type [GrantBuilder](#)

build()

Builds the new Grant object from the entered data.

Returns The new Grant object.

Return type *Grant*

create_profile (*template=None*)

Returns either a new Profile, or a ProfileBuilder to begin the process of adding profile to the new grant.

Parameters **template** (*dict*) – Optional template to use for creating the profile object.

Returns

If a template was specified, return the new Profile object.

ProfileBuilder: If template was None, returns a ProfileBuilder object. Call methods on it to set up the new profile, and then call build() to create the new profile.

Return type *Profile*

set_org (*org*)

Sets the organization reference to be associated with the new grant.

Parameters **org** (*str*) – Organization key or URN of the organization.

Returns This object.

Return type *GrantBuilder*

set_principal_name (*name*)

Sets the principal name to be associated with the new object.

Parameters **name** (*str*) – Principal name to be used.

Returns This object.

Return type *GrantBuilder*

set_roles (*roles*)

Sets the roles to be associated with the new grant.

Parameters **roles** (*list*) – List of role URNs.

Returns This object.

Return type *GrantBuilder*

class Profile (*cb, grant, model_unique_id, initial_data=None*)

Bases: *cbc_sdk.base.MutableBaseModel*

Represents an access profile assigned to a grant.

Parameters

- **profile_uuid** – UUID identifying this profile
- **orgs** – Organization references for this profile
- **org_groups** – Organization groups added to this grant (TBD)
- **roles** – URNs of roles assigned to profile
- **conditions** – Access conditions to be imposed on this profile
- **can_manage** – True if can manage (TBD)

Initialize the Profile object.

Parameters

- **cb** (*BaseAPI*) – Reference to API object used to communicate with the server.
- **grant** (*Grant*) – Reference to the Grant containing this Profile.
- **model_unique_id** (*str*) – UUID of this profile.
- **initial_data** (*dict*) – Initial data used to populate the profile.

allowed_orgs
Returns the list of organization URNs allowed by this profile.

can_manage = None

conditions = {}

matches_template (*template*)
Returns whether or not the profile matches the given template.
Parameters **template** (*dict*) – The profile template to match against.
Returns True if this profile matches the template, False if not.
Return type bool

org_groups = []

orgs = {}

primary_key = 'profile_uuid'

profile_uuid = None

roles = []

set_disabled (*flag*)
Sets the “disabled” flag on a profile.
Parameters **flag** (*bool*) – True to disable the profile, False to enable it.

set_expiration (*expiration*)
Sets the expiration time on a profile.
Parameters **expiration** (*str*) – Expiration time to set on the profile (ISO 8601 format).

urlobject = '/access/v2/orgs/{0}/grants/{1}/profiles'

urlobject_single = '/access/v2/orgs/{0}/grants/{1}/profiles/{2}'

class ProfileBuilder (*grant*)
Bases: object
Auxiliary object used to construct a new profile on a grant.
Create the empty ProfileBuilder object.
Parameters **grant** (*Grant/GrantBuilder*) – The grant or GrantBuilder the new profile will be attached to.

add_org (*org*)
Adds the specified organization to the list of organizations for which the new profile is allowed.
Parameters **org** (*str*) – Organization key or URN of the organization to be added.
Returns This object.
Return type *ProfileBuilder*

add_role (*role*)
Adds a role identifier to the list of roles associated with the new profile.
Parameters **role** (*str*) – URN of the role to add.
Returns This object.
Return type *ProfileBuilder*

build ()
Builds the new Profile object from the entered data.
Returns The new Profile object.
Return type *Profile*

set_conditions (*conditions_structure*)
Sets the access conditions associated with the new profile.

Parameters **conditions_structure** (*dict*) – The conditions associated with the new profile, with 'cidr', 'expiration', and 'disabled' members.

Returns This object.

Return type *ProfileBuilder*

set_disabled (*flag*)

Sets whether or not the new profile is disabled.

Parameters **flag** (*bool*) – True if this profile is disabled, False if not.

Returns This object.

Return type *ProfileBuilder*

set_expiration (*expiration*)

Sets the expiration time on the new profile.

Parameters **expiration** (*str*) – The expiration time, specified as ISO 8601.

Returns This object.

Return type *ProfileBuilder*

set_orgs (*orgs_list*)

Set the list of organizations to which the new profile is allowed access.

Parameters **orgs_list** (*list*) – List of organization keys or URNs.

Returns This object.

Return type *ProfileBuilder*

set_roles (*roles_list*)

Sets the list of roles associated with the new profile.

Parameters **roles_list** (*list*) – A list of role URNs.

Returns This object.

Return type *ProfileBuilder*

can_manage = None

classmethod **create** (*cb, template=None, **kwargs*)

Returns either a new Grant, or a GrantBuilder to begin the process of creating a new grant.

Parameters

- **cb** (*CBCloudAPI*) – A reference to the CBCloudAPI object.
- **template** (*dict*) – Optional template to use for creating the grant object.
- **kwargs** (*dict*) – Additional arguments to be used to specify the principal, if template is None.
- **arguments to be used are 'org_key' and 'userid' for the two parts of the ID. (The)** –

Returns

The new grant object, if the template is specified.

GrantBuilder: If template was None, returns a GrantBuilder object. Call methods on it to set up the new grant, and then call build() to create the new grant.

Return type *Grant*

Raises *ApiError* – If the principal is inadequately specified (whether for the Grant or GrantBuilder).

create_profile (*template=None*)

Returns either a new Profile, or a ProfileBuilder to begin the process of adding a new profile to this grant.

Parameters **template** (*dict*) – Optional template to use for creating the profile object.

Returns

If a template was specified, return the new Profile object.

ProfileBuilder: If template was None, returns a ProfileBuilder object. Call methods on it to set up the new profile, and then call build() to create the new profile.

Return type *Profile*

create_time = None

created_by = None

expires = None

classmethod get_permitted_role_urns (*cb*)

Returns a list of the URNs of all permitted roles that we can assign to a user.

Parameters **cb** (*CBCloudAPI*) – A reference to the CBCloudAPI object.

Returns A list of string role URNs that we are permitted to manage (assign to users).

Return type list

org_ref = None

primary_key = 'principal'

principal = None

principal_name = None

profiles = []

profiles_

Return the profiles associated with this grant.

Returns The profiles associated with this grant, each represented as a Profile object.

Return type list

roles = []

update_time = None

updated_by = None

urlobject = '/access/v2/orgs/{0}/grants'

urlobject_single = '/access/v2/orgs/{0}/grants/{1}'

class GrantQuery (*doc_class, cb*)

Bases: *cbc_sdk.base.BaseQuery*, *cbc_sdk.base.IterableQueryMixin*, *cbc_sdk.base.AsyncQueryMixin*

Query for retrieving grants in bulk.

Initialize the Query object.

Parameters

- **doc_class** (*class*) – The class of the model this query returns.
- **cb** (*CBCloudAPI*) – A reference to the CBCloudAPI object.

add_principal (*principal_urn, org_urn*)

Add a new principal to the query.

Parameters

- **principal_urn** (*str*) – URN of the principal to search for grants on.
- **org_urn** (*str*) – URN of the organization to which the principal belongs.

Returns This object.

Return type *GrantQuery*

log = <Logger **cbc_sdk.platform.grants** (**WARNING**)>
Grant and Profile Models

normalize_org (*org*)
Internal function to normalize an org reference to a URN.

5.6.7 cbc_sdk.platform.jobs module

Model and Query Classes for Jobs API

class Job (*cb, model_unique_id, initial_data=None*)
Bases: *cbc_sdk.base.NewBaseModel*

Represents a job currently executing in the background.

Parameters

- **connector_id** – Connector ID for the job
- **create_time** – Time this job was created
- **errors** – Errors for the job
- **id** – ID of the job
- **job_parameters** – Parameters that were used for this job
- **last_update_time** – Last time this job was updated
- **org_key** – Organization key of the org this job is being run against
- **owner_id** – ID of the job owner
- **status** – Current job status
- **type** – Type of job this is

Initialize the Job object.

Parameters

- **cb** (*BaseAPI*) – Reference to API object used to communicate with the server.
- **model_unique_id** (*int*) – ID of the job.
- **initial_data** (*dict*) – Initial data used to populate the job.

await_completion ()

Create a Python Future to check for job completion and return results when available.

Returns a Future object which can be used to await results that are ready to fetch. This function call does not block.

Required Permissions: jobs.status(READ)

Returns

A future which can be used to wait for this job's completion. When complete, the result of the Future will be this object.

Return type Future

connector_id = None

create_time = None

errors = None

get_output_as_file (*filename*)

Export the results from the job, writing the results to the given file.

Required Permissions: jobs.status(READ)

Parameters **filename** (*str*) – Name of the file to write the results to.

get_output_as_lines ()

Export the results from the job, returning the data as iterated lines of text.

This is only intended for output that can reasonably be represented as lines of text, such as plain text or CSV. If a job outputs structured text like JSON or XML, this method should not be used.

Required Permissions: jobs.status(READ)

Returns An iterable that can be used to get each line of text in turn as a string.

Return type iterable

get_output_as_stream (*output*)

Export the results from the job, writing the results to the given stream.

Required Permissions: jobs.status(READ)

Parameters **output** (*RawIOBase*) – Stream to write the CSV data from the request to.

get_output_as_string ()

Export the results from the job, returning the results as a string.

Required Permissions: jobs.status(READ)

Returns The results from the job.

Return type str

get_progress ()

Get and return the current progress information for the job.

Required Permissions: jobs.status(READ)

Returns Total number of items to be operated on by this job. int: Total number of items for which operation has been completed. str: Current status message for the job.

Return type int

id = None

job_parameters = {}

last_update_time = None

org_key = None

owner_id = None

```

primary_key = 'id'
progress = {}
status = None
type = None
urlobject = '/jobs/v1/orgs/{0}/jobs'
urlobject_single = '/jobs/v1/orgs/{0}/jobs/{1}'

```

class JobQuery (*doc_class, cb*)

Bases: *cbc_sdk.base.BaseQuery, cbc_sdk.base.IterableQueryMixin, cbc_sdk.base.AsyncQueryMixin*

Query for retrieving current jobs.

Initialize the Query object.

Parameters

- **doc_class** (*class*) – The class of the model this query returns.
- **cb** (*CBCloudAPI*) – A reference to the CBCloudAPI object.

5.6.8 cbc_sdk.platform.network_threat_metadata module

Model Class for NetworkThreatMetadata

```

class NetworkThreatMetadata (cb, model_unique_id=None, initial_data=None, force_init=False, full_doc=True)

```

Bases: *cbc_sdk.base.NewBaseModel*

Represents a NetworkThreatMetadata

Parameters

- **detector_abstract** – Abstract or description of the detector
- **detector_goal** – Description of what the detector is achieving
- **false_negatives** – Highlights why detector could not have been triggered
- **false_positives** – Highlights why detector could have been triggered
- **threat_public_comment** – Public comment of the threat

Initialize the NetworkThreatMetadata object.

Required Permissions: org.xdr.metadata (READ)

Parameters

- **cb** (*CBCloudAPI*) – A reference to the CBCloudAPI object.
- **model_unique_id** (*Any*) – The unique ID for this particular instance of the model object.
- **initial_data** (*dict*) – Not used, retained for compatibility.
- **force_init** (*bool*) – False to not force object initialization.
- **full_doc** (*bool*) – True to mark the object as fully initialized.

Raises *ApiError* – if model_unique_id is not provided

```
detector_abstract = None
detector_goal = None
false_negatives = None
false_positives = None
primary_key = 'tms_rule_id'
threat_public_comment = None
urlobject = '/threatmetadata/v1/orgs/{0}/detectors/{1}'
```

5.6.9 cbc_sdk.platform.observations module

Model and Query Classes for Observations

class **Observation** (*cb, model_unique_id=None, initial_data=None, force_init=False, full_doc=False*)
Bases: *cbc_sdk.base.NewBaseModel*

Represents an Observation

Initialize the Observation object.

Required Permissions: org.search.events (READ)

Parameters

- **cb** (*CBCloudAPI*) – A reference to the CBCloudAPI object.
- **model_unique_id** (*Any*) – The unique ID for this particular instance of the model object.
- **initial_data** (*dict*) – The data to use when initializing the model object.
- **force_init** (*bool*) – True to force object initialization.
- **full_doc** (*bool*) – False to mark the object as not fully initialized.

```
alert_category = []
```

```
alert_id = []
```

```
backend_timestamp = None
```

static bulk_get_details (*cb, alert_id=None, observation_ids=None, timeout=0*)
Bulk get details

Parameters

- **cb** (*CBCloudAPI*) – A reference to the CBCloudAPI object.
- **alert_id** (*str*) – An alert id to fetch associated observations
- **observation_ids** (*list*) – A list of observation ids to fetch
- **timeout** (*int*) – Observations details request timeout in milliseconds.

Returns list of Observations

Return type list

Raises *ApiError* – if cb is not instance of CBCloudAPI

```
device_group_id = None
```

```
device_id = None
device_name = None
device_policy = None
device_policy_id = None
device_timestamp = None
enriched = None
enriched_event_type = None
event_description = None
event_id = None
event_network_inbound = None
event_network_local_ipv4 = None
event_network_location = None
event_network_protocol = None
event_network_remote_ipv4 = None
event_network_remote_port = None
event_type = []
get_details (timeout=0, async_mode=False)
    Requests detailed results.
```

Parameters

- **timeout** (*int*) – Observations details request timeout in milliseconds.
- **async_mode** (*bool*) – True to request details in an asynchronous manner.

Returns Observation object enriched with the details fields

Return type *Observation*

Note:

- When using asynchronous mode, this method returns a python future. You can call result() on the future object to wait for completion and get the results.
-

Examples

```
>>> observation = api.select(Observation, observation_id)
>>> observation.get_details()
```

```
>>> observations = api.select(Observation.where(process_pid=2000)
>>> observations[0].get_details()
```

get_network_threat_metadata()

Requests Network Threat Metadata.

Returns Get the metadata for a given detector (rule).

Return type *NetworkThreatMetadata*

Raises `ApiError` – when `rule_id` is not returned for the Observation

Examples

```
>>> observation = api.select(Observation, observation_id)
>>> threat_metadata = observation.get_network_threat_metadata()
```

`ingress_time = None`

`legacy = None`

`observation_description = None`

`observation_id = None`

`observation_type = None`

`org_id = None`

`parent_guid = None`

`parent_pid = None`

`primary_key = 'observation_id'`

`process_guid = None`

`process_hash = []`

`process_name = None`

`process_pid = []`

`process_username = []`

`rule_id = None`

static `search_suggestions` (*cb, query, count=None*)

Returns suggestions for keys and field values that can be used in a search.

Parameters

- **cb** (`CBCloudAPI`) – A reference to the CBCloudAPI object.
- **query** (*str*) – A search query to use.
- **count** (*int*) – (optional) Number of suggestions to be returned

Returns A list of search suggestions expressed as dict objects.

Return type list

Raises `ApiError` – if `cb` is not instance of `CBCloudAPI`

`validation_url = '/api/investigate/v2/orgs/{}/observations/search_validation'`

class `ObservationFacet` (*cb, model_unique_id, initial_data*)

Bases: `cbc_sdk.base.UnrefreshableModel`

Represents an observation facet retrieved.

Parameters

- **terms** – Contains the Observations Facet search results
- **ranges** – Groupings for search result properties that are ISO 8601 timestamps or numbers

- **contacted** – The number of searchers contacted for this query
- **completed** – The number of searchers that have reported their results

Initialize the Terms object with initial data.

class Ranges (*cb, initial_data*)

Bases: *cbc_sdk.base.UnrefreshableModel*

Represents the range (bucketed) facet fields and values associated with an Observation Facet query.

Initialize an ObservationFacet Ranges object with initial_data.

facets

Returns the reified *ObservationFacet.Terms._facets* for this result.

fields

Returns the ranges fields for this result.

class Terms (*cb, initial_data*)

Bases: *cbc_sdk.base.UnrefreshableModel*

Represents the facet fields and values associated with an Observation Facet query.

Initialize an ObservationFacet Terms object with initial_data.

facets

Returns the terms' facets for this result.

fields

Returns the terms facets' fields for this result.

completed = None

contacted = None

num_found = None

primary_key = 'job_id'

ranges = []

ranges_

Returns the reified *ObservationFacet.Ranges* for this result.

result_url = '/api/investigate/v2/orgs/{}/observations/facet_jobs/{}/results'

submit_url = '/api/investigate/v2/orgs/{}/observations/facet_jobs'

terms = []

terms_

Returns the reified *ObservationFacet.Terms* for this result.

class ObservationGroup (*cb, initial_data=None*)

Bases: *object*

Represents ObservationGroup

Initialize ObservationGroup object

Parameters

- **cb** (*CBCloudAPI*) – A reference to the CBCloudAPI object.
- **initial_data** (*dict*) – The data to use when initializing the model object.

Notes

The constructed object will have the following data: - group_start_timestamp - group_end_timestamp - group_key - group_value

class ObservationQuery (*doc_class*, *cb*)

Bases: *cbc_sdk.base.Query*

Represents the query logic for an Observation query.

This class specializes *Query* to handle the particulars of observations querying.

Initialize the ObservationQuery object.

Parameters

- **doc_class** (*class*) – The class of the model this query returns.
- **cb** (*CBCloudAPI*) – A reference to the CBCloudAPI object.

VALID_GROUP_FIELDS = ['observation_type', 'device_name', 'process_username', 'attack_t.

get_group_results (*fields*, *max_events_per_group=None*, *rows=500*, *start=None*,
range_duration=None, *range_field=None*, *range_method=None*)

Get group results grouped by provided fields.

Parameters

- **fields** (*str / list*) – field or fields by which to perform the grouping
- **max_events_per_group** (*int*) – Maximum number of events in a group, if not provided, all events will be returned
- **rows** (*int*) – Number of rows to request, can be paginated
- **start** (*int*) – First row to use for pagination
- **ranges** (*dict*) – dict with information about duration, field, method

Returns grouped results

Return type dict

Examples

```
>>> for group in api.select(Observation).where(process_pid=2000).get_group_  
↪ results("device_name"):  
>>>     ...
```

or_ (***kwargs*)

or_() criteria are explicitly provided to Observation queries.

This method overrides the base class in order to provide *or_()* functionality rather than raising an exception.

set_rows (*rows*)

Sets the ‘rows’ query body parameter to the ‘start search’ API call, determining how many rows to request.

Parameters **rows** (*int*) – How many rows to request.

Returns ObservationQuery object

Return type *Query*

Example

```
>>> cb.select(Observation).where(process_name="foo.exe").set_rows(50)
```

timeout (*msecs*)

Sets the timeout on a observation query.

Parameters *msecs* (*int*) – Timeout duration, in milliseconds.

Returns

The Query object with new **milliseconds** parameter.

Return type *Query* (*ObservationQuery*)

Example

```
>>> cb.select(Observation).where(process_name="foo.exe").timeout(5000)
```

5.6.10 cbc_sdk.platform.policies module

Policy implementation as part of Platform API

class Policy (*cb*, *model_unique_id=None*, *initial_data=None*, *force_init=False*, *full_doc=False*)

Bases: *cbc_sdk.base.MutableBaseModel*

Represents a policy within the organization.

Create one of these objects (either directly or with the `CBCloudAPI.create()` method) and set its properties, then call its `save()` method to create the policy on the server. This requires the `org.policies(CREATE)` permission.

Alternatively, you may call `Policy.create()` to get a `PolicyBuilder`, use its methods to set the properties of the new policy, call its `build()` method to build the populated `Policy`, then call the policy `save()` method.

To update a `Policy`, change the values of its property fields, then call the policy's `save()` method. This requires the `org.policies(UPDATE)` permission.

To delete an existing `Policy`, call its `delete()` method. This requires the `org.policies(DELETE)` permission.

For information on values for policy settings including enumeration values, see the Policy Service API page: <https://developer.carbonblack.com/reference/carbon-black-cloud/platform/latest/policy-service/#fields>

Parameters

- **id** – The policy identifier
- **name** – Defined name for the policy
- **org_key** – The organization key associated with the console instance
- **priority_level** – The priority level designated for policy
- **is_system** – Indicates that the policy was created by VMware
- **description** – The description of the policy

- **auto_deregister_inactive_vdi_interval_ms** – The time in milliseconds to wait after a VDI is inactive before setting the VDI to a DEREGISTERED state
- **auto_delete_known_bad_hashes_delay** – Enables the Carbon Black Cloud to automatically delete known malware after a specified time in milliseconds
- **av_settings** – Anti-Virus settings for endpoints and workloads assigned to the policy
- **rules** – Permission or prevention rules
- **directory_action_rules** – Rules to deny or allow the deployed sensors to send uploads from specific paths
- **sensor_settings** – Settings to configure sensor behavior and capabilities
- **managed_detection_response_permissions** – Permissions for Managed Detection and Response analysts to perform remediations on endpoints and workloads assigned to the policy
- **version** – Version of the policy

Initialize the Policy object.

Required Permissions: org.policies (READ)

Parameters

- **cb** ([BaseAPI](#)) – Reference to API object used to communicate with the server.
- **model_unique_id** (*int*) – ID of the policy.
- **initial_data** (*dict*) – Initial data used to populate the policy.
- **force_init** (*bool*) – If True, forces the object to be refreshed after constructing. Default False.
- **full_doc** (*bool*) – If True, object is considered “fully” initialized. Default False.

class PolicyBuilder (*cb*)

Bases: `object`

Builder object to simplify the creation of new Policy objects.

To use, call `Policy.create()` to get a `PolicyBuilder`, use its methods to set the properties of the new policy, call its `build()` method to build the populated `Policy`, then call the `policy save()` method. The `org.policy(CREATE)` permission is required.

Examples

```
>>> builder = Policy.create(api)
>>> builder.set_name("New Policy").set_priority("MEDIUM").set_description(
    ↪ "New policy description")
>>> # more calls here to set up rules, sensor settings, etc.
>>> policy = builder.build()
>>> policy.save()
```

Initialize the `PolicyBuilder` object.

Parameters **cb** ([BaseAPI](#)) – Reference to API object used to communicate with the server.

add_directory_action_rule (*path, file_upload, protection*)

Add a directory action rule to the new policy.

Parameters

- **path** (*str*) – Path to the file or directory.
- **file_upload** (*bool*) – True to allow the deployed sensor to upload from that path.
- **protection** (*bool*) – True to deny the deployed sensor to upload from that path.

Returns This object.

Return type *PolicyBuilder*

add_rule (*app_type, app_value, operation, action, required=True*)

Add a new rule as discrete data elements to the new policy.

Parameters

- **app_type** (*str*) – Specifies “NAME_PATH”, “SIGNED_BY”, or “REPUTATION”.
- **app_value** (*str*) – Value of the attribute specified by *app_type* to be matched.
- **operation** (*str*) – The type of behavior the application is performing.
- **action** (*str*) – The action the sensor will take when the application performs the specified action.
- **required** (*bool*) – True if this rule is required, False if not.

Returns This object.

Return type *PolicyBuilder*

Raises *InvalidObjectError* – If the rule data passed in is not valid.

add_rule_config (*config_id, name, category, **kwargs*)

Add a new rule configuration as discrete data elements to the new policy.

Parameters

- **config_id** (*str*) – ID of the rule configuration object (a GUID).
- **name** (*str*) – Name of the rule configuration object.
- **category** (*str*) – Category of the rule configuration object.
- ****kwargs** (*dict*) – Parameter values for the rule configuration object.

Returns This object.

Return type *PolicyBuilder*

Raises *InvalidObjectError* – If the rule configuration data passed in is not valid.

add_rule_config_copy (*rule_config*)

Adds a copy of an existing rule configuration to this new policy.

Parameters **rule_config** (*PolicyRuleConfig*) – The rule configuration to copy and add to this object.

Returns This object.

Return type *PolicyBuilder*

Raises *InvalidObjectError* – If the rule configuration data passed in is not valid.

add_rule_copy (*rule*)

Adds a copy of an existing rule to this new policy.

Parameters **rule** (*PolicyRule*) – The rule to copy and add to this object.

Returns This object.

Return type *PolicyBuilder*

Raises *InvalidObjectError* – If the rule data passed in is not valid.

add_sensor_setting (*name, value*)

Add a sensor setting to the policy.

Parameters

- **name** (*str*) – Sensor setting name.
- **value** (*str*) – Sensor setting value.

Returns This object.

Return type *PolicyBuilder*

Raises *ApiError* – If the sensor setting name is not a valid one.

build ()

Build a new Policy object using the contents of this builder.

The new policy must have *save()* called on it to be saved to the server.

Returns The new Policy object.

Return type *Policy*

set_auto_delete_bad_hash_delay (*delay*)

Set the delay in milliseconds after which known malware will be deleted.

Parameters **delay** (*int*) – The desired delay interval in milliseconds.

Returns This object.

Return type *PolicyBuilder*

set_auto_deregister_interval (*interval*)

Set the time in milliseconds after a VDI goes inactive to deregister it.

Parameters **interval** (*int*) – The desired interval in milliseconds.

Returns This object.

Return type *PolicyBuilder*

set_avira_protection_cloud (*enabled*, *max_exe_delay=None*, *max_file_size=None*,
risk_level=None)

Set the settings for third-party unknown binary reputation analysis.

Parameters

- **enabled** (*bool*) – True to enable unknown binary reputation analysis.
- **max_exe_delay** (*int*) – Time before sending unknown binary for analysis, in seconds.
- **max_file_size** (*int*) – Maximum size of file to send for analysis, in megabytes.
- **risk_level** (*int*) – Risk level to send for analysis (0-7).

Returns This object.

Return type *PolicyBuilder*

set_description (*descr*)

Set the new policy description.

Parameters **descr** (*str*) – The new policy description.

Returns This object.

Return type *PolicyBuilder*

set_managed_detection_response_permissions (*policy_mod*, *quarantine*)

Set the permissions for managed detection and response.

Parameters

- **policy_mod** (*bool*) – True to allow MDR team to modify the policy.
- **quarantine** (*bool*) – True to allow MDR team to quarantine endpoints/workloads associated with the policy.

Returns This object.

Return type *PolicyBuilder*

set_name (*name*)

Set the new policy name.

Parameters **name** (*str*) – The new policy name.

Returns This object.

Return type *PolicyBuilder*

set_on_access_scan (*enabled*, *mode='NORMAL'*)

Sets the local scan settings.

Parameters

- **enabled** (*bool*) – True to enable local scan.
- **mode** (*str*) – The mode to operate in, either “NORMAL” or “AGGRESSIVE”.

Returns This object.

Return type *PolicyBuilder*

Raises *ApiError* – If an invalid value is passed for the “mode” parameter.

set_on_demand_scan (*enabled*, *profile*='NORMAL', *scan_usb*='AUTOSCAN',
scan_cd_dvd='AUTOSCAN')

Sets the background scan settings.

Parameters

- **enabled** (*bool*) – True to enable background scan.
- **profile** (*str*) – The background scan mode, either “NORMAL” or “AGGRESSIVE”.
- **scan_usb** (*str*) – Either “AUTOSCAN” to scan USB devices, or “DISABLED” to not do so.
- **scan_cd_dvd** (*str*) – Either “AUTOSCAN” to scan CDs and DVDs, or “DISABLED” to not do so.

Returns This object.

Return type *PolicyBuilder*

Raises *ApiError* – If an invalid value is passed for any parameter.

set_on_demand_scan_schedule (*days*, *start_hour*, *range_hours*, *recover_if_missed*=True)

Sets the schedule for when background scans will be performed.

Parameters

- **days** (*list[str]*) – The days on which to perform background scans.
- **start_hour** (*int*) – The hour of the day at which to perform the scans.
- **range_hours** (*int*) – The range of hours over which to perform the scans.
- **recover_if_missed** (*bool*) – True if the background scan should be performed ASAP if it’s been missed.

Returns This object.

Return type *PolicyBuilder*

Raises *ApiError* – If an invalid value is passed for a day of the week.

set_priority (*priority*)

Set the new policy’s priority. Default is MEDIUM.

Parameters **priority** (*str*) – The priority, either “LOW”, “MEDIUM”, “HIGH”, or “MISSION_CRITICAL”.

Returns This object.

Return type *PolicyBuilder*

Raises *ApiError* – If an invalid priority value is passed in.

set_signature_update (*enabled*)

Set the enable status for signature updates.

Parameters **enabled** (*bool*) – True to enable signature updates.

Returns This object.

Return type *PolicyBuilder*

set_signature_update_schedule (*full_interval_hours*, *initial_random_delay_hours*, *interval_hours*)

Set the signature update schedule.

Parameters

- **full_interval_hours** (*int*) – The interval in hours between signature updates.
- **initial_random_delay_hours** (*int*) – The initial delay in hours before the first signature update.
- **interval_hours** (*int*) – The interval in hours between signature updates.

Returns This object.

Return type *PolicyBuilder*

set_update_servers_offsite (*names*)

Sets the list of update servers for offsite devices.

Parameters **names** (*list[str]*) – The list of update servers, as URIs.

Returns This object.

Return type *PolicyBuilder*

set_update_servers_onsite (*names, preferred_servers=None*)

Sets the list of update servers for internal devices.

Parameters

- **names** (*list[str]*) – The list of available update servers, as URIs.
- **preferred_servers** (*list[str]*) – The list of update servers to be considered “preferred,” as URIs.

Returns This object.

Return type *PolicyBuilder*

set_update_servers_override (*names*)

Sets the list of update servers to override offsite/onsite settings.

Parameters **names** (*list[str]*) – The server names to use, as a list of URIs.

Returns This object.

Return type *PolicyBuilder*

VALID_DAYS = ['MONDAY', 'TUESDAY', 'WEDNESDAY', 'THURSDAY', 'FRIDAY', 'SATURDAY', 'SUNDAY']

VALID_PRIORITIES = ['LOW', 'MEDIUM', 'HIGH', 'MISSION_CRITICAL']

VALID_SCAN_MODES = ['NORMAL', 'AGGRESSIVE']

VALID_SCAN_OPTIONS = ['AUTOSCAN', 'DISABLED']

VALID_SENSOR_SETTINGS = ['SHOW_UI', 'ALLOW_UNINSTALL', 'ALLOW_UPLOAD', 'QUARANTINE_DEV']

add_rule (*new_rule*)

Adds a rule to this Policy.

Parameters **new_rule** (*dict(str, str)*) – The new rule to add to this Policy.

Notes

- The new rule must conform to this dictionary format:

```
{“action”: “ACTION”, “application”: {“type”: “TYPE”, “value”: “VALUE”}, “operation”:  
“OPERATION”, “required”: “REQUIRED”}
```

- The dictionary keys have these possible values:

“action”: [“IGNORE”, “ALLOW”, “DENY”, “TERMINATE_PROCESS”, “TERMINATE_THREAD”, “TERMINATE”]

“type”: [“NAME_PATH”, “SIGNED_BY”, “REPUTATION”]

“value”: Any string value to match on

“operation”: [“BYPASS_ALL”, “INVOKE_SCRIPT”, “INVOKE_SYSAPP”, “POL_INVOKE_NOT_TRUSTED”, “INVOKE_CMD_INTERPRETER”, “RANSOM”, “NETWORK”, “PROCESS_ISOLATION”, “CODE_INJECTION”, “MEMORY_SCRAPE”, “RUN_INMEMORY_CODE”, “ESCALATE”, “RUN”]

“required”: [True, False]

auto_delete_known_bad_hashes_delay = None

auto_deregister_inactive_vdi_interval_ms = None

av_settings = {}

core_prevention_rule_configs

Returns a dictionary of core prevention rule configuration IDs and objects for this Policy.

Returns

A dictionary with core prevention rule configuration IDs as keys and CorePreventionRuleConfig objects as values.

Return type dict

core_prevention_rule_configs_list

Returns a list of core prevention rule configuration objects for this Policy.

Returns A list of CorePreventionRuleConfig objects.

Return type list

classmethod create (*cb*)

Begins creating a policy by returning a PolicyBuilder.

Parameters *cb* (*BaseAPI*) – Reference to API object used to communicate with the server.

Returns The new policy builder object.

Return type *PolicyBuilder*

delete_rule (*rule_id*)

Deletes a rule from this Policy.

Parameters *rule_id* (*int*) – The ID of the rule to be deleted.

Raises *ApiError* – If the rule ID does not exist in this policy.

delete_rule_config (*rule_config_id*)

Deletes a rule configuration from this Policy.

Parameters *rule_config_id* (*str*) – The ID of the rule configuration to be deleted.

Raises *ApiError* – If the rule configuration ID does not exist in this policy.

description = None

directory_action_rules = []

get_ruleconfig_parameter_schema (*ruleconfig_id*)

Returns the parameter schema for a specified rule configuration.

Uses cached rule configuration presentation data if present.

Parameters *ruleconfig_id* (*str*) – The rule configuration ID (UUID).

Returns The parameter schema for this particular rule configuration (a JSON schema).

Return type dict

Raises *InvalidObjectError* – If the rule configuration ID is not valid.

id = None

is_system = None

latestRevision

Returns the latest revision of this policy (compatibility method).

managed_detection_response_permissions = {}

name = None

object_rule_configs

Returns a dictionary of rule configuration IDs and objects for this Policy.

Returns A dictionary with rule configuration IDs as keys and PolicyRuleConfig objects as values.

Return type dict

object_rule_configs_list

Returns a list of rule configuration objects for this Policy.

Returns A list of PolicyRuleConfig objects.

Return type list

object_rules

Returns a dictionary of rule objects and rule IDs for this Policy.

Returns A dictionary with rule IDs as keys and PolicyRule objects as values.

Return type dict

org_key = None

policy

Returns the contents of this policy [compatibility method].

position = None

primary_key = 'id'

priorityLevel

Returns the priority level of this policy (compatibility method).

priority_level = None

replace_rule (*rule_id*, *new_rule*)

Replaces a rule in this policy.

Parameters

- **rule_id** (*int*) – The ID of the rule to be replaced.
- **new_rule** (*dict*) – The data for the new rule.

Raises `ApiError` – If the rule ID does not exist in this policy.

replace_rule_config (*rule_config_id*, *new_rule_config*)

Replaces a rule configuration in this policy.

Parameters

- **rule_config_id** (*str*) – The ID of the rule configuration to be replaced.
- **new_rule_config** (*dict*) – The data for the new rule configuration.

Raises `ApiError` – If the rule configuration ID does not exist in this policy.

rules = []

sensor_settings = []

systemPolicy

Returns whether or not this is a system policy (compatibility method).

urlobject = '/policyservice/v1/orgs/{0}/policies'

urlobject_single = '/policyservice/v1/orgs/{0}/policies/{1}'

valid_rule_configs ()

Returns a dictionary identifying all valid rule configurations for this policy.

Returns

A dictionary mapping string ID values (UUIDs) to dicts containing entries for name, description, and category.

Return type dict

version = None

class PolicyQuery(*doc_class*, *cb*)

Bases: *cbc_sdk.base.BaseQuery*, *cbc_sdk.base.IterableQueryMixin*, *cbc_sdk.base.AsyncQueryMixin*

Query for retrieving policies (summary info only).

Initialize the Query object.

Parameters

- **doc_class** (*class*) – The class of the model this query returns.
- **cb** (*CBCloudAPI*) – A reference to the CBCloudAPI object.

add_descriptions (*descrs*)

Add policy description(s) to the list to search for.

Parameters **descrs** (*str/list*) – Either a single policy description or a list of descriptions.

Returns This object instance.

Return type *PolicyQuery*

Raises *ApiError* – If not supplied with a string or a list of strings.

add_names (*names*)

Add policy name(s) to the list to search for.

Parameters **names** (*str/list*) – Either a single policy name or a list of names.

Returns This object instance.

Return type *PolicyQuery*

Raises *ApiError* – If not supplied with a string or a list of strings.

add_policy_ids (*ids*)

Add policy ID(s) to the list to search for.

Parameters **ids** (*int/list*) – Either a single policy ID or a list of IDs.

Returns This object instance.

Return type *PolicyQuery*

Raises *ApiError* – If not supplied with an int or a list of ints.

add_priorities (*priorities*)

Add policy priority/priorities to the list to search for.

Parameters **priorities** (*str/list*) – Either a single policy priority value or a list of priority values.

Returns This object instance.

Return type *PolicyQuery*

Raises *ApiError* – If not supplied with a string priority value or a list of string priority values.

set_system (*system*)

Set to look for either system or non-system policies.

Parameters **system** (*bool*) – True to look for system policies, False to look for non-system policies.

Returns This object instance.

Return type *PolicyQuery*

Raises *ApiError* – If not supplied with a Boolean.

```
class PolicyRule(cb, parent, model_unique_id=None, initial_data=None, force_init=False,
                 full_doc=False)
```

Bases: *cbc_sdk.base.MutableBaseModel*

Represents a rule in the policy.

Create one of these objects, associating it with a Policy, and set its properties, then call its `save()` method to add the rule to the policy. This requires the `org.policies(UPDATE)` permission.

To update a PolicyRule, change the values of its property fields, then call the rule's `save()` method. This requires the `org.policies(UPDATE)` permission.

To delete an existing PolicyRule, call its `delete()` method. This requires the `org.policies(UPDATE)` permission.

Parameters

- **id** – The identifier of the rule
- **action** – The action the sensor will take when an application attempts to perform the selected operation
- **application** – The path, signature or reputation of the application
- **operation** – The type of behavior an application is performing

Initialize the PolicyRule object.

Parameters

- **cb** (*BaseAPI*) – Reference to API object used to communicate with the server.
- **parent** (*Policy*) – The “parent” policy of this rule.
- **model_unique_id** (*int*) – ID of the rule.
- **initial_data** (*dict*) – Initial data used to populate the rule.
- **force_init** (*bool*) – If True, forces the object to be refreshed after constructing. Default False.
- **full_doc** (*bool*) – If True, object is considered “fully” initialized. Default False.

```
VALID_ACTIONS = ['IGNORE', 'ALLOW', 'TERMINATE_PROCESS', 'TERMINATE_THREAD', 'TERMINATE']
```

```
VALID_APP_KEYS = {'type', 'value'}
```

```
VALID_APP_TYPES = ['NAME_PATH', 'SIGNED_BY', 'REPUTATION']
```

```
VALID_OPERATIONS = ['BYPASS_ALL', 'BYPASS_API', 'INVOKE_SCRIPT', 'INVOKE_SYSAPP', 'POL']
```

```
VALID_REPUTATIONS = ['ADAPTIVE_WHITE_LIST', 'ADWARE', 'COMMON_WHITE_LIST', 'COMPANY_BL']
```

```
action = None
```

```
application = {}
```

```
id = None
```

```

is_deleted
    Returns True if this rule object has been deleted.

operation = None

primary_key = 'id'

required = None

validate()
    Validates this rule against its constraints.

    Raises InvalidObjectError – If the rule object is not valid.

```

5.6.11 cbc_sdk.platform.policy_ruleconfigs module

Policy rule configuration implementation as part of Platform API

```

class CorePreventionRuleConfig(cb, parent, model_unique_id=None, initial_data=None,
                               force_init=False, full_doc=False)
    Bases: cbc_sdk.platform.policy_ruleconfigs.PolicyRuleConfig

```

Represents a core prevention rule configuration in the policy.

Create one of these objects, associating it with a Policy, and set its properties, then call its `save()` method to add the rule configuration to the policy. This requires the `org.policies(UPDATE)` permission.

To update a `CorePreventionRuleConfig`, change the values of its property fields, then call its `save()` method. This requires the `org.policies(UPDATE)` permission.

To delete an existing `CorePreventionRuleConfig`, call its `delete()` method. This requires the `org.policies(DELETE)` permission.

Parameters

- **id** – The ID of this rule config
- **name** – The name of this rule config
- **description** – The description of this rule config
- **inherited_from** – Indicates where the rule config was inherited from
- **category** – The category for this rule config
- **parameters** – The parameters associated with this rule config

Initialize the `CorePreventionRuleConfig` object.

Parameters

- **cb** (`BaseAPI`) – Reference to API object used to communicate with the server.
- **parent** (`Policy`) – The “parent” policy of this rule configuration.
- **model_unique_id** (`str`) – ID of the rule configuration.
- **initial_data** (`dict`) – Initial data used to populate the rule configuration.
- **force_init** (`bool`) – If True, forces the object to be refreshed after constructing. Default False.
- **full_doc** (`bool`) – If True, object is considered “fully” initialized. Default False.

category = None

description = None

get_assignment_mode()

Returns the assignment mode of this core prevention rule configuration.

Returns The assignment mode, either “REPORT” or “BLOCK”.

Return type str

id = None

inherited_from = None

name = None

parameters = {}

set_assignment_mode(mode)

Sets the assignment mode of this core prevention rule configuration.

Parameters mode (str) – The new mode to set, either “REPORT” or “BLOCK”. The default is “BLOCK”.

class PolicyRuleConfig(cb, parent, model_unique_id=None, initial_data=None, force_init=False, full_doc=False)

Bases: *cbc_sdk.base.MutableBaseModel*

Represents a rule configuration in the policy.

Create one of these objects, associating it with a Policy, and set its properties, then call its save() method to add the rule configuration to the policy. This requires the org.policies(UPDATE) permission.

To update a PolicyRuleConfig, change the values of its property fields, then call its save() method. This requires the org.policies(UPDATE) permission.

To delete an existing PolicyRuleConfig, call its delete() method. This requires the org.policies(DELETE) permission.

Parameters

- **id** – The ID of this rule config
- **name** – The name of this rule config
- **description** – The description of this rule config
- **inherited_from** – Indicates where the rule config was inherited from
- **category** – The category for this rule config
- **parameters** – The parameters associated with this rule config

Initialize the PolicyRuleConfig object.

Parameters

- **cb** (BaseAPI) – Reference to API object used to communicate with the server.
- **parent** (Policy) – The “parent” policy of this rule configuration.
- **model_unique_id** (str) – ID of the rule configuration.
- **initial_data** (dict) – Initial data used to populate the rule configuration.

- **force_init** (*bool*) – If True, forces the object to be refreshed after constructing. Default False.
- **full_doc** (*bool*) – If True, object is considered “fully” initialized. Default False.

category = None

description = None

get_parameter (*name*)

Returns a parameter value from the rule configuration.

Parameters **name** (*str*) – The parameter name.

Returns The parameter value, or None if there is no value.

Return type Any

id = None

inherited_from = None

name = None

parameters = {}

primary_key = 'id'

set_parameter (*name, value*)

Sets a parameter value into the rule configuration.

Parameters

- **name** (*str*) – The parameter name.
- **value** (*Any*) – The new value to be set.

urlobject = '/policyengine/v1/orgs/{0}/policies'

validate ()

Validates this rule configuration against its constraints.

Raises `InvalidObjectError` – If the rule object is not valid.

5.6.12 cbc_sdk.platform.processes module

Model and Query Classes for Processes

class AsyncProcessQuery (*doc_class, cb*)

Bases: `cbc_sdk.base.Query`

Represents the query logic for an asynchronous Process query.

This class specializes *Query* to handle the particulars of process querying.

Initialize the AsyncProcessQuery object.

Parameters

- **doc_class** (*class*) – The class of the model this query returns.
- **cb** (`CBCloudAPI`) – A reference to the `CBCloudAPI` object.

set_rows (*rows*)

Sets the ‘rows’ query parameter to the ‘results’ API call, determining how many rows to request per batch.

This will not limit the total results to rows instead the batch size will use rows and all of the num_available will be fetched.

Parameters `rows` (*int*) – How many rows to request.

timeout (*msecs*)

Sets the timeout on a process query.

Parameters `msecs` (*int*) – Timeout duration, in milliseconds.

Returns

The Query object with new `milliseconds` parameter.

Return type *Query* (*AsyncProcessQuery*)

Example

```
>>> cb.select(Process).where(process_name="foo.exe").timeout(5000)
```

class `Process` (*cb*, *model_unique_id=None*, *initial_data=None*, *force_init=False*, *full_doc=False*)

Bases: *cbc_sdk.base.UnrefreshableModel*

Represents a process retrieved by one of the Enterprise EDR endpoints.

Examples

use the Process GUID directly

```
>>> process = api.select(Process, "WNEXFKQ7-00050603-0000066c-00000000-1d6c9acb43e29bb")
```

use the Process GUID in a where() clause

```
>>> process_query = (api.select(Process).where(process_guid="WNEXFKQ7-00050603-0000066c-00000000-1d6c9acb43e29bb"))
>>> process_query_results = [proc for proc in process_query]
>>> process_2 = process_query_results[0]
```

Initialize the Process object.

Parameters

- **cb** (*CBCloudAPI*) – A reference to the CBCloudAPI object.
- **model_unique_id** (*str*) – The unique ID (GUID) for this process.
- **initial_data** (*dict*) – The data to use when initializing the model object.
- **force_init** (*bool*) – True to force object initialization.
- **full_doc** (*bool*) – True to mark the object as fully initialized.

class `Summary` (*cb*, *model_unique_id=None*, *initial_data=None*, *force_init=False*, *full_doc=True*)

Bases: *cbc_sdk.base.UnrefreshableModel*

Represents a summary of organization-specific information for a process.

The preferred interface for interacting with Summary models is *Process.summary*.

Example

```
>>> process = api.select(Process, "WNEXFKQ7-00050603-0000066c-00000000-
↳1d6c9acb43e29bb")
>>> summary = process.summary
```

Initialize the Summary object.

Parameters

- **cb** ([CBCloudAPI](#)) – A reference to the CBCloudAPI object.
- **model_unique_id** (*str*) – The unique ID for this particular instance of the model object.
- **initial_data** (*dict*) – The data to use when initializing the model object.
- **force_init** (*bool*) – True to force object initialization.
- **full_doc** (*bool*) – True to mark the object as fully initialized.

```
SHOW_ATTR = {'children': {'fields': ['process_name', 'process_guid', 'process_has
```

```
default_sort = 'last_update desc'
```

```
primary_key = 'process_guid'
```

```
result_url = '/api/investigate/v2/orgs/{}/processes/summary_jobs/{}/results'
```

```
summary_format = 'summary'
```

```
urlobject = '/api/investigate/v2/orgs/{}/processes/summary_jobs'
```

```
class Tree(cb, model_unique_id=None, initial_data=None, force_init=False, full_doc=True)
```

```
Bases: cbc\_sdk.base.UnrefreshableModel
```

Represents a summary of organization-specific information for a process.

The preferred interface for interacting with Tree models is *Process.tree*.

Example

```
>>> process = api.select(Process, "WNEXFKQ7-00050603-0000066c-00000000-
↳1d6c9acb43e29bb")
>>> tree = process.tree
```

Initialize the Tree object.

Parameters

- **cb** ([CBCloudAPI](#)) – A reference to the CBCloudAPI object.
- **model_unique_id** (*str*) – The unique ID for this particular instance of the model object.
- **initial_data** (*dict*) – The data to use when initializing the model object.
- **force_init** (*bool*) – True to force object initialization.
- **full_doc** (*bool*) – True to mark the object as fully initialized.

```
SHOW_ATTR = {'children': ['process_name', 'process_guid', 'process_hash', 'process
```

```
default_sort = 'last_update desc'
```

```
primary_key = 'process_guid'
result_url = '/api/investigate/v2/orgs/{}/processes/summary_jobs/{}/results'
summary_format = 'tree'
urlobject = '/api/investigate/v2/orgs/{}/processes/summary_jobs'
```

approve_process_sha256 (*description*=")

Approves the application by adding the process_sha256 to the WHITE_LIST

Parameters **description** – The justification for why the application was added to the WHITE_LIST

Returns

ReputationOverride object created in the Carbon Black Cloud

Return type *ReputationOverride* (cbc_sdk.platform.ReputationOverride)

ban_process_sha256 (*description*=")

Bans the application by adding the process_sha256 to the BLACK_LIST

Parameters **description** – The justification for why the application was added to the BLACK_LIST

Returns

ReputationOverride object created in the Carbon Black Cloud

Return type *ReputationOverride* (cbc_sdk.platform.ReputationOverride)

children

Returns a list of child processes for this process.

Returns

List of Processes, one for each child of the parent Process.

Return type children ([*Process*])

default_sort = 'last_update desc'

events (***kwargs*)

Returns a query for events associated with this process's process GUID.

Parameters **kwargs** – Arguments to filter the event query with.

Returns

Query object with the appropriate search parameters for events

Return type query (cbc_sdk.enterprise_edr.Query)

Example

```
>>> [print(event) for event in process.events()]
>>> [print(event) for event in process.events(event_type="modload")]
```

facets ()

Returns a FacetQuery for a Process.

This represents the search for a summary of result groupings (facets). The returned AsyncFacetQuery object must have facet fields or ranges specified before it can be submitted, using the *add_facet_field()* or *add_range()* methods.

get_details (*timeout=0, async_mode=False*)

Requests detailed results.

Parameters

- **timeout** (*int*) – Event details request timeout in milliseconds.
- **async_mode** (*bool*) – True to request details in an asynchronous manner.

Note:

- When using asynchronous mode, this method returns a python future. You can call result() on the future object to wait for completion and get the results.
-

parents

Returns a parent process associated with this process.

Returns Parent Process if one exists, None if the process has no recorded parent.

Return type parent (*Process*)

primary_key = 'process_guid'

process_md5

Returns a string representation of the MD5 hash for this process.

Returns MD5 hash of the process.

Return type hash (str)

process_pids

Returns a list of PIDs associated with this process.

Returns List of integer PIDs. None if there are no associated PIDs.

Return type pids ([int])

process_sha256

Returns a string representation of the SHA256 hash for this process.

Returns SHA256 hash of the process.

Return type hash (str)

siblings

Returns a list of sibling processes for this process.

Returns

List of Processes, one for each sibling of the parent Process.

Return type siblings ([*Process*])

summary

Returns organization-specific information about this process.

tree

Returns a Process Tree associated with this process.

Returns Tree with children (and possibly siblings).

Return type *Tree* (cbc_sdk.enterprise_edr.Tree)

Example

```
>>> tree = process.tree
```

```
urlobject = ''
```

```
validation_url = '/api/investigate/v1/orgs/{}/processes/search_validation'
```

```
class ProcessFacet (cb, model_unique_id, initial_data)
```

```
Bases: cbc_sdk.base.UnrefreshableModel
```

Represents the results of an AsyncFacetQuery.

ProcessFacet objects contain both Terms and Ranges. Each of those contain facet fields and values.

Access all of the Terms facet data with `ProcessFacet.Terms.facets()` or see just the field names with `ProcessFacet.Terms.fields()`.

Access all of the Ranges facet data with `ProcessFacet.Ranges.facets()` or see just the field names with `ProcessFacet.Ranges.fields()`.

Process Facets can be queried for via `CBCloudAPI.select(ProcessFacet)`. Specify facet field(s) with `.add_facet_field("my_facet_field")`.

Optionally you can limit the facet query to a single process with the following two options. Using the solrq builder specify Process GUID with `.where(process_guid="example_guid")` and modify the query with `.or_(parent_effective_reputation="KNOWN_MALWARE")` and `.and_(parent_effective_reputation="KNOWN_MALWARE")`.

If you want full control over the query string specify Process Guid in the query string `.where("process_guid: example_guid OR parent_effective_reputation: KNOWN_MALWARE")` Examples:

```
>>> process_facet_query = (api.select(ProcessFacet).where(process_
↳guid=
                                "WNEXFKQ7-00050603-0000066c-00000000-
↳1d6c9acb43e29bb"))
>>> process_facet_query.add_facet_field("device_name")
```

retrieve results synchronously

```
>>> facet = process_facet_query.results
```

retrieve results asynchronously

```
>>> future = process_facet_query.execute_async()
>>> result = future.result()
```

result is a list with one item, so access the first item

```
>>> facet = result[0]
```

Parameters

- **job_id** – The Job ID assigned to this query
- **terms** – Contains the Process Facet search results
- **ranges** – Groupings for search result properties that are ISO 8601 timestamps or numbers
- **contacted** – The number of searchers contacted for this query

- **completed** – The number of searchers that have reported their results

Initialize a ResultFacet object with initial_data.

class Ranges (*cb, initial_data*)

Bases: *cbc_sdk.base.UnrefreshableModel*

Represents the range (bucketed) facet fields and values associated with a Process Facet query.

Initialize a ProcessFacet Ranges object with initial_data.

facets

Returns the reified *ProcessFacet.Terms._facets* for this result.

fields

Returns the ranges fields for this result.

class Terms (*cb, initial_data*)

Bases: *cbc_sdk.base.UnrefreshableModel*

Represents the facet fields and values associated with a Process Facet query.

Initialize a ProcessFacet Terms object with initial_data.

facets

Returns the terms' facets for this result.

fields

Returns the terms facets' fields for this result.

completed = None

contacted = None

job_id = None

num_found = None

primary_key = 'job_id'

ranges = []

ranges_

Returns the reified *ProcessFacet.Ranges* for this result.

result_url = '/api/investigate/v2/orgs/{}/processes/facet_jobs/{}/results'

submit_url = '/api/investigate/v2/orgs/{}/processes/facet_jobs'

terms = {}

terms_

Returns the reified *ProcessFacet.Terms* for this result.

class SummaryQuery (*doc_class, cb*)

Bases: *cbc_sdk.base.BaseQuery*, *cbc_sdk.base.AsyncQueryMixin*, *cbc_sdk.base.QueryBuilderSupportMixin*

Represents the logic for a Process Summary or Process Tree query.

Initialize the SummaryQuery object.

Parameters

- **doc_class** (*class*) – The class of the model this query returns.
- **cb** (*CBCloudAPI*) – A reference to the CBCloudAPI object.

results

Save query results to `self._results` with `self._search()` method.

set_time_range (*start=None, end=None, window=None*)

Sets the 'time_range' query body parameter, determining a time window based on 'device_timestamp'.

Parameters

- **start** (*str in ISO 8601 timestamp*) – When to start the result search.
- **end** (*str in ISO 8601 timestamp*) – When to end the result search.
- **window** (*str*) – Time window to execute the result search, ending on the current time. Should be in the form “-2w”, where y=year, w=week, d=day, h=hour, m=minute, s=second.

Note:

- *window* will take precedent over *start* and *end* if provided.
-

Examples

```
>>> query = api.select(Event).set_time_range(start="2020-10-20T20:34:07Z")
>>> second_query = api.select(Event).set_time_range
...     (start="2020-10-20T20:34:07Z", end="2020-10-30T20:34:07Z")
>>> third_query = api.select(Event).set_time_range(window='-3d')
```

timeout (*msecs*)

Sets the timeout on a process query.

Parameters **msecs** (*int*) – Timeout duration, in milliseconds.

Returns

The Query object with new **milliseconds** parameter.

Return type *Query* (*AsyncProcessQuery*)

Example

```
>>> cb.select(Process).where(process_name="foo.exe").timeout(5000)
```

5.6.13 cbc_sdk.platform.reputation module

Model and Query Classes for Reputation

class ReputationOverride (*cb, model_unique_id, initial_data=None*)

Bases: *cbc_sdk.platform.base.PlatformModel*

Represents a reputation override.

Parameters

- **id** – An identifier for a reputation override
- **created_by** – Creator of the override

- **create_time** – Time the override was created
- **description** – Justification for override
- **override_list** – The override list to add a new reputation (BLACK_LIST only valid for SHA256)
- **override_type** – Process property match when applying override
- **sha256_hash** – A hexadecimal string of length 64 characters representing the SHA-256 hash of the application
- **filename** – An application name for the hash
- **signed_by** – Name of the signer for the application
- **certificate_authority** – Certificate authority that authorizes the validity of the certificate
- **path** – The absolute path to file or directory where tool exists on disk
- **include_child_processes** – Include tool's child processes on approved list

Initialize the ReputationOverride object.

Parameters

- **cb** ([BaseAPI](#)) – Reference to API object used to communicate with the server.
- **model_unique_id** (*str*) – ID of the alert represented.
- **initial_data** (*dict*) – Initial data used to populate the alert.

classmethod bulk_delete (*cb, overrides*)

Deletes reputation overrides in bulk by id.

Parameters

- **cb** ([BaseAPI](#)) – Reference to API object used to communicate with the server.
- **overrides** (*List*) – List of reputation override ids

Example

```
>>>
[
    "e9410b754ea011ebbfd0db2585a41b07"
]
```

certificate_authority = None

classmethod create (*cb, initial_data*)

Returns all vendors and products that have been seen for the organization.

Parameters

- **cb** ([BaseAPI](#)) – Reference to API object used to communicate with the server.
- **initial_data** (*Object*) – The initial data for a ReputationOverride

Example

```
>>>
{
    "description": "Banned as known malware",
    "override_list": "BLACK_LIST",
    "override_type": "SHA256",
    "sha256_hash":
    ↪ "dd191a5b23df92e13a8852291f9fb5ed594b76a28a5a464418442584afd1e048",
    "filename": "foo.exe"
}
```

Returns The created ReputationOverride object based on the specified properties

Return type *ReputationOverride*

create_time = None

created_by = None

delete()

Delete this object.

description = None

filename = None

id = None

include_child_processes = None

override_list = None

override_type = None

path = None

primary_key = 'id'

sha256_hash = None

signed_by = None

urlobject = '/appservices/v6/orgs/{0}/reputations/overrides'

urlobject_single = '/appservices/v6/orgs/{0}/reputations/overrides/{1}'

class ReputationOverrideQuery (*doc_class, cb*)

Bases: *cbc_sdk.base.BaseQuery, cbc_sdk.base.QueryBuilderSupportMixin, cbc_sdk.base.IterableQueryMixin, cbc_sdk.base.AsyncQueryMixin*

Represents a query that is used to locate ReputationOverride objects.

Initialize the ReputationOverrideQuery.

Parameters

- **doc_class** (*class*) – The model class that will be returned by this query.
- **cb** (*BaseAPI*) – Reference to API object used to communicate with the server.

VALID_DIRECTIONS = ['ASC', 'DESC', 'asc', 'desc']

set_override_list (*override_list*)

Sets the override_list criteria filter.

Parameters `override_list` (*str*) – Override List to filter on.

Returns The ReputationOverrideQuery with specified `override_list`.

set_override_type (*override_type*)

Sets the `override_type` criteria filter.

Parameters `override_type` (*str*) – Override List to filter on.

Returns The ReputationOverrideQuery with specified `override_type`.

sort_by (*key*, *direction*='ASC')

Sets the sorting behavior on a query's results.

Example

```
>>> cb.select(ReputationOverride).sort_by("create_time")
```

Parameters

- **key** (*str*) – The key in the schema to sort by.
- **direction** (*str*) – The sort order, either “ASC” or “DESC”.

Returns This instance.

Return type *ReputationOverrideQuery*

Raises `ApiError` – If an invalid direction value is passed.

5.6.14 cbc_sdk.platform.users module

Model and Query Classes for Users

class `User` (*cb*, *model_unique_id*, *initial_data*=None)

Bases: `cbc_sdk.base.MutableBaseModel`

Represents a user in the Carbon Black Cloud.

Parameters

- **org_key** – Organization key for this user
- **auth_method** – Method to be used for the user to authenticate
- **admin_login_version** – Version number of the user information
- **email** – User's E-mail address
- **login_name** – Login name for the user
- **login_id** – Login ID (user ID) for this user
- **phone** – User's phone number
- **first_name** – User's first name
- **last_name** – User's last name
- **org_id** – ID of the organization the user is in
- **org_admin_version** – TBD
- **role** – Not used, always “DEPRECATED”

- **contact_id** – ID of the user’s contact information
- **contact_version** – Version of the user’s contact information

Initialize the User object.

Parameters

- **cb** ([BaseAPI](#)) – Reference to API object used to communicate with the server.
- **model_unique_id** (*int*) – Login ID of this user.
- **initial_data** (*dict*) – Initial data used to populate the user.

class UserBuilder (*cb*)

Bases: `object`

Auxiliary object used to construct a new User.

Create the empty UserBuilder object.

Parameters **cb** ([BaseAPI](#)) – Reference to API object used to communicate with the server.

add_grant_profile (*orgs, roles*)

Adds a grant profile for the new user.

Parameters

- **orgs** (*list[str]*) – List of organizations to be allowed, specified as keys or URNs.
- **roles** (*list[str]*) – List of roles to be granted, specified as URNs.

Returns This object.

Return type [UserBuilder](#)

build ()

Builds the new user.

Notes

The new user will not be “findable” by other API functions until it has been activated and its initial password has been set.

set_auth_method (*method*)

Sets the authentication method for the new user. The default is ‘PASSWORD’.

Parameters **method** (*str*) – The authentication method for the new user.

Returns This object.

Return type [UserBuilder](#)

set_email (*email*)

Sets the E-mail address for the new user.

Parameters **email** (*str*) – The E-mail address for the new user.

Returns This object.

Return type [UserBuilder](#)

set_first_name (*first_name*)

Sets the first name for the new user.

Parameters **first_name** (*str*) – The first name for the new user.

Returns This object.

Return type [UserBuilder](#)

set_last_name (*last_name*)

Sets the last name for the new user.

Parameters **last_name** (*str*) – The last name for the new user.

Returns This object.

Return type *UserBuilder*

set_phone (*phone*)

Sets the phone number for the new user.

Parameters **phone** (*str*) – The phone number for the new user.

Returns This object.

Return type *UserBuilder*

set_role (*role*)

Sets the role URN for the new user.

Parameters **role** (*str*) – The URN of the role to set for the user.

Returns This object.

Return type *UserBuilder*

add_profiles (*profile_templates*)

Add the specified profiles to the user's grant.

Parameters **profile_templates** (*list[dict]*) – List of profile templates to be added to the user.

admin_login_version = None

auth_method = None

classmethod bulk_add_profiles (*users, profile_templates*)

Add the specified profiles to the specified users' grants.

Parameters

- **users** (*list[User]*) – List of User objects specifying users to be modified.
- **profile_templates** (*list[dict]*) – List of profile templates to be added to the users.

classmethod bulk_create (*cb, user_templates, profile_templates*)

Creates a series of new users.

Parameters

- **cb** (*CBCloudAPI*) – A reference to the CBCloudAPI object.
- **user_templates** (*list[dict]*) – List of templates for users to be created.
- **profile_templates** (*list[dict]*) – List of profile templates to be applied to each user.

classmethod bulk_delete (*users*)

Deletes all the listed users.

Parameters **users** (*list[User]*) – List of User objects specifying users to be deleted.

classmethod bulk_disable_all_access (*users*)

Disables all access profiles held by the listed users.

Parameters **users** (*list[User]*) – List of User objects specifying users to be disabled.

classmethod bulk_disable_profiles (*users, profile_templates*)

Disable the specified profiles in the specified users' grants.

Parameters

- **users** (*list[User]*) – List of User objects specifying users to be modified.
- **profile_templates** (*list[dict]*) – List of profile templates to be disabled.

change_role (*role_urn*, *org=None*)

Add the specified role to the user (either to the grant or the profiles).

Parameters

- **role_urn** (*str*) – URN of the role to be added.
- **org** (*str*) – If specified, only profiles that match this organization will have the role added. Organization may be specified as either an org key or a URN.

Raises `ApiError` – If the user is a “legacy” user that has no grant.

contact_id = `None`

contact_version = `None`

classmethod create (*cb*, *template=None*)

Creates a new user.

Parameters

- **cb** (`CBCloudAPI`) – A reference to the CBCloudAPI object.
- **template** (*dict*) – Optional template data for creating the new user.

Returns

If **template** is `None`, returns an instance of this object. Call methods on the object to set the values associated with the new user, and then call `build()` to create it.

Return type *UserBuilder*

delete ()

Delete this object.

disable_all_access ()

Disables all access profiles held by this user.

Raises `ApiError` – If the user is a “legacy” user that has no grant.

disable_profiles (*profile_templates*)

Disable the specified profiles in the user’s grant.

Parameters **profile_templates** (*list[dict]*) – List of profile templates to be disabled.

Raises `ApiError` – If the user is a “legacy” user that has no grant.

email = `None`

first_name = `None`

grant ()

Locates the access grant for this user.

Returns Access grant for this user, or `None` if the user has none.

Return type *Grant*

last_name = `None`

login_id = `None`

login_name = `None`

org_admin_version = `None`

org_id = `None`

org_key = None

org_urn

Returns the URN for this user's organization (used in accessing Grants).

Returns URN for this user's organization.

Return type str

phone = None

primary_key = 'login_id'

reset_google_authenticator_registration()

Forces Google Authenticator registration to be reset for this user.

role = None

set_profile_expiration(*profile_templates*, *expiration_date*)

Set the expiration time for the specified profiles in the user's grant.

Parameters

- **profile_templates** (*list[dict]*) – List of profile templates to be reset.
- **expiration_date** (*str*) – New expiration date, in ISO 8601 format.

Raises `ApiError` – If the user is a “legacy” user that has no grant.

urlobject = '/appservices/v6/orgs/{0}/users'

urlobject_single = '/appservices/v6/orgs/{0}/users/{1}'

urn

Returns the URN for this user (used in accessing Grants).

Returns URN for this user.

Return type str

class UserQuery (*doc_class*, *cb*)

Bases: `cbc_sdk.base.BaseQuery`, `cbc_sdk.base.IterableQueryMixin`, `cbc_sdk.base.AsyncQueryMixin`

Query for retrieving users in bulk.

Initialize the Query object.

Parameters

- **doc_class** (*class*) – The class of the model this query returns.
- **cb** (`CBCloudAPI`) – A reference to the CBCloudAPI object.

email_addresses (*addrs*)

Limit the query to users with the specified E-mail addresses. Call multiple times to add multiple addresses.

Parameters **addrs** (*list[str]*) – List of addresses to be added to the query.

Returns This object.

Return type `UserQuery`

user_ids (*userids*)

Limit the query to users with the specified user IDs. Call multiple times to add multiple user IDs.

Parameters **userids** (*list[str]*) – List of user IDs to be added to the query.

Returns This object.

Return type *UserQuery*

`log = <Logger cbc_sdk.platform.users (WARNING)>`
User Models

`normalize_profile_list(profile_templates)`
Internal function to normalize a list of profile templates.

5.6.15 cbc_sdk.platform.vulnerability_assessment module

Model and Query Classes for Vulnerability Assessment API

class AffectedAssetQuery (*vulnerability, cb*)
Bases: *cbc_sdk.platform.vulnerability_assessment.VulnerabilityQuery*
Query Class for the Vulnerability
Initialize the AffectedAssetQuery.

Parameters

- **vulnerability** (*class*) – The vulnerability that will be returned by this query.
- **cb** (*BaseAPI*) – Reference to API object used to communicate with the server.

set_os_product_id (*os_product_id, operator*)
Restricts the vulnerabilities that this query is performed on to the specified os_product_id.

Parameters

- **os_product_id** (*str*) – os_product_id.
- **operator** (*str*) – logic operator to apply to property value.

Returns This instance.

Return type *AffectedAssetQuery*

class Vulnerability (*cb, model_unique_id, os_product_id=None, initial_data=None*)
Bases: *cbc_sdk.base.NewBaseModel*
Represents a vulnerability

Parameters

- **affected_assets** – List of affected assets
- **category** – Vulnerability category
- **device_count** – Number of affected devices
- **os_info** – Information about the operating system associated with the vulnerability
- **os_product_id** – Operating system product ID
- **product_info** – Information about the vulnerable product
- **vuln_info** – Information about the vulnerability

Initialize the Vulnerability object.

Parameters

- **cb** (*BaseAPI*) – Reference to API object used to communicate with the server.
- **model_unique_id** (*str*) – ID of the vulnerability represented.

- **os_product_id** (*str*) – os_product_id of the vulnerability used to uniquely identify a CVE with multiple OS/Product instances
- **initial_data** (*dict*) – Initial data used to populate the alert.

class AssetView (*cb*, *initial_data=None*)

Bases: `list`

Represents a list of Vulnerability for an organization.

Initialize Vulnerability.AssetView object

Parameters

- **cb** (`BaseAPI`) – Reference to API object used to communicate with the server.
- **initial_data** (`list[dict]`) – list of assets and their vulnerability view

swagger_meta_file = 'workload/models/vulnerabilityAssetView.yaml'

urlobject = '/vulnerability/assessment/api/v1/orgs/{'

class OrgSummary (*cb*, *initial_data=None*)

Bases: `cbc_sdk.base.UnrefreshableModel`

Represents a vulnerability summary for an organization.

Parameters

- **monitored_assets** – Number of assets being monitored
- **severity_summary** – Information about vulnerabilities at each severity level

Initialize Vulnerability.OrgSummary object

Parameters

- **cb** (`BaseAPI`) – Reference to API object used to communicate with the server.
- **initial_data** (*dict*) – dictionary of the data

monitored_assets = None

severity_levels ()

Returns the severity levels

Returns List of severities

Return type Severities (list[str])

severity_summary = {}

urlobject = '/vulnerability/assessment/api/v1/orgs/{'

affected_assets = []

category = None

device_count = None

get_affected_assets ()

Returns an AffectedAssetQuery to fetch the list of devices affected by the Vulnerability.

Args: os_product_id (str) operating system product ID

Returns AffectedAssetQuery

os_info = {}

os_product_id = None

perform_action (*type*, *reason=None*, *notes=None*)

Take an action to manage the Vulnerability.

Parameters

- **type** (*str*) – The type of action. (supports DISMISS, DISMISS_EDIT, or UNDISMISS)
- **reason** (*str*) – The reason the vulnerability is dismissed. Required when type is DISMISS or DISMISS_EDIT. (supports FALSE_POSITIVE, RESOLUTION_DEFERRED, NON_ISSUE, NON_CRITICAL_ASSET, UNDER_RESOLUTION, OTHER)
- **notes** (*str*) – Notes to be associated with the dismissal. Required when reason is OTHER.

Returns The action response

Return type *obj*

Raises *ApiError* – If the request is invalid or missing required properties

```
primary_key = 'cve_id'
```

```
product_info = {}
```

```
urlobject = '/vulnerability/assessment/api/v1/orgs/{}'
```

```
vuln_info = {}
```

class VulnerabilityAssetViewQuery (*doc_class*, *cb*)

Bases: *cbc_sdk.platform.vulnerability_assessment.VulnerabilityQuery*

Represents a query that is used fetch the Vulnerability Asset View

Initialize the VulnerabilityAssetViewQuery.

Parameters

- **doc_class** (*class*) – The model class that will be returned by this query.
- **cb** (*BaseAPI*) – Reference to API object used to communicate with the server.

export ()

Performs the query and export the results in the form of a Job.

Returns The export job.

Return type *Job*

class VulnerabilityOrgSummaryQuery (*doc_class*, *cb*, *device=None*)

Bases: *cbc_sdk.base.BaseQuery*

Represents a query that is used fetch the VulnerabilitySummary

Initialize the VulnerabilityQuery.

Parameters

- **doc_class** (*class*) – The model class that will be returned by this query.
- **cb** (*BaseAPI*) – Reference to API object used to communicate with the server.
- **device** (*Device*) – Optional Device object to indicate VulnerabilityQuery is for a specific device

```
VALID_SEVERITY = ['CRITICAL', 'IMPORTANT', 'MODERATE', 'LOW']
```

```
VALID_VISIBILITY = ['DISMISSED', 'ACTIVE']
```


set_severity (*severity*)

Restricts the vulnerability summary to a severity level

Parameters **severity** (*str*) – filters the vulnerability summary per severity (CRITICAL, IMPORTANT, MODERATE, LOW)

Returns This instance.

Return type *VulnerabilityOrgSummaryQuery*

set_vcenter (*vcenter_uuid*)

Restricts the vulnerability summary to a specific vcenter

Parameters **vcenter_uuid** (*str*) – vcenter uuid.

Returns This instance.

Return type *VulnerabilityOrgSummaryQuery*

set_visibility (*visibility*)

Restricts the vulnerabilities that this query is performed on to the specified visibility

Parameters **visibility** (*str*) – The visibility state of the vulnerability. (supports ACTIVE, DISMISSED)

Returns This instance.

Return type *VulnerabilityOrgSummaryQuery*

submit ()

Performs the query and returns the Vulnerability.OrgSummary

Returns The vulnerability summary for the organization

Return type *Vulnerability.OrgSummary*

class VulnerabilityQuery (*doc_class, cb, device=None*)

Bases: *cbc_sdk.base.BaseQuery*, *cbc_sdk.base.QueryBuilderSupportMixin*, *cbc_sdk.base.IterableQueryMixin*, *cbc_sdk.base.AsyncQueryMixin*

Represents a query that is used to locate Vulnerability objects.

Initialize the VulnerabilityQuery.

Parameters

- **doc_class** (*class*) – The model class that will be returned by this query.
- **cb** (*BaseAPI*) – Reference to API object used to communicate with the server.
- **device** (*Device*) – Optional Device object to indicate VulnerabilityQuery is for a specific device

VALID_DEVICE_TYPE = ['WORKLOAD', 'ENDPOINT']

VALID_DIRECTIONS = ['ASC', 'DESC']

VALID_OS_TYPE = ['CENTOS', 'RHEL', 'SLES', 'UBUNTU', 'WINDOWS']

VALID_SEVERITY = ['CRITICAL', 'IMPORTANT', 'MODERATE', 'LOW']

VALID_SYNC_STATUS = ['NOT_STARTED', 'MATCHED', 'ERROR', 'NOT_MATCHED', 'NOT_SUPPORTED']

VALID_SYNC_TYPE = ['MANUAL', 'SCHEDULED']

VALID_VISIBILITY = ['DISMISSED', 'ACTIVE']

add_criteria (*key*, *value*, *operator*=*'EQUALS'*)

Restricts the vulnerabilities that this query is performed on to the specified key value pair.

Parameters

- **key** (*str*) – Property from the vulnerability object
- **value** (*str*) – Value of the property to filter by
- **operator** (*str*) – (optional) logic operator to apply to property value.

Returns This instance.

Return type *VulnerabilityQuery*

export ()

Performs the query and export the results in the form of a Job.

Example

```
>>> # Create the Vulnerability query
>>> query = cb.select(Vulnerability).set_severity('CRITICAL')
>>> # Export the results
>>> job = query.export()
>>> # wait for the export to finish
>>> job.await_completion()
>>> # write the results to a file
>>> job.get_output_as_file("vulnerabilities.csv")
```

Returns The export job.

Return type *Job*

set_deployment_type (*deployment_type*, *operator*)

Restricts the vulnerabilities that this query is performed on to the specified deployment type.

Parameters

- **deployment_type** (*str*) – deployment type (“ENDPOINT”, “AWS”)
- **operator** (*str*) – logic operator to apply to property value.

Returns This instance.

Return type *VulnerabilityQuery*

set_device_type (*device_type*, *operator*)

Restricts the vulnerabilities that this query is performed on to the specified device type.

Parameters

- **device_type** (*str*) – device type (“WORKLOAD”, “ENDPOINT”)
- **operator** (*str*) – logic operator to apply to property value.

Returns This instance.

Return type *VulnerabilityQuery*

set_highest_risk_score (*highest_risk_score*, *operator*)

Restricts the vulnerabilities that this query is performed on to the specified highest_risk_score.

Parameters

- **highest_risk_score** (*double*) – highest_risk_score.
- **operator** (*str*) – logic operator to apply to property value.

Returns This instance.

Return type *VulnerabilityQuery*

set_last_sync_ts (*last_sync_ts, operator*)

Restricts the vulnerabilities that this query is performed on to the specified last_sync_ts.

Parameters

- **last_sync_ts** (*str*) – last_sync_ts.
- **operator** (*str*) – logic operator to apply to property value.

Returns This instance.

Return type *VulnerabilityQuery*

set_name (*name, operator*)

Restricts the vulnerabilities that this query is performed on to the specified name.

Parameters

- **name** (*str*) – name.
- **operator** (*str*) – logic operator to apply to property value.

Returns This instance.

Return type *VulnerabilityQuery*

set_os_arch (*os_arch, operator*)

Restricts the vulnerabilities that this query is performed on to the specified os_arch.

Parameters

- **os_arch** (*str*) – os_arch.
- **operator** (*str*) – logic operator to apply to property value.

Returns This instance.

Return type *VulnerabilityQuery*

set_os_name (*os_name, operator*)

Restricts the vulnerabilities that this query is performed on to the specified os_name.

Parameters

- **os_name** (*str*) – os_name.
- **operator** (*str*) – logic operator to apply to property value.

Returns This instance.

Return type *VulnerabilityQuery*

set_os_type (*os_type, operator*)

Restricts the vulnerabilities that this query is performed on to the specified os type.

Parameters

- **os_type** (*str*) – os type (“CENTOS”, “RHEL”, “SLES”, “UBUNTU”, “WINDOWS”)
- **operator** (*str*) – logic operator to apply to property value.

Returns This instance.

Return type *VulnerabilityQuery*

set_os_version (*os_version*, *operator*)

Restricts the vulnerabilities that this query is performed on to the specified *os_version*.

Parameters

- **os_version** (*str*) – *os_version*.
- **operator** (*str*) – logic operator to apply to property value.

Returns This instance.

Return type *VulnerabilityQuery*

set_severity (*severity*, *operator*)

Restricts the vulnerabilities that this query is performed on to the specified *severity*.

Parameters

- **severity** (*str*) – severity (“CRITICAL”, “IMPORTANT”, “MODERATE”, “LOW”)
- **operator** (*str*) – logic operator to apply to property value.

Returns This instance.

Return type *VulnerabilityQuery*

set_sync_status (*sync_status*, *operator*)

Restricts the vulnerabilities that this query is performed on to the specified *sync_status*.

Parameters

- **sync_status** (*str*) – *sync_status* (“NOT_STARTED”, “MATCHED”, “ERROR”, “NOT_MATCHED”, “NOT_SUPPORTED”, “CANCELLED”, “IN_PROGRESS”, “ACTIVE”, “COMPLETED”)
- **operator** (*str*) – logic operator to apply to property value.

Returns This instance.

Return type *VulnerabilityQuery*

set_sync_type (*sync_type*, *operator*)

Restricts the vulnerabilities that this query is performed on to the specified *sync_type*.

Parameters

- **sync_type** (*str*) – *sync_type* (“MANUAL”, “SCHEDULED”)
- **operator** (*str*) – logic operator to apply to property value.

Returns This instance.

Return type *VulnerabilityQuery*

set_vcenter (*vcenter_uuid*)

Restricts the vulnerabilities that this query is performed on to the specified *vcenter id*.

Parameters **vcenter_uuid** (*str*) – *vcenter uuid*.

Returns This instance.

Return type *VulnerabilityQuery*

set_visibility (*visibility*)

Restricts the vulnerabilities that this query is performed on to the specified *visibility*

Parameters **visibility** (*str*) – The visibility state of the vulnerability. (supports ACTIVE, DISMISSED)

Returns This instance.

Return type *VulnerabilityQuery*

set_vm_id (*vm_id*, *operator*)

Restricts the vulnerabilities that this query is performed on to the specified *vm_id*.

Parameters

- **vm_id** (*str*) – *vm_id*.
- **operator** (*str*) – logic operator to apply to property value.

Returns This instance.

Return type *VulnerabilityQuery*

set_vuln_count (*vuln_count*, *operator*)

Restricts the vulnerabilities that this query is performed on to the specified *vuln_count*.

Parameters

- **vuln_count** (*str*) – *vuln_count*.
- **operator** (*str*) – logic operator to apply to property value.

Returns This instance.

Return type *VulnerabilityQuery*

sort_by (*key*, *direction*=‘ASC’)

Sets the sorting behavior on a query’s results.

Example

```
>>> cb.select(Vulnerability).sort_by("status")
```

Parameters

- **key** (*str*) – The key in the schema to sort by.
- **direction** (*str*) – The sort order, either “ASC” or “DESC”.

Returns This instance.

Return type *VulnerabilityQuery*

Raises *ApiError* – If an invalid direction value is passed.

```
log = <Logger cbc_sdk.platform.vulnerability_assessment (WARNING)>
Vulnerability models
```

5.6.16 Module contents

5.7 Workload

5.7.1 Submodules

5.7.2 cbc_sdk.workload.nsx_remediation module

NSX Remediation for Workloads

class `NSXRemediationJob` (*cb, running_job_ids*)

Bases: `object`

An object that runs and monitors an NSX Remediation operation.

Creates a new NSXRemediationJob object.

Parameters

- **cb** (`BaseAPI`) – Reference to API object used to communicate with the server.
- **running_job_ids** (*list[str]*) – The list of running job IDs.

`RUNNING_STATUSES = ['UNASSIGNED', 'SCHEDULED', 'RUNNING', 'RUNNING_UNDELIVERED']`

`VALID_TAGS = ['CB-NSX-Quarantine', 'CB-NSX-Isolate', 'CB-NSX-Custom']`

async_wait_result ()

Sets up a Future which can be used to wait asynchronously for all running jobs to be completed.

Required Permissions: appliances.registration(READ)

Returns A future representing the job and its results.

Return type Future

await_result ()

Waits for all running jobs to be completed and returns the final status.

Required Permissions: appliances.registration(READ)

Returns The final status, mapping individual job IDs to status value dicts.

Return type dict

classmethod `start_request` (*cb, device_ids, tag, set_tag=True*)

Starts an NSX Remediation request and returns the job object.

Required Permissions: appliances.nsx.remediation(EXECUTE)

Parameters

- **cb** (`BaseAPI`) – Reference to API object used to communicate with the server.
- **device_ids** (*int/list*) – The device ID(s) to run the remediation request on.
- **tag** (*str*) – The NSX tag to apply to specified devices. Valid values are “CB-NSX-Quarantine”, “CB-NSX-Isolate”, and “CB-NSX-Custom”.
- **set_tag** (*bool*) – True to toggle the specified tag on, False to toggle it off. Default True.

Returns The object representing all running jobs.

Return type *NSXRemediationJob*

Raises

- *ApiError* – If the parameters to start the request are incorrect.
- *ServerError* – If the request could not be successfully started.

status

Returns the current status.

Returns The current status, mapping individual job IDs to status value dicts.

Return type dict

5.7.3 cbc_sdk.workload.sensor_lifecycle module

Sensor Lifecycle Management for Workloads

class *SensorKit* (*cb*, *initial_data=None*)

Bases: *cbc_sdk.base.UnrefreshableModel*

Represents the information about a sensor, including installation file URLs.

Parameters

- **sensor_type** – The type of information this sensor is for.
- **sensor_url** – The URL for downloading the sensor installation package.
- **sensor_config_url** – The URL for downloading the sensor configuration information.
- **error_code** – Code for any error that occurred while getting the sensor information.
- **message** – Message for any error that occurred while getting the sensor information.

Initialize the SensorKit object.

Parameters

- **cb** (*BaseAPI*) – Reference to API object used to communicate with the server.
- **initial_data** (*dict*) – Initial data used to populate the sensor kit data.

```
COMPUTE_RESOURCE_MAP = {'CENTOS': 'RHEL', 'ORACLE': 'RHEL', 'SLES': 'SUSE'}
```

```
VALID_ARCHITECTURES = ['32', '64', 'OTHER']
```

```
VALID_DEVICE_TYPES = ['WINDOWS', 'LINUX', 'MAC']
```

```
VALID_TYPES = ['WINDOWS', 'MAC', 'RHEL', 'UBUNTU', 'SUSE', 'AMAZON_LINUX']
```

```
error_code = None
```

classmethod *from_type* (*cb*, *device_type*, *architecture*, *sensor_type*, *version*)

Helper method used to create a temporary SensorKit object from its four components.

This method CANNOT be used to create an object that will be persisted to the server.

Parameters

- **cb** (*BaseAPI*) – Reference to API object used to communicate with the server.
- **device_type** (*str*) – Device type to be used. Valid values are “WINDOWS”, “LINUX”, and “MAC”.

- **architecture** (*str*) – Architecture to be used. Valid values are “32”, “64”, and “OTHER”.
- **sensor_type** (*str*) – Sensor type to be used. Valid values are “WINDOWS”, “MAC”, “RHEL”, “UBUNTU”, “SUSE”, and “AMAZON_LINUX”.
- **version** (*str*) – Sensor version number to be used.

Returns A `SensorType` object with those specified values.

Return type `SensorType`

Raises `ApiError` – If an invalid value was used for one of the three limited values.

classmethod **get_config_template** (*cb*)

Retrieve the sample config.ini file with the properties populated from the server.

Parameters **cb** (`BaseAPI`) – Reference to API object used to communicate with the server.

Returns Text of the sample configuration file.

Return type `str`

`message = None`

`sensor_config_url = None`

`sensor_type = {}`

`sensor_url = None`

class **SensorKitQuery** (*doc_class, cb*)

Bases: `cbc_sdk.base.BaseQuery`, `cbc_sdk.base.CriteriaBuilderSupportMixin`, `cbc_sdk.base.IterableQueryMixin`, `cbc_sdk.base.AsyncQueryMixin`

Query class used to read in SensorKit objects.

Initialize the `SensorKitQuery`.

Parameters

- **doc_class** (*class*) – The model class that will be returned by this query.
- **cb** (`BaseAPI`) – Reference to API object used to communicate with the server.

add_sensor_kit_type (*skit=None, **kwargs*)

Add a sensor kit type to the request.

Parameters

- **skit** (`SensorKit`) – The sensor kit type to be added to the request.
- ****kwargs** (*dict*) – If `skit` is `None`, the keyword arguments ‘device_type’, ‘architecture’, ‘sensor_type’, and ‘version’ are used to create the sensor kit type to be added.

Returns Reference to this object.

Return type `SensorKitQuery`

config_params (*params*)

Sets the configuration parameters for the sensor kit query request.

Parameters **params** (*str*) – The text of a config.ini file with a list of sensor properties to configure on installation.

Returns Reference to this object.

Return type `SensorKitQuery`

expires (*expiration_date_time*)

Sets the expiration date and time for the sensor kit query request.

Parameters **expiration_date_time** (*str*) – The time at which the sensor download link will expire, expressed as ISO 8601 UTC.

Returns Reference to this object.

Return type *SensorKitQuery*

5.7.4 cbc_sdk.workload.vm_workloads_search module

Model and Query Classes for VM Workloads Search API

class **AWSComputeResource** (*cb, model_unique_id, initial_data=None*)

Bases: *cbc_sdk.workload.vm_workloads_search.BaseComputeResource*

Models an AWS compute resource.

Initialize the AWSComputeResource object.

Parameters

- **cb** (*BaseAPI*) – Reference to API object used to communicate with the server.
- **model_unique_id** (*str*) – ID of the alert represented.
- **initial_data** (*dict*) – Initial data used to populate the alert.

classmethod **bulk_install** (*cb, compute_resources, sensor_kit_types, config_file=None*)

Install a sensor on a list of compute resources.

Parameters

- **cb** (*BaseAPI*) – Reference to API object used to communicate with the server.
- **compute_resources** (*list*) – A list of ComputeResource objects used to specify compute resources to install sensors on.
- **sensor_kit_types** (*list*) – A list of SensorKit objects used to specify sensor types to choose from in installation.
- **config_file** (*str*) – The text of a config.ini file with a list of sensor properties to configure on installation.

Returns A dict with two members, ‘type’ and ‘code’, indicating the status of the installation.

Return type dict

Raises *NotImplementedError* – Always, for BaseComputeResource.

classmethod **bulk_install_by_id** (*cb, compute_resources, sensor_kit_types, config_file=None*)

Install a sensor on a list of compute resources, specified by ID.

Parameters

- **cb** (*BaseAPI*) – Reference to API object used to communicate with the server.
- **compute_resources** (*list*) – A list of dicts, each of which contains the keys ‘vcenter_uuid’ and ‘compute_resource_id’, specifying the compute resources to install sensors on.
- **sensor_kit_types** (*list*) – A list of SensorKit objects used to specify sensor types to choose from in installation.

- **config_file** (*str*) – The text of a config.ini file with a list of sensor properties to configure on installation.

Returns A dict with two members, 'type' and 'code', indicating the status of the installation.

Return type dict

Raises `NotImplementedError` – Always, for `BaseComputeResource`.

install_sensor (*sensor_version*, *config_file=None*)

Install a sensor on this compute resource.

Parameters

- **sensor_version** (*str*) – The version number of the sensor to be used.
- **config_file** (*str*) – The text of a config.ini file with a list of sensor properties to configure on installation.

Returns A dict with two members, 'type' and 'code', indicating the status of the installation.

Return type dict

Raises `NotImplementedError` – Always, for `BaseComputeResource`.

class `AWSComputeResourceQuery` (*doc_class*, *cb*)

Bases: `cbc_sdk.workload.vm_workloads_search.BaseComputeResourceQuery`

Represents a query that is used to locate `AWSComputeResource` objects.

Initialize the `ComputeResourceQuery`.

Parameters

- **doc_class** (*class*) – The model class that will be returned by this query.
- **cb** (`BaseAPI`) – Reference to API object used to communicate with the server.

VALID_INSTALLATION_STATUS = ('SUCCESS', 'ERROR', 'PENDING', 'NOT_INSTALLED')

exclude_auto_scaling_group_name (*auto_scaling_group_name*)

Excludes the specified auto scaling group name from appearing in the search results.

Parameters **auto_scaling_group_name** (*list*) – List of string auto scaling group names.

Returns This instance.

Return type `AWSComputeResourceQuery`

exclude_availability_zone (*availability_zone*)

Excludes the specified availability zone from appearing in the search results.

Parameters **availability_zone** (*list*) – List of string availability zones.

Returns This instance.

Return type `AWSComputeResourceQuery`

exclude_cloud_provider_account_id (*cloud_provider_account_id*)

Excludes the specified cloud provider account ID from appearing in the search results.

Parameters **cloud_provider_account_id** (*list*) – List of string cloud provider account IDs.

Returns This instance.

Return type `AWSComputeResourceQuery`

exclude_cloud_provider_resource_id (*cloud_provider_resource_id*)

Excludes the specified cloud provider resource ID from appearing in the search results.

Parameters **cloud_provider_resource_id** (*list*) – List of string cloud provider resource IDs.

Returns This instance.

Return type *AWSComputeResourceQuery*

exclude_cloud_provider_tags (*cloud_provider_tags*)

Excludes the specified cloud provider tags from appearing in the search results.

Parameters **cloud_provider_tags** (*list*) – List of string cloud provider tags.

Returns This instance.

Return type *AWSComputeResourceQuery*

exclude_id (*id_value*)

Excludes the specified compute resource ID from appearing in the search results.

Parameters **id_value** (*list*) – List of string compute resource IDs.

Returns This instance.

Return type *AWSComputeResourceQuery*

exclude_installation_status (*installation_status*)

Excludes the specified installation status from appearing in the search results.

Parameters **installation_status** (*list*) – List of string installation statuses.

Returns This instance.

Return type *AWSComputeResourceQuery*

exclude_name (*name*)

Excludes the specified compute resource name from appearing in the search results.

Parameters **name** (*list*) – List of string compute resource names.

Returns This instance.

Return type *AWSComputeResourceQuery*

exclude_platform (*platform*)

Excludes the specified platform from appearing in the search results.

Parameters **platform** (*list*) – List of string platforms.

Returns This instance.

Return type *AWSComputeResourceQuery*

exclude_platform_details (*platform_details*)

Excludes the specified platform details from appearing in the search results.

Parameters **platform_details** (*list*) – List of string platform details.

Returns This instance.

Return type *AWSComputeResourceQuery*

exclude_region (*region*)

Excludes the specified region from appearing in the search results.

Parameters **region** (*list*) – List of string regions.

Returns This instance.

Return type *AWSComputeResourceQuery*

exclude_subnet_id (*subnet_id*)

Excludes the specified subnet ID from appearing in the search results.

Parameters **subnet_id** (*list*) – List of string subnet IDs.

Returns This instance.

Return type *AWSComputeResourceQuery*

exclude_virtual_private_cloud_id (*virtual_private_cloud_id*)

Excludes the specified virtual private cloud ID from appearing in the search results.

Parameters **virtual_private_cloud_id** (*list*) – List of string virtual private cloud IDs.

Returns This instance.

Return type *AWSComputeResourceQuery*

set_auto_scaling_group_name (*auto_scaling_group_name*)

Restricts the search that this query is performed on to the specified auto scaling group name.

Parameters **auto_scaling_group_name** (*list*) – List of string auto scaling group names.

Returns This instance.

Return type *AWSComputeResourceQuery*

set_availability_zone (*availability_zone*)

Restricts the search that this query is performed on to the specified availability zone.

Parameters **availability_zone** (*list*) – List of string availability zones.

Returns This instance.

Return type *AWSComputeResourceQuery*

set_cloud_provider_account_id (*cloud_provider_account_id*)

Restricts the search that this query is performed on to the specified cloud provider account ID.

Parameters **cloud_provider_account_id** (*list*) – List of string cloud provider account IDs.

Returns This instance.

Return type *AWSComputeResourceQuery*

set_cloud_provider_resource_id (*cloud_provider_resource_id*)

Restricts the search that this query is performed on to the specified cloud provider resource ID.

Parameters **cloud_provider_resource_id** (*list*) – List of string cloud provider resource IDs.

Returns This instance.

Return type *AWSComputeResourceQuery*

set_cloud_provider_tags (*cloud_provider_tags*)

Restricts the search that this query is performed on to the specified cloud provider tags.

Parameters **cloud_provider_tags** (*list*) – List of string cloud provider tags.

Returns This instance.

Return type *AWSComputeResourceQuery*

set_id (*id_value*)

Restricts the search that this query is performed on to the specified compute resource ID.

Parameters **id_value** (*list*) – List of string compute resource IDs.

Returns This instance.

Return type *AWSComputeResourceQuery*

set_installation_status (*installation_status*)

Restricts the search that this query is performed on to the specified installation status.

Parameters **installation_status** (*list*) – List of string installation statuses.

Returns This instance.

Return type *AWSComputeResourceQuery*

set_name (*name*)

Restricts the search that this query is performed on to the specified compute resource name.

Parameters **name** (*list*) – List of string compute resource names.

Returns This instance.

Return type *AWSComputeResourceQuery*

set_platform (*platform*)

Restricts the search that this query is performed on to the specified platform.

Parameters **platform** (*list*) – List of string platforms.

Returns This instance.

Return type *AWSComputeResourceQuery*

set_platform_details (*platform_details*)

Restricts the search that this query is performed on to the specified platform details.

Parameters **platform_details** (*list*) – List of string platform details.

Returns This instance.

Return type *AWSComputeResourceQuery*

set_region (*region*)

Restricts the search that this query is performed on to the specified region.

Parameters **region** (*list*) – List of string regions.

Returns This instance.

Return type *AWSComputeResourceQuery*

set_subnet_id (*subnet_id*)

Restricts the search that this query is performed on to the specified subnet ID.

Parameters **subnet_id** (*list*) – List of string subnet IDs.

Returns This instance.

Return type *AWSComputeResourceQuery*

set_virtual_private_cloud_id (*virtual_private_cloud_id*)

Restricts the search that this query is performed on to the specified virtual private cloud ID.

Parameters `virtual_private_cloud_id` (*list*) – List of string virtual private cloud IDs.

Returns This instance.

Return type *AWSComputeResourceQuery*

summarize (*summary_fields*)

Get compute resource summaries on required fields of the resources with the specified criteria.

Example

```
>>> from cbc_sdk import CBCloudAPI
>>> from cbc_sdk.workload import AWSComputeResource
>>> cbc = CBCloudAPI()
>>> query = cbc.select(AWSComputeResource)
>>> summary = query.summarize(['availability_zone', 'region', 'virtual_
↳private_cloud_id'])
```

Required Permissions: public.cloud.inventory(READ) or _API.Public.Cloud:Public.cloud.inventory:READ

Parameters `summary_fields` (*list[str]*) – The fields to be summarized.

Returns A mapping of field names to the number of resources with that field.

Return type `map[str, int]`

class `BaseComputeResource` (*cb, model_unique_id, initial_data=None*)

Bases: *cbc_sdk.base.NewBaseModel*

Internal BaseComputeResource model

Initialize the BaseComputeResource object.

Parameters

- **cb** (*BaseAPI*) – Reference to API object used to communicate with the server.
- **model_unique_id** (*str*) – ID of the compute resource represented.
- **initial_data** (*dict*) – Initial data used to populate the resource object.

classmethod `bulk_install` (*cb, compute_resources, sensor_kit_types, config_file=None*)

Install a sensor on a list of compute resources.

Parameters

- **cb** (*BaseAPI*) – Reference to API object used to communicate with the server.
- **compute_resources** (*list*) – A list of ComputeResource objects used to specify compute resources to install sensors on.
- **sensor_kit_types** (*list*) – A list of SensorKit objects used to specify sensor types to choose from in installation.
- **config_file** (*str*) – The text of a config.ini file with a list of sensor properties to configure on installation.

Returns A dict with two members, ‘type’ and ‘code’, indicating the status of the installation.

Return type `dict`

Raises `NotImplementedError` – Always, for BaseComputeResource.

classmethod bulk_install_by_id(*cb*, *compute_resources*, *sensor_kit_types*, *config_file=None*)

Install a sensor on a list of compute resources, specified by ID.

Parameters

- **cb** (*BaseAPI*) – Reference to API object used to communicate with the server.
- **compute_resources** (*list*) – A list of dicts, each of which contains the keys 'vcenter_uuid' and 'compute_resource_id', specifying the compute resources to install sensors on.
- **sensor_kit_types** (*list*) – A list of SensorKit objects used to specify sensor types to choose from in installation.
- **config_file** (*str*) – The text of a config.ini file with a list of sensor properties to configure on installation.

Returns A dict with two members, 'type' and 'code', indicating the status of the installation.

Return type dict

Raises *NotImplementedError* – Always, for *BaseComputeResource*.

install_sensor(*sensor_version*, *config_file=None*)

Install a sensor on this compute resource.

Parameters

- **sensor_version** (*str*) – The version number of the sensor to be used.
- **config_file** (*str*) – The text of a config.ini file with a list of sensor properties to configure on installation.

Returns A dict with two members, 'type' and 'code', indicating the status of the installation.

Return type dict

Raises *NotImplementedError* – Always, for *BaseComputeResource*.

primary_key = 'id'

urlobject = '/lcm/view/v2/orgs/{0}/compute_resources'

urlobject_single = '/lcm/view/v2/orgs/{0}/compute_resources/{1}?deployment_type={2}'

class BaseComputeResourceQuery(*doc_class*, *cb*)

Bases: *cbc_sdk.base.BaseQuery*, *cbc_sdk.base.QueryBuilderSupportMixin*, *cbc_sdk.base.CriteriaBuilderSupportMixin*, *cbc_sdk.base.IterableQueryMixin*, *cbc_sdk.base.AsyncQueryMixin*

Base class for compute resource queries, not intended for direct use.

Initialize the *BaseComputeResourceQuery*.

Parameters

- **doc_class** (*class*) – The model class that will be returned by this query.
- **cb** (*BaseAPI*) – Reference to API object used to communicate with the server.

DEFAULT_FACET_ROWS = 20

VALID_DEPLOYMENT_TYPE = ('WORKLOAD', 'AWS')

VALID_DIRECTIONS = ('ASC', 'DESC')

VALID_DOWNLOAD_FORMATS = ('JSON', 'CSV')

download (*download_format=None*)

Downloads all compute resources matching the specific criteria.

Example

```
>>> from cbc_sdk import CBCloudAPI
>>> from cbc_sdk.workload import VCenterComputeResource
>>> cbc = CBCloudAPI()
>>> query = cbc.select(VCenterComputeResource).set_os_type(["UBUNTU"]).set_
<-eligibility(["ELIGIBLE"])
>>> query.set_installation_status(["ERROR"])
>>> job = query.download("CSV")
>>> job.await_completion()
>>> print(job.get_output_as_string())
```

Required Permissions: public.cloud.inventory(READ) or _API.Public.Cloud:Public.cloud.inventory:READ, jobs.status(READ)

Parameters **download_format** (*str*) – The download format to be used. Valid values are “JSON” (the default) and “CSV”.

Returns Asynchronous job which will supply the results of the download when they’re complete.

Return type *Job*

Raises *ApiError* – If the format specified was not valid, or if the server did not properly return the job.

facet (*fields, rows=None*)

Facets all compute resources matching the specified criteria and returns the facet results.

Example

```
>>> from cbc_sdk import CBCloudAPI
>>> from cbc_sdk.workload import AWSComputeResource
>>> cbc = CBCloudAPI()
>>> query = cbc.select(AWSComputeResource)
>>> facets = query.facet(['platform', 'virtual_private_cloud_id'])
```

Required Permissions: public.cloud.inventory(READ) or _API.Public.Cloud:Public.cloud.inventory:READ

Parameters

- **fields** (*list[str]*) – List of the fields to be faceted on.
- **rows** (*int*) – Number of the top entries to return. Default is 20.

Returns The facet data.

Return type *list[ComputeResourceFacet]*

sort_by (*key, direction='ASC'*)

Sets the sorting behavior on a query’s results.

Example

```
>>> cb.select(ComputeResource).sort_by("name")
```

Parameters

- **key** (*str*) – The key in the schema to sort by.
- **direction** (*str*) – The sort order.

Returns This instance.

Return type *BaseComputeResourceQuery*

class ComputeResourceFacet (*cb, model_unique_id, initial_data=None*)

Bases: *cbc_sdk.base.UnrefreshableModel*

Facet data returned by the facet() method of the query.

Initialize the ComputeResourceFacet object.

Parameters

- **cb** (*BaseAPI*) – Reference to API object used to communicate with the server.
- **model_unique_id** (*str*) – ID of the facet represented.
- **initial_data** (*dict*) – Initial data used to populate the facet.

class ComputeResourceFacetValue (*cb, model_unique_id, initial_data=None*)

Bases: *cbc_sdk.base.UnrefreshableModel*

Represents a single facet value inside a ComputeResourceFacet.

Initialize the ComputeResourceFacetValue object.

Parameters

- **cb** (*BaseAPI*) – Reference to API object used to communicate with the server.
- **model_unique_id** (*str*) – ID of the facet value represented.
- **initial_data** (*dict*) – Initial data used to populate the facet value.

values

Returns the values for this particular facet.

Returns The values of this facet.

Return type list[*ComputeResourceFacet.ComputeResourceFacetValue*]

class VCenterComputeResource (*cb, model_unique_id, initial_data=None*)

Bases: *cbc_sdk.workload.vm_workloads_search.BaseComputeResource*

Models a vCenter compute resource.

Initialize the VCenterComputeResource object.

Parameters

- **cb** (*BaseAPI*) – Reference to API object used to communicate with the server.
- **model_unique_id** (*str*) – ID of the alert represented.
- **initial_data** (*dict*) – Initial data used to populate the alert.

classmethod `bulk_install` (*cb*, *compute_resources*, *sensor_kit_types*, *config_file=None*)

Install a sensor on a list of compute resources.

Parameters

- **cb** (*BaseAPI*) – Reference to API object used to communicate with the server.
- **compute_resources** (*list*) – A list of `ComputeResource` objects used to specify compute resources to install sensors on.
- **sensor_kit_types** (*list*) – A list of `SensorKit` objects used to specify sensor types to choose from in installation.
- **config_file** (*str*) – The text of a `config.ini` file with a list of sensor properties to configure on installation.

Returns A dict with two members, ‘type’ and ‘code’, indicating the status of the installation.

Return type dict

classmethod `bulk_install_by_id` (*cb*, *compute_resources*, *sensor_kit_types*, *config_file=None*)

Install a sensor on a list of compute resources, specified by ID.

Parameters

- **cb** (*BaseAPI*) – Reference to API object used to communicate with the server.
- **compute_resources** (*list*) – A list of dicts, each of which contains the keys ‘vcenter_uuid’ and ‘compute_resource_id’, specifying the compute resources to install sensors on.
- **sensor_kit_types** (*list*) – A list of `SensorKit` objects used to specify sensor types to choose from in installation.
- **config_file** (*str*) – The text of a `config.ini` file with a list of sensor properties to configure on installation.

Returns A dict with two members, ‘type’ and ‘code’, indicating the status of the installation.

Return type dict

install_sensor (*sensor_version*, *config_file=None*)

Install a sensor on this compute resource.

Parameters

- **sensor_version** (*str*) – The version number of the sensor to be used.
- **config_file** (*str*) – The text of a `config.ini` file with a list of sensor properties to configure on installation.

Returns A dict with two members, ‘type’ and ‘code’, indicating the status of the installation.

Return type dict

Raises `ApiError` – If the compute node is not eligible or is of an invalid type.

class `VCenterComputeResourceQuery` (*doc_class*, *cb*)

Bases: `cbc_sdk.workload.vm_workloads_search.BaseComputeResourceQuery`

Represents a query that is used to locate `ComputeResource` objects.

Initialize the `ComputeResourceQuery`.

Parameters

- **doc_class** (*class*) – The model class that will be returned by this query.

- **cb** ([BaseAPI](#)) – Reference to API object used to communicate with the server.

VALID_ELIGIBILITY = ('ELIGIBLE', 'NOT_ELIGIBLE', 'UNSUPPORTED')

VALID_INSTALLATION_STATUS = ('SUCCESS', 'ERROR', 'PENDING', 'NOT_INSTALLED')

VALID_OS_ARCHITECTURE = ('32', '64')

VALID_OS_TYPE = ('WINDOWS', 'RHEL', 'UBUNTU', 'SUSE', 'SLES', 'CENTOS', 'OTHER', 'AMAZON')

exclude_appliance_uuid (*appliance_uuid*)

Excludes the specified appliance UUID from appearing in the search results.

Parameters **appliance_uuid** (*list*) – List of string appliance uuids.

Returns This instance.

Return type [VCenterComputeResourceQuery](#)

exclude_cluster_name (*cluster_name*)

Excludes the specified cluster name from appearing in the search results.

Parameters **cluster_name** (*list*) – List of string cluster names.

Returns This instance.

Return type [VCenterComputeResourceQuery](#)

exclude_datacenter_name (*datacenter_name*)

Excludes the specified datacenter name from appearing in the search results.

Parameters **datacenter_name** (*list*) – List of string datacenter names.

Returns This instance.

Return type [VCenterComputeResourceQuery](#)

exclude_device_guid (*device_guid*)

Excludes the specified device GUID from appearing in the search results.

Parameters **device_guid** (*list*) – List of string device GUIDs.

Returns This instance.

Return type [VCenterComputeResourceQuery](#)

exclude_eligibility (*eligibility*)

Excludes the specified eligibility from appearing in the search results.

Parameters **eligibility** (*list*) – List of string eligibilities.

Returns This instance.

Return type [VCenterComputeResourceQuery](#)

exclude_eligibility_code (*eligibility_code*)

Excludes the specified eligibility code from appearing in the search results.

Parameters **eligibility_code** (*list*) – List of string eligibility codes.

Returns This instance.

Return type [VCenterComputeResourceQuery](#)

exclude_esx_host_name (*esx_host_name*)

Excludes the specified ESX host name from appearing in the search results.

Parameters **esx_host_name** (*list*) – List of string ESX host names.

Returns This instance.

Return type *VCenterComputeResourceQuery*

exclude_esx_host_uuid (*esx_host_uuid*)

Excludes the specified ESX host UUID from appearing in the search results.

Parameters **esx_host_uuid** (*list*) – List of string ESX host UUIDs.

Returns This instance.

Return type *VCenterComputeResourceQuery*

exclude_host_name (*host_name*)

Excludes the specified host name from appearing in the search results.

Parameters **host_name** (*list*) – List of string host names.

Returns This instance.

Return type *VCenterComputeResourceQuery*

exclude_installation_status (*installation_status*)

Excludes the specified installation status from appearing in the search results.

Parameters **installation_status** (*list*) – List of string installation statuses.

Returns This instance.

Return type *VCenterComputeResourceQuery*

exclude_installation_type (*installation_type*)

Excludes the specified installation type from appearing in the search results.

Parameters **installation_type** (*list*) – List of string installation types.

Returns This instance.

Return type *VCenterComputeResourceQuery*

exclude_ip_address (*ip_address*)

Excludes the specified IP address from appearing in the search results.

Parameters **ip_address** (*list*) – List of string IP addresses.

Returns This instance.

Return type *VCenterComputeResourceQuery*

exclude_name (*name*)

Excludes the specified name from appearing in the search results.

Parameters **name** (*list*) – List of string names.

Returns This instance.

Return type *VCenterComputeResourceQuery*

exclude_os_architecture (*os_architecture*)

Excludes the specified OS architecture from appearing in the search results.

Parameters **os_architecture** (*list*) – List of string OS architectures.

Returns This instance.

Return type *VCenterComputeResourceQuery*

exclude_os_description (*os_description*)

Excludes the specified OS description from appearing in the search results.

Parameters `os_description` (*list*) – List of string OS descriptions.

Returns This instance.

Return type *VCenterComputeResourceQuery*

exclude_os_type (*os_type*)

Excludes the specified OS type from appearing in the search results.

Parameters `os_type` (*list*) – List of string OS types.

Returns This instance.

Return type *VCenterComputeResourceQuery*

exclude_registration_id (*registration_id*)

Excludes the specified registration ID from appearing in the search results.

Parameters `registration_id` (*list*) – List of string registration IDs.

Returns This instance.

Return type *VCenterComputeResourceQuery*

exclude_uuid (*uuid*)

Excludes the specified UUID from appearing in the search results.

Parameters `uuid` (*list*) – List of string UUIDs.

Returns This instance.

Return type *VCenterComputeResourceQuery*

exclude_vcenter_host_url (*vcenter_host_url*)

Excludes the specified vCenter host URL from appearing in the search results.

Parameters `vcenter_host_url` (*list*) – List of string vCenter host URLs.

Returns This instance.

Return type *VCenterComputeResourceQuery*

exclude_vcenter_name (*vcenter_name*)

Excludes the specified vCenter name from appearing in the search results.

Parameters `vcenter_name` (*list*) – List of string vCenter names.

Returns This instance.

Return type *VCenterComputeResourceQuery*

exclude_vcenter_uuid (*vcenter_uuid*)

Excludes the specified vCenter UUID from appearing in the search results.

Parameters `vcenter_uuid` (*list*) – List of string vCenter UUIDs.

Returns This instance.

Return type *VCenterComputeResourceQuery*

exclude_vmwaretools_version (*vmwaretools_version*)

Excludes the specified VMware Tools version from appearing in the search results.

Parameters `vmwaretools_version` (*list*) – List of string VMware Tools versions.

Returns This instance.

Return type *VCenterComputeResourceQuery*

set_appliance_uuid (*appliance_uuid*)

Restricts the search that this query is performed on to the specified appliance uuid.

Parameters **appliance_uuid** (*list*) – List of string appliance uuids.

Returns This instance.

Return type *VCenterComputeResourceQuery*

set_cluster_name (*cluster_name*)

Restricts the search that this query is performed on to the specified cluster name.

Parameters **cluster_name** (*list*) – List of string cluster names.

Returns This instance.

Return type *VCenterComputeResourceQuery*

set_datacenter_name (*datacenter_name*)

Restricts the search that this query is performed on to the specified datacenter name.

Parameters **datacenter_name** (*list*) – List of string datacenter names.

Returns This instance.

Return type *VCenterComputeResourceQuery*

set_device_guid (*device_guid*)

Restricts the search that this query is performed on to the specified device GUID.

Parameters **device_guid** (*list*) – List of string device GUIDs.

Returns This instance.

Return type *VCenterComputeResourceQuery*

set_eligibility (*eligibility*)

Restricts the search that this query is performed on to the specified eligibility.

Parameters **eligibility** (*list*) – List of string eligibilities.

Returns This instance.

Return type *VCenterComputeResourceQuery*

set_eligibility_code (*eligibility_code*)

Restricts the search that this query is performed on to the specified eligibility code.

Parameters **eligibility_code** (*list*) – List of string eligibility codes.

Returns This instance.

Return type *VCenterComputeResourceQuery*

set_esx_host_name (*esx_host_name*)

Restricts the search that this query is performed on to the specified ESX host name.

Parameters **esx_host_name** (*list*) – List of string ESX host names.

Returns This instance.

Return type *VCenterComputeResourceQuery*

set_esx_host_uuid (*esx_host_uuid*)

Restricts the search that this query is performed on to the specified ESX host UUID.

Parameters **esx_host_uuid** (*list*) – List of string ESX host UUIDs.

Returns This instance.

Return type *VCenterComputeResourceQuery*

set_host_name (*host_name*)

Restricts the search that this query is performed on to the specified host name.

Parameters **host_name** (*list*) – List of string host names.

Returns This instance.

Return type *VCenterComputeResourceQuery*

set_installation_status (*installation_status*)

Restricts the search that this query is performed on to the specified installation status.

Parameters **installation_status** (*list*) – List of string installation status.

Returns This instance.

Return type *VCenterComputeResourceQuery*

set_installation_type (*installation_type*)

Restricts the search that this query is performed on to the specified installation type.

Parameters **installation_type** (*list*) – List of string installation types.

Returns This instance.

Return type *VCenterComputeResourceQuery*

set_ip_address (*ip_address*)

Restricts the search that this query is performed on to the specified ip address.

Parameters **ip_address** (*list*) – List of string ip addresses.

Returns This instance.

Return type *VCenterComputeResourceQuery*

set_name (*name*)

Restricts the search that this query is performed on to the specified name.

Parameters **name** (*list*) – List of string names.

Returns This instance.

Return type *VCenterComputeResourceQuery*

set_os_architecture (*os_architecture*)

Restricts the search that this query is performed on to the specified os architecture.

Parameters **os_architecture** (*list*) – List of string os architecture.

Returns This instance.

Return type *VCenterComputeResourceQuery*

set_os_description (*os_description*)

Restricts the search that this query is performed on to the specified os description.

Parameters **os_description** (*list*) – List of string os description.

Returns This instance.

Return type *VCenterComputeResourceQuery*

set_os_type (*os_type*)

Restricts the search that this query is performed on to the specified os type.

Parameters **os_type** (*list*) – List of string os type.

Returns This instance.

Return type *VCenterComputeResourceQuery*

set_registration_id (*registration_id*)

Restricts the search that this query is performed on to the specified registration ID.

Parameters **registration_id** (*list*) – List of string registration IDs.

Returns This instance.

Return type *VCenterComputeResourceQuery*

set_uuid (*uuid*)

Restricts the search that this query is performed on to the specified uuid.

Parameters **uuid** (*list*) – List of string uuid.

Returns This instance.

Return type *VCenterComputeResourceQuery*

set_vcenter_host_url (*vcenter_host_url*)

Restricts the search that this query is performed on to the specified vCenter host URL.

Parameters **vcenter_host_url** (*list*) – List of string vCenter host URLs.

Returns This instance.

Return type *VCenterComputeResourceQuery*

set_vcenter_name (*vcenter_name*)

Restricts the search that this query is performed on to the specified vCenter name.

Parameters **vcenter_name** (*list*) – List of string vCenter names.

Returns This instance.

Return type *VCenterComputeResourceQuery*

set_vcenter_uuid (*vcenter_uuid*)

Restricts the search that this query is performed on to the specified vCenter UUID.

Parameters **vcenter_uuid** (*list*) – List of string vCenter UUIDs.

Returns This instance.

Return type *VCenterComputeResourceQuery*

set_vmwaretools_version (*vmwaretools_version*)

Restricts the search that this query is performed on to the specified VMware Tools version.

Parameters **vmwaretools_version** (*list*) – List of string VMware Tools versions.

Returns This instance.

Return type *VCenterComputeResourceQuery*

log = <Logger **cbc_sdk.workload.vm_workloads_search** (WARNING)>

Workloads Search model

5.7.5 Module contents

5.8 CBC SDK

5.8.1 Subpackages

cbc_sdk.cache package

Submodules

cbc_sdk.cache.lru module

LRU cache based on stucchio's py-lru-cache module

original copy at <https://github.com/stucchio/Python-LRU-cache> licensed under MIT

class LRUCacheDict (*max_size=1024, expiration=900, thread_clear=False, concurrent=True*)

Bases: object

A dictionary-like object, supporting LRU caching semantics.

```
>>> d = LRUCacheDict(max_size=3, expiration=3)
>>> d['foo'] = 'bar'
>>> d['foo']
'bar'
>>> import time
>>> time.sleep(4) # 4 seconds > 3 second cache expiry of d
>>> d['foo']
Traceback (most recent call last):
...
KeyError: 'foo'
>>> d['a'] = 'A'
>>> d['b'] = 'B'
>>> d['c'] = 'C'
>>> d['d'] = 'D'
>>> d['a'] # Should return value error, since we exceeded the max cache size
Traceback (most recent call last):
...
KeyError: 'a'
```

By default, this cache will only expire items whenever you poke it - all methods on this class will result in a cleanup. If the `thread_clear` option is specified, a background thread will clean it up every `thread_clear_min_check` seconds.

If this class must be used in a multithreaded environment, the option `concurrent` should be set to true. Note that the cache will always be concurrent if a background cleanup thread is used.

Initialize the LRUCacheDict object.

Parameters

- **max_size** (*int*) – Maximum number of elements in the cache.
- **expiration** (*int*) – Number of seconds an item can be in the cache before it expires.
- **thread_clear** (*bool*) – True if we want to use a background thread to keep the cache clear.
- **concurrent** (*bool*) – True to make access to the cache thread-safe.

```
class EmptyCacheThread (cache, peek_duration=60)
```

Bases: threading.Thread

Background thread that expires elements out of the cache.

Initialize the EmptyCacheThread.

Parameters

- **cache** (LRUCacheDict) – The cache to be monitored.
- **peek_duration** (int) – The delay between “sweeps” of the cache.

daemon = True

run ()

Execute the background cleanup.

cleanup (*args, **kwargs)

clear (*args, **kwargs)

has_key (*args, **kwargs)

size (*args, **kwargs)

```
class LRUCachedFunction (function, cache=None)
```

Bases: object

A memoized function, backed by an LRU cache.

```
>>> def f(x):
...     print "Calling f(" + str(x) + ")"
...     return x
>>> f = LRUCachedFunction(f, LRUCacheDict(max_size=3, expiration=3) )
>>> f(3)
Calling f(3)
3
>>> f(3)
3
>>> import time
>>> time.sleep(4) #Cache should now be empty, since expiration time is 3.
>>> f(3)
Calling f(3)
3
>>> f(4)
Calling f(4)
4
>>> f(5)
Calling f(5)
5
>>> f(3) #Still in cache, so no print statement. At this point, 4 is the least_
↪recently used.
3
>>> f(6)
Calling f(6)
6
>>> f(4) #No longer in cache - 4 is the least recently used, and there are at_
↪least 3 others
items in cache [3,4,5,6].
Calling f(4)
4
```

Initialize the LRUCachedFunction object.

Parameters

- **function** (*func*) – The function to be used to create new items in the cache.
- **cache** (`LRUCacheDict`) – The internal cache structure.

lru_cache_function (*max_size=1024, expiration=900*)
Least recently used cache function

```
>>> @lru_cache_function(3, 1)
... def f(x):
...     print "Calling f(" + str(x) + ")"
...     return x
>>> f(3)
Calling f(3)
3
>>> f(3)
3
```

Module contents

5.8.2 Submodules

5.8.3 cbc_sdk.base module

Models and Queries for the Base Carbon Black Cloud SDK

class ArrayFieldDescriptor (*field_name, coerce_to=None, default_value=None*)
Bases: `cbc_sdk.base.FieldDescriptor`

Field descriptor for fields of ‘array’ type.

Initialize the FieldDescriptor object.

Parameters

- **field_name** (*str*) – The name of the field.
- **coerce_to** (*class*) – The type to which the value should be coerced, or None.
- **default_value** (*Any*) – The default value of the field.

class AsyncQueryMixin

Bases: `object`

A mix-in which provides support for asynchronous queries.

execute_async ()

Executes the current query in an asynchronous fashion.

Returns A future representing the query and its results.

Return type Future

class BaseQuery (*query=None*)

Bases: `object`

The base query for finding objects via the API.

Initializes the BaseQuery object.

Parameters `query` (*solrq.Q*) – The parent query of this one.

class `BinaryFieldDescriptor` (*field_name, coerce_to=None, default_value=None*)

Bases: `cbc_sdk.base.FieldDescriptor`

Field descriptor for fields of 'byte' type.

Initialize the FieldDescriptor object.

Parameters

- **field_name** (*str*) – The name of the field.
- **coerce_to** (*class*) – The type to which the value should be coerced, or None.
- **default_value** (*Any*) – The default value of the field.

class `CbMetaModel`

Bases: `type`

Meta-model for NewBaseModel and its subclasses.

Creates a new instance of a class, setting up the field descriptors based on the metafile.

Parameters

- **name** (*str*) – The name of the class.
- **bases** (*list*) – Base classes of the class to be created.
- **clsdict** (*dict*) – Elements defined in the new class.

```
model_base_directory = '/home/docs/checkouts/readthedocs.org/user_builds/carbon-black-
```

```
model_classes = [<class 'cbc_sdk.base.NewBaseModel'>, <class 'cbc_sdk.base.Unrefreshab
```

class `CreatableModelMixin`

Bases: `object`

Mixin for all objects which are creatable.

class `CriteriaBuilderSupportMixin`

Bases: `object`

A mixin that supplies wrapper methods to access the `_criteria`.

add_criteria (*key, newlist*)

Add to the criteria on this query with a custom criteria key.

Will overwrite any existing criteria for the specified key.

Parameters

- **key** (*str*) – The key for the criteria item to be set.
- **newlist** (*str or list[str]*) – Value or list of values to be set for the criteria item.

Returns The query object with specified custom criteria.

Example

```
>>> query = api.select(Event).add_criteria("event_type", ["filemod",  
↪ "scriptload"])  
>>> query = api.select(Event).add_criteria("event_type", "filemod")
```

update_criteria (*key, newlist*)

Update the criteria on this query with a custom criteria key.

Parameters

- **key** (*str*) – The key for the criteria item to be set.
- **newlist** (*list*) – List of values to be set for the criteria item.

Returns The query object with specified custom criteria.

Example

```
>>> query = api.select(Alert).update_criteria("my.criteria.key", ["criteria_
↪value"])
```

Note: Use this method if there is no implemented method for your desired criteria.

class EpochDateTimeFieldDescriptor (*field_name, multiplier=1.0*)

Bases: *cbc_sdk.base.FieldDescriptor*

Field descriptor for fields of ‘epoch-ms-date-time’ type.

Initialize the EpochDateTimeFieldDescriptor object.

Parameters

- **field_name** (*str*) – The name of the field.
- **multiplier** (*float*) – Unused.

class FacetQuery (*cls, cb, query=None*)

Bases: *cbc_sdk.base.BaseQuery, cbc_sdk.base.AsyncQueryMixin, cbc_sdk.base.QueryBuilderSupportMixin, cbc_sdk.base.CriteriaBuilderSupportMixin*

Query class for asynchronous Facet API calls.

These API calls return one result, and are not paginated or iterable.

Initialize the FacetQuery object.

add_exclusions (*key, newlist*)

Add to the exclusions on this query with a custom exclusion key.

Parameters

- **key** (*str*) – The key for the exclusion item to be set.
- **newlist** (*str or list[str]*) – Value or list of values to be set for the exclusion item.

Returns The ResultQuery with specified custom exclusion.

Example

```
>>> query = api.select(Event).add_exclusions("netconn_domain", ["www.google.
↪com"])
>>> query = api.select(Event).add_exclusions("netconn_domain", "www.google.com
↪")
```

add_facet_field (*field*)

Sets the facet fields to be received by this query.

Parameters **field** (*str* or [*str*]) – Field(s) to be received.

Returns The Query object that will receive the specified field(s).

Return type [Query](#) (AsyncQuery)

Example

```
>>> cb.select(ProcessFacet).add_facet_field(["process_name", "process_username", "process_username"])
```

add_range (*range*)

Sets the facet ranges to be received by this query.

Parameters **range** (*dict* or [*dict*]) – Range(s) to be received.

Returns The Query object that will receive the specified range(s).

Return type [Query](#) (AsyncQuery)

Note: The range parameter must be in this dictionary format:

```
{
    "bucket_size": "<object>",
    "start": "<object>",
    "end": "<object>",
    "field": "<string>"
},
```

where “bucket_size”, “start”, and “end” can be numbers or ISO 8601 timestamps.

Examples

```
>>> cb.select(ProcessFacet).add_range({"bucket_size": 5, "start": 0, "end": 10, "field": "netconn_count"})
>>> cb.select(ProcessFacet).add_range({"bucket_size": "+1DAY", "start": "2020-11-01T00:00:00Z", "end": "2020-11-12T00:00:00Z", "field": "backend_timestamp"})
```

limit (*limit*)

Sets the maximum number of facets per category (i.e. any Process Search Fields in self._fields).

The default limit for Process Facet searches in the Carbon Black Cloud backend is 100.

Parameters **limit** (*int*) – Maximum number of facets per category.

Returns The Query object with new limit parameter.

Return type [Query](#) (AsyncQuery)

Example

```
>>> cb.select(ProcessFacet).where(process_name="foo.exe").limit(50)
```

results

Save query results to self._results with self._search() method.

set_rows (rows)

Sets the number of facet results to return with the query.

Parameters **rows** (*int*) – Number of rows to return.

Returns The Query object with the new rows parameter.

Return type *Query* (AsyncQuery)

Example

```
>>> cb.select(ProcessFacet).set_rows(50)
```

set_time_range (start=None, end=None, window=None)

Sets the 'time_range' query body parameter, determining a time window based on 'device_timestamp'.

Parameters

- **start** (*str in ISO 8601 timestamp*) – When to start the result search.
- **end** (*str in ISO 8601 timestamp*) – When to end the result search.
- **window** (*str*) – Time window to execute the result search, ending on the current time.
- **be in the form "-2w", where y=year, w=week, d=day, h=hour, m=minute, s=second. (Should)–**

Note:

- *window* will take precedent over *start* and *end* if provided.

Examples

```
>>> query = api.select(Event).set_time_range(start="2020-10-20T20:34:07Z")
>>> second_query = api.select(Event).
...     set_time_range(start="2020-10-20T20:34:07Z", end="2020-10-30T20:34:07Z
... ↵)
>>> third_query = api.select(Event).set_time_range(window='-3d')
```

timeout (msecs)

Sets the timeout on an AsyncQuery. By default, there is no timeout.

Parameters **msecs** (*int*) – Timeout duration, in milliseconds.

Returns The Query object with new milliseconds parameter.

Return type *Query* (AsyncQuery)

Example

```
>>> cb.select(ProcessFacet).where(process_name="foo.exe").timeout(5000)
```

class FieldDescriptor (*field_name, coerce_to=None, default_value=None*)

Bases: object

Object that describes a field within a model instance.

Initialize the FieldDescriptor object.

Parameters

- **field_name** (*str*) – The name of the field.
- **coerce_to** (*class*) – The type to which the value should be coerced, or None.
- **default_value** (*Any*) – The default value of the field.

class ForeignKeyFieldDescriptor (*field_name, join_model, join_field=None*)

Bases: *cbc_sdk.base.FieldDescriptor*

Field descriptor for fields that are foreign keys.

Initialize the ForeignKeyFieldDescriptor object.

Parameters

- **field_name** (*str*) – The name of the field.
- **join_model** (*class*) – The class for which this field value is a foreign key.
- **join_field** (*str*) – The name fo the field in the joined class for which this field value is a foreign key.

class IsoDateTimeFieldDescriptor (*field_name*)

Bases: *cbc_sdk.base.FieldDescriptor*

Field descriptor for fields of ‘iso-date-time’ type.

Initialize the IsoDateTimeFieldDescriptor object.

Parameters **field_name** (*str*) – The name of the field.

class IterableQueryMixin

Bases: object

A mix-in to provide iterability to a query.

all()

Returns all the items of a query as a list.

Returns List of query items

Return type list

first()

Returns the first item that would be returned as the result of a query.

Returns First query item

Return type obj

one()

Returns the only item that would be returned by a query.

Returns Sole query return item

Return type obj

Raises

- `MoreThanOneResultError` – If the query returns more than one item
- `ObjectNotFoundError` – If the query returns zero items

```
class MutableBaseModel (cb,    model_unique_id=None,    initial_data=None,    force_init=False,
                        full_doc=False)
```

Bases: `cbc_sdk.base.NewBaseModel`

Base model for objects that can have properties changed and then saved back to the server.

Initialize the `NewBaseModel` object.

Parameters

- **cb** (`CBCloudAPI`) – A reference to the `CBCloudAPI` object.
- **model_unique_id** (*Any*) – The unique ID for this particular instance of the model object.
- **initial_data** (*dict*) – The data to use when initializing the model object.
- **force_init** (*bool*) – True to force object initialization.
- **full_doc** (*bool*) – True to mark the object as fully initialized.

delete()

Delete this object.

is_dirty()

Returns whether or not any fields of this object have been changed.

Returns True if any fields of this object have been changed, False if not.

Return type bool

refresh()

Reload this object from the server.

reset()

Undo any changes made to this object's fields.

save()

Save any changes made to this object's fields.

Returns This object.

Return type `MutableBaseModel`

touch (*fulltouch=False*)

Force this object to be considered as changed.

validate()

Validates this object.

Returns True if the object is validated.

Return type bool

Raises `InvalidObjectError` – If the object has missing fields.

```
class NewBaseModel (cb,    model_unique_id=None,    initial_data=None,    force_init=False,
                    full_doc=False)
```

Bases: `object`

Base class of all model objects within the Carbon Black Cloud SDK.

Initialize the NewBaseModel object.

Parameters

- **cb** (`CBCloudAPI`) – A reference to the CBCloudAPI object.
- **model_unique_id** (*Any*) – The unique ID for this particular instance of the model object.
- **initial_data** (*dict*) – The data to use when initializing the model object.
- **force_init** (*bool*) – True to force object initialization.
- **full_doc** (*bool*) – True to mark the object as fully initialized.

MAX_LIST_ITEM_RENDER = 3

MAX_VALUE_WIDTH = 50

get (*attrname*, *default_val=None*)

Return an attribute of this object.

Parameters

- **attrname** (*str*) – Name of the attribute to be returned.
- **default_val** (*Any*) – Default value to be used if the attribute is not set.

Returns The returned attribute value, which may be defaulted.

Return type Any

classmethod new_object (*cb*, *item*, ***kwargs*)

Create a new object of a model class.

Parameters

- **cb** (`CBCloudAPI`) – Reference to the CBCloudAPI object.
- **item** (*dict*) – Item data to use to create the object.
- ****kwargs** (*dict*) – Additional keyword arguments.

Returns The new object instance.

Return type object

original_document

Returns the original meta-information about the object.

Returns The original meta-information about the object.

Return type object

primary_key = 'id'

refresh ()

Reload this object from the server.

class ObjectFieldDescriptor (*field_name*, *coerce_to=None*, *default_value=None*)

Bases: `cbc_sdk.base.FieldDescriptor`

Field descriptor for fields of 'object' type.

Initialize the FieldDescriptor object.

Parameters

- **field_name** (*str*) – The name of the field.
- **coerce_to** (*class*) – The type to which the value should be coerced, or None.
- **default_value** (*Any*) – The default value of the field.

class PaginatedQuery (*cls, cb, query=None*)

Bases: *cbc_sdk.base.BaseQuery, cbc_sdk.base.IterableQueryMixin*

A query that returns objects in a paginated fashion.

Initialize the PaginatedQuery object.

Parameters

- **cls** (*class*) – The class of objects being returned by this query.
- **cb** (*CBCloudAPI*) – Reference to the CBCloudAPI object.
- **query** (*BaseQuery*) – The query that we are paginating.

batch_size (*new_batch_size*)

Set the batch size of the paginated query.

Parameters **new_batch_size** (*int*) – The new batch size.

Returns A new query with the updated batch size.

Return type *PaginatedQuery*

class Query (*doc_class, cb*)

Bases: *cbc_sdk.base.PaginatedQuery, cbc_sdk.base.QueryBuilderSupportMixin, cbc_sdk.base.IterableQueryMixin, cbc_sdk.base.AsyncQueryMixin, cbc_sdk.base.CriteriaBuilderSupportMixin*

Represents a prepared query to the Carbon Black Cloud.

This object is returned as part of a *CBCCloudAPI.select* operation on models requested from the Carbon Black Cloud backend. You should not have to create this class yourself.

The query is not executed on the server until it's accessed, either as an iterator (where it will generate values on demand as they're requested) or as a list (where it will retrieve the entire result set and save to a list). You can also call the Python built-in `len()` on this object to retrieve the total number of items matching the query.

```
>>> from cbc_sdk import CBCloudAPI
>>> from cbc_sdk.enterprise_edr import Report
>>> cb = CBCloudAPI()
>>> query = cb.select(Report)
>>> query = query.where(report_id="ABCDEFG1234")
>>> # alternatively:
>>> query = query.where("report_id:ABCDEFG1234")
```

Notes

- The slicing operator only supports start and end parameters, but not step. `[1:-1]` is legal, but `[1:2:-1]` is not.
- You can chain where clauses together to create AND queries; only objects that match all where clauses will be returned.

Initialize the Query object.

Parameters

- **doc_class** (*class*) – The class of the model this query returns.
- **cb** ([CBCloudAPI](#)) – A reference to the CBCloudAPI object.

add_exclusions (*key, newlist*)

Add to the exclusions on this query with a custom exclusion key.

Parameters

- **key** (*str*) – The key for the exclusion item to be set.
- **newlist** (*str or list[str]*) – Value or list of values to be set for the exclusion item.

Returns The ResultQuery with specified custom exclusion.

Example

```
>>> query = api.select(Event).add_exclusions("netconn_domain", ["www.google.com"])
>>> query = api.select(Event).add_exclusions("netconn_domain", "www.google.com")
```

set_fields (*fields*)

Sets the fields to be returned with the response.

Parameters **fields** (*str or list[str]*) – Field or list of fields to be returned.

set_rows (*rows*)

Sets the ‘rows’ query body parameter, determining how many rows of results to request.

Parameters **rows** (*int*) – How many rows to request.

set_start (*start*)

Sets the ‘start’ query body parameter, determining where to begin retrieving results from.

Parameters **start** (*int*) – Where to start results from.

set_time_range (*start=None, end=None, window=None*)

Sets the ‘time_range’ query body parameter, determining a time window based on ‘device_timestamp’.

Parameters

- **start** (*str in ISO 8601 timestamp*) – When to start the result search.
- **end** (*str in ISO 8601 timestamp*) – When to end the result search.
- **window** (*str*) – Time window to execute the result search, ending on the current time. Should be in the form “-2w”, where y=year, w=week, d=day, h=hour, m=minute, s=second.

Note:

- *window* will take precedent over *start* and *end* if provided.
-

Examples

```
>>> query = api.select(Event).set_time_range(start="2020-10-20T20:34:07Z")
>>> second_query = api.select(Event).
...     set_time_range(start="2020-10-20T20:34:07Z", end="2020-10-30T20:34:07Z
↪")
>>> third_query = api.select(Event).set_time_range(window='-3d')
```

sort_by (*key*, *direction*='ASC')

Sets the sorting behavior on a query's results.

Parameters

- **key** (*str*) – The key in the schema to sort by.
- **direction** (*str*) – The sort order, either “ASC” or “DESC”.

Returns The query with sorting parameters.

Return type *Query*

Example

```
>>> cb.select(Process).where(process_name="cmd.exe").sort_by("device_timestamp
↪")
```

class QueryBuilder (***kwargs*)

Bases: *object*

Provides a flexible interface for building prepared queries for the CB Cloud backend.

This object can be instantiated directly, or can be managed implicitly through the CBCloudAPI.select API.

Examples

```
>>> from cbc_sdk.base import QueryBuilder
>>> # build a query with chaining
>>> query = QueryBuilder().where(process_name="malicious.exe").and_(device_name=
↪"suspect")
>>> # start with an initial query, and chain another condition to it
>>> query = QueryBuilder(device_os="WINDOWS").or_(process_username="root")
```

Initialize the QueryBuilder object.

Parameters ***kwargs* (*dict*) – If present, these are used to construct a Solrq Query.

and_ (*q*, ***kwargs*)

Adds a conjunctive filter to a QueryBuilder.

Parameters

- **q** (*object*) – Either a string or solrq.Q object representing the query to be added.
- ****kwargs** (*dict*) – Arguments with which to construct a solrq.Q object.

Returns This object.

Return type *QueryBuilder*

Raises *ApiError* – If the *q* parameter is of an invalid type.

not_ (*q*, ***kwargs*)

Adds a negative filter to a QueryBuilder.

Parameters

- **q** (*object*) – Either a string or `solrq.Q` object representing the query to be added.
- ****kwargs** (*dict*) – Arguments with which to construct a `solrq.Q` object.

Returns This object.

Return type [*QueryBuilder*](#)

Raises `ApiError` – If the `q` parameter is of an invalid type.

or_ (*q*, ***kwargs*)

Adds a disjunctive filter to a `QueryBuilder`.

Parameters

- **q** (*object*) – Either a string or `solrq.Q` object representing the query to be added.
- ****kwargs** (*dict*) – Arguments with which to construct a `solrq.Q` object.

Returns This object.

Return type [*QueryBuilder*](#)

Raises `ApiError` – If the `q` parameter is of an invalid type.

where (*q*, ***kwargs*)

Adds a conjunctive filter to a `QueryBuilder`.

Parameters

- **q** (*object*) – Either a string or `solrq.Q` object representing the query to be added.
- ****kwargs** (*dict*) – Arguments with which to construct a `solrq.Q` object.

Returns This object.

Return type [*QueryBuilder*](#)

Raises `ApiError` – If the `q` parameter is of an invalid type.

class `QueryBuilderSupportMixin`

Bases: `object`

A mixin that supplies wrapper methods to access the `_query_builder`.

and_ (*q=None*, ***kwargs*)

Add a conjunctive filter to this query.

Parameters

- **q** (*Any*) – Query string or `solrq.Q` object
- ****kwargs** (*dict*) – Arguments to construct a `solrq.Q` with

Returns This Query object.

Return type [*Query*](#)

not_ (*q=None*, ***kwargs*)

Adds a negated filter to this query.

Parameters

- **q** (*solrq.Q*) – Query object.
- ****kwargs** (*dict*) – Arguments to construct a `solrq.Q` with.

Returns This Query object.

Return type *Query*

or_ (*q=None, **kwargs*)

Add a disjunctive filter to this query.

Parameters

- **q** (*solrq.Q*) – Query object.
- ****kwargs** (*dict*) – Arguments to construct a *solrq.Q* with.

Returns This Query object.

Return type *Query*

where (*q=None, **kwargs*)

Add a filter to this query.

Parameters

- **q** (*Any*) – Query string, *QueryBuilder*, or *solrq.Q* object
- ****kwargs** (*dict*) – Arguments to construct a *solrq.Q* with

Returns This Query object.

Return type *Query*

class SimpleQuery (*cls, cb, urlobject=None, returns_fulldoc=True*)

Bases: *cbc_sdk.base.BaseQuery*, *cbc_sdk.base.IterableQueryMixin*

A simple query object.

Initialize the SimpleQuery object.

Parameters

- **cls** (*class*) – Class of the object to be returned by the query.
- **cb** (*CBCCloudAPI*) – Reference to the CBCCloudAPI object.
- **urlobject** (*str*) – URL to be used in making the query.
- **returns_fulldoc** (*bool*) – Whether the result of the Query yields objects that have been fully initialized.

and_ (*new_query*)

Add an additional “where” clause to this query.

Parameters **new_query** (*object*) – The additional “where” clause, as a string or *solrq.Q* object.

Returns A new query with the extra “where” clause specified.

Return type *SimpleQuery*

results

Collect and return the results of this query.

Returns The results of this query.

Return type list

sort (*new_sort*)

Set the sorting for this query.

Parameters **new_sort** (*object*) – The new sort criteria for this query.

Returns A new query with the sort parameter specified.

Return type *SimpleQuery*

where (*new_query*)

Add a “where” clause to this query.

Parameters **new_query** (*object*) – The “where” clause, as a string or solrq.Q object.

Returns A new query with the “where” clause specified.

Return type *SimpleQuery*

class **SwaggerLoader** (*stream*)

Bases: `yaml.loader.SafeLoader`

YAML loader class for loading Swagger metafiles.

yaml_constructors = {'tag:yaml.org,2002:null': <function SafeConstructor.construct_ya

class **UnrefreshableModel** (*cb, model_unique_id=None, initial_data=None, force_init=False, full_doc=False*)

Bases: *cbc_sdk.base.NewBaseModel*

Represents a model that can’t be refreshed, i.e. for which `reset()` is not a valid operation.

Initialize the *NewBaseModel* object.

Parameters

- **cb** (*CBCCloudAPI*) – A reference to the *CBCCloudAPI* object.
- **model_unique_id** (*Any*) – The unique ID for this particular instance of the model object.
- **initial_data** (*dict*) – The data to use when initializing the model object.
- **force_init** (*bool*) – True to force object initialization.
- **full_doc** (*bool*) – True to mark the object as fully initialized.

refresh ()

Reload this object from the server.

construct_include (*loader, node*)

Include the file referenced by the node.

Parameters

- **loader** (*yaml.Loader*) – YAML loader object.
- **node** (*yaml.Node*) – Current node being loaded.

Returns The data to be included in the YAML loader output.

Return type *Any*

log = <Logger *cbc_sdk.base (WARNING)*>

Base Models

5.8.4 cbc_sdk.connection module

Manages the CBC SDK connection to the server.

class **BaseAPI** (**args, **kwargs*)

Bases: *object*

The base API object used by all CBC SDK objects to communicate with the server.

Initialize the base API information.

Parameters

- ***args** – Unused.
- ****kwargs** – Additional arguments.

api_json_request (*method, uri, **kwargs*)

Submit a request to the server.

Parameters

- **method** (*str*) – HTTP method to use.
- **uri** (*str*) – URI to submit the request to.
- ****kwargs** (*dict*) – Additional arguments.

Returns Result of the operation.

Return type object

Raises `ServerError` – If there's an error output from the server.

api_request_iterate (*method, uri, **kwargs*)

Submit a request to the specified URI and iterate over the response as lines of text.

Should only be used for requests that can be expressed as large amounts of text that can be broken into lines. Since this is an iterator, call it with the 'yield from' syntax.

Parameters

- **method** (*str*) – HTTP method to use.
- **uri** (*str*) – The URI to send the request to.
- ****kwargs** (*dict*) – Additional arguments for the request.

Returns An iterable that can be used to get each line of text in turn as a string.

Return type iterable

api_request_stream (*method, uri, stream_output, **kwargs*)

Submit a request to the specified URI and stream the results back into the given stream object.

Parameters

- **method** (*str*) – HTTP method to use.
- **uri** (*str*) – The URI to send the request to.
- **stream_output** (*RawIOBase*) – The output stream to write the data to.
- ****kwargs** (*dict*) – Additional arguments for the request.

Returns The return data from the request.

Return type object

create (*cls, data=None*)

Create a new object.

Parameters

- **cls** (*class*) – The Model class (only some models can be created, for example, Feed, Notification, ...)
- **data** (*object*) – The data used to initialize the new object

Returns An empty instance of the model class.

Return type Model

Raises `ApiError` – If the Model cannot be created.

delete_object (*uri*)

Send a DELETE request to the specified URI.

Parameters **uri** (*str*) – The URI to send the DELETE request to.

Returns The return data from the DELETE request.

Return type object

get_object (*uri, query_parameters=None, default=None*)

Submit a GET request to the server and parse the result as JSON before returning.

Parameters

- **uri** (*str*) – The URI to send the GET request to.
- **query_parameters** (*object*) – Parameters for the query.
- **default** (*object*) – What gets returned in the event of an empty response.

Returns Result of the GET request.

Return type object

get_raw_data (*uri, query_parameters=None, default=None, **kwargs*)

Submit a GET request to the server and return the result without parsing it.

Parameters

- **uri** (*str*) – The URI to send the GET request to.
- **query_parameters** (*object*) – Parameters for the query.
- **default** (*object*) – What gets returned in the event of an empty response.
- ****kwargs** –

Returns Result of the GET request.

Return type object

post_multipart (*uri, param_table, **kwargs*)

Send a POST request to the specified URI, with parameters sent as multipart form data.

Parameters

- **uri** (*str*) – The URI to send the POST request to.
- **param_table** (*dict*) – A dict of known parameters to the underlying method, each element of which is a parameter name mapped to a dict, which contains elements ‘filename’ and ‘type’ representing the pseudo-filename to be used for the data and the MIME type of the data.
- ****kwargs** (*dict*) – Arguments to pass to the API. Except for “headers,” these will all be added as parameters to the form data sent.

Returns The return data from the POST request.

Return type object

post_object (*uri, body, **kwargs*)

Send a POST request to the specified URI.

Parameters

- **uri** (*str*) – The URI to send the POST request to.
- **body** (*object*) – The data to be sent in the body of the POST request.
- ****kwargs** (*dict*) – Additional arguments for the HTTP POST.

Returns The return data from the POST request.

Return type object

put_object (*uri, body, **kwargs*)

Send a PUT request to the specified URI.

Parameters

- **uri** (*str*) – The URI to send the PUT request to.
- **body** (*object*) – The data to be sent in the body of the PUT request.
- ****kwargs** –

Returns The return data from the PUT request.

Return type object

raise_unless_json (*ret, expected*)

Raise a `ServerError` unless we got back an HTTP 200 response with JSON containing all the expected values.

Parameters

- **ret** (*object*) – Return value to be checked.
- **expected** (*dict*) – Expected keys and values that need to be found in the JSON response.

Raises `ServerError` – If the HTTP response is anything but 200, or if the expected values are not found.

select (*cls, unique_id=None, *args, **kwargs*)

Prepare a query against the Carbon Black data store.

Parameters

- **cls** (*class | str*) – The Model class (for example, `Computer`, `Process`, `Binary`, `File-Instance`) to query
- **unique_id** (*optional*) – The unique id of the object to retrieve, to retrieve a single object by ID
- ***args** –
- ****kwargs** –

Returns An instance of the Model class if a `unique_id` is provided, otherwise a `Query` object

Return type object

url

Return the connection URL.

Returns The connection URL.

Return type str

```
class CBCSDKSessionAdapter (verify_hostname=True,    force_tls_1_2=False,    max_retries=0,
                           **pool_kwargs)
```

Bases: `requests.adapters.HTTPAdapter`

Adapter object used to handle TLS connections to the CB server.

Initialize the CBCSDKSessionManager.

Parameters

- **verify_hostname** (*boolean*) – True if we want to verify the hostname.
- **force_tls_1_2** (*boolean*) – True to force the use of TLS 1.2.
- **max_retries** (*int*) – Maximum number of retries.
- ****pool_kwargs** – Additional arguments.

Raises `ApiError` – If the library versions are too old to force the use of TLS 1.2.

```
init_poolmanager (connections, maxsize, block=False, **pool_kwargs)
```

Initialize the connection pool manager.

Parameters

- **connections** (*int*) – Initial number of connections to be used.
- **maxsize** (*int*) – Maximum size of the connection pool.
- **block** (*object*) – Blocking policy.
- ****pool_kwargs** – Additional arguments for the connection pool.

Returns `None`

```
class Connection (credentials,    integration_name=None,    timeout=None,    max_retries=None,
                  proxy_session=None, **pool_kwargs)
```

Bases: `object`

Object that encapsulates the HTTP connection to the CB server.

Initialize the Connection object.

Parameters

- **credentials** (*object*) – The credentials to use for the connection.
- **integration_name** (*str*) – The integration name being used.
- **timeout** (*int*) – The timeout value to use for HTTP requests on this connection.
- **max_retries** (*int*) – The maximum number of times to retry a request.
- **proxy_session** (*requests.Session*) –
- ****pool_kwargs** – Additional arguments to be used to initialize connection pooling.

Raises

- `ApiError` – If there's an internal error initializing the connection.
- `ConnectionError` – If there's a problem with the credentials.

```
delete (url, **kwargs)
```

Submit a DELETE request on this connection.

Parameters

- **url** (*str*) – The URL to submit the request to.

- ****kwargs** – Additional arguments for the request.

Returns Result of the HTTP request.

Return type object

get (*url*, ***kwargs*)

Submit a GET request on this connection.

Parameters

- **url** (*str*) – The URL to submit the request to.
- ****kwargs** – Additional arguments for the request.

Returns Result of the HTTP request.

Return type object

http_request (*method*, *url*, ***kwargs*)

Submit a HTTP request to the server.

Parameters

- **method** (*str*) – The method name to use for the HTTP request.
- **url** (*str*) – The URL to submit the request to.
- ****kwargs** – Additional arguments for the request.

Returns Result of the HTTP request.

Return type object

Raises

- **ApiError** – An unknown problem was detected.
- **ClientError** – The server returned an error code in the 4xx range, indicating a problem with the request.
- **ConnectionError** – A problem was seen with the HTTP connection.
- **ObjectNotFoundError** – The specified object was not found on the server.
- **QuerySyntaxError** – The query passed in had invalid syntax.
- **ServerError** – The server returned an error code in the 5xx range, indicating a problem on the server side.
- **TimeoutError** – The HTTP request timed out.
- **UnauthorizedError** – The stored credentials do not permit access to the specified request.

post (*url*, ***kwargs*)

Submit a POST request on this connection.

Parameters

- **url** (*str*) – The URL to submit the request to.
- ****kwargs** – Additional arguments for the request.

Returns Result of the HTTP request.

Return type object

put (*url*, ****kwargs**)

Submit a PUT request on this connection.

Parameters

- **url** (*str*) – The URL to submit the request to.
- ****kwargs** – Additional arguments for the request.

Returns Result of the HTTP request.

Return type object

check_python_tls_compatibility ()

Verify which level of TLS/SSL that this version of the code is compatible with.

Returns The maximum level of TLS/SSL that this version is compatible with.

Return type str

select_class_instance (*cls: str*)

Selecting the appropriate class based on the passed string.

Parameters **cls** – The class name represented in a string.

Returns

Return type Object[]

try_json (*resp*)

Return a parsed JSON representation of the input.

Parameters **resp** (*Response*) – Input to be parsed.

Returns The parsed JSON result, or an empty dict if the value is not valid JSON.

Return type object

5.8.5 cbc_sdk.credentials module

Credentials management for the CBC SDK.

class CredentialProvider

Bases: object

The interface implemented by a credential provider.

get_credentials (*section=None*)

Return a Credentials object containing the configured credentials.

Parameters **section** (*str*) – The credential section to retrieve.

Returns The credentials retrieved from that source.

Return type *Credentials*

Raises *CredentialError* – If there is any error retrieving the credentials.

class CredentialValue

Bases: `enum.Enum`

All possible credential values.

CSP_API_TOKEN = 13

CSP_OAUTH_APP_ID = 11

```
CSP_OAUTH_APP_SECRET = 12
```

```
CSP_URL_OVERRIDE = 14
```

```
IGNORE_SYSTEM_PROXY = 9
```

```
INTEGRATION = 10
```

```
ORG_KEY = 3
```

```
PROXY = 8
```

```
SSL_CERT_FILE = 6
```

```
SSL_FORCE_TLS_1_2 = 7
```

```
SSL_VERIFY = 4
```

```
SSL_VERIFY_HOSTNAME = 5
```

```
TOKEN = 2
```

```
URL = 1
```

```
requires_boolean_value()
```

Return whether or not this credential requires a boolean value.

Returns True if the credential requires a Boolean value, False if not.

Return type bool

```
class Credentials(values=None)
```

Bases: object

The object that contains credentials retrieved from the credential provider.

Initialize the Credentials object.

Parameters *values* (*dict*) – Dictionary containing values to be set in the credentials.

Raises `CredentialError` – If the value is not correct for any credential of boolean type.

```
get_token()
```

Get token required to authenticate with VMware Carbon Black Cloud

Returns Token string for VMware Carbon Black Cloud

Return type str

```
get_token_type()
```

Get token type API_KEY or BEARER

Returns The token type

Return type str

```
get_value(key)
```

Get the value of a credential.

Parameters *key* (*CredentialValues*) – The credential to be retrieved.

Returns The credential's value, or a default value if the value was not explicitly set.

Return type object

```
to_dict()
```

Serializes the credentials into a dictionary.

Returns Dictionary with the credentials.

Return type dict

5.8.6 cbc_sdk.errors module

Exceptions that are thrown by CBC SDK operations.

exception ApiError (*message=None, original_exception=None*)

Bases: `Exception`

Base class for all CBC SDK errors; also raised for generic internal errors.

Initialize the ApiError.

Parameters

- **message** (*str*) – The actual error message.
- **original_exception** (*Exception*) – The exception that caused this one to be raised.

exception ClientError (*error_code, message, result=None, original_exception=None*)

Bases: `cbc_sdk.errors.ApiError`

A ClientError is raised when an HTTP 4xx error code is returned from the Carbon Black server.

Initialize the ClientError.

Parameters

- **error_code** (*int*) – The error code that was received from the server.
- **message** (*str*) – The actual error message.
- **result** (*object*) – The result of the operation from the server.
- **original_exception** (*Exception*) – The exception that caused this one to be raised.

exception ConnectionError (*message=None, original_exception=None*)

Bases: `cbc_sdk.errors.ApiError`

There was an error in the connection to the server.

Initialize the ApiError.

Parameters

- **message** (*str*) – The actual error message.
- **original_exception** (*Exception*) – The exception that caused this one to be raised.

exception CredentialError (*message=None, original_exception=None*)

Bases: `cbc_sdk.errors.ApiError`

The credentials had an unspecified error.

Initialize the ApiError.

Parameters

- **message** (*str*) – The actual error message.
- **original_exception** (*Exception*) – The exception that caused this one to be raised.

exception FunctionalityDecommissioned (*functionality_tag, alternate=None*)

Bases: `cbc_sdk.errors.ApiError`

Raised when a piece of decommissioned functionality is used.

Initialize the FunctionalityDecommissioned exception.

Parameters

- **functionality_tag** (*str*) – Should indicate which functionality has been decommisioned.
- **alternate** (*str*) – Optional indication of what the replacement for this functionality is.

exception InvalidHashError

Bases: `Exception`

An invalid hash value was used.

exception InvalidObjectError (*message=None, original_exception=None*)

Bases: `cbc_sdk.errors.ApiError`

An invalid object was received by the server.

Initialize the ApiError.

Parameters

- **message** (*str*) – The actual error message.
- **original_exception** (*Exception*) – The exception that caused this one to be raised.

exception ModelNotFound

Bases: `Exception`

Exception for not finding a model while selecting dynamically.

exception MoreThanOneResultError (*message=None, original_exception=None, results=None*)

Bases: `cbc_sdk.errors.ApiError`

Only one object was requested, but multiple matches were found in the Carbon Black datastore.

Initialize the MoreThanOneResultError.

Parameters

- **message** (*str*) – The actual error message.
- **original_exception** (*Exception*) – The exception that caused this one to be raised.
- **results** (*list*) – List of results returned

exception NSXJobError (*message=None, original_exception=None*)

Bases: `cbc_sdk.errors.ApiError`

NSX remediation jobs were not started

Initialize the ApiError.

Parameters

- **message** (*str*) – The actual error message.
- **original_exception** (*Exception*) – The exception that caused this one to be raised.

exception NonQueryableModel (*message=None, original_exception=None*)

Bases: `cbc_sdk.errors.ApiError`

A model that attempted to be queried which is not queryable

Initialize the ApiError.

Parameters

- **message** (*str*) – The actual error message.

- **original_exception** (*Exception*) – The exception that caused this one to be raised.

exception ObjectNotFoundError (*uri, message=None, original_exception=None*)

Bases: *cbc_sdk.errors.ApiError*

The requested object could not be found in the Carbon Black datastore.

Initialize the ObjectNotFoundError.

Parameters

- **uri** (*str*) – The URI of the action that failed.
- **message** (*str*) – The error message.
- **original_exception** (*Exception*) – The exception that caused this one to be raised.

exception OperationCancelled (*message=None, original_exception=None*)

Bases: *cbc_sdk.errors.ApiError*

An operation in the background was canceled.

Initialize the ApiError.

Parameters

- **message** (*str*) – The actual error message.
- **original_exception** (*Exception*) – The exception that caused this one to be raised.

exception QuerySyntaxError (*uri, message=None, original_exception=None*)

Bases: *cbc_sdk.errors.ApiError*

The request contains a query with malformed syntax.

Initialize the QuerySyntaxError.

Parameters

- **uri** (*str*) – The URI of the action that failed.
- **message** (*str*) – The error message.
- **original_exception** (*Exception*) – The exception that caused this one to be raised.

exception ServerError (*error_code, message, result=None, original_exception=None*)

Bases: *cbc_sdk.errors.ApiError*

A ServerError is raised when an HTTP 5xx error code is returned from the Carbon Black server.

Initialize the ServerError.

Parameters

- **error_code** (*int*) – The error code that was received from the server.
- **message** (*str*) – The actual error message.
- **result** (*object*) – The result of the operation from the server.
- **original_exception** (*Exception*) – The exception that caused this one to be raised.

exception TimeoutError (*uri=None, error_code=None, message=None, original_exception=None*)

Bases: *cbc_sdk.errors.ApiError*

A requested operation timed out.

Initialize the TimeoutError.

Parameters

- **uri** (*str*) – The URI of the action that timed out.
- **error_code** (*int*) – The error code that was received from the server.
- **message** (*str*) – The error message.
- **original_exception** (*Exception*) – The exception that caused this one to be raised.

exception UnauthorizedError (*uri, message=None, action='read', original_exception=None*)

Bases: *cbc_sdk.errors.ApiError*

The action that was attempted was not authorized.

Initialize the UnauthorizedError.

Parameters

- **uri** (*str*) – The URI of the action that was not authorized.
- **message** (*str*) – The error message.
- **action** (*str*) – The action that was being performed that was not authorized.
- **original_exception** (*Exception*) – The exception that caused this one to be raised.

5.8.7 cbc_sdk.helpers module

Helper functions which are not strictly part of the SDK API, but which are used by many of the examples.

build_cli_parser (*description='Cb Example Script'*)

Build a basic CLI parser containing the arguments needed to create a CBCloudAPI. Additional arguments may be added.

Parameters **description** (*str*) – Description of the script, for use in help messages.

Returns The new argument parser.

Return type ArgumentParser

disable_insecure_warnings ()

Disable warnings about insecure URLs.

eprint (**args, **kwargs*)

Print to standard error output.

Parameters

- ***args** (*list*) – Arguments to the print function.
- ****kwargs** (*dict*) – Keyword arguments to the print function.

get_cb_cloud_object (*args*)

Based on parsed command line arguments, create and return a CBCloudAPI object.

Parameters **args** (*Namespace*) – Arguments parsed from the command line.

Returns The CBCloudAPI object.

Return type *CBCloudAPI*

get_object_by_name_or_id (*cb, cls, name_field='name', id=None, name=None*)

Locate an object in the API by either ID or name.

Parameters

- **cb** (*CBCloudAPI*) – Reference to the CBCloudAPI.

- **cls** (*class*) – Class of object to be found.
- **name_field** (*str*) – Name field to search on.
- **id** (*int*) – ID of object to search for. May be None to do name searching.
- **name** (*str*) – Object name to search on.
- **force_init** (*bool*) – True to force a new object found by ID to be initialized.

Returns List of objects that match the search criteria.

Return type list

read_iocs (*cb*, *file*=<_io.TextIOWrapper name='<stdin>' mode='r' encoding='UTF-8'>)

Read indicators of compromise from standard input.

Parameters

- **cb** ([CBCloudAPI](#)) – Reference to the CBCloudAPI.
- **file** – Not used.

Returns New report ID to be used. dict: The indicators of compromise that were read in.

Return type str

5.8.8 cbc_sdk.live_response_api module

The Live Response API and associated objects.

class CbLRManagerBase (*cb*, *timeout*=30, *keepalive_sessions*=False, *thread_pool_count*=5)

Bases: object

Live Response manager object.

Initialize the CbLRManagerBase object.

Parameters

- **cb** ([BaseAPI](#)) – The CBC SDK object reference.
- **timeout** (*int*) – Timeout to use for requests, in seconds.
- **keepalive_sessions** (*bool*) – If True, “ping” sessions occasionally to ensure they stay alive.
- **thread_pool_count** (*int*) – number of workers for async commands (optional)

cbldr_base = ''

cbldr_session_cls = [NotImplemented](#)

close_session (*device_id*, *session_id*)

Close the specified Live Response session.

Parameters

- **device_id** (*int*) – ID of the device.
- **session_id** (*int*) – ID of the session.

request_session (*device_id*, *async_mode*=False)

Initiate a new Live Response session.

Parameters **device_id** (*int*) – The device ID to use.

Returns The new Live Response session.

Return type *CbLRSessionBase*

stop_keepalive_thread()

Stops the keepalive thread.

submit_job(*job, device*)

Submit a new job to be executed as a Live Response.

Parameters

- **job** (*func*) – The job function to be scheduled.
- **device** (*int*) – ID of the device to use for job execution.

Returns A reference to the running job.

Return type Future

```
class CbLRSessionBase(cblr_manager, session_id, device_id, session_data=None,  
                      thread_pool_count=5)
```

Bases: object

A Live Response session that interacts with a remote machine.

Initialize the CbLRSessionBase.

Parameters

- **cblr_manager** (*CbLRManagerBase*) – The Live Response manager governing this session.
- **session_id** (*str*) – The ID of this session.
- **device_id** (*int*) – The ID of the device (remote machine) we’re connected to.
- **session_data** (*dict*) – Additional session data.
- **thread_pool_count** (*int*) – number of workers for async commands (optional)

MAX_RETRY_COUNT = 5

cancel_command(*command_id*)

Cancel command if it is in status PENDING.

Parameters **command_id** (*int*) – command_id

close()

Close the Live Response session.

command_status(*command_id*)

Check the status of async command

Parameters **command_id** (*int*) – command_id

Returns status of the command

create_directory(*dir_name, async_mode=False*)

Create a directory on the remote machine.

Parameters

- **dir_name** (*str*) – The new directory name.
- **async_mode** (*bool*) – Flag showing whether the command should be executed asynchronously

Returns command_id, future if ran async

create_process (*command_string*, *wait_for_output=True*, *remote_output_file_name=None*, *working_directory=None*, *wait_timeout=30*, *wait_for_completion=True*, *async_mode=False*)

Create a new process on the remote machine with the specified command string.

Example

```
>>> with c.select(Device, 1).lr_session() as lr_session:
...     print(lr_session.create_process(r'cmd.exe /c "ping.exe 192.168.1.1"'))
Pinging 192.168.1.1 with 32 bytes of data:
Reply from 192.168.1.1: bytes=32 time<1ms TTL=64
```

Parameters

- **command_string** (*str*) – Command string used for the create process operation.
- **wait_for_output** (*bool*) – True to block on output from the new process (execute in foreground). This will also set **wait_for_completion** (below).
- **remote_output_file_name** (*str*) – The remote output file name used for process output.
- **working_directory** (*str*) – The working directory of the create process operation.
- **wait_timeout** (*int*) – Timeout used for this command.
- **wait_for_completion** (*bool*) – True to wait until the process is completed before returning.
- **async_mode** (*bool*) – Flag showing whether the command should be executed asynchronously

Returns *command_id*, future if ran async *str*: The output of the process.

create_registry_key (*regkey*, *async_mode=False*)

Create a new registry key on the remote machine.

Parameters

- **regkey** (*str*) – The registry key to create.
- **async_mode** (*bool*) – Flag showing whether the command should be executed asynchronously

Returns *command_id*, future if ran async

delete_file (*filename*, *async_mode=False*)

Delete the specified file name on the remote machine.

Parameters

- **filename** (*str*) – Name of the file to be deleted.
- **async_mode** (*bool*) – Flag showing whether the command should be executed asynchronously

Returns *command_id*, future if ran async

delete_registry_key (*regkey*, *async_mode=False*)

Delete a registry key on the remote machine.

Parameters

- **regkey** (*str*) – The registry key to delete.
- **async_mode** (*bool*) – Flag showing whether the command should be executed asynchronously

Returns command_id, future if ran async

delete_registry_value (*regkey, async_mode=False*)

Delete a registry value on the remote machine.

Parameters

- **regkey** (*str*) – The registry value to delete.
- **async_mode** (*bool*) – Flag showing whether the command should be executed asynchronously

Returns command_id, future if ran async

get_file (*file_name, timeout=None, delay=None, async_mode=False*)

Retrieve contents of the specified file on the remote machine.

Parameters

- **file_name** (*str*) – Name of the file to be retrieved.
- **timeout** (*int*) – Timeout for the operation.
- **delay** (*float*) – Delay in seconds to wait before command complete.
- **async_mode** (*bool*) – Flag showing whether the command should be executed asynchronously

Returns command_id, future if ran async str: Contents of the specified file.

get_raw_file (*file_name, timeout=None, delay=None, async_mode=False*)

Retrieve contents of the specified file on the remote machine.

Parameters

- **file_name** (*str*) – Name of the file to be retrieved.
- **timeout** (*int*) – Timeout for the operation.
- **delay** (*float*) – Delay in seconds to wait before command complete.
- **async_mode** (*bool*) – Flag showing whether the command should be executed asynchronously

Returns command_id, future if ran async or object: Contains the data of the file.

get_registry_value (*regkey, async_mode=False*)

Return the associated value of the specified registry key on the remote machine.

Example

```
>>> with c.select(Device, 1).lr_session() as lr_session:
>>>     pprint.pprint(lr_session.
...     get_registry_value(
↪ 'HKLM\\SYSTEM\\CurrentControlSet\\services\\ACPI\\Start'))
{'value_data': 0, 'value_name': u'Start', 'value_type': u'REG_DWORD'}
```

Parameters

- **regkey** (*str*) – The registry key to retrieve.
- **async_mode** (*bool*) – Flag showing whether the command should be executed asynchronously

Returns command_id, future if ran async or dict: A dictionary with keys of: value_data, value_name, value_type.

kill_process (*pid*, *async_mode=False*)

Terminate a process on the remote machine.

Parameters

- **pid** (*int*) – Process ID to be terminated.
- **async_mode** (*bool*) – Flag showing whether the command should be executed asynchronously

Returns command_id, future if ran async bool: True if success, False if failure.

list_directory (*dir_name*, *async_mode=False*)

List the contents of a directory on the remote machine.

Example

```
>>> with c.select(Device, 1).lr_session() as lr_session:
...     pprint.pprint(lr_session.list_directory('C:\\\\temp\\\\'))
[{u'attributes': [u'DIRECTORY'],
  u'create_time': 1471897244,
  u'filename': u'..',
  u'last_access_time': 1476390670,
  u'last_write_time': 1476390670,
  u'size': 0},
```

```
{u'attributes': [u'DIRECTORY'], u'create_time': 1471897244, u'filename': u'..',
  u'last_access_time': 1476390670, u'last_write_time': 1476390670, u'size': 0},
```

```
{u'attributes': [u'ARCHIVE'], u'create_time': 1476390668, u'filename': u'test.txt',
  u'last_access_time': 1476390668, u'last_write_time': 1476390668, u'size': 0}]
```

Parameters

- **dir_name** (*str*) – Directory to list. This parameter should end with the path separator.
- **async_mode** (*bool*) – Flag showing whether the command should be executed asynchronously

Returns command_id, future if ran async or list: A list of dicts, each one describing a directory entry.

list_processes (*async_mode=False*)

List currently running processes on the remote machine.

Example


```
>>> with c.select(Device, 1).lr_session() as lr_session:
...     print(lr_session.list_processes()[0])
{'command_line': u'',
 'create_time': 1476260500,
 'parent': 0,
 'parent_guid': u'00000001-0000-0000-0000-000000000000',
 'path': u'',
 'pid': 4,
 'proc_guid': u'00000001-0000-0004-01d2-2461a85e4546',
 'sid': u's-1-5-18',
 'username': u'NT AUTHORITY\\SYSTEM'}
```

Parameters `async_mode` (*bool*) – Flag showing whether the command should be executed asynchronously

Returns `command_id`, future if ran async or list: A list of dicts describing the processes.

list_registry_keys_and_values (*regkey, async_mode=False*)

Enumerate subkeys and values of the specified registry key on the remote machine.

Example

```
>>> with c.select(Device, 1).lr_session() as lr_session:
>>> pprint.pprint(lr_session.
...     list_registry_keys_and_values(
...         ↪'HKLM\\SYSTEM\\CurrentControlSet\\services\\ACPI'))
{'sub_keys': [u'Parameters', u'Enum'],
 'values': [{u'value_data': 0,
             u'value_name': u'Start',
             u'value_type': u'REG_DWORD'},
            {u'value_data': 1,
             u'value_name': u'Type',
             u'value_type': u'REG_DWORD'},
            {u'value_data': 3,
             u'value_name': u'ErrorControl',
             u'value_type': u'REG_DWORD'},
            {u'value_data': u'system32\\drivers\\ACPI.sys',
             u'value_name': u'ImagePath',
             u'value_type': u'REG_EXPAND_SZ'},
            {u'value_data': u'Microsoft ACPI Driver',
             u'value_name': u'DisplayName',
             u'value_type': u'REG_SZ'},
            {u'value_data': u'Boot Bus Extender',
             u'value_name': u'Group',
             u'value_type': u'REG_SZ'},
            {u'value_data': u'acpi.inf_x86_neutral_ddd3c514822f1b21',
             u'value_name': u'DriverPackageId',
             u'value_type': u'REG_SZ'},
            {u'value_data': 1,
             u'value_name': u'Tag',
             u'value_type': u'REG_DWORD'}]}
```

Parameters

- **regkey** (*str*) – The registry key to enumerate.

- **async_mode** (*bool*) – Flag showing whether the command should be executed asynchronously

Returns

command_id, future if ran async

or

dict: A dictionary with two keys, 'sub_keys' (a list of subkey names) and 'values' (a list of dicts containing value data, name, and type).

list_registry_values (*regkey*, *async_mode=False*)

Enumerate all registry values from the specified registry key on the remote machine.

Parameters

- **regkey** (*str*) – The registry key to enumerate.
- **async_mode** (*bool*) – Flag showing whether the command should be executed asynchronously

Returns command_id, future if ran async or list: List of values for the registry key.

memdump (*local_filename*, *remote_filename=None*, *compress=False*, *async_mode=False*)

Perform a memory dump operation on the remote machine.

Parameters

- **local_filename** (*str*) – Name of the file the memory dump will be transferred to on the local machine.
- **remote_filename** (*str*) – Name of the file the memory dump will be stored in on the remote machine.
- **compress** (*bool*) – True to compress the file on the remote system.
- **async_mode** (*bool*) – Flag showing whether the command should be executed asynchronously

Returns command_id, future if ran async

put_file (*infp*, *remote_filename*, *async_mode=False*)

Create a new file on the remote machine with the specified data.

Example

```
>>> with c.select(Device, 1).lr_session() as lr_session:
...     lr_session.put_file(open("test.txt", "rb"), r"c:\test.txt")
```

Parameters

- **infp** (*object*) – Python file-like containing data to upload to the remote endpoint.
- **remote_filename** (*str*) – File name to create on the remote endpoint.
- **async_mode** (*bool*) – Flag showing whether the command should be executed asynchronously

Returns command_id, future if ran async

set_registry_value (*regkey*, *value*, *overwrite=True*, *value_type=None*, *async_mode=False*)

Set a registry value on the specified registry key on the remote machine.

Example

```
>>> with c.select(Device, 1).lr_session() as lr_session:
...     lr_session.
...     set_registry_value(
↪ 'HKLM\\SYSTEM\\CurrentControlSet\\services\\ACPI\\testvalue', 1)
```

Parameters

- **regkey** (*str*) – The registry key to set.
- **value** (*object*) – The value data.
- **overwrite** (*bool*) – If True, any existing value will be overwritten.
- **value_type** (*str*) – The type of value. Examples: REG_DWORD, REG_MULTI_SZ, REG_SZ
- **async_mode** (*bool*) – Flag showing whether the command should be executed asynchronously

Returns command_id, future if ran async

start_memdump (*remote_filename=None, compress=True*)

Start a memory dump operation on the remote machine.

Parameters

- **remote_filename** (*str*) – Name of the file the memory dump will be stored in on the remote machine.
- **compress** (*bool*) – True to compress the file on the remote system.

Returns Controlling object for the memory dump operation.

Return type *LiveResponseMemdump*

walk (*top, topdown=True, onerror=None, followlinks=False*)

Perform a full directory walk with recursion into subdirectories on the remote machine.

Note: walk does not support `async_mode` due to its behaviour, it can only be invoked synchronously

Example

```
>>> with c.select(Device, 1).lr_session() as lr_session:
...     for entry in lr_session.walk(directory_name):
...         print(entry)
('C:\\temp\\', [u'dir1', u'dir2'], [u'file1.txt'])
```

Parameters

- **top** (*str*) – Directory to recurse on.
- **topdown** (*bool*) – If True, start output from top level directory.
- **onerror** (*func*) – Callback if an error occurs. This function is called with one argument (the exception that occurred).
- **followlinks** (*bool*) – True to follow symbolic links.

Returns List of tuples containing directory name, subdirectory names, file names.

Return type list

class CompletionNotification (*device_id*)

Bases: object

The notification that an operation is complete.

Initialize the CompletionNotification.

Parameters **device_id** (*int*) – The device ID this notification is for.

class GetFileJob (*file_name*)

Bases: object

Object that retrieves a file via Live Response.

Initialize the GetFileJob.

Parameters **file_name** (*str*) – The name of the file to be fetched.

run (*session*)

Execute the file transfer.

Parameters **session** (*CbLRSessionBase*) – The Live Response session being used.

Returns The contents of the file being retrieved.

Return type str

class JobWorker (*cb, device_id, result_queue*)

Bases: threading.Thread

Thread object that executes individual Live Response jobs.

Initialize the JobWorker.

Parameters

- **cb** (*BaseAPI*) – The CBC SDK object reference.
- **device_id** (*int*) – The ID of the device being used.
- **result_queue** (*Queue*) – The queue where results are placed.

run ()

Execute the job worker.

run_job (*work_item*)

Execute an individual WorkItem.

Parameters **work_item** (*WorkItem*) – The work item to execute.

exception LiveResponseError (*details*)

Bases: Exception

Exception raised for errors with Live Response.

Initialize the LiveResponseError.

Parameters **details** (*object*) – Details of the specific error.

class LiveResponseJobScheduler (*cb, max_workers=10*)

Bases: threading.Thread

Thread that schedules Live Response jobs.

Initialize the LiveResponseJobScheduler.

Parameters

- **cb** ([BaseAPI](#)) – The CBC SDK object reference.
- **max_workers** (*int*) – Maximum number of JobWorker threads to use.

daemon = **True**

run ()

Execute the job scheduler.

submit_job (*work_item*)

Submit a new job to be processed.

Parameters **work_item** ([WorkItem](#)) – New job to be processed.

class LiveResponseMemdump (*lr_session, memdump_id, remote_filename*)

Bases: [object](#)

Object managing a memory dump on a remote machine.

Initialize the LiveResponseMemdump.

Parameters

- **lr_session** (*Session*) – The Live Response session to the machine doing the memory dump.
- **memdump_id** (*str*) – The ID of the memory dump being performed.
- **remote_filename** (*str*) – The file name the memory dump will be stored in on the remote machine.

delete ()

Delete the memory dump file.

get (*local_filename*)

Retrieve the remote memory dump to a local file.

Parameters **local_filename** (*str*) – Filename locally that will receive the memory dump.

wait ()

Wait for the remote memory dump to complete.

class LiveResponseSession (*cbl_manager, session_id, device_id, session_data=None*)

Bases: [cbc_sdk.live_response_api.CbLRSessionBase](#)

Public face of the Live Response session object.

Initializes the LiveResponseSession.

Parameters

- **cbl_manager** ([LiveResponseSessionManager](#)) – Reference to the session manager.
- **session_id** (*str*) – The ID of this session.
- **device_id** (*int*) – The ID of the device (remote machine) we're connected to.
- **session_data** (*dict*) – Additional session data.

class LiveResponseSessionManager (*cb, timeout=30, keepalive_sessions=False*)

Bases: [cbc_sdk.live_response_api.CbLRManagerBase](#)

Session manager for Live Response sessions.

Initialize the LiveResponseSessionManager - only needed to format cbl_base

cblr_base = '/appservices/v6/orgs/{}/liveresponse'

cblr_session_cls

alias of *LiveResponseSession*

session_status (*session_id*)

Check the status of a lr session

Parameters **session_id** (*str*) – The id of the session.

Returns Status of the session

Return type *str*

submit_job (*job*, *device*)

Submit a job for execution by the job scheduler.

Parameters

- **job** (*func*) – The job function to be executed.
- **device** (*object*) – The device ID or Device object the job will be executed on.

Returns A Future that will allow waiting until the job is complete.

Return type Future

class WorkItem (*fn*, *device_id*)

Bases: *object*

Work item for scheduling.

Initialize the WorkItem.

Parameters

- **fn** (*func*) – The function to be called to do the actual work.
- **device_id** (*object*) – The device ID or Device object the work item is directed for.

class WorkerStatus (*device_id*, *status*='READY', *exception*=None)

Bases: *object*

Holds the status of an individual worker.

Initialize the WorkerStatus.

Parameters

- **device_id** (*int*) – The device ID this status is for.
- **status** (*str*) – The current status value.
- **exception** (*Exception*) – Any exception that happened.

jobrunner (*callable*, *cb*, *device_id*)

Wrap a callable object with a live response session.

Parameters

- **callable** (*object*) – The object to be wrapped.
- **cb** (*BaseAPI*) – The CBC SDK object reference.
- **device_id** (*int*) – The device ID to use to get the session.

Returns The wrapped object.

Return type *object*

poll_status (*cb*, *url*, *desired_status*='COMPLETE', *timeout*=None, *delay*=None)

Poll the status of a Live Response query.

Parameters

- **cb** (*BaseAPI*) – The CBC SDK object reference.
- **url** (*str*) – The URL to poll.
- **desired_status** (*str*) – The status we’re looking for.
- **timeout** (*int*) – The timeout value in seconds.
- **delay** (*float*) – The delay between attempts in seconds.

Returns The result of the Live Response query that has the desired status.

Return type object

Raises *LiveResponseError* – If an error response was encountered.

5.8.9 cbc_sdk.rest_api module

Definition of the CBCloudAPI object, the core object for interacting with the Carbon Black Cloud SDK.

class CBCloudAPI (*args, **kwargs)
Bases: *cbc_sdk.connection.BaseAPI*

The main entry point into the CBCloudAPI.

Usage:

```
>>> from cbc_sdk import CBCloudAPI
>>> cb = CBCloudAPI(profile="production")
```

Initialize the CBCloudAPI object.

Parameters

- ***args** (*list*) – List of arguments to pass to the API object.
- ****kwargs** (*dict*) – Keyword arguments to pass to the API object.

Keyword Arguments

- **profile** (*str*) – Use the credentials in the named profile when connecting to the Carbon Black server. Uses the profile named ‘default’ when not specified.
- **threat_pool_count** (*int*) – The number of threads to create for asynchronous queries. Defaults to 3.

alert_search_suggestions (*query*)

Returns suggestions for keys and field values that can be used in a search.

Parameters **query** (*str*) – A search query to use.

Returns A list of search suggestions expressed as dict objects.

Return type list

audit_remediation (*sql*)

Run an audit-remediation query.

Parameters **sql** (*str*) – The SQL for the query.

Returns The query object.

Return type *cbc_sdk.base.Query*

audit_remediation_history (*query=None*)

Run an audit-remediation history query.

Parameters **query** (*str*) – The SQL for the query.

Returns The query object.

Return type *cbc_sdk.base.Query*

bulk_threat_dismiss (*threat_ids, remediation=None, comment=None*)

Dismiss the alerts associated with multiple threat IDs. The alerts will be left in a DISMISSED state.

Parameters

- **threat_ids** (*list*) – List of string threat IDs.
- **remediation** (*str*) – The remediation state to set for all alerts.
- **comment** (*str*) – The comment to set for all alerts.

Returns The request ID of the pending request, which may be used to select a WorkflowStatus object.

Return type *str*

bulk_threat_update (*threat_ids, remediation=None, comment=None*)

Update the alert status of alerts associated with multiple threat IDs. The alerts will be left in an OPEN state

Parameters

- **threat_ids** (*list*) – List of string threat IDs.
- **remediation** (*str*) – The remediation state to set for all alerts.
- **comment** (*str*) – The comment to set for all alerts.

Returns The request ID of the pending request, which may be used to select a WorkflowStatus object.

Return type *str*

convert_feed_query (*query*)

Converts a legacy CB Response query to a ThreatHunter query.

Parameters **query** (*str*) – The query to convert.

Returns The converted query.

Return type *str*

create (*cls, data=None*)

Creates a new model.

Parameters

- **cls** (*class*) – The model being created.
- **data** (*dict*) – The data to pre-populate the model with.

Returns An instance of *cls*.

Return type *object*

Examples

```
>>> feed = cb.create(Feed, feed_data)
```

custom_severities

Returns a list of active ReportSeverity instances.

device_background_scan (*device_ids*, *scan*)

Set the background scan option for the specified devices.

Parameters

- **device_ids** (*list*) – List of IDs of devices to be set.
- **scan** (*bool*) – True to turn background scan on, False to turn it off.

Returns The parsed JSON output from the request.

Return type dict

Raises `ServerError` – If the API method returns an HTTP error code.

device_bypass (*device_ids*, *enable*)

Set the bypass option for the specified devices.

Parameters

- **device_ids** (*list*) – List of IDs of devices to be set.
- **enable** (*bool*) – True to enable bypass, False to disable it.

Returns The parsed JSON output from the request.

Return type dict

Raises `ServerError` – If the API method returns an HTTP error code.

device_delete_sensor (*device_ids*)

Delete the specified sensor devices.

Parameters **device_ids** (*list*) – List of IDs of devices to be deleted.

Returns The parsed JSON output from the request.

Return type dict

Raises `ServerError` – If the API method returns an HTTP error code.

device_quarantine (*device_ids*, *enable*)

Set the quarantine option for the specified devices.

Parameters

- **device_ids** (*list*) – List of IDs of devices to be set.
- **enable** (*bool*) – True to enable quarantine, False to disable it.

Returns The parsed JSON output from the request.

Return type dict

Raises `ServerError` – If the API method returns an HTTP error code.

device_uninstall_sensor (*device_ids*)

Uninstall the specified sensor devices.

Parameters **device_ids** (*list*) – List of IDs of devices to be uninstalled.

Returns The parsed JSON output from the request.

Return type dict

Raises `ServerError` – If the API method returns an HTTP error code.

`device_update_policy(device_ids, policy_id)`

Set the current policy for the specified devices.

Parameters

- **`device_ids`** (*list*) – List of IDs of devices to be changed.
- **`policy_id`** (*int*) – ID of the policy to set for the devices.

Returns The parsed JSON output from the request.

Return type dict

Raises `ServerError` – If the API method returns an HTTP error code.

`device_update_sensor_version(device_ids, sensor_version)`

Update the sensor version for the specified devices.

Parameters

- **`device_ids`** (*list*) – List of IDs of devices to be changed.
- **`sensor_version`** (*dict*) – New version properties for the sensor.

Returns The parsed JSON output from the request.

Return type dict

Raises `ServerError` – If the API method returns an HTTP error code.

`fetch_process_queries()`

Retrieves a list of query IDs, active or complete, known by the ThreatHunter server.

`get_auditlogs()`

Retrieve queued audit logs from the Carbon Black Cloud Endpoint Standard server.

Note that this can only be used with a ‘API’ key generated in the CBC console.

Returns list of dictionary objects representing the audit logs, or an empty list if none available.

`get_notifications()`

Retrieve queued notifications (alerts) from the Cb Endpoint Standard server.

Note that this can only be used with a ‘SIEM’ key generated in the Cb Endpoint Standard console.

Returns List of dictionary objects representing the notifications, or an empty list if none available.

Return type list

`get_policy_ruleconfig_parameter_schema(ruleconfig_id)`

Returns the parameter schema for a specified rule configuration.

Parameters

- **`cb`** (`BaseAPI`) – Reference to API object used to communicate with the server.
- **`ruleconfig_id`** (*str*) – The rule configuration ID (UUID).

Returns The parameter schema for this particular rule configuration (as a JSON schema).

Return type dict

Raises `InvalidObjectError` – If the rule configuration ID is not valid.

live_response

Create and return the Live Response session manager.

Returns The session manager object.

Return type `LiveResponseSessionManager`

notification_listener (*interval=60*)

Generator to continually poll the Cb Endpoint Standard server for notifications (alerts).

Note that this can only be used with a ‘SIEM’ key generated in the Cb Endpoint Standard console.

org_urn

Returns the URN based on the configured `org_key`.

Returns The URN based on the configured `org_key`.

Return type `str`

process_limits ()

Returns a dictionary containing API limiting information.

Examples

```
>>> cb.process_limits()
{'u'status_code': 200, u'time_bounds': {'u'upper': 1545335070095, u'lower':
↪ 1542779216139}}
```

validate_process_query (*query*)

Validates the given IOC query.

Parameters `query` (*str*) – The query to validate.

Returns True if the query is valid, False if not.

Return type `bool`

Examples

```
>>> cb.validate_process_query("process_name:chrome.exe") # True
```

5.8.10 cbc_sdk.utils module

Utility functions for use within the CBC SDK.

convert_from_cb (*s*)

Parse a date and time value into a datetime object.

Parameters `s` (*str*) – The date and time string to parse. If this is None, we use the UNIX epoch timestamp.

Returns The parsed date and time.

Return type `datetime`

convert_to_cb (*dt*)

Convert a date and time to a string in the Carbon Black format.

Parameters `dt` (*datetime*) – The date and time to be converted.

Returns The date and time as a string.

Return type `str`

5.8.11 `cbc_sdk.winerror` module

Error related constants for win32

Generated by h2py from winerror.h

class `CommDlgError`

Bases: `cbc_sdk.winerror.ErrorBaseClass`

Collects all the common dialog error codes.

`CCERR_CHOOSECOLORCODES` = 20480

`CDERR_DIALOGFAILURE` = 65535

`CDERR_FINDRESFAILURE` = 6

`CDERR_GENERALCODES` = 0

`CDERR_INITIALIZATION` = 2

`CDERR_LOADRESFAILURE` = 7

`CDERR_LOADSTRFAILURE` = 5

`CDERR_LOCKRESFAILURE` = 8

`CDERR_MEMALLOCFAILURE` = 9

`CDERR_MEMLOCKFAILURE` = 10

`CDERR_NOINSTANCE` = 4

`CDERR_NOHOOK` = 11

`CDERR_NOTEMPLATE` = 3

`CDERR_REGISTERMSGFAIL` = 12

`CDERR_STRUCTSIZE` = 1

`CFERR_CHOOSEFONTCODES` = 8192

`CFERR_MAXLESSTHANMIN` = 8194

`CFERR_NOFONTS` = 8193

`FNERR_BUFFERTOOSMALL` = 12291

`FNERR_FILENAMECODES` = 12288

`FNERR_INVALIDFILENAME` = 12290

`FNERR_SUBCLASSFAILURE` = 12289

`FRERR_BUFFERLENGTHZERO` = 16385

`FRERR_FINDREPLACECODES` = 16384

`PDERR_CREATEICFAILURE` = 4106

`PDERR_DEFAULTDIFFERENT` = 4108

```
PDERR_DNDMMISMATCH = 4105
PDERR_GETDEVMODEFAIL = 4101
PDERR_INITFAILURE = 4102
PDERR_LOADDRVFAILURE = 4100
PDERR_NODEFAULTPRN = 4104
PDERR_NODEVICES = 4103
PDERR_PARSEFAILURE = 4098
PDERR_PRINTERCODES = 4096
PDERR_PRINTERNOTFOUND = 4107
PDERR_RETDEFFAILURE = 4099
PDERR_SETUPFAILURE = 4097
```

```
class DirectoryStorageError
```

```
    Bases: cbc_sdk.winerror.ErrorBaseClass
```

```
    Collects all the directory storage error codes.
```

```
ERROR_DS_ADD_REPLICA_INHIBITED = 8302
ERROR_DS_ADMIN_LIMIT_EXCEEDED = 8228
ERROR_DS_AFFECTS_MULTIPLE_DSAS = 8249
ERROR_DS_AG_CANT_HAVE_UNIVERSAL_MEMBER = 8578
ERROR_DS_ALIASED_OBJ_MISSING = 8334
ERROR_DS_ALIAS_DEREF_PROBLEM = 8244
ERROR_DS_ALIAS_POINTS_TO_ALIAS = 8336
ERROR_DS_ALIAS_PROBLEM = 8241
ERROR_DS_ATTRIBUTE_OR_VALUE_EXISTS = 8205
ERROR_DS_ATTRIBUTE_OWNED_BY_SAM = 8346
ERROR_DS_ATTRIBUTE_TYPE_UNDEFINED = 8204
ERROR_DS_ATT_ALREADY_EXISTS = 8318
ERROR_DS_ATT_IS_NOT_ON_OBJ = 8310
ERROR_DS_ATT_NOT_DEF_FOR_CLASS = 8317
ERROR_DS_ATT_NOT_DEF_IN_SCHEMA = 8303
ERROR_DS_ATT_SCHEMA_REQ_ID = 8399
ERROR_DS_ATT_SCHEMA_REQ_SYNTAX = 8416
ERROR_DS_ATT_VAL_ALREADY_EXISTS = 8323
ERROR_DS_AUTHORIZATION_FAILED = 8599
ERROR_DS_AUTH_METHOD_NOT_SUPPORTED = 8231
ERROR_DS_AUTH_UNKNOWN = 8234
ERROR_DS_AUX_CLS_TEST_FAIL = 8389
```

```
ERROR_DS_BACKLINK_WITHOUT_LINK = 8482
ERROR_DS_BAD_ATT_SCHEMA_SYNTAX = 8400
ERROR_DS_BAD_HIERARCHY_FILE = 8425
ERROR_DS_BAD_INSTANCE_TYPE = 8313
ERROR_DS_BAD_NAME_SYNTAX = 8335
ERROR_DS_BAD_RDN_ATT_ID_SYNTAX = 8392
ERROR_DS_BUILD_HIERARCHY_TABLE_FAILED = 8426
ERROR_DS_BUSY = 8206
ERROR_DS_CANT_ACCESS_REMOTE_PART_OF_AD = 8585
ERROR_DS_CANT_ADD_ATT_VALUES = 8320
ERROR_DS_CANT_ADD_SYSTEM_ONLY = 8358
ERROR_DS_CANT_ADD_TO_GC = 8550
ERROR_DS_CANT_CACHE_ATT = 8401
ERROR_DS_CANT_CACHE_CLASS = 8402
ERROR_DS_CANT_CREATE_IN_NONDOMAIN_NC = 8553
ERROR_DS_CANT_CREATE_UNDER_SCHEMA = 8510
ERROR_DS_CANT_DELETE = 8398
ERROR_DS_CANT_DELETE_DSA_OBJ = 8340
ERROR_DS_CANT_DEL_MASTER_CROSSREF = 8375
ERROR_DS_CANT_DEMOTE_WITH_WRITEABLE_NC = 8604
ERROR_DS_CANT_DEREF_ALIAS = 8337
ERROR_DS_CANT_DERIVE_SPN_FOR_DELETED_DOMAIN = 8603
ERROR_DS_CANT_DERIVE_SPN_WITHOUT_SERVER_REF = 8589
ERROR_DS_CANT_FIND_DC_FOR_SRC_DOMAIN = 8537
ERROR_DS_CANT_FIND_DSA_OBJ = 8419
ERROR_DS_CANT_FIND_EXPECTED_NC = 8420
ERROR_DS_CANT_FIND_NC_IN_CACHE = 8421
ERROR_DS_CANT_MIX_MASTER_AND_REPS = 8331
ERROR_DS_CANT_MOD_OBJ_CLASS = 8215
ERROR_DS_CANT_MOD_PRIMARYGROUPID = 8506
ERROR_DS_CANT_MOD_SYSTEM_ONLY = 8369
ERROR_DS_CANT_MOVE_ACCOUNT_GROUP = 8498
ERROR_DS_CANT_MOVE_APP_BASIC_GROUP = 8608
ERROR_DS_CANT_MOVE_APP_QUERY_GROUP = 8609
ERROR_DS_CANT_MOVE_DELETED_OBJECT = 8489
ERROR_DS_CANT_MOVE_RESOURCE_GROUP = 8499
```

`ERROR_DS_CANT_ON_NON_LEAF = 8213`
`ERROR_DS_CANT_ON_RDN = 8214`
`ERROR_DS_CANT_REMOVE_ATT_CACHE = 8403`
`ERROR_DS_CANT_REMOVE_CLASS_CACHE = 8404`
`ERROR_DS_CANT_REM_MISSING_ATT = 8324`
`ERROR_DS_CANT_REM_MISSING_ATT_VAL = 8325`
`ERROR_DS_CANT_REPLACE_HIDDEN_REC = 8424`
`ERROR_DS_CANT_RETRIEVE_ATTS = 8481`
`ERROR_DS_CANT_RETRIEVE_CHILD = 8422`
`ERROR_DS_CANT_RETRIEVE_DN = 8405`
`ERROR_DS_CANT_RETRIEVE_INSTANCE = 8407`
`ERROR_DS_CANT_RETRIEVE_SD = 8526`
`ERROR_DS_CANT_START = 8531`
`ERROR_DS_CANT_TREE_DELETE_CRITICAL_OBJ = 8560`
`ERROR_DS_CANT_WITH_ACCT_GROUP_MEMBERSHPS = 8493`
`ERROR_DS_CHILDREN_EXIST = 8332`
`ERROR_DS_CLASS_MUST_BE_CONCRETE = 8359`
`ERROR_DS_CLASS_NOT_DSA = 8343`
`ERROR_DS_CLIENT_LOOP = 8259`
`ERROR_DS_CODE_INCONSISTENCY = 8408`
`ERROR_DS_COMPARE_FALSE = 8229`
`ERROR_DS_COMPARE_TRUE = 8230`
`ERROR_DS_CONFIDENTIALITY_REQUIRED = 8237`
`ERROR_DS_CONFIG_PARAM_MISSING = 8427`
`ERROR_DS_CONSTRAINT_VIOLATION = 8239`
`ERROR_DS_CONSTRUCTED_ATT_MOD = 8475`
`ERROR_DS_CONTROL_NOT_FOUND = 8258`
`ERROR_DS_COULDNT_CONTACT_FSMO = 8367`
`ERROR_DS_COULDNT_IDENTIFY_OBJECTS_FOR_TREE_DELETE = 8503`
`ERROR_DS_COULDNT_LOCK_TREE_FOR_DELETE = 8502`
`ERROR_DS_COULDNT_UPDATE_SPNS = 8525`
`ERROR_DS_COUNTING_AB_INDICES_FAILED = 8428`
`ERROR_DS_CROSS_DOMAIN_CLEANUP_REQD = 8491`
`ERROR_DS_CROSS_DOM_MOVE_ERROR = 8216`
`ERROR_DS_CROSS_NC_DN_RENAME = 8368`
`ERROR_DS_CROSS_REF_BUSY = 8602`

```
ERROR_DS_CROSS_REF_EXISTS = 8374
ERROR_DS_CR_IMPOSSIBLE_TO_VALIDATE = 8495
ERROR_DS_CR_IMPOSSIBLE_TO_VALIDATE_V2 = 8586
ERROR_DS_DATABASE_ERROR = 8409
ERROR_DS_DECODING_ERROR = 8253
ERROR_DS_DESTINATION_AUDITING_NOT_ENABLED = 8536
ERROR_DS_DESTINATION_DOMAIN_NOT_IN_FOREST = 8535
ERROR_DS_DIFFERENT_REPL_EPOCHS = 8593
ERROR_DS_DISALLOWED_IN_SYSTEM_CONTAINER = 8615
ERROR_DS_DNS_LOOKUP_FAILURE = 8524
ERROR_DS_DOMAIN_RENAME_IN_PROGRESS = 8612
ERROR_DS_DOMAIN_VERSION_TOO_HIGH = 8564
ERROR_DS_DOMAIN_VERSION_TOO_LOW = 8566
ERROR_DS_DRA_ABANDON_SYNC = 8462
ERROR_DS_DRA_ACCESS_DENIED = 8453
ERROR_DS_DRA_BAD_DN = 8439
ERROR_DS_DRA_BAD_INSTANCE_TYPE = 8445
ERROR_DS_DRA_BAD_NC = 8440
ERROR_DS_DRA_BUSY = 8438
ERROR_DS_DRA_CONNECTION_FAILED = 8444
ERROR_DS_DRA_DB_ERROR = 8451
ERROR_DS_DRA_DN_EXISTS = 8441
ERROR_DS_DRA_EARLIER_SCHEMA_CONFLICT = 8544
ERROR_DS_DRA_EXTN_CONNECTION_FAILED = 8466
ERROR_DS_DRA_GENERIC = 8436
ERROR_DS_DRA_INCOMPATIBLE_PARTIAL_SET = 8464
ERROR_DS_DRA_INCONSISTENT_DIT = 8443
ERROR_DS_DRA_INTERNAL_ERROR = 8442
ERROR_DS_DRA_INVALID_PARAMETER = 8437
ERROR_DS_DRA_MAIL_PROBLEM = 8447
ERROR_DS_DRA_MISSING_PARENT = 8460
ERROR_DS_DRA_NAME_COLLISION = 8458
ERROR_DS_DRA_NOT_SUPPORTED = 8454
ERROR_DS_DRA_NO_REPLICA = 8452
ERROR_DS_DRA_OBJ_IS_REP_SOURCE = 8450
ERROR_DS_DRA_OBJ_NC_MISMATCH = 8545
```


`ERROR_DS_DRA_OUT_OF_MEM = 8446`
`ERROR_DS_DRA_OUT_SCHEDULE_WINDOW = 8617`
`ERROR_DS_DRA_PREEMPTED = 8461`
`ERROR_DS_DRA_REF_ALREADY_EXISTS = 8448`
`ERROR_DS_DRA_REF_NOT_FOUND = 8449`
`ERROR_DS_DRA_REPL_PENDING = 8477`
`ERROR_DS_DRA_RPC_CANCELLED = 8455`
`ERROR_DS_DRA_SCHEMA_CONFLICT = 8543`
`ERROR_DS_DRA_SCHEMA_INFO_SHIP = 8542`
`ERROR_DS_DRA_SCHEMA_MISMATCH = 8418`
`ERROR_DS_DRA_SHUTDOWN = 8463`
`ERROR_DS_DRA_SINK_DISABLED = 8457`
`ERROR_DS_DRA_SOURCE_DISABLED = 8456`
`ERROR_DS_DRA_SOURCE_IS_PARTIAL_REPLICA = 8465`
`ERROR_DS_DRA_SOURCE_REINSTALLED = 8459`
`ERROR_DS_DRS_EXTENSIONS_CHANGED = 8594`
`ERROR_DS_DSA_MUST_BE_INT_MASTER = 8342`
`ERROR_DS_DST_DOMAIN_NOT_NATIVE = 8496`
`ERROR_DS_DST_NC_MISMATCH = 8486`
`ERROR_DS_DS_REQUIRED = 8478`
`ERROR_DS_DUPLICATE_ID_FOUND = 8605`
`ERROR_DS_DUP_LDAP_DISPLAY_NAME = 8382`
`ERROR_DS_DUP_LINK_ID = 8468`
`ERROR_DS_DUP_MAPI_ID = 8380`
`ERROR_DS_DUP_MSDS_INTID = 8597`
`ERROR_DS_DUP_OID = 8379`
`ERROR_DS_DUP_RDN = 8378`
`ERROR_DS_DUP_SCHEMA_ID_GUID = 8381`
`ERROR_DS_ENCODING_ERROR = 8252`
`ERROR_DS_EPOCH_MISMATCH = 8483`
`ERROR_DS_EXISTING_AD_CHILD_NC = 8613`
`ERROR_DS_EXISTS_IN_AUX_CLS = 8393`
`ERROR_DS_EXISTS_IN_MAY_HAVE = 8386`
`ERROR_DS_EXISTS_IN_MUST_HAVE = 8385`
`ERROR_DS_EXISTS_IN_POSS_SUP = 8395`
`ERROR_DS_EXISTS_IN_RDNATTID = 8598`

`ERROR_DS_EXISTS_IN_SUB_CLS = 8394`
`ERROR_DS_FILTER_UNKNOWN = 8254`
`ERROR_DS_FILTER_USES_CONSTRUCTED_ATTRS = 8555`
`ERROR_DS_FOREST_VERSION_TOO_HIGH = 8563`
`ERROR_DS_FOREST_VERSION_TOO_LOW = 8565`
`ERROR_DS_GCVERIFY_ERROR = 8417`
`ERROR_DS_GC_NOT_AVAILABLE = 8217`
`ERROR_DS_GC_REQUIRED = 8547`
`ERROR_DS_GENERIC_ERROR = 8341`
`ERROR_DS_GLOBAL_CANT_HAVE_CROSSDOMAIN_MEMBER = 8519`
`ERROR_DS_GLOBAL_CANT_HAVE_LOCAL_MEMBER = 8516`
`ERROR_DS_GLOBAL_CANT_HAVE_UNIVERSAL_MEMBER = 8517`
`ERROR_DS_GOVERNSID_MISSING = 8410`
`ERROR_DS_GROUP_CONVERSION_ERROR = 8607`
`ERROR_DS_HAVE_PRIMARY_MEMBERS = 8521`
`ERROR_DS_HIERARCHY_TABLE_MALLOC_FAILED = 8429`
`ERROR_DS_ILLEGAL_BASE_SCHEMA_MOD = 8507`
`ERROR_DS_ILLEGAL_MOD_OPERATION = 8311`
`ERROR_DS_ILLEGAL_SUPERIOR = 8345`
`ERROR_DS_ILLEGAL_XDOM_MOVE_OPERATION = 8492`
`ERROR_DS_INAPPROPRIATE_AUTH = 8233`
`ERROR_DS_INAPPROPRIATE_MATCHING = 8238`
`ERROR_DS_INCOMPATIBLE_CONTROLS_USED = 8574`
`ERROR_DS_INCOMPATIBLE_VERSION = 8567`
`ERROR_DS_INCORRECT_ROLE_OWNER = 8210`
`ERROR_DS_INIT_FAILURE = 8532`
`ERROR_DS_INIT_FAILURE_CONSOLE = 8561`
`ERROR_DS_INSTALL_NO_SCH_VERSION_IN_INIFILE = 8512`
`ERROR_DS_INSTALL_NO_SRC_SCH_VERSION = 8511`
`ERROR_DS_INSTALL_SCHEMA_MISMATCH = 8467`
`ERROR_DS_INSUFFICIENT_ATTR_TO_CREATE_OBJECT = 8606`
`ERROR_DS_INSUFF_ACCESS_RIGHTS = 8344`
`ERROR_DS_INTERNAL_FAILURE = 8430`
`ERROR_DS_INVALID_ATTRIBUTE_SYNTAX = 8203`
`ERROR_DS_INVALID_DMD = 8360`
`ERROR_DS_INVALID_DN_SYNTAX = 8242`

```
ERROR_DS_INVALID_GROUP_TYPE = 8513
ERROR_DS_INVALID_LDAP_DISPLAY_NAME = 8479
ERROR_DS_INVALID_NAME_FOR_SPN = 8554
ERROR_DS_INVALID_ROLE_OWNER = 8366
ERROR_DS_INVALID_SCRIPT = 8600
ERROR_DS_INVALID_SEARCH_FLAG = 8500
ERROR_DS_IS_LEAF = 8243
ERROR_DS_KEY_NOT_UNIQUE = 8527
ERROR_DS_LDAP_SEND_QUEUE_FULL = 8616
ERROR_DS_LINK_ID_NOT_AVAILABLE = 8577
ERROR_DS_LOCAL_CANT_HAVE_CROSSDOMAIN_LOCAL_MEMBER = 8520
ERROR_DS_LOCAL_ERROR = 8251
ERROR_DS_LOCAL_MEMBER_OF_LOCAL_ONLY = 8548
ERROR_DS_LOOP_DETECT = 8246
ERROR_DS_LOW_DSA_VERSION = 8568
ERROR_DS_MACHINE_ACCOUNT_CREATED_PRENT4 = 8572
ERROR_DS_MACHINE_ACCOUNT_QUOTA_EXCEEDED = 8557
ERROR_DS_MASTERDSA_REQUIRED = 8314
ERROR_DS_MAX_OBJ_SIZE_EXCEEDED = 8304
ERROR_DS_MEMBERSHIP_EVALUATED_LOCALLY = 8201
ERROR_DS_MISSING_EXPECTED_ATT = 8411
ERROR_DS_MISSING_FSMO_SETTINGS = 8434
ERROR_DS_MISSING_INFRASTRUCTURE_CONTAINER = 8497
ERROR_DS_MISSING_REQUIRED_ATT = 8316
ERROR_DS_MISSING_SUPREF = 8406
ERROR_DS_MODIFYDN_DISALLOWED_BY_FLAG = 8581
ERROR_DS_MODIFYDN_DISALLOWED_BY_INSTANCE_TYPE = 8579
ERROR_DS_MODIFYDN_WRONG_GRANDPARENT = 8582
ERROR_DS_MUST_BE_RUN_ON_DST_DC = 8558
ERROR_DS_NAME_ERROR_DOMAIN_ONLY = 8473
ERROR_DS_NAME_ERROR_NOT_FOUND = 8470
ERROR_DS_NAME_ERROR_NOT_UNIQUE = 8471
ERROR_DS_NAME_ERROR_NO_MAPPING = 8472
ERROR_DS_NAME_ERROR_NO_SYNTACTICAL_MAPPING = 8474
ERROR_DS_NAME_ERROR_RESOLVING = 8469
ERROR_DS_NAME_ERROR_TRUST_REFERRAL = 8583
```

ERROR_DS_NAME_NOT_UNIQUE = 8571
ERROR_DS_NAME_REFERENCE_INVALID = 8373
ERROR_DS_NAME_TOO_LONG = 8348
ERROR_DS_NAME_TOO_MANY_PARTS = 8347
ERROR_DS_NAME_TYPE_UNKNOWN = 8351
ERROR_DS_NAME_UNPARSEABLE = 8350
ERROR_DS_NAME_VALUE_TOO_LONG = 8349
ERROR_DS_NAMING_MASTER_GC = 8523
ERROR_DS_NAMING_VIOLATION = 8247
ERROR_DS_NCNAME_MISSING_CR_REF = 8412
ERROR_DS_NCNAME_MUST_BE_NC = 8357
ERROR_DS_NC_MUST_HAVE_NC_PARENT = 8494
ERROR_DS_NC_STILL_HAS_DSAS = 8546
ERROR_DS_NONEXISTENT_MAY_HAVE = 8387
ERROR_DS_NONEXISTENT_MUST_HAVE = 8388
ERROR_DS_NONEXISTENT_POSS_SUP = 8390
ERROR_DS_NONSAFE_SCHEMA_CHANGE = 8508
ERROR_DS_NON_BASE_SEARCH = 8480
ERROR_DS_NOTIFY_FILTER_TOO_COMPLEX = 8377
ERROR_DS_NOT_AN_OBJECT = 8352
ERROR_DS_NOT_AUTHORITY_FOR_DST_NC = 8487
ERROR_DS_NOT_CLOSEST = 8588
ERROR_DS_NOT_INSTALLED = 8200
ERROR_DS_NOT_ON_BACKLINK = 8362
ERROR_DS_NOT_SUPPORTED = 8256
ERROR_DS_NOT_SUPPORTED_SORT_ORDER = 8570
ERROR_DS_NO_ATTRIBUTE_OR_VALUE = 8202
ERROR_DS_NO_BEHAVIOR_VERSION_IN_MIXEDDOMAIN = 8569
ERROR_DS_NO_CHAINED_EVAL = 8328
ERROR_DS_NO_CHAINING = 8327
ERROR_DS_NO_CHECKPOINT_WITH_PDC = 8551
ERROR_DS_NO_CROSSREF_FOR_NC = 8363
ERROR_DS_NO_DELETED_NAME = 8355
ERROR_DS_NO_FPO_IN_UNIVERSAL_GROUPS = 8549
ERROR_DS_NO_MORE_RIDS = 8209
ERROR_DS_NO_MSDFS_INTID = 8596

ERROR_DS_NO_NEST_GLOBALGROUP_IN_MIXEDDOMAIN = 8514
ERROR_DS_NO_NEST_LOCALGROUP_IN_MIXEDDOMAIN = 8515
ERROR_DS_NO_OBJECT_MOVE_IN_SCHEMA_NC = 8580
ERROR_DS_NO_PARENT_OBJECT = 8329
ERROR_DS_NO_PKT_PRIVACY_ON_CONNECTION = 8533
ERROR_DS_NO_RDN_DEFINED_IN_SCHEMA = 8306
ERROR_DS_NO_REF_DOMAIN = 8575
ERROR_DS_NO_REQUESTED_ATTNS_FOUND = 8308
ERROR_DS_NO_RESULTS_RETURNED = 8257
ERROR_DS_NO_RIDS_ALLOCATED = 8208
ERROR_DS_NO_SUCH_OBJECT = 8240
ERROR_DS_NO_TREE_DELETE_ABOVE_NC = 8501
ERROR_DS_NTDSRIPT_PROCESS_ERROR = 8592
ERROR_DS_NTDSRIPT_SYNTAX_ERROR = 8591
ERROR_DS_OBJECT_BEING_REMOVED = 8339
ERROR_DS_OBJECT_CLASS_REQUIRED = 8315
ERROR_DS_OBJECT_RESULTS_TOO_LARGE = 8248
ERROR_DS_OBJ_CLASS_NOT_DEFINED = 8371
ERROR_DS_OBJ_CLASS_NOT_SUBCLASS = 8372
ERROR_DS_OBJ_CLASS_VIOLATION = 8212
ERROR_DS_OBJ_GUID_EXISTS = 8361
ERROR_DS_OBJ_NOT_FOUND = 8333
ERROR_DS_OBJ_STRING_NAME_EXISTS = 8305
ERROR_DS_OBJ_TOO_LARGE = 8312
ERROR_DS_OFFSET_RANGE_ERROR = 8262
ERROR_DS_OPERATIONS_ERROR = 8224
ERROR_DS_OUT_OF_SCOPE = 8338
ERROR_DS_OUT_OF_VERSION_STORE = 8573
ERROR_DS_PARAM_ERROR = 8255
ERROR_DS_PARENT_IS_AN_ALIAS = 8330
ERROR_DS_PDC_OPERATION_IN_PROGRESS = 8490
ERROR_DS_PROTOCOL_ERROR = 8225
ERROR_DS_RANGE_CONSTRAINT = 8322
ERROR_DS_RDN_DOESNT_MATCH_SCHEMA = 8307
ERROR_DS_RECALCSHEMA_FAILED = 8396
ERROR_DS_REFERRAL = 8235

ERROR_DS_REFERRAL_LIMIT_EXCEEDED = 8260
ERROR_DS_REFUSING_FSMO_ROLES = 8433
ERROR_DS_REMOTE_CROSSREF_OP_FAILED = 8601
ERROR_DS_REPLICATOR_ONLY = 8370
ERROR_DS_REPLICA_SET_CHANGE_NOT_ALLOWED_ON_DISABLED_CR = 8595
ERROR_DS_REPL_LIFETIME_EXCEEDED = 8614
ERROR_DS_RESERVED_LINK_ID = 8576
ERROR_DS_RIDMGR_INIT_ERROR = 8211
ERROR_DS_ROLE_NOT_VERIFIED = 8610
ERROR_DS_ROOT_CANT_BE_SUBREF = 8326
ERROR_DS_ROOT_MUST_BE_NC = 8301
ERROR_DS_ROOT_REQUIRES_CLASS_TOP = 8432
ERROR_DS_SAM_INIT_FAILURE = 8504
ERROR_DS_SAM_INIT_FAILURE_CONSOLE = 8562
ERROR_DS_SAM_NEED_BOOTKEY_FLOPPY = 8530
ERROR_DS_SAM_NEED_BOOTKEY_PASSWORD = 8529
ERROR_DS_SCHEMA_ALLOC_FAILED = 8415
ERROR_DS_SCHEMA_NOT_LOADED = 8414
ERROR_DS_SCHEMA_UPDATE_DISALLOWED = 8509
ERROR_DS_SECURITY_CHECKING_ERROR = 8413
ERROR_DS_SECURITY_ILLEGAL_MODIFY = 8423
ERROR_DS_SEC_DESC_INVALID = 8354
ERROR_DS_SEC_DESC_TOO_SHORT = 8353
ERROR_DS_SEMANTIC_ATT_TEST = 8383
ERROR_DS_SENSITIVE_GROUP_VIOLATION = 8505
ERROR_DS_SERVER_DOWN = 8250
ERROR_DS_SHUTTING_DOWN = 8364
ERROR_DS_SINGLE_USER_MODE_FAILED = 8590
ERROR_DS_SINGLE_VALUE_CONSTRAINT = 8321
ERROR_DS_SIZELIMIT_EXCEEDED = 8227
ERROR_DS_SORT_CONTROL_MISSING = 8261
ERROR_DS_SOURCE_AUDITING_NOT_ENABLED = 8552
ERROR_DS_SOURCE_DOMAIN_IN_FOREST = 8534
ERROR_DS_SRC_AND_DST_NC_IDENTICAL = 8485
ERROR_DS_SRC_AND_DST_OBJECT_CLASS_MISMATCH = 8540
ERROR_DS_SRC_DC_MUST_BE_SP4_OR_GREATER = 8559

```
ERROR_DS_SRC_GUID_MISMATCH = 8488
ERROR_DS_SRC_NAME_MISMATCH = 8484
ERROR_DS_SRC_OBJ_NOT_GROUP_OR_USER = 8538
ERROR_DS_SRC_SID_EXISTS_IN_FOREST = 8539
ERROR_DS_STRING_SD_CONVERSION_FAILED = 8522
ERROR_DS_STRONG_AUTH_REQUIRED = 8232
ERROR_DS_SUBREF_MUST_HAVE_PARENT = 8356
ERROR_DS_SUBTREE_NOTIFY_NOT_NC_HEAD = 8376
ERROR_DS_SUB_CLS_TEST_FAIL = 8391
ERROR_DS_SYNTAX_MISMATCH = 8384
ERROR_DS_THREAD_LIMIT_EXCEEDED = 8587
ERROR_DS_TIMELIMIT_EXCEEDED = 8226
ERROR_DS_TREE_DELETE_NOT_FINISHED = 8397
ERROR_DS_UNABLE_TO_SURRENDER_ROLES = 8435
ERROR_DS_UNAVAILABLE = 8207
ERROR_DS_UNAVAILABLE_CRIT_EXTENSION = 8236
ERROR_DS_UNICODEPWD_NOT_IN_QUOTES = 8556
ERROR_DS_UNIVERSAL_CANT_HAVE_LOCAL_MEMBER = 8518
ERROR_DS_UNKNOWN_ERROR = 8431
ERROR_DS_UNKNOWN_OPERATION = 8365
ERROR_DS_UNWILLING_TO_PERFORM = 8245
ERROR_DS_USER_BUFFER_TO_SMALL = 8309
ERROR_DS_WKO_CONTAINER_CANNOT_BE_SPECIAL = 8611
ERROR_DS_WRONG_LINKED_ATT_SYNTAX = 8528
ERROR_DS_WRONG_OM_OBJ_CLASS = 8476
ERROR_NOT_SUPPORTED_ON_STANDARD_SERVER = 8584
ERROR_NO_PROMOTION_ACTIVE = 8222
ERROR_POLICY_OBJECT_NOT_FOUND = 8219
ERROR_POLICY_ONLY_IN_DS = 8220
ERROR_PROMOTION_ACTIVE = 8221
ERROR_SAM_INIT_FAILURE = 8541
ERROR_SHARED_POLICY = 8218
```

```
class ErrorBaseClass
```

```
    Bases: object
```

```
    Base class for repositories of error codes.
```

```
    classmethod lookup_error(error_code)
```

```
        Look up an error code by value.
```

Parameters `error_code` (*int*) – The error code to be looked up.

Returns The error code name.

Return type `str`

class `ErrorMetaClass`

Bases: `type`

Metaclass which establishes an easy means of looking up error codes in a collection.

Creates a new instance of a class, setting up the dict to make it easy to look up error codes.

Parameters

- **name** (*str*) – The name of the class.
- **bases** (*list*) – Base classes of the class to be created.
- **clsdict** (*dict*) – Elements defined in the new class.

FAILED (*Status*)

Return True iff a HRESULT/SCODE status represents failure.

class `Facility`

Bases: `cbc_sdk.winerror.ErrorBaseClass`

Collects all known facility codes.

`FACILITY_AAF = 18`

`FACILITY_ACS = 20`

`FACILITY_BACKGROUNDCOPY = 32`

`FACILITY_CERT = 11`

`FACILITY_CMI = 54`

`FACILITY_COMPLUS = 17`

`FACILITY_CONFIGURATION = 33`

`FACILITY_CONTROL = 10`

`FACILITY_DIRECTORYSERVICE = 37`

`FACILITY_DISPATCH = 2`

`FACILITY_DPLAY = 21`

`FACILITY_FVE = 49`

`FACILITY_FWP = 50`

`FACILITY_GRAPHICS = 38`

`FACILITY_HTTP = 25`

`FACILITY_INTERNET = 12`

`FACILITY_ITF = 4`

`FACILITY_MEDIASERVER = 13`

`FACILITY_METADIRECTORY = 35`

`FACILITY_MSMQ = 14`

`FACILITY_NDIS = 52`


```
FACILITY_NULL = 0
FACILITY_PLA = 48
FACILITY_RPC = 1
FACILITY_SCARD = 16
FACILITY_SECURITY = 9
FACILITY_SETUPAPI = 15
FACILITY_SHELL = 39
FACILITY_SSPI = 9
FACILITY_STATE_MANAGEMENT = 34
FACILITY_STORAGE = 3
FACILITY_SXS = 23
FACILITY_TPM_SERVICES = 40
FACILITY_TPM_SOFTWARE = 41
FACILITY_UMI = 22
FACILITY_URT = 19
FACILITY_USERMODE_COMMONLOG = 26
FACILITY_USERMODE_FILTER_MANAGER = 31
FACILITY_USERMODE_HYPERVISOR = 53
FACILITY_WIN32 = 7
FACILITY_WINDOWS = 8
FACILITY_WINDOWSUPDATE = 36
FACILITY_WINDOWS_CE = 24
FACILITY_WINDOWS_DEFENDER = 80
FACILITY_WINRM = 51
```

GetScore (*hr*)

Turn a HRESULT into a SCORE.

HRESULT_CODE (*hr*)

Return the error code field of a HRESULT.

HRESULT_FACILITY (*hr*)

Return the facility field of a HRESULT.

HRESULT_FROM_NT (*x*)

Turn an NT error code into a HRESULT.

HRESULT_FROM_WIN32 (*scode*)

Return the HRESULT corresponding to a Win32 error code.

HRESULT_SEVERITY (*hr*)

Return the severity field of a HRESULT.

class RawErrorCodeBases: *cbc_sdk.winerror.ErrorBaseClass*

Collects all known error codes defined as raw SCODEs (from COM, OLE, etc.)

```
CACHE_E_FIRST = -2147221136
CACHE_E_LAST = -2147221121
CACHE_E_NOCACHE_UPDATED = -2147221136
CACHE_S_FIRST = 262512
CACHE_S_LAST = 262527
CAT_E_CATIDNOEXIST = -2147221152
CAT_E_FIRST = -2147221152
CAT_E_LAST = -2147221151
CAT_E_NODESCRIPTION = -2147221151
CERTDB_E_JET_ERROR = -2146873344
CERTSRV_E_BAD_REQUESTSTATUS = -2146877437
CERTSRV_E_BAD_REQUESTSUBJECT = -2146877439
CERTSRV_E_NO_REQUEST = -2146877438
CERTSRV_E_PROPERTY_EMPTY = -2146877436
CERT_E_CHAINING = -2146762486
CERT_E_CN_NO_MATCH = -2146762481
CERT_E_CRITICAL = -2146762491
CERT_E_EXPIRED = -2146762495
CERT_E_ISSUERCHAINING = -2146762489
CERT_E_MALFORMED = -2146762488
CERT_E_PATHLENCONST = -2146762492
CERT_E_PURPOSE = -2146762490
CERT_E_REVOCATION_FAILURE = -2146762482
CERT_E_REVOKED = -2146762484
CERT_E_ROLE = -2146762493
CERT_E_UNTRUSTEDROOT = -2146762487
CERT_E_UNTRUSTEDTESTROOT = -2146762483
CERT_E_VALIDITYPERIODNESTING = -2146762494
CERT_E_WRONG_USAGE = -2146762480
CLASSFACTORY_E_FIRST = -2147221232
CLASSFACTORY_E_LAST = -2147221217
CLASSFACTORY_S_FIRST = 262416
CLASSFACTORY_S_LAST = 262431
```

```
CLASS_E_CLASSNOTAVAILABLE = -2147221231
CLASS_E_NOAGGREGATION = -2147221232
CLASS_E_NOTLICENSED = -2147221230
CLIENTSITE_E_FIRST = -2147221104
CLIENTSITE_E_LAST = -2147221089
CLIENTSITE_S_FIRST = 262544
CLIENTSITE_S_LAST = 262559
CLIPBRD_E_BAD_DATA = -2147221037
CLIPBRD_E_CANT_CLOSE = -2147221036
CLIPBRD_E_CANT_EMPTY = -2147221039
CLIPBRD_E_CANT_OPEN = -2147221040
CLIPBRD_E_CANT_SET = -2147221038
CLIPBRD_E_FIRST = -2147221040
CLIPBRD_E_LAST = -2147221025
CLIPBRD_S_FIRST = 262608
CLIPBRD_S_LAST = 262623
CONVERT10_E_FIRST = -2147221056
CONVERT10_E_LAST = -2147221041
CONVERT10_E_OLESTREAM_BITMAP_TO_DIB = -2147221053
CONVERT10_E_OLESTREAM_FMT = -2147221054
CONVERT10_E_OLESTREAM_GET = -2147221056
CONVERT10_E_OLESTREAM_PUT = -2147221055
CONVERT10_E_STG_DIB_TO_BITMAP = -2147221050
CONVERT10_E_STG_FMT = -2147221052
CONVERT10_E_STG_NO_STD_STREAM = -2147221051
CONVERT10_S_FIRST = 262592
CONVERT10_S_LAST = 262607
CO_E_ACCESSCHECKFAILED = -2147220985
CO_E_ACESINWRONGORDER = -2147220969
CO_E_ACNOTINITIALIZED = -2147220965
CO_E_ALREADYINITIALIZED = -2147221007
CO_E_APPDIDNTREG = -2147220994
CO_E_APPNOTFOUND = -2147221003
CO_E_APPSINGLEUSE = -2147221002
CO_E_BAD_PATH = -2146959356
CO_E_BAD_SERVER_NAME = -2147467244
```

```
CO_E_CANTDETERMINECLASS = -2147221006
CO_E_CANT_REMOTE = -2147467245
CO_E_CLASSSTRING = -2147221005
CO_E_CLASS_CREATE_FAILED = -2146959359
CO_E_CLSREG_INCONSISTENT = -2147467233
CO_E_CONVERSIONFAILED = -2147220981
CO_E_CREATEPROCESS_FAILURE = -2147467240
CO_E_DECODEFAILED = -2147220966
CO_E_DLLNOTFOUND = -2147221000
CO_E_ERRORINAPP = -2147221001
CO_E_ERRORINDLL = -2147220999
CO_E_EXCEEDSYSACLLIMIT = -2147220970
CO_E_FAILEDTOCLOSEHANDLE = -2147220971
CO_E_FAILEDTOCREATEFILE = -2147220972
CO_E_FAILEDTOGENUUID = -2147220973
CO_E_FAILEDTOGETSECCTX = -2147220991
CO_E_FAILEDTOGETTOKENINFO = -2147220989
CO_E_FAILEDTOGETWINDIR = -2147220975
CO_E_FAILEDTOIMPERSONATE = -2147220992
CO_E_FAILEDTOOPENPROCESSTOKEN = -2147220967
CO_E_FAILEDTOOPENTHREADTOKEN = -2147220990
CO_E_FAILEDTOQUERYCLIENTBLANKET = -2147220987
CO_E_FAILEDTOSETDACL = -2147220986
CO_E_FIRST = -2147221008
CO_E_IIDREG_INCONSISTENT = -2147467232
CO_E_IIDSTRING = -2147221004
CO_E_INCOMPATIBLESTREAMVERSION = -2147220968
CO_E_INIT_CLASS_CACHE = -2147467255
CO_E_INIT_MEMORY_ALLOCATOR = -2147467256
CO_E_INIT_ONLY_SINGLE_THREADED = -2147467246
CO_E_INIT_RPC_CHANNEL = -2147467254
CO_E_INIT_SCM_EXEC_FAILURE = -2147467247
CO_E_INIT_SCM_FILE_MAPPING_EXISTS = -2147467249
CO_E_INIT_SCM_MAP_VIEW_OF_FILE = -2147467248
CO_E_INIT_SCM_MUTEX_EXISTS = -2147467250
CO_E_INIT_SHARED_ALLOCATOR = -2147467257
```

```
CO_E_INIT_TLS = -2147467258
CO_E_INIT_TLS_CHANNEL_CONTROL = -2147467252
CO_E_INIT_TLS_SET_CHANNEL_CONTROL = -2147467253
CO_E_INIT_UNACCEPTED_USER_ALLOCATOR = -2147467251
CO_E_INVALIDSID = -2147220982
CO_E_LAST = -2147220993
CO_E_LAUNCH_PERMSSION_DENIED = -2147467237
CO_E_LOOKUPACCFNAMEFAILED = -2147220977
CO_E_LOOKUPACCSIDFAILED = -2147220979
CO_E_MSI_ERROR = -2147467229
CO_E_NETACCESSAPIFAILED = -2147220984
CO_E_NOMATCHINGNAMEFOUND = -2147220978
CO_E_NOMATCHINGSIDFOUND = -2147220980
CO_E_NOTINITIALIZED = -2147221008
CO_E_NOT_SUPPORTED = -2147467231
CO_E_OBJISREG = -2147220996
CO_E_OBJNOTCONNECTED = -2147220995
CO_E_OBJNOTREG = -2147220997
CO_E_OBJSRV_RPC_FAILURE = -2146959354
CO_E_OLE1DDE_DISABLED = -2147467242
CO_E_PATHTOOLONG = -2147220974
CO_E_RELEASED = -2147220993
CO_E_RELOAD_DLL = -2147467230
CO_E_REMOTE_COMMUNICATION_FAILURE = -2147467235
CO_E_RUNAS_CREATEPROCESS_FAILURE = -2147467239
CO_E_RUNAS_LOGON_FAILURE = -2147467238
CO_E_RUNAS_SYNTAX = -2147467241
CO_E_SCM_ERROR = -2146959358
CO_E_SCM_RPC_FAILURE = -2146959357
CO_E_SERVER_EXEC_FAILURE = -2146959355
CO_E_SERVER_START_TIMEOUT = -2147467234
CO_E_SERVER_STOPPING = -2146959352
CO_E_SETSERLHNDLFAILED = -2147220976
CO_E_START_SERVICE_FAILURE = -2147467236
CO_E_TRUSTEEDOESNTMATCHCLIENT = -2147220988
CO_E_WRONGOSFORAPP = -2147220998
```

```
CO_E_WRONGTRUSTEENAMESYNTAX = -2147220983
CO_E_WRONG_SERVER_IDENTITY = -2147467243
CO_S_FIRST = 262640
CO_S_LAST = 262655
CO_S_NOTALLINTERFACES = 524306
CRYPT_E_ALREADY_DECRYPTED = -2146889719
CRYPT_E_ATTRIBUTES_MISSING = -2146889713
CRYPT_E_AUTH_ATTR_MISSING = -2146889722
CRYPT_E_BAD_ENCODE = -2146885630
CRYPT_E_BAD_LEN = -2146885631
CRYPT_E_BAD_MSG = -2146885619
CRYPT_E_CONTROL_TYPE = -2146889716
CRYPT_E_DELETED_PREV = -2146885624
CRYPT_E_EXISTS = -2146885627
CRYPT_E_FILERESIZED = -2146885595
CRYPT_E_FILE_ERROR = -2146885629
CRYPT_E_HASH_VALUE = -2146889721
CRYPT_E_INVALID_IA5_STRING = -2146885598
CRYPT_E_INVALID_INDEX = -2146889720
CRYPT_E_INVALID_MSG_TYPE = -2146889724
CRYPT_E_INVALID_NUMERIC_STRING = -2146885600
CRYPT_E_INVALID_PRINTABLE_STRING = -2146885599
CRYPT_E_INVALID_X500_STRING = -2146885597
CRYPT_E_ISSUER_SERIALNUMBER = -2146889715
CRYPT_E_MSG_ERROR = -2146889727
CRYPT_E_NOT_CHAR_STRING = -2146885596
CRYPT_E_NOT_DECRYPTED = -2146889718
CRYPT_E_NOT_FOUND = -2146885628
CRYPT_E_NOT_IN_CTL = -2146885590
CRYPT_E_NOT_IN_REVOCATION_DATABASE = -2146885612
CRYPT_E_NO_DECRYPT_CERT = -2146885620
CRYPT_E_NO_KEY_PROPERTY = -2146885621
CRYPT_E_NO_MATCH = -2146885623
CRYPT_E_NO_PROVIDER = -2146885626
CRYPT_E_NO_REVOCATION_CHECK = -2146885614
CRYPT_E_NO_REVOCATION_DLL = -2146885615
```

```
CRYPT_E_NO_SIGNER = -2146885618
CRYPT_E_NO_TRUSTED_SIGNER = -2146885589
CRYPT_E_NO_VERIFY_USAGE_CHECK = -2146885592
CRYPT_E_NO_VERIFY_USAGE_DLL = -2146885593
CRYPT_E_OID_FORMAT = -2146889725
CRYPT_E_OSS_ERROR = -2146881536
CRYPT_E_PENDING_CLOSE = -2146885617
CRYPT_E_RECIPIENT_NOT_FOUND = -2146889717
CRYPT_E_REVOCATION_OFFLINE = -2146885613
CRYPT_E_REVOKED = -2146885616
CRYPT_E_SECURITY_SETTINGS = -2146885594
CRYPT_E_SELF_SIGNED = -2146885625
CRYPT_E_SIGNER_NOT_FOUND = -2146889714
CRYPT_E_STREAM_INSUFFICIENT_DATA = -2146889711
CRYPT_E_STREAM_MSG_NOT_READY = -2146889712
CRYPT_E_UNEXPECTED_ENCODING = -2146889723
CRYPT_E_UNEXPECTED_MSG_TYPE = -2146885622
CRYPT_E_UNKNOWN_ALGO = -2146889726
CRYPT_E_VERIFY_USAGE_OFFLINE = -2146885591
CS_E_CLASS_NOTFOUND = -2147221146
CS_E_FIRST = -2147221148
CS_E_INVALID_VERSION = -2147221145
CS_E_LAST = -2147221144
CS_E_NOT_DELETABLE = -2147221147
CS_E_NO_CLASSSTORE = -2147221144
CS_E_PACKAGE_NOTFOUND = -2147221148
DATA_E_FIRST = -2147221200
DATA_E_LAST = -2147221185
DATA_S_FIRST = 262448
DATA_S_LAST = 262463
DIGSIG_E_CRYPTO = -2146762744
DIGSIG_E_DECODE = -2146762746
DIGSIG_E_ENCODE = -2146762747
DIGSIG_E_EXTENSIBILITY = -2146762745
DISP_E_ARRAYISLOCKED = -2147352563
DISP_E_BADCALLEE = -2147352560
```

```
DISP_E_BADINDEX = -2147352565
DISP_E_BADPARAMCOUNT = -2147352562
DISP_E_BADVARTYPE = -2147352568
DISP_E_DIVBYZERO = -2147352558
DISP_E_EXCEPTION = -2147352567
DISP_E_MEMBERNOTFOUND = -2147352573
DISP_E_NONAMEDARGS = -2147352569
DISP_E_NOTACCOLLECTION = -2147352559
DISP_E_OVERFLOW = -2147352566
DISP_E_PARAMNOTFOUND = -2147352572
DISP_E_PARAMNOTOPTIONAL = -2147352561
DISP_E_TYPERISMATCH = -2147352571
DISP_E_UNKNOWNINTERFACE = -2147352575
DISP_E_UNKNOWNLCID = -2147352564
DISP_E_UNKNOWNNAME = -2147352570
DRAGDROP_E_ALREADYREGISTERED = -2147221247
DRAGDROP_E_FIRST = -2147221248
DRAGDROP_E_INVALIDHWND = -2147221246
DRAGDROP_E_LAST = -2147221233
DRAGDROP_E_NOTREGISTERED = -2147221248
DRAGDROP_S_FIRST = 262400
DRAGDROP_S_LAST = 262415
DV_E_CLIPFORMAT = -2147221398
DV_E_DVASPECT = -2147221397
DV_E_DVTARGETDEVICE = -2147221403
DV_E_DVTARGETDEVICE_SIZE = -2147221396
DV_E_FORMATETC = -2147221404
DV_E_LINDEX = -2147221400
DV_E_NOVIEWOBJECT = -2147221395
DV_E_STATDATA = -2147221401
DV_E_STGMEDIUM = -2147221402
DV_E_TYMED = -2147221399
ENUM_E_FIRST = -2147221072
ENUM_E_LAST = -2147221057
ENUM_S_FIRST = 262576
ENUM_S_LAST = 262591
```



```
E_ABORT = -2147467260
E_ACCESSDENIED = -2147024891
E_FAIL = -2147467259
E_HANDLE = -2147024890
E_INVALIDARG = -2147024809
E_NOINTERFACE = -2147467262
E_NOTIMPL = -2147467263
E_OUTOFMEMORY = -2147024882
E_PENDING = -2147483638
E_POINTER = -2147467261
E_UNEXPECTED = -2147418113
INPLACE_E_FIRST = -2147221088
INPLACE_E_LAST = -2147221073
INPLACE_E_NOTOOLSPACE = -2147221087
INPLACE_E_NOTUNDOABLE = -2147221088
INPLACE_S_FIRST = 262560
INPLACE_S_LAST = 262575
MARSHAL_E_FIRST = -2147221216
MARSHAL_E_LAST = -2147221201
MARSHAL_S_FIRST = 262432
MARSHAL_S_LAST = 262447
MEM_E_INVALID_LINK = -2146959344
MEM_E_INVALID_ROOT = -2146959351
MEM_E_INVALID_SIZE = -2146959343
MK_E_CANTOPENFILE = -2147221014
MK_E_CONNECTMANUALLY = -2147221024
MK_E_ENUMERATION_FAILED = -2147221009
MK_E_EXCEEDEDDEADLINE = -2147221023
MK_E_FIRST = -2147221024
MK_E_INTERMEDIATEINTERFACENOTSUPPORTED = -2147221017
MK_E_INVALIDEXTENSION = -2147221018
MK_E_LAST = -2147221009
MK_E_MUSTBOTHERUSER = -2147221013
MK_E_NEEDGENERIC = -2147221022
MK_E_NOINVERSE = -2147221012
MK_E_NOOBJECT = -2147221019
```

```
MK_E_NOPREFIX = -2147221010
MK_E_NOSTORAGE = -2147221011
MK_E_NOTBINDABLE = -2147221016
MK_E_NOTBOUND = -2147221015
MK_E_NO_NORMALIZED = -2146959353
MK_E_SYNTAX = -2147221020
MK_E_UNAVAILABLE = -2147221021
MK_S_FIRST = 262624
MK_S_LAST = 262639
NTE_BAD_ALGID = -2146893816
NTE_BAD_DATA = -2146893819
NTE_BAD_FLAGS = -2146893815
NTE_BAD_HASH = -2146893822
NTE_BAD_HASH_STATE = -2146893812
NTE_BAD_KEY = -2146893821
NTE_BAD_KEYSET = -2146893802
NTE_BAD_KEYSET_PARAM = -2146893793
NTE_BAD_KEY_STATE = -2146893813
NTE_BAD_LEN = -2146893820
NTE_BAD_PROVIDER = -2146893805
NTE_BAD_PROV_TYPE = -2146893804
NTE_BAD_PUBLIC_KEY = -2146893803
NTE_BAD_SIGNATURE = -2146893818
NTE_BAD_TYPE = -2146893814
NTE_BAD_UID = -2146893823
NTE_BAD_VER = -2146893817
NTE_DOUBLE_ENCRYPT = -2146893806
NTE_EXISTS = -2146893809
NTE_FAIL = -2146893792
NTE_KEYSET_ENTRY_BAD = -2146893798
NTE_KEYSET_NOT_DEF = -2146893799
NTE_NOT_FOUND = -2146893807
NTE_NO_KEY = -2146893811
NTE_NO_MEMORY = -2146893810
NTE_OP_OK = 0
NTE_PERM = -2146893808
```

```
NTE_PROVIDER_DLL_FAIL = -2146893795
NTE_PROV_DLL_NOT_FOUND = -2146893794
NTE_PROV_TYPE_ENTRY_BAD = -2146893800
NTE_PROV_TYPE_NOT_DEF = -2146893801
NTE_PROV_TYPE_NO_MATCH = -2146893797
NTE_SIGNATURE_FILE_BAD = -2146893796
NTE_SYS_ERR = -2146893791
OLEOBJ_E_FIRST = -2147221120
OLEOBJ_E_INVALIDVERB = -2147221119
OLEOBJ_E_LAST = -2147221105
OLEOBJ_E_NOVERBS = -2147221120
OLEOBJ_S_FIRST = 262528
OLEOBJ_S_LAST = 262543
OLE_E_ADVFE = -2147221503
OLE_E_ADVISENOTSUPPORTED = -2147221501
OLE_E_BLANK = -2147221497
OLE_E_CANTCONVERT = -2147221487
OLE_E_CANT_BINDTOSOURCE = -2147221494
OLE_E_CANT_GETMONIKER = -2147221495
OLE_E_CLASSDIFF = -2147221496
OLE_E_ENUM_NOMORE = -2147221502
OLE_E_FIRST = -2147221504
OLE_E_INVALIDHWND = -2147221489
OLE_E_INVALIDRECT = -2147221491
OLE_E_LAST = -2147221249
OLE_E_NOCACHE = -2147221498
OLE_E_NOCONNECTION = -2147221500
OLE_E_NOSTORAGE = -2147221486
OLE_E_NOTRUNNING = -2147221499
OLE_E_NOT_INPLACEACTIVE = -2147221488
OLE_E_OLEVERB = -2147221504
OLE_E_PROMPTSAVECANCELLED = -2147221492
OLE_E_STATIC = -2147221493
OLE_E_WRONGCOMPOBJ = -2147221490
OLE_S_FIRST = 262144
OLE_S_LAST = 262399
```

```
PERSIST_E_NOTSELSIZING = -2146762741
PERSIST_E_SIZEDEFINITE = -2146762743
PERSIST_E_SIZEINDEFINITE = -2146762742
REGDB_E_CLASSNOTREG = -2147221164
REGDB_E_FIRST = -2147221168
REGDB_E_IIDNOTREG = -2147221163
REGDB_E_INVALIDVALUE = -2147221165
REGDB_E_KEYMISSING = -2147221166
REGDB_E_LAST = -2147221153
REGDB_E_READREGDB = -2147221168
REGDB_E_WRITEREGDB = -2147221167
REGDB_S_FIRST = 262480
REGDB_S_LAST = 262495
RPC_E_ACCESS_DENIED = -2147417829
RPC_E_ATTEMPTED_MULTITHREAD = -2147417854
RPC_E_CALL_CANCELED = -2147418110
RPC_E_CALL_COMPLETE = -2147417833
RPC_E_CALL_REJECTED = -2147418111
RPC_E_CANTCALLOUT_AGAIN = -2147418095
RPC_E_CANTCALLOUT_INASYNCCALL = -2147418108
RPC_E_CANTCALLOUT_INEXTERNALCALL = -2147418107
RPC_E_CANTCALLOUT_ININPUTSYNCCALL = -2147417843
RPC_E_CANTPOST_INSENDCALL = -2147418109
RPC_E_CANTTRANSMIT_CALL = -2147418102
RPC_E_CHANGED_MODE = -2147417850
RPC_E_CLIENT_CANTMARSHAL_DATA = -2147418101
RPC_E_CLIENT_CANTUNMARSHAL_DATA = -2147418100
RPC_E_CLIENT_DIED = -2147418104
RPC_E_CONNECTION_TERMINATED = -2147418106
RPC_E_DISCONNECTED = -2147417848
RPC_E_FAULT = -2147417852
RPC_E_INVALIDMETHOD = -2147417849
RPC_E_INVALID_CALldata = -2147417844
RPC_E_INVALID_DATA = -2147418097
RPC_E_INVALID_DATAPACKET = -2147418103
RPC_E_INVALID_EXTENSION = -2147417838
```

```
RPC_E_INVALID_HEADER = -2147417839
RPC_E_INVALID_IPID = -2147417837
RPC_E_INVALID_OBJECT = -2147417836
RPC_E_INVALID_OBJREF = -2147417827
RPC_E_INVALID_PARAMETER = -2147418096
RPC_E_NOT_REGISTERED = -2147417853
RPC_E_NO_CONTEXT = -2147417826
RPC_E_NO_GOOD_SECURITY_PACKAGES = -2147417830
RPC_E_NO_SYNC = -2147417824
RPC_E_OUT_OF_RESOURCES = -2147417855
RPC_E_REMOTE_DISABLED = -2147417828
RPC_E_RETRY = -2147417847
RPC_E_SERVERCALL_REJECTED = -2147417845
RPC_E_SERVERCALL_RETRYLATER = -2147417846
RPC_E_SERVERFAULT = -2147417851
RPC_E_SERVER_CANTMARSHAL_DATA = -2147418099
RPC_E_SERVER_CANTUNMARSHAL_DATA = -2147418098
RPC_E_SERVER_DIED = -2147418105
RPC_E_SERVER_DIED_DNE = -2147418094
RPC_E_SYS_CALL_FAILED = -2147417856
RPC_E_THREAD_NOT_INIT = -2147417841
RPC_E_TIMEOUT = -2147417825
RPC_E_TOO_LATE = -2147417831
RPC_E_UNEXPECTED = -2147352577
RPC_E_UNSECURE_CALL = -2147417832
RPC_E_VERSION_MISMATCH = -2147417840
RPC_E_WRONG_THREAD = -2147417842
RPC_S_CALLPENDING = -2147417835
RPC_S_WAITONTIMER = -2147417834
SPAPI_E_BAD_INTERFACE_INSTALLSECT = -2146500067
SPAPI_E_BAD_SECTION_NAME_LINE = -2146500607
SPAPI_E_BAD_SERVICE_INSTALLSECT = -2146500073
SPAPI_E_CANT_LOAD_CLASS_ICON = -2146500084
SPAPI_E_CLASS_MISMATCH = -2146500095
SPAPI_E_DEVICE_INTERFACE_ACTIVE = -2146500069
SPAPI_E_DEVICE_INTERFACE_REMOVED = -2146500068
```

```
SPAPI_E_DEVINFO_DATA_LOCKED = -2146500077
SPAPI_E_DEVINFO_LIST_LOCKED = -2146500078
SPAPI_E_DEVINFO_NOT_REGISTERED = -2146500088
SPAPI_E_DEVINST_ALREADY_EXISTS = -2146500089
SPAPI_E_DI_BAD_PATH = -2146500076
SPAPI_E_DI_DONT_INSTALL = -2146500053
SPAPI_E_DI_DO_DEFAULT = -2146500082
SPAPI_E_DI_NOFILECOPY = -2146500081
SPAPI_E_DI_POSTPROCESSING_REQUIRED = -2146500058
SPAPI_E_DUPLICATE_FOUND = -2146500094
SPAPI_E_ERROR_NOT_INSTALLED = -2146496512
SPAPI_E_EXPECTED_SECTION_NAME = -2146500608
SPAPI_E_FILEQUEUE_LOCKED = -2146500074
SPAPI_E_GENERAL_SYNTAX = -2146500605
SPAPI_E_INVALID_CLASS = -2146500090
SPAPI_E_INVALID_CLASS_INSTALLER = -2146500083
SPAPI_E_INVALID_COINSTALLER = -2146500057
SPAPI_E_INVALID_DEVINST_NAME = -2146500091
SPAPI_E_INVALID_FILTER_DRIVER = -2146500052
SPAPI_E_INVALID_HWPROFILE = -2146500080
SPAPI_E_INVALID_INF_LOGCONFIG = -2146500054
SPAPI_E_INVALID_MACHINENAME = -2146500064
SPAPI_E_INVALID_PROPPAGE_PROVIDER = -2146500060
SPAPI_E_INVALID_REFERENCE_STRING = -2146500065
SPAPI_E_INVALID_REG_PROPERTY = -2146500087
SPAPI_E_KEY_DOES_NOT_EXIST = -2146500092
SPAPI_E_LINE_NOT_FOUND = -2146500350
SPAPI_E_MACHINE_UNAVAILABLE = -2146500062
SPAPI_E_NO_ASSOCIATED_CLASS = -2146500096
SPAPI_E_NO_ASSOCIATED_SERVICE = -2146500071
SPAPI_E_NO_CLASSINSTALL_PARAMS = -2146500075
SPAPI_E_NO_CLASS_DRIVER_LIST = -2146500072
SPAPI_E_NO_COMPAT_DRIVERS = -2146500056
SPAPI_E_NO_CONFIGMGR_SERVICES = -2146500061
SPAPI_E_NO_DEFAULT_DEVICE_INTERFACE = -2146500070
SPAPI_E_NO_DEVICE_ICON = -2146500055
```

```
SPAPI_E_NO_DEVICE_SELECTED = -2146500079
SPAPI_E_NO_DRIVER_SELECTED = -2146500093
SPAPI_E_NO_INF = -2146500086
SPAPI_E_NO_SUCH_DEVICE_INTERFACE = -2146500059
SPAPI_E_NO_SUCH_DEVINST = -2146500085
SPAPI_E_NO_SUCH_INTERFACE_CLASS = -2146500066
SPAPI_E_REMOTE_COMM_FAILURE = -2146500063
SPAPI_E_SECTION_NAME_TOO_LONG = -2146500606
SPAPI_E_SECTION_NOT_FOUND = -2146500351
SPAPI_E_WRONG_INF_STYLE = -2146500352
STG_E_ABNORMALAPIEXIT = -2147286790
STG_E_ACCESSDENIED = -2147287035
STG_E_BADBASEADDRESS = -2147286768
STG_E_CANTSAVE = -2147286781
STG_E_DISKISWRITEPROTECTED = -2147287021
STG_E_DOCFILECORRUPT = -2147286775
STG_E_EXTANTMARSHALLINGS = -2147286776
STG_E_FILEALREADYEXISTS = -2147286960
STG_E_FILENOTFOUND = -2147287038
STG_E_INCOMPLETE = -2147286527
STG_E_INSUFFICIENTMEMORY = -2147287032
STG_E_INUSE = -2147286784
STG_E_INVALIDFLAG = -2147286785
STG_E_INVALIDFUNCTION = -2147287039
STG_E_INVALIDHANDLE = -2147287034
STG_E_INVALIDHEADER = -2147286789
STG_E_INVALIDNAME = -2147286788
STG_E_INVALIDPARAMETER = -2147286953
STG_E_INVALIDPOINTER = -2147287031
STG_E_LOCKVIOLATION = -2147287007
STG_E_MEDIUMFULL = -2147286928
STG_E_NOMOREFILES = -2147287022
STG_E_NOTCURRENT = -2147286783
STG_E_NOTFILEBASEDSTORAGE = -2147286777
STG_E_OLDDLL = -2147286779
STG_E_OLDFORMAT = -2147286780
```

```
STG_E_PATHNOTFOUND = -2147287037
STG_E_PROPSETMISMATCHED = -2147286800
STG_E_READFAULT = -2147287010
STG_E_REVERTED = -2147286782
STG_E_SEEKERROR = -2147287015
STG_E_SHAREREQUIRED = -2147286778
STG_E_SHAREVIOLATION = -2147287008
STG_E_TERMINATED = -2147286526
STG_E_TOOMANYOPENFILES = -2147287036
STG_E_UNIMPLEMENTEDFUNCTION = -2147286786
STG_E_UNKNOWN = -2147286787
STG_E_WRITEFAULT = -2147287011
STG_S_BLOCK = 197121
STG_S_CANNOTCONSOLIDATE = 197126
STG_S_CONSOLIDATIONFAILED = 197125
STG_S_CONVERTED = 197120
STG_S_MONITORING = 197123
STG_S_MULTIPLEOPENS = 197124
STG_S_RETRYNOW = 197122
TRUST_E_ACTION_UNKNOWN = -2146762750
TRUST_E_BAD_DIGEST = -2146869232
TRUST_E_BASIC_CONSTRAINTS = -2146869223
TRUST_E_CERT_SIGNATURE = -2146869244
TRUST_E_COUNTER_SIGNER = -2146869245
TRUST_E_FAIL = -2146762485
TRUST_E_FINANCIAL_CRITERIA = -2146869218
TRUST_E_NOSIGNATURE = -2146762496
TRUST_E_NO_SIGNER_CERT = -2146869246
TRUST_E_PROVIDER_UNKNOWN = -2146762751
TRUST_E_SUBJECT_FORM_UNKNOWN = -2146762749
TRUST_E_SUBJECT_NOT_TRUSTED = -2146762748
TRUST_E_SYSTEM_ERROR = -2146869247
TRUST_E_TIME_STAMP = -2146869243
TYPE_E_AMBIGUOUSNAME = -2147319764
TYPE_E_BADMODULEKIND = -2147317571
TYPE_E_BUFFERTOOSMALL = -2147319786
```



```
TYPE_E_CANTCREATETMPFILE = -2147316573
TYPE_E_CANTLOADLIBRARY = -2147312566
TYPE_E_CIRCULARTYPE = -2147312508
TYPE_E_DLLFUNCTIONNOTFOUND = -2147319761
TYPE_E_DUPLICATEID = -2147317562
TYPE_E_ELEMENTNOTFOUND = -2147319765
TYPE_E_FIELDNOTFOUND = -2147319785
TYPE_E_INCONSISTENTPROPFUNCS = -2147312509
TYPE_E_INVALIDID = -2147317553
TYPE_E_INVALIDSTATE = -2147319767
TYPE_E_INVDATAREAD = -2147319784
TYPE_E_IOERROR = -2147316574
TYPE_E_LIBNOTREGISTERED = -2147319779
TYPE_E_NAMECONFLICT = -2147319763
TYPE_E_OUTOFBOUNDS = -2147316575
TYPE_E_QUALIFIEDNAMEDISALLOWED = -2147319768
TYPE_E_REGISTRYACCESS = -2147319780
TYPE_E_SIZETOOBIG = -2147317563
TYPE_E_TYPEMISMATCH = -2147316576
TYPE_E_UNDEFINEDTYPE = -2147319769
TYPE_E_UNKNOWNLCID = -2147319762
TYPE_E_UNSUPFORMAT = -2147319783
TYPE_E_WRONGTYPEKIND = -2147319766
VIEW_E_DRAW = -2147221184
VIEW_E_FIRST = -2147221184
VIEW_E_LAST = -2147221169
VIEW_S_FIRST = 262464
VIEW_S_LAST = 262479
win16_E_ABORT = -2147483641
win16_E_ACCESSDENIED = -2147483639
win16_E_FAIL = -2147483640
win16_E_HANDLE = -2147483642
win16_E_INVALIDARG = -2147483645
win16_E_NOINTERFACE = -2147483644
win16_E_NOTIMPL = -2147483647
win16_E_OUTOFMEMORY = -2147483646
```

win16_E_POINTER = -2147483643

ResultFromScore (*sc*)

Turn a SCORE into a HRESULT.

SCORE_CODE (*sc*)

Return the error code field of a SCORE.

SCORE_FACILITY (*sc*)

Return the facility field of a SCORE.

SCORE_SEVERITY (*sc*)

Return the severity field of a SCORE.

SUCCEEDED (*Status*)

Return True iff a HRESULT/SCORE status represents success.

class Win32Error

Bases: *cbc_sdk.winerror.ErrorBaseClass*

Collects all the Win32 error codes.

DS_S_SUCCESS = 0

EPT_S_CANT_CREATE = 1899

EPT_S_CANT_PERFORM_OP = 1752

EPT_S_INVALID_ENTRY = 1751

EPT_S_NOT_REGISTERED = 1753

ERROR_ABANDONED_WAIT_0 = 735

ERROR_ABANDONED_WAIT_63 = 736

ERROR_ABANDON_HIBERFILE = 787

ERROR_ABIOS_ERROR = 538

ERROR_ACCESS_AUDIT_BY_POLICY = 785

ERROR_ACCESS_DENIED = 5

ERROR_ACCESS_DISABLED_NO_SAFER_UI_BY_POLICY = 786

ERROR_ACCOUNT_DISABLED = 1331

ERROR_ACCOUNT_EXPIRED = 1793

ERROR_ACCOUNT_LOCKED_OUT = 1909

ERROR_ACCOUNT_RESTRICTION = 1327

ERROR_ACPI_ERROR = 669

ERROR_ACTIVATION_COUNT_EXCEEDED = 7059

ERROR_ACTIVE_CONNECTIONS = 2402

ERROR_ADAP_HDW_ERR = 57

ERROR_ADDRESS_ALREADY_ASSOCIATED = 1227

ERROR_ADDRESS_NOT_ASSOCIATED = 1228

ERROR_ALERTED = 739

ERROR_ALIAS_EXISTS = 1379

```
ERROR_ALLOCATE_BUCKET = 602
ERROR_ALLOTTED_SPACE_EXCEEDED = 1344
ERROR_ALL_NODES_NOT_AVAILABLE = 5037
ERROR_ALL_USER_TRUST_QUOTA_EXCEEDED = 1933
ERROR_ALREADY_ASSIGNED = 85
ERROR_ALREADY_EXISTS = 183
ERROR_ALREADY_INITIALIZED = 1247
ERROR_ALREADY_REGISTERED = 1242
ERROR_ALREADY_RUNNING_LKG = 1074
ERROR_ALREADY_WAITING = 1904
ERROR_ALREADY_WIN32 = 719
ERROR_APP_INIT_FAILURE = 575
ERROR_APP_WRONG_OS = 1151
ERROR_ARBITRATION_UNHANDLED = 723
ERROR_ARENA_TRASHED = 7
ERROR_ARITHMETIC_OVERFLOW = 534
ERROR_ASSERTION_FAILURE = 668
ERROR_ATOMIC_LOCKS_NOT_SUPPORTED = 174
ERROR_AUDIT_FAILED = 606
ERROR_AUTHENTICATION_FIREWALL_FAILED = 1935
ERROR_AUTHIP_FAILURE = 1469
ERROR_AUTODATASEG_EXCEEDS_64k = 199
ERROR_BACKUP_CONTROLLER = 586
ERROR_BADDB = 1009
ERROR_BADKEY = 1010
ERROR_BADSTARTPOSITION = 778
ERROR_BAD_ACCESSOR_FLAGS = 773
ERROR_BAD_ARGUMENTS = 160
ERROR_BAD_CLUSTERS = 6849
ERROR_BAD_COMMAND = 22
ERROR_BAD_COMPRESSION_BUFFER = 605
ERROR_BAD_CONFIGURATION = 1610
ERROR_BAD_CURRENT_DIRECTORY = 703
ERROR_BAD_DATABASE_VERSION = 1613
ERROR_BAD_DESCRIPTOR_FORMAT = 1361
ERROR_BAD_DEVICE = 1200
```

```
ERROR_BAD_DEV_TYPE = 66
ERROR_BAD_DLL_ENTRYPOINT = 609
ERROR_BAD_DRIVER = 2001
ERROR_BAD_DRIVER_LEVEL = 119
ERROR_BAD_ENVIRONMENT = 10
ERROR_BAD_EXE_FORMAT = 193
ERROR_BAD_FILE_TYPE = 222
ERROR_BAD_FORMAT = 11
ERROR_BAD_FUNCTION_TABLE = 559
ERROR_BAD_IMPERSONATION_LEVEL = 1346
ERROR_BAD_INHERITANCE_ACL = 1340
ERROR_BAD_LENGTH = 24
ERROR_BAD_LOGON_SESSION_STATE = 1365
ERROR_BAD_MCFG_TABLE = 791
ERROR_BAD_NETPATH = 53
ERROR_BAD_NET_NAME = 67
ERROR_BAD_NET_RESP = 58
ERROR_BAD_PATHNAME = 161
ERROR_BAD_PIPE = 230
ERROR_BAD_PROFILE = 1206
ERROR_BAD_PROVIDER = 1204
ERROR_BAD_QUERY_SYNTAX = 1615
ERROR_BAD_RECOVERY_POLICY = 6012
ERROR_BAD_REM_ADAP = 60
ERROR_BAD_SERVICE_ENTRYPOINT = 610
ERROR_BAD_STACK = 543
ERROR_BAD_THREADID_ADDR = 159
ERROR_BAD_TOKEN_TYPE = 1349
ERROR_BAD_UNIT = 20
ERROR_BAD_USERNAME = 2202
ERROR_BAD_VALIDATION_CLASS = 1348
ERROR_BEGINNING_OF_MEDIA = 1102
ERROR_BIOS_FAILED_TO_CONNECT_INTERRUPT = 585
ERROR_BOOT_ALREADY_ACCEPTED = 1076
ERROR_BROKEN_PIPE = 109
ERROR_BUFFER_ALL_ZEROS = 754
```

```
ERROR_BUFFER_OVERFLOW = 111
ERROR_BUSY = 170
ERROR_BUSY_DRIVE = 142
ERROR_BUS_RESET = 1111
ERROR_CACHE_PAGE_LOCKED = 752
ERROR_CALLBACK_POP_STACK = 768
ERROR_CALL_NOT_IMPLEMENTED = 120
ERROR_CANCELLED = 1223
ERROR_CANCEL_VIOLATION = 173
ERROR_CANNOT_ABORT_TRANSACTIONS = 6848
ERROR_CANNOT_ACCEPT_TRANSACTED_WORK = 6847
ERROR_CANNOT_COPY = 266
ERROR_CANNOT_DETECT_DRIVER_FAILURE = 1080
ERROR_CANNOT_DETECT_PROCESS_ABORT = 1081
ERROR_CANNOT_EXECUTE_FILE_IN_TRANSACTION = 6838
ERROR_CANNOT_FIND_WND_CLASS = 1407
ERROR_CANNOT_IMPERSONATE = 1368
ERROR_CANNOT_LOAD_REGISTRY_FILE = 589
ERROR_CANNOT_MAKE = 82
ERROR_CANNOT_OPEN_PROFILE = 1205
ERROR_CANTFETCHBACKWARDS = 770
ERROR_CANTOPEN = 1011
ERROR_CANTREAD = 1012
ERROR_CANTSCROLLBACKWARDS = 771
ERROR_CANTWRITE = 1013
ERROR_CANT_ACCESS_DOMAIN_INFO = 1351
ERROR_CANT_ACCESS_FILE = 1920
ERROR_CANT_BREAK_TRANSACTIONAL_DEPENDENCY = 6824
ERROR_CANT_CREATE_MORE_STREAM_MINIVERSIONS = 6812
ERROR_CANT_CROSS_RM_BOUNDARY = 6825
ERROR_CANT_DELETE_LAST_ITEM = 4335
ERROR_CANT_DISABLE_MANDATORY = 1310
ERROR_CANT_ENABLE_DENY_ONLY = 629
ERROR_CANT_EVICT_ACTIVE_NODE = 5009
ERROR_CANT_OPEN_ANONYMOUS = 1347
ERROR_CANT_OPEN_MINIVERSION_WITH_MODIFY_INTENT = 6811
```

```
ERROR_CANT_RECOVER_WITH_HANDLE_OPEN = 6818
ERROR_CANT_RESOLVE_FILENAME = 1921
ERROR_CANT_TERMINATE_SELF = 555
ERROR_CANT_WAIT = 554
ERROR_CAN_NOT_COMPLETE = 1003
ERROR_CAN_NOT_DEL_LOCAL_WINS = 4001
ERROR_CARDBUS_NOT_SUPPORTED = 724
ERROR_CHECKING_FILE_SYSTEM = 712
ERROR_CHECKOUT_REQUIRED = 221
ERROR_CHILD_MUST_BE_VOLATILE = 1021
ERROR_CHILD_NOT_COMPLETE = 129
ERROR_CHILD_WINDOW_MENU = 1436
ERROR_CIRCULAR_DEPENDENCY = 1059
ERROR_CLASS_ALREADY_EXISTS = 1410
ERROR_CLASS_DOES_NOT_EXIST = 1411
ERROR_CLASS_HAS_WINDOWS = 1412
ERROR_CLEANER_CARTRIDGE_INSTALLED = 4340
ERROR_CLEANER_CARTRIDGE_SPENT = 4333
ERROR_CLEANER_SLOT_NOT_SET = 4332
ERROR_CLEANER_SLOT_SET = 4331
ERROR_CLIENT_SERVER_PARAMETERS_INVALID = 597
ERROR_CLIPBOARD_NOT_OPEN = 1418
ERROR_CLIPPING_NOT_SUPPORTED = 2005
ERROR_CLUSCFG_ALREADY_COMMITTED = 5901
ERROR_CLUSCFG_ROLLBACK_FAILED = 5902
ERROR_CLUSCFG_SYSTEM_DISK_DRIVE_LETTER_CONFLICT = 5903
ERROR_CLUSTERLOG_CHKPOINT_NOT_FOUND = 5032
ERROR_CLUSTERLOG_CORRUPT = 5029
ERROR_CLUSTERLOG_EXCEEDS_MAXSIZE = 5031
ERROR_CLUSTERLOG_NOT_ENOUGH_SPACE = 5033
ERROR_CLUSTERLOG_RECORD_EXCEEDS_MAXSIZE = 5030
ERROR_CLUSTER_CANT_CREATE_DUP_CLUSTER_NAME = 5900
ERROR_CLUSTER_CANT_DESERIALIZE_DATA = 5923
ERROR_CLUSTER_DATABASE_SEQMISMATCH = 5083
ERROR_CLUSTER_DATABASE_TRANSACTION_IN_PROGRESS = 5918
ERROR_CLUSTER_DATABASE_TRANSACTION_NOT_IN_PROGRESS = 5919
```

`ERROR_CLUSTER_EVICT_WITHOUT_CLEANUP = 5896`
`ERROR_CLUSTER_GROUP_MOVING = 5908`
`ERROR_CLUSTER_GUM_NOT_LOCKER = 5085`
`ERROR_CLUSTER_INCOMPATIBLE_VERSIONS = 5075`
`ERROR_CLUSTER_INSTANCE_ID_MISMATCH = 5893`
`ERROR_CLUSTER_INTERNAL_INVALID_FUNCTION = 5912`
`ERROR_CLUSTER_INVALID_IPV6_NETWORK = 5926`
`ERROR_CLUSTER_INVALID_IPV6_TUNNEL_NETWORK = 5927`
`ERROR_CLUSTER_INVALID_NETWORK = 5054`
`ERROR_CLUSTER_INVALID_NETWORK_PROVIDER = 5049`
`ERROR_CLUSTER_INVALID_NODE = 5039`
`ERROR_CLUSTER_INVALID_REQUEST = 5048`
`ERROR_CLUSTER_INVALID_STRING_FORMAT = 5917`
`ERROR_CLUSTER_INVALID_STRING_TERMINATION = 5916`
`ERROR_CLUSTER_IPADDR_IN_USE = 5057`
`ERROR_CLUSTER_JOIN_ABORTED = 5074`
`ERROR_CLUSTER_JOIN_IN_PROGRESS = 5041`
`ERROR_CLUSTER_JOIN_NOT_IN_PROGRESS = 5053`
`ERROR_CLUSTER_LAST_INTERNAL_NETWORK = 5066`
`ERROR_CLUSTER_LOCAL_NODE_NOT_FOUND = 5043`
`ERROR_CLUSTER_MAXNUM_OF_RESOURCES_EXCEEDED = 5076`
`ERROR_CLUSTER_MEMBERSHIP_HALT = 5892`
`ERROR_CLUSTER_MEMBERSHIP_INVALID_STATE = 5890`
`ERROR_CLUSTER_MISMATCHED_COMPUTER_ACCT_NAME = 5905`
`ERROR_CLUSTER_NETINTERFACE_EXISTS = 5046`
`ERROR_CLUSTER_NETINTERFACE_NOT_FOUND = 5047`
`ERROR_CLUSTER_NETWORK_ALREADY_OFFLINE = 5064`
`ERROR_CLUSTER_NETWORK_ALREADY_ONLINE = 5063`
`ERROR_CLUSTER_NETWORK_EXISTS = 5044`
`ERROR_CLUSTER_NETWORK_HAS_DEPENDENTS = 5067`
`ERROR_CLUSTER_NETWORK_NOT_FOUND = 5045`
`ERROR_CLUSTER_NETWORK_NOT_FOUND_FOR_IP = 5894`
`ERROR_CLUSTER_NETWORK_NOT_INTERNAL = 5060`
`ERROR_CLUSTER_NODE_ALREADY_DOWN = 5062`
`ERROR_CLUSTER_NODE_ALREADY_HAS_DFS_ROOT = 5088`
`ERROR_CLUSTER_NODE_ALREADY_MEMBER = 5065`

```
ERROR_CLUSTER_NODE_ALREADY_UP = 5061
ERROR_CLUSTER_NODE_DOWN = 5050
ERROR_CLUSTER_NODE_EXISTS = 5040
ERROR_CLUSTER_NODE_NOT_FOUND = 5042
ERROR_CLUSTER_NODE_NOT_MEMBER = 5052
ERROR_CLUSTER_NODE_NOT_PAUSED = 5058
ERROR_CLUSTER_NODE_NOT_READY = 5072
ERROR_CLUSTER_NODE_PAUSED = 5070
ERROR_CLUSTER_NODE_SHUTTING_DOWN = 5073
ERROR_CLUSTER_NODE_UNREACHABLE = 5051
ERROR_CLUSTER_NODE_UP = 5056
ERROR_CLUSTER_NOT_INSTALLED = 5932
ERROR_CLUSTER_NO_NET_ADAPTERS = 5906
ERROR_CLUSTER_NO_QUORUM = 5925
ERROR_CLUSTER_NO_RPC_PACKAGES_REGISTERED = 5081
ERROR_CLUSTER_NO_SECURITY_CONTEXT = 5059
ERROR_CLUSTER_NULL_DATA = 5920
ERROR_CLUSTER_OLD_VERSION = 5904
ERROR_CLUSTER_OWNER_NOT_IN_PREFLIST = 5082
ERROR_CLUSTER_PARAMETER_MISMATCH = 5897
ERROR_CLUSTER_PARAMETER_OUT_OF_BOUNDS = 5913
ERROR_CLUSTER_PARTIAL_READ = 5921
ERROR_CLUSTER_PARTIAL_SEND = 5914
ERROR_CLUSTER_PARTIAL_WRITE = 5922
ERROR_CLUSTER_POISONED = 5907
ERROR_CLUSTER_PROPERTY_DATA_TYPE_MISMATCH = 5895
ERROR_CLUSTER_QUORUMLOG_NOT_FOUND = 5891
ERROR_CLUSTER_REGISTRY_INVALID_FUNCTION = 5915
ERROR_CLUSTER_RESNAME_NOT_FOUND = 5080
ERROR_CLUSTER_RESOURCES_MUST_BE_ONLINE_ON_THE_SAME_NODE = 5933
ERROR_CLUSTER_RESOURCE_TYPE_BUSY = 5909
ERROR_CLUSTER_RESOURCE_TYPE_NOT_FOUND = 5078
ERROR_CLUSTER_RESTYPE_NOT_SUPPORTED = 5079
ERROR_CLUSTER_RHS_FAILED_INITIALIZATION = 5931
ERROR_CLUSTER_SHUTTING_DOWN = 5022
ERROR_CLUSTER_SYSTEM_CONFIG_CHANGED = 5077
```



```
ERROR_CLUSTER_WRONG_OS_VERSION = 5899
ERROR_COLORSPACE_MISMATCH = 2021
ERROR_COMMITMENT_LIMIT = 1455
ERROR_COMMITMENT_MINIMUM = 635
ERROR_COMPRESSION_DISABLED = 769
ERROR_COMPRESSION_NOT_ALLOWED_IN_TRANSACTION = 6850
ERROR_CONNECTED_OTHER_PASSWORD = 2108
ERROR_CONNECTED_OTHER_PASSWORD_DEFAULT = 2109
ERROR_CONNECTION_ABORTED = 1236
ERROR_CONNECTION_ACTIVE = 1230
ERROR_CONNECTION_COUNT_LIMIT = 1238
ERROR_CONNECTION_INVALID = 1229
ERROR_CONNECTION_REFUSED = 1225
ERROR_CONNECTION_UNAVAIL = 1201
ERROR_CONTEXT_EXPIRED = 1931
ERROR_CONTINUE = 1246
ERROR_CONTROLLING_IEPORT = 4329
ERROR_CONTROL_C_EXIT = 572
ERROR_CONTROL_ID_NOT_FOUND = 1421
ERROR_CONVERT_TO_LARGE = 600
ERROR_CORE_DRIVER_PACKAGE_NOT_FOUND = 3016
ERROR_CORE_RESOURCE = 5026
ERROR_CORRUPT_SYSTEM_FILE = 634
ERROR_COULD_NOT_INTERPRET = 552
ERROR_COULD_NOT_RESIZE_LOG = 6629
ERROR_COUNTER_TIMEOUT = 1121
ERROR_CRASH_DUMP = 753
ERROR_CRC = 23
ERROR_CREATE_FAILED = 1631
ERROR_CRM_PROTOCOL_ALREADY_EXISTS = 6710
ERROR_CRM_PROTOCOL_NOT_FOUND = 6712
ERROR_CS_ENCRYPTION_EXISTING_ENCRYPTED_FILE = 6019
ERROR_CS_ENCRYPTION_FILE_NOT_CSE = 6021
ERROR_CS_ENCRYPTION_INVALID_SERVER_RESPONSE = 6017
ERROR_CS_ENCRYPTION_NEW_ENCRYPTED_FILE = 6020
ERROR_CS_ENCRYPTION_UNSUPPORTED_SERVER = 6018
```

```
ERROR_CTX_ACCOUNT_RESTRICTION = 7064
ERROR_CTX_BAD_VIDEO_MODE = 7025
ERROR_CTX_CANNOT_MAKE_EVENTLOG_ENTRY = 7005
ERROR_CTX_CDM_CONNECT = 7066
ERROR_CTX_CDM_DISCONNECT = 7067
ERROR_CTX_CLIENT_LICENSE_IN_USE = 7052
ERROR_CTX_CLIENT_LICENSE_NOT_SET = 7053
ERROR_CTX_CLIENT_QUERY_TIMEOUT = 7040
ERROR_CTX_CLOSE_PENDING = 7007
ERROR_CTX_CONSOLE_CONNECT = 7042
ERROR_CTX_CONSOLE_DISCONNECT = 7041
ERROR_CTX_ENCRYPTION_LEVEL_REQUIRED = 7061
ERROR_CTX_GRAPHICS_INVALID = 7035
ERROR_CTX_INVALID_MODEMNAME = 7010
ERROR_CTX_INVALID_PD = 7002
ERROR_CTX_INVALID_WD = 7049
ERROR_CTX_LICENSE_CLIENT_INVALID = 7055
ERROR_CTX_LICENSE_EXPIRED = 7056
ERROR_CTX_LICENSE_NOT_AVAILABLE = 7054
ERROR_CTX_LOGON_DISABLED = 7037
ERROR_CTX_MODEM_INF_NOT_FOUND = 7009
ERROR_CTX_MODEM_RESPONSE_BUSY = 7015
ERROR_CTX_MODEM_RESPONSE_ERROR = 7011
ERROR_CTX_MODEM_RESPONSE_NO_CARRIER = 7013
ERROR_CTX_MODEM_RESPONSE_NO_DIALTONE = 7014
ERROR_CTX_MODEM_RESPONSE_TIMEOUT = 7012
ERROR_CTX_MODEM_RESPONSE_VOICE = 7016
ERROR_CTX_NOT_CONSOLE = 7038
ERROR_CTX_NO_FORCE_LOGOFF = 7063
ERROR_CTX_NO_OUTBUF = 7008
ERROR_CTX_PD_NOT_FOUND = 7003
ERROR_CTX_SECURITY_LAYER_ERROR = 7068
ERROR_CTX_SERVICE_NAME_COLLISION = 7006
ERROR_CTX_SESSION_IN_USE = 7062
ERROR_CTX_SHADOW_DENIED = 7044
ERROR_CTX_SHADOW_DISABLED = 7051
```

ERROR_CTX_SHADOW_ENDED_BY_MODE_CHANGE = 7058
ERROR_CTX_SHADOW_INVALID = 7050
ERROR_CTX_SHADOW_NOT_RUNNING = 7057
ERROR_CTX_TD_ERROR = 7017
ERROR_CTX_WD_NOT_FOUND = 7004
ERROR_CTX_WINSTATIONS_DISABLED = 7060
ERROR_CTX_WINSTATION_ACCESS_DENIED = 7045
ERROR_CTX_WINSTATION_ALREADY_EXISTS = 7023
ERROR_CTX_WINSTATION_BUSY = 7024
ERROR_CTX_WINSTATION_NAME_INVALID = 7001
ERROR_CTX_WINSTATION_NOT_FOUND = 7022
ERROR_CURRENT_DIRECTORY = 16
ERROR_CURRENT_TRANSACTION_NOT_VALID = 6714
ERROR_DATABASE_BACKUP_CORRUPT = 5087
ERROR_DATABASE_DOES_NOT_EXIST = 1065
ERROR_DATABASE_FAILURE = 4313
ERROR_DATABASE_FULL = 4314
ERROR_DATATYPE_MISMATCH = 1629
ERROR_DATA_LOST_REPAIR = 6843
ERROR_DATA_NOT_ACCEPTED = 592
ERROR_DBG_COMMAND_EXCEPTION = 697
ERROR_DBG_CONTINUE = 767
ERROR_DBG_CONTROL_BREAK = 696
ERROR_DBG_CONTROL_C = 693
ERROR_DBG_EXCEPTION_HANDLED = 766
ERROR_DBG_EXCEPTION_NOT_HANDLED = 688
ERROR_DBG_PRINTEXCEPTION_C = 694
ERROR_DBG_REPLY_LATER = 689
ERROR_DBG_RIPEXCEPTION = 695
ERROR_DBG_TERMINATE_PROCESS = 692
ERROR_DBG_TERMINATE_THREAD = 691
ERROR_DBG_UNABLE_TO_PROVIDE_HANDLE = 690
ERROR_DC_NOT_FOUND = 1425
ERROR_DDE_FAIL = 1156
ERROR_DEBUG_ATTACH_FAILED = 590
ERROR_DECRYPTION_FAILED = 6001

```
ERROR_DELETE_PENDING = 303
ERROR_DELETING_ICM_XFORM = 2309
ERROR_DEPENDENCY_ALREADY_EXISTS = 5003
ERROR_DEPENDENCY_NOT_ALLOWED = 5069
ERROR_DEPENDENCY_NOT_FOUND = 5002
ERROR_DEPENDENCY_TREE_TOO_COMPLEX = 5929
ERROR_DEPENDENT_RESOURCE_EXISTS = 5001
ERROR_DEPENDENT_RESOURCE_PROPERTY_CONFLICT = 5924
ERROR_DEPENDENT_SERVICES_RUNNING = 1051
ERROR_DESTINATION_ELEMENT_FULL = 1161
ERROR_DESTROY_OBJECT_OF_OTHER_THREAD = 1435
ERROR_DEVICE_ALREADY_ATTACHED = 548
ERROR_DEVICE_ALREADY_REMEMBERED = 1202
ERROR_DEVICE_DOOR_OPEN = 1166
ERROR_DEVICE_ENUMERATION_ERROR = 648
ERROR_DEVICE_IN_USE = 2404
ERROR_DEVICE_NOT_AVAILABLE = 4319
ERROR_DEVICE_NOT_CONNECTED = 1167
ERROR_DEVICE_NOT_PARTITIONED = 1107
ERROR_DEVICE_REINITIALIZATION_NEEDED = 1164
ERROR_DEVICE_REMOVED = 1617
ERROR_DEVICE_REQUIRES_CLEANING = 1165
ERROR_DEV_NOT_EXIST = 55
ERROR_DHCP_ADDRESS_CONFLICT = 4100
ERROR_DIFFERENT_SERVICE_ACCOUNT = 1079
ERROR_DIRECTORY = 267
ERROR_DIRECTORY_NOT_RM = 6803
ERROR_DIRECT_ACCESS_HANDLE = 130
ERROR_DIR_EFS_DISALLOWED = 6010
ERROR_DIR_NOT_EMPTY = 145
ERROR_DIR_NOT_ROOT = 144
ERROR_DISCARDED = 157
ERROR_DISK_CHANGE = 107
ERROR_DISK_CORRUPT = 1393
ERROR_DISK_FULL = 112
ERROR_DISK_OPERATION_FAILED = 1127
```

ERROR_DISK_RECALIBRATE_FAILED = 1126
ERROR_DISK_REPAIR_DISABLED = 780
ERROR_DISK_RESET_FAILED = 1128
ERROR_DISK_TOO_FRAGMENTED = 302
ERROR_DLL_INIT_FAILED = 1114
ERROR_DLL_INIT_FAILED_LOGOFF = 624
ERROR_DLL_MIGHT_BE_INCOMPATIBLE = 687
ERROR_DLL_MIGHT_BE_INSECURE = 686
ERROR_DLL_NOT_FOUND = 1157
ERROR_DOMAIN_CONTROLLER_EXISTS = 1250
ERROR_DOMAIN_CONTROLLER_NOT_FOUND = 1908
ERROR_DOMAIN_CTRLR_CONFIG_ERROR = 581
ERROR_DOMAIN_EXISTS = 1356
ERROR_DOMAIN_LIMIT_EXCEEDED = 1357
ERROR_DOMAIN_TRUST_INCONSISTENT = 1810
ERROR_DRIVERS_LEAKING_LOCKED_PAGES = 729
ERROR_DRIVER_CANCEL_TIMEOUT = 594
ERROR_DRIVER_DATABASE_ERROR = 652
ERROR_DRIVER_FAILED_PRIOR_UNLOAD = 654
ERROR_DRIVER_FAILED_SLEEP = 633
ERROR_DRIVE_LOCKED = 108
ERROR_DRIVE_MEDIA_MISMATCH = 4303
ERROR_DS_ADD_REPLICA_INHIBITED = 8302
ERROR_DS_ADMIN_LIMIT_EXCEEDED = 8228
ERROR_DS_AFFECTS_MULTIPLE_DSAS = 8249
ERROR_DS_AG_CANT_HAVE_UNIVERSAL_MEMBER = 8578
ERROR_DS_ALIASIED_OBJ_MISSING = 8334
ERROR_DS_ALIAS_DEREF_PROBLEM = 8244
ERROR_DS_ALIAS_POINTS_TO_ALIAS = 8336
ERROR_DS_ALIAS_PROBLEM = 8241
ERROR_DS_ATTRIBUTE_OR_VALUE_EXISTS = 8205
ERROR_DS_ATTRIBUTE_OWNED_BY_SAM = 8346
ERROR_DS_ATTRIBUTE_TYPE_UNDEFINED = 8204
ERROR_DS_ATT_ALREADY_EXISTS = 8318
ERROR_DS_ATT_IS_NOT_ON_OBJ = 8310
ERROR_DS_ATT_NOT_DEF_FOR_CLASS = 8317

```
ERROR_DS_ATT_NOT_DEF_IN_SCHEMA = 8303
ERROR_DS_ATT_SCHEMA_REQ_ID = 8399
ERROR_DS_ATT_SCHEMA_REQ_SYNTAX = 8416
ERROR_DS_ATT_VAL_ALREADY_EXISTS = 8323
ERROR_DS_AUDIT_FAILURE = 8625
ERROR_DS_AUTHORIZATION_FAILED = 8599
ERROR_DS_AUTH_METHOD_NOT_SUPPORTED = 8231
ERROR_DS_AUTH_UNKNOWN = 8234
ERROR_DS_AUX_CLS_TEST_FAIL = 8389
ERROR_DS_BACKLINK_WITHOUT_LINK = 8482
ERROR_DS_BAD_ATT_SCHEMA_SYNTAX = 8400
ERROR_DS_BAD_HIERARCHY_FILE = 8425
ERROR_DS_BAD_INSTANCE_TYPE = 8313
ERROR_DS_BAD_NAME_SYNTAX = 8335
ERROR_DS_BAD_RDN_ATT_ID_SYNTAX = 8392
ERROR_DS_BUILD_HIERARCHY_TABLE_FAILED = 8426
ERROR_DS_BUSY = 8206
ERROR_DS_CANT_ACCESS_REMOTE_PART_OF_AD = 8585
ERROR_DS_CANT_ADD_ATT_VALUES = 8320
ERROR_DS_CANT_ADD_SYSTEM_ONLY = 8358
ERROR_DS_CANT_ADD_TO_GC = 8550
ERROR_DS_CANT_CACHE_ATT = 8401
ERROR_DS_CANT_CACHE_CLASS = 8402
ERROR_DS_CANT_CREATE_IN_NONDOMAIN_NC = 8553
ERROR_DS_CANT_CREATE_UNDER_SCHEMA = 8510
ERROR_DS_CANT_DELETE = 8398
ERROR_DS_CANT_DELETE_DSA_OBJ = 8340
ERROR_DS_CANT_DEL_MASTER_CROSSREF = 8375
ERROR_DS_CANT_DEMOTE_WITH_WRITEABLE_NC = 8604
ERROR_DS_CANT_DEREF_ALIAS = 8337
ERROR_DS_CANT_DERIVE_SPN_FOR_DELETED_DOMAIN = 8603
ERROR_DS_CANT_DERIVE_SPN_WITHOUT_SERVER_REF = 8589
ERROR_DS_CANT_FIND_DC_FOR_SRC_DOMAIN = 8537
ERROR_DS_CANT_FIND_DSA_OBJ = 8419
ERROR_DS_CANT_FIND_EXPECTED_NC = 8420
ERROR_DS_CANT_FIND_NC_IN_CACHE = 8421
```

`ERROR_DS_CANT_MIX_MASTER_AND_REPS = 8331`
`ERROR_DS_CANT_MOD_OBJ_CLASS = 8215`
`ERROR_DS_CANT_MOD_PRIMARYGROUPID = 8506`
`ERROR_DS_CANT_MOD_SYSTEM_ONLY = 8369`
`ERROR_DS_CANT_MOVE_ACCOUNT_GROUP = 8498`
`ERROR_DS_CANT_MOVE_APP_BASIC_GROUP = 8608`
`ERROR_DS_CANT_MOVE_APP_QUERY_GROUP = 8609`
`ERROR_DS_CANT_MOVE_DELETED_OBJECT = 8489`
`ERROR_DS_CANT_MOVE_RESOURCE_GROUP = 8499`
`ERROR_DS_CANT_ON_NON_LEAF = 8213`
`ERROR_DS_CANT_ON_RDN = 8214`
`ERROR_DS_CANT_REMOVE_ATT_CACHE = 8403`
`ERROR_DS_CANT_REMOVE_CLASS_CACHE = 8404`
`ERROR_DS_CANT_REM_MISSING_ATT = 8324`
`ERROR_DS_CANT_REM_MISSING_ATT_VAL = 8325`
`ERROR_DS_CANT_REPLACE_HIDDEN_REC = 8424`
`ERROR_DS_CANT_RETRIEVE_ATTS = 8481`
`ERROR_DS_CANT_RETRIEVE_CHILD = 8422`
`ERROR_DS_CANT_RETRIEVE_DN = 8405`
`ERROR_DS_CANT_RETRIEVE_INSTANCE = 8407`
`ERROR_DS_CANT_RETRIEVE_SD = 8526`
`ERROR_DS_CANT_START = 8531`
`ERROR_DS_CANT_TREE_DELETE_CRITICAL_OBJ = 8560`
`ERROR_DS_CANT_WITH_ACCT_GROUP_MEMBERSHPS = 8493`
`ERROR_DS_CHILDREN_EXIST = 8332`
`ERROR_DS_CLASS_MUST_BE_CONCRETE = 8359`
`ERROR_DS_CLASS_NOT_DSA = 8343`
`ERROR_DS_CLIENT_LOOP = 8259`
`ERROR_DS_CODE_INCONSISTENCY = 8408`
`ERROR_DS_COMPARE_FALSE = 8229`
`ERROR_DS_COMPARE_TRUE = 8230`
`ERROR_DS_CONFIDENTIALITY_REQUIRED = 8237`
`ERROR_DS_CONFIG_PARAM_MISSING = 8427`
`ERROR_DS_CONSTRAINT_VIOLATION = 8239`
`ERROR_DS_CONSTRUCTED_ATT_MOD = 8475`
`ERROR_DS_CONTROL_NOT_FOUND = 8258`

ERROR_DS_COULDNT_CONTACT_FSMO = 8367
ERROR_DS_COULDNT_IDENTIFY_OBJECTS_FOR_TREE_DELETE = 8503
ERROR_DS_COULDNT_LOCK_TREE_FOR_DELETE = 8502
ERROR_DS_COULDNT_UPDATE_SPNS = 8525
ERROR_DS_COUNTING_AB_INDICES_FAILED = 8428
ERROR_DS_CROSS_DOMAIN_CLEANUP_REQD = 8491
ERROR_DS_CROSS_DOM_MOVE_ERROR = 8216
ERROR_DS_CROSS_NC_DN_RENAME = 8368
ERROR_DS_CROSS_REF_BUSY = 8602
ERROR_DS_CROSS_REF_EXISTS = 8374
ERROR_DS_CR_IMPOSSIBLE_TO_VALIDATE = 8495
ERROR_DS_CR_IMPOSSIBLE_TO_VALIDATE_V2 = 8586
ERROR_DS_DATABASE_ERROR = 8409
ERROR_DS_DECODING_ERROR = 8253
ERROR_DS_DESTINATION_AUDITING_NOT_ENABLED = 8536
ERROR_DS_DESTINATION_DOMAIN_NOT_IN_FOREST = 8535
ERROR_DS_DIFFERENT_REPL_EPOCHS = 8593
ERROR_DS_DISALLOWED_IN_SYSTEM_CONTAINER = 8615
ERROR_DS_DNS_LOOKUP_FAILURE = 8524
ERROR_DS_DOMAIN_RENAME_IN_PROGRESS = 8612
ERROR_DS_DOMAIN_VERSION_TOO_HIGH = 8564
ERROR_DS_DOMAIN_VERSION_TOO_LOW = 8566
ERROR_DS_DRA_ABANDON_SYNC = 8462
ERROR_DS_DRA_ACCESS_DENIED = 8453
ERROR_DS_DRA_BAD_DN = 8439
ERROR_DS_DRA_BAD_INSTANCE_TYPE = 8445
ERROR_DS_DRA_BAD_NC = 8440
ERROR_DS_DRA_BUSY = 8438
ERROR_DS_DRA_CONNECTION_FAILED = 8444
ERROR_DS_DRA_DB_ERROR = 8451
ERROR_DS_DRA_DN_EXISTS = 8441
ERROR_DS_DRA_EARLIER_SCHEMA_CONFLICT = 8544
ERROR_DS_DRA_EXTN_CONNECTION_FAILED = 8466
ERROR_DS_DRA_GENERIC = 8436
ERROR_DS_DRA_INCOMPATIBLE_PARTIAL_SET = 8464
ERROR_DS_DRA_INCONSISTENT_DIT = 8443

ERROR_DS_DRA_INTERNAL_ERROR = 8442
ERROR_DS_DRA_INVALID_PARAMETER = 8437
ERROR_DS_DRA_MAIL_PROBLEM = 8447
ERROR_DS_DRA_MISSING_PARENT = 8460
ERROR_DS_DRA_NAME_COLLISION = 8458
ERROR_DS_DRA_NOT_SUPPORTED = 8454
ERROR_DS_DRA_NO_REPLICA = 8452
ERROR_DS_DRA_OBJ_IS_REP_SOURCE = 8450
ERROR_DS_DRA_OBJ_NC_MISMATCH = 8545
ERROR_DS_DRA_OUT_OF_MEM = 8446
ERROR_DS_DRA_OUT_SCHEDULE_WINDOW = 8617
ERROR_DS_DRA_PREEMPTED = 8461
ERROR_DS_DRA_REF_ALREADY_EXISTS = 8448
ERROR_DS_DRA_REF_NOT_FOUND = 8449
ERROR_DS_DRA_REPL_PENDING = 8477
ERROR_DS_DRA_RPC_CANCELLED = 8455
ERROR_DS_DRA_SCHEMA_CONFLICT = 8543
ERROR_DS_DRA_SCHEMA_INFO_SHIP = 8542
ERROR_DS_DRA_SCHEMA_MISMATCH = 8418
ERROR_DS_DRA_SHUTDOWN = 8463
ERROR_DS_DRA_SINK_DISABLED = 8457
ERROR_DS_DRA_SOURCE_DISABLED = 8456
ERROR_DS_DRA_SOURCE_IS_PARTIAL_REPLICA = 8465
ERROR_DS_DRA_SOURCE_REINSTALLED = 8459
ERROR_DS_DRS_EXTENSIONS_CHANGED = 8594
ERROR_DS_DSA_MUST_BE_INT_MASTER = 8342
ERROR_DS_DST_DOMAIN_NOT_NATIVE = 8496
ERROR_DS_DST_NC_MISMATCH = 8486
ERROR_DS_DS_REQUIRED = 8478
ERROR_DS_DUPLICATE_ID_FOUND = 8605
ERROR_DS_DUP_LDAP_DISPLAY_NAME = 8382
ERROR_DS_DUP_LINK_ID = 8468
ERROR_DS_DUP_MAPI_ID = 8380
ERROR_DS_DUP_MSDS_INTID = 8597
ERROR_DS_DUP_OID = 8379
ERROR_DS_DUP_RDN = 8378

```
ERROR_DS_DUP_SCHEMA_ID_GUID = 8381
ERROR_DS_ENCODING_ERROR = 8252
ERROR_DS_EPOCH_MISMATCH = 8483
ERROR_DS_EXISTING_AD_CHILD_NC = 8613
ERROR_DS_EXISTS_IN_AUX_CLS = 8393
ERROR_DS_EXISTS_IN_MAY_HAVE = 8386
ERROR_DS_EXISTS_IN_MUST_HAVE = 8385
ERROR_DS_EXISTS_IN_POSS_SUP = 8395
ERROR_DS_EXISTS_IN_RDNATTID = 8598
ERROR_DS_EXISTS_IN_SUB_CLS = 8394
ERROR_DS_FILTER_UNKNOWN = 8254
ERROR_DS_FILTER_USES_CONSTRUCTED_ATTRS = 8555
ERROR_DS_FOREST_VERSION_TOO_HIGH = 8563
ERROR_DS_FOREST_VERSION_TOO_LOW = 8565
ERROR_DS_GCVERIFY_ERROR = 8417
ERROR_DS_GC_NOT_AVAILABLE = 8217
ERROR_DS_GC_REQUIRED = 8547
ERROR_DS_GENERIC_ERROR = 8341
ERROR_DS_GLOBAL_CANT_HAVE_CROSSDOMAIN_MEMBER = 8519
ERROR_DS_GLOBAL_CANT_HAVE_LOCAL_MEMBER = 8516
ERROR_DS_GLOBAL_CANT_HAVE_UNIVERSAL_MEMBER = 8517
ERROR_DS_GOVERNSID_MISSING = 8410
ERROR_DS_GROUP_CONVERSION_ERROR = 8607
ERROR_DS_HAVE_PRIMARY_MEMBERS = 8521
ERROR_DS_HIERARCHY_TABLE_MALLOC_FAILED = 8429
ERROR_DS_HIERARCHY_TABLE_TOO_DEEP = 8628
ERROR_DS_ILLEGAL_BASE_SCHEMA_MOD = 8507
ERROR_DS_ILLEGAL_MOD_OPERATION = 8311
ERROR_DS_ILLEGAL_SUPERIOR = 8345
ERROR_DS_ILLEGAL_XDOM_MOVE_OPERATION = 8492
ERROR_DS_INAPPROPRIATE_AUTH = 8233
ERROR_DS_INAPPROPRIATE_MATCHING = 8238
ERROR_DS_INCOMPATIBLE_CONTROLS_USED = 8574
ERROR_DS_INCOMPATIBLE_VERSION = 8567
ERROR_DS_INCORRECT_ROLE_OWNER = 8210
ERROR_DS_INIT_FAILURE = 8532
```

`ERROR_DS_INIT_FAILURE_CONSOLE = 8561`
`ERROR_DS_INSTALL_NO_SCH_VERSION_IN_INIFILE = 8512`
`ERROR_DS_INSTALL_NO_SRC_SCH_VERSION = 8511`
`ERROR_DS_INSTALL_SCHEMA_MISMATCH = 8467`
`ERROR_DS_INSUFFICIENT_ATTR_TO_CREATE_OBJECT = 8606`
`ERROR_DS_INSUFF_ACCESS_RIGHTS = 8344`
`ERROR_DS_INTERNAL_FAILURE = 8430`
`ERROR_DS_INVALID_ATTRIBUTE_SYNTAX = 8203`
`ERROR_DS_INVALID_DMD = 8360`
`ERROR_DS_INVALID_DN_SYNTAX = 8242`
`ERROR_DS_INVALID_GROUP_TYPE = 8513`
`ERROR_DS_INVALID_LDAP_DISPLAY_NAME = 8479`
`ERROR_DS_INVALID_NAME_FOR_SPN = 8554`
`ERROR_DS_INVALID_ROLE_OWNER = 8366`
`ERROR_DS_INVALID_SCRIPT = 8600`
`ERROR_DS_INVALID_SEARCH_FLAG = 8500`
`ERROR_DS_INVALID_SEARCH_FLAG_SUBTREE = 8626`
`ERROR_DS_INVALID_SEARCH_FLAG_TUPLE = 8627`
`ERROR_DS_IS_LEAF = 8243`
`ERROR_DS_KEY_NOT_UNIQUE = 8527`
`ERROR_DS_LDAP_SEND_QUEUE_FULL = 8616`
`ERROR_DS_LINK_ID_NOT_AVAILABLE = 8577`
`ERROR_DS_LOCAL_CANT_HAVE_CROSSDOMAIN_LOCAL_MEMBER = 8520`
`ERROR_DS_LOCAL_ERROR = 8251`
`ERROR_DS_LOCAL_MEMBER_OF_LOCAL_ONLY = 8548`
`ERROR_DS_LOOP_DETECT = 8246`
`ERROR_DS_LOW_DSA_VERSION = 8568`
`ERROR_DS_MACHINE_ACCOUNT_CREATED_PRENT4 = 8572`
`ERROR_DS_MACHINE_ACCOUNT_QUOTA_EXCEEDED = 8557`
`ERROR_DS_MASTERDSA_REQUIRED = 8314`
`ERROR_DS_MAX_OBJ_SIZE_EXCEEDED = 8304`
`ERROR_DS_MEMBERSHIP_EVALUATED_LOCALLY = 8201`
`ERROR_DS_MISSING_EXPECTED_ATT = 8411`
`ERROR_DS_MISSING_FSMO_SETTINGS = 8434`
`ERROR_DS_MISSING_INFRASTRUCTURE_CONTAINER = 8497`
`ERROR_DS_MISSING_REQUIRED_ATT = 8316`

```
ERROR_DS_MISSING_SUPREF = 8406
ERROR_DS_MODIFYDN_DISALLOWED_BY_FLAG = 8581
ERROR_DS_MODIFYDN_DISALLOWED_BY_INSTANCE_TYPE = 8579
ERROR_DS_MODIFYDN_WRONG_GRANDPARENT = 8582
ERROR_DS_MUST_BE_RUN_ON_DST_DC = 8558
ERROR_DS_NAME_ERROR_DOMAIN_ONLY = 8473
ERROR_DS_NAME_ERROR_NOT_FOUND = 8470
ERROR_DS_NAME_ERROR_NOT_UNIQUE = 8471
ERROR_DS_NAME_ERROR_NO_MAPPING = 8472
ERROR_DS_NAME_ERROR_NO_SYNTACTICAL_MAPPING = 8474
ERROR_DS_NAME_ERROR_RESOLVING = 8469
ERROR_DS_NAME_ERROR_TRUST_REFERRAL = 8583
ERROR_DS_NAME_NOT_UNIQUE = 8571
ERROR_DS_NAME_REFERENCE_INVALID = 8373
ERROR_DS_NAME_TOO_LONG = 8348
ERROR_DS_NAME_TOO_MANY_PARTS = 8347
ERROR_DS_NAME_TYPE_UNKNOWN = 8351
ERROR_DS_NAME_UNPARSEABLE = 8350
ERROR_DS_NAME_VALUE_TOO_LONG = 8349
ERROR_DS_NAMING_MASTER_GC = 8523
ERROR_DS_NAMING_VIOLATION = 8247
ERROR_DS_NCNAME_MISSING_CR_REF = 8412
ERROR_DS_NCNAME_MUST_BE_NC = 8357
ERROR_DS_NC_MUST_HAVE_NC_PARENT = 8494
ERROR_DS_NC_STILL_HAS_DSAS = 8546
ERROR_DS_NONEXISTENT_MAY_HAVE = 8387
ERROR_DS_NONEXISTENT_MUST_HAVE = 8388
ERROR_DS_NONEXISTENT_POSS_SUP = 8390
ERROR_DS_NONSAFE_SCHEMA_CHANGE = 8508
ERROR_DS_NON_ASQ_SEARCH = 8624
ERROR_DS_NON_BASE_SEARCH = 8480
ERROR_DS_NOTIFY_FILTER_TOO_COMPLEX = 8377
ERROR_DS_NOT_AN_OBJECT = 8352
ERROR_DS_NOT_AUTHORITY_FOR_DST_NC = 8487
ERROR_DS_NOT_CLOSEST = 8588
ERROR_DS_NOT_INSTALLED = 8200
```

ERROR_DS_NOT_ON_BACKLINK = 8362
ERROR_DS_NOT_SUPPORTED = 8256
ERROR_DS_NOT_SUPPORTED_SORT_ORDER = 8570
ERROR_DS_NO_ATTRIBUTE_OR_VALUE = 8202
ERROR_DS_NO_BEHAVIOR_VERSION_IN_MIXEDDOMAIN = 8569
ERROR_DS_NO_CHAINED_EVAL = 8328
ERROR_DS_NO_CHAINING = 8327
ERROR_DS_NO_CHECKPOINT_WITH_PDC = 8551
ERROR_DS_NO_CROSSREF_FOR_NC = 8363
ERROR_DS_NO_DELETED_NAME = 8355
ERROR_DS_NO_FPO_IN_UNIVERSAL_GROUPS = 8549
ERROR_DS_NO_MORE_RIDS = 8209
ERROR_DS_NO_MSDS_INTID = 8596
ERROR_DS_NO_NEST_GLOBALGROUP_IN_MIXEDDOMAIN = 8514
ERROR_DS_NO_NEST_LOCALGROUP_IN_MIXEDDOMAIN = 8515
ERROR_DS_NO_NTDSA_OBJECT = 8623
ERROR_DS_NO_OBJECT_MOVE_IN_SCHEMA_NC = 8580
ERROR_DS_NO_PARENT_OBJECT = 8329
ERROR_DS_NO_PKT_PRIVACY_ON_CONNECTION = 8533
ERROR_DS_NO_RDN_DEFINED_IN_SCHEMA = 8306
ERROR_DS_NO_REF_DOMAIN = 8575
ERROR_DS_NO_REQUESTED_ATTS_FOUND = 8308
ERROR_DS_NO_RESULTS_RETURNED = 8257
ERROR_DS_NO_RIDS_ALLOCATED = 8208
ERROR_DS_NO_SERVER_OBJECT = 8622
ERROR_DS_NO_SUCH_OBJECT = 8240
ERROR_DS_NO_TREE_DELETE_ABOVE_NC = 8501
ERROR_DS_NTDSRIPT_PROCESS_ERROR = 8592
ERROR_DS_NTDSRIPT_SYNTAX_ERROR = 8591
ERROR_DS_OBJECT_BEING_REMOVED = 8339
ERROR_DS_OBJECT_CLASS_REQUIRED = 8315
ERROR_DS_OBJECT_RESULTS_TOO_LARGE = 8248
ERROR_DS_OBJ_CLASS_NOT_DEFINED = 8371
ERROR_DS_OBJ_CLASS_NOT_SUBCLASS = 8372
ERROR_DS_OBJ_CLASS_VIOLATION = 8212
ERROR_DS_OBJ_GUID_EXISTS = 8361

```
ERROR_DS_OBJ_NOT_FOUND = 8333
ERROR_DS_OBJ_STRING_NAME_EXISTS = 8305
ERROR_DS_OBJ_TOO_LARGE = 8312
ERROR_DS_OFFSET_RANGE_ERROR = 8262
ERROR_DS_OPERATIONS_ERROR = 8224
ERROR_DS_OUT_OF_SCOPE = 8338
ERROR_DS_OUT_OF_VERSION_STORE = 8573
ERROR_DS_PARAM_ERROR = 8255
ERROR_DS_PARENT_IS_AN_ALIAS = 8330
ERROR_DS_PDC_OPERATION_IN_PROGRESS = 8490
ERROR_DS_POLICY_NOT_KNOWN = 8618
ERROR_DS_PROTOCOL_ERROR = 8225
ERROR_DS_RANGE_CONSTRAINT = 8322
ERROR_DS_RDN_DOESNT_MATCH_SCHEMA = 8307
ERROR_DS_RECALCSHEMA_FAILED = 8396
ERROR_DS_REFERRAL = 8235
ERROR_DS_REFERRAL_LIMIT_EXCEEDED = 8260
ERROR_DS_REFUSING_FSMO_ROLES = 8433
ERROR_DS_REMOTE_CROSSREF_OP_FAILED = 8601
ERROR_DS_REPLICATOR_ONLY = 8370
ERROR_DS_REPLICA_SET_CHANGE_NOT_ALLOWED_ON_DISABLED_CR = 8595
ERROR_DS_REPL_LIFETIME_EXCEEDED = 8614
ERROR_DS_RESERVED_LINK_ID = 8576
ERROR_DS RIDMGR_INIT_ERROR = 8211
ERROR_DS_ROLE_NOT_VERIFIED = 8610
ERROR_DS_ROOT_CANT_BE_SUBREF = 8326
ERROR_DS_ROOT_MUST_BE_NC = 8301
ERROR_DS_ROOT_REQUIRES_CLASS_TOP = 8432
ERROR_DS_SAM_INIT_FAILURE = 8504
ERROR_DS_SAM_INIT_FAILURE_CONSOLE = 8562
ERROR_DS_SAM_NEED_BOOTKEY_FLOPPY = 8530
ERROR_DS_SAM_NEED_BOOTKEY_PASSWORD = 8529
ERROR_DS_SCHEMA_ALLOC_FAILED = 8415
ERROR_DS_SCHEMA_NOT_LOADED = 8414
ERROR_DS_SCHEMA_UPDATE_DISALLOWED = 8509
ERROR_DS_SECURITY_CHECKING_ERROR = 8413
```

`ERROR_DS_SECURITY_ILLEGAL_MODIFY = 8423`
`ERROR_DS_SEC_DESC_INVALID = 8354`
`ERROR_DS_SEC_DESC_TOO_SHORT = 8353`
`ERROR_DS_SEMANTIC_ATT_TEST = 8383`
`ERROR_DS_SENSITIVE_GROUP_VIOLATION = 8505`
`ERROR_DS_SERVER_DOWN = 8250`
`ERROR_DS_SHUTTING_DOWN = 8364`
`ERROR_DS_SINGLE_USER_MODE_FAILED = 8590`
`ERROR_DS_SINGLE_VALUE_CONSTRAINT = 8321`
`ERROR_DS_SIZELIMIT_EXCEEDED = 8227`
`ERROR_DS_SORT_CONTROL_MISSING = 8261`
`ERROR_DS_SOURCE_AUDITING_NOT_ENABLED = 8552`
`ERROR_DS_SOURCE_DOMAIN_IN_FOREST = 8534`
`ERROR_DS_SRC_AND_DST_NC_IDENTICAL = 8485`
`ERROR_DS_SRC_AND_DST_OBJECT_CLASS_MISMATCH = 8540`
`ERROR_DS_SRC_DC_MUST_BE_SP4_OR_GREATER = 8559`
`ERROR_DS_SRC_GUID_MISMATCH = 8488`
`ERROR_DS_SRC_NAME_MISMATCH = 8484`
`ERROR_DS_SRC_OBJ_NOT_GROUP_OR_USER = 8538`
`ERROR_DS_SRC_SID_EXISTS_IN_FOREST = 8539`
`ERROR_DS_STRING_SD_CONVERSION_FAILED = 8522`
`ERROR_DS_STRONG_AUTH_REQUIRED = 8232`
`ERROR_DS_SUBREF_MUST_HAVE_PARENT = 8356`
`ERROR_DS_SUBTREE_NOTIFY_NOT_NC_HEAD = 8376`
`ERROR_DS_SUB_CLS_TEST_FAIL = 8391`
`ERROR_DS_SYNTAX_MISMATCH = 8384`
`ERROR_DS_THREAD_LIMIT_EXCEEDED = 8587`
`ERROR_DS_TIMELIMIT_EXCEEDED = 8226`
`ERROR_DS_TREE_DELETE_NOT_FINISHED = 8397`
`ERROR_DS_UNABLE_TO_SURRENDER_ROLES = 8435`
`ERROR_DS_UNAVAILABLE = 8207`
`ERROR_DS_UNAVAILABLE_CRIT_EXTENSION = 8236`
`ERROR_DS_UNICODEPWD_NOT_IN_QUOTES = 8556`
`ERROR_DS_UNIVERSAL_CANT_HAVE_LOCAL_MEMBER = 8518`
`ERROR_DS_UNKNOWN_ERROR = 8431`
`ERROR_DS_UNKNOWN_OPERATION = 8365`

`ERROR_DS_UNWILLING_TO_PERFORM = 8245`
`ERROR_DS_USER_BUFFER_TOO_SMALL = 8309`
`ERROR_DS_VERSION_CHECK_FAILURE = 643`
`ERROR_DS_WKO_CONTAINER_CANNOT_BE_SPECIAL = 8611`
`ERROR_DS_WRONG_LINKED_ATT_SYNTAX = 8528`
`ERROR_DS_WRONG_OM_OBJ_CLASS = 8476`
`ERROR_DUPLICATE_SERVICE_NAME = 1078`
`ERROR_DUPLICATE_TAG = 2304`
`ERROR_DUP_DOMAINNAME = 1221`
`ERROR_DUP_NAME = 52`
`ERROR_DYNLINK_FROM_INVALID_RING = 196`
`ERROR_EAS_DIDNT_FIT = 275`
`ERROR_EAS_NOT_SUPPORTED = 282`
`ERROR_EA_ACCESS_DENIED = 994`
`ERROR_EA_FILE_CORRUPT = 276`
`ERROR_EA_LIST_INCONSISTENT = 255`
`ERROR_EA_TABLE_FULL = 277`
`ERROR_EFS_ALG_BLOB_TOO_BIG = 6013`
`ERROR_EFS_DISABLED = 6015`
`ERROR_EFS_NOT_ALLOWED_IN_TRANSACTION = 6831`
`ERROR_EFS_SERVER_NOT_TRUSTED = 6011`
`ERROR_EFS_VERSION_NOT_SUPPORT = 6016`
`ERROR_ELEVATION_REQUIRED = 740`
`ERROR_EMPTY = 4306`
`ERROR_ENCRYPTION_FAILED = 6000`
`ERROR_END_OF_MEDIA = 1100`
`ERROR_ENLISTMENT_NOT_FOUND = 6717`
`ERROR_ENLISTMENT_NOT_SUPERIOR = 6820`
`ERROR_ENVVAR_NOT_FOUND = 203`
`ERROR_EOM_OVERFLOW = 1129`
`ERROR_ERRORS_ENCOUNTERED = 774`
`ERROR_EVALUATION_EXPIRATION = 622`
`ERROR_EVENTLOG_CANT_START = 1501`
`ERROR_EVENTLOG_FILE_CHANGED = 1503`
`ERROR_EVENTLOG_FILE_CORRUPT = 1500`
`ERROR_EVENT_DONE = 710`


```
ERROR_EVENT_PENDING = 711
ERROR_EXCEPTION_IN_RESOURCE_CALL = 5930
ERROR_EXCEPTION_IN_SERVICE = 1064
ERROR_EXCL_SEM_ALREADY_OWNED = 101
ERROR_EXE_CANNOT_MODIFY_SIGNED_BINARY = 217
ERROR_EXE_CANNOT_MODIFY_STRONG_SIGNED_BINARY = 218
ERROR_EXE_MACHINE_TYPE_MISMATCH = 216
ERROR_EXE_MARKED_INVALID = 192
ERROR_EXTENDED_ERROR = 1208
ERROR_EXTRANEIOUS_INFORMATION = 677
ERROR_FAILED_DRIVER_ENTRY = 647
ERROR_FAILED_SERVICE_CONTROLLER_CONNECT = 1063
ERROR_FAIL_I24 = 83
ERROR_FAIL_NOACTION_REBOOT = 350
ERROR_FAIL_REBOOT_INITIATED = 3018
ERROR_FAIL_REBOOT_REQUIRED = 3017
ERROR_FAIL_RESTART = 352
ERROR_FAIL_SHUTDOWN = 351
ERROR_FATAL_APP_EXIT = 713
ERROR_FILEMARK_DETECTED = 1101
ERROR_FILENAME_EXCED_RANGE = 206
ERROR_FILE_CHECKED_OUT = 220
ERROR_FILE_CORRUPT = 1392
ERROR_FILE_ENCRYPTED = 6002
ERROR_FILE_EXISTS = 80
ERROR_FILE_IDENTITY_NOT_PERSISTENT = 6823
ERROR_FILE_INVALID = 1006
ERROR_FILE_NOT_ENCRYPTED = 6007
ERROR_FILE_NOT_FOUND = 2
ERROR_FILE_OFFLINE = 4350
ERROR_FILE_READ_ONLY = 6009
ERROR_FILE_SYSTEM_LIMITATION = 665
ERROR_FILE_TOO_LARGE = 223
ERROR_FIRMWARE_UPDATED = 728
ERROR_FLOATED_SECTION = 6846
ERROR_FLOAT_MULTIPLE_FAULTS = 630
```

```
ERROR_FLOAT_MULTIPLE_TRAPS = 631
ERROR_FLOPPY_BAD_REGISTERS = 1125
ERROR_FLOPPY_ID_MARK_NOT_FOUND = 1122
ERROR_FLOPPY_UNKNOWN_ERROR = 1124
ERROR_FLOPPY_VOLUME = 584
ERROR_FLOPPY_WRONG_CYLINDER = 1123
ERROR_FORMS_AUTH_REQUIRED = 224
ERROR_FOUND_OUT_OF_SCOPE = 601
ERROR_FSFILTER_OP_COMPLETED_SUCCESSFULLY = 762
ERROR_FS_DRIVER_REQUIRED = 588
ERROR_FT_READ_RECOVERY_FROM_BACKUP = 704
ERROR_FT_WRITE_RECOVERY = 705
ERROR_FULLSCREEN_MODE = 1007
ERROR_FULL_BACKUP = 4004
ERROR_FUNCTION_FAILED = 1627
ERROR_FUNCTION_NOT_CALLED = 1626
ERROR_GENERIC_NOT_MAPPED = 1360
ERROR_GEN_FAILURE = 31
ERROR_GLOBAL_ONLY_HOOK = 1429
ERROR_GRACEFUL_DISCONNECT = 1226
ERROR_GROUP_EXISTS = 1318
ERROR_GROUP_NOT_AVAILABLE = 5012
ERROR_GROUP_NOT_FOUND = 5013
ERROR_GROUP_NOT_ONLINE = 5014
ERROR_GUID_SUBSTITUTION_MADE = 680
ERROR_HANDLES_CLOSED = 676
ERROR_HANDLE_DISK_FULL = 39
ERROR_HANDLE_EOF = 38
ERROR_HANDLE_NO_LONGER_VALID = 6815
ERROR_HIBERNATED = 726
ERROR_HIBERNATION_FAILURE = 656
ERROR_HOOK_NEEDS_HMOD = 1428
ERROR_HOOK_NOT_INSTALLED = 1431
ERROR_HOOK_TYPE_NOT_ALLOWED = 1458
ERROR_HOST_NODE_NOT_AVAILABLE = 5005
ERROR_HOST_NODE_NOT_GROUP_OWNER = 5016
```

```
ERROR_HOST_NODE_NOT_RESOURCE_OWNER = 5015
ERROR_HOST_UNREACHABLE = 1232
ERROR_HOTKEY_ALREADY_REGISTERED = 1409
ERROR_HOTKEY_NOT_REGISTERED = 1419
ERROR_HWNDS_HAVE_DIFF_PARENT = 1441
ERROR_ICM_NOT_ENABLED = 2308
ERROR_IEPORT_FULL = 4341
ERROR_ILLEGAL_CHARACTER = 582
ERROR_ILLEGAL_DLL_RELOCATION = 623
ERROR_ILLEGAL_ELEMENT_ADDRESS = 1162
ERROR_ILLEGAL_FLOAT_CONTEXT = 579
ERROR_ILL_FORMED_PASSWORD = 1324
ERROR_IMAGE_MACHINE_TYPE_MISMATCH = 706
ERROR_IMAGE_MACHINE_TYPE_MISMATCH_EXE = 720
ERROR_IMAGE_NOT_AT_BASE = 700
ERROR_IMPLICIT_TRANSACTION_NOT_SUPPORTED = 6725
ERROR_INCORRECT_ADDRESS = 1241
ERROR_INCORRECT_SIZE = 1462
ERROR_INC_BACKUP = 4003
ERROR_INDEX_ABSENT = 1611
ERROR_INDIGENOUS_TYPE = 4338
ERROR_INDOUBT_TRANSACTIONS_EXIST = 6827
ERROR_INFLOOP_IN_RELOC_CHAIN = 202
ERROR_INSTALL_ALREADY_RUNNING = 1618
ERROR_INSTALL_FAILURE = 1603
ERROR_INSTALL_LANGUAGE_UNSUPPORTED = 1623
ERROR_INSTALL_LOG_FAILURE = 1622
ERROR_INSTALL_NOTUSED = 1634
ERROR_INSTALL_PACKAGE_INVALID = 1620
ERROR_INSTALL_PACKAGE_OPEN_FAILED = 1619
ERROR_INSTALL_PACKAGE_REJECTED = 1625
ERROR_INSTALL_PLATFORM_UNSUPPORTED = 1633
ERROR_INSTALL_REMOTE_DISALLOWED = 1640
ERROR_INSTALL_REMOTE_PROHIBITED = 1645
ERROR_INSTALL_SERVICE = 1601
ERROR_INSTALL_SERVICE_SAFEBOOT = 1652
```

```
ERROR_INSTALL_SOURCE_ABSENT = 1612
ERROR_INSTALL_SUSPEND = 1604
ERROR_INSTALL_TEMP_UNWRITABLE = 1632
ERROR_INSTALL_TRANSFORM_FAILURE = 1624
ERROR_INSTALL_TRANSFORM_REJECTED = 1644
ERROR_INSTALL_UI_FAILURE = 1621
ERROR_INSTALL_USEREXIT = 1602
ERROR_INSTRUCTION_MISALIGNMENT = 549
ERROR_INSUFFICIENT_BUFFER = 122
ERROR_INSUFFICIENT_LOGON_INFO = 608
ERROR_INSUFFICIENT_POWER = 639
ERROR_INSUFFICIENT_RESOURCE_FOR_SPECIFIED_SHARED_SECTION_SIZE = 781
ERROR_INTERNAL_DB_CORRUPTION = 1358
ERROR_INTERNAL_DB_ERROR = 1383
ERROR_INTERNAL_ERROR = 1359
ERROR_INTERRUPT_STILL_CONNECTED = 764
ERROR_INTERRUPT_VECTOR_ALREADY_CONNECTED = 763
ERROR_INVALID_ACCEL_HANDLE = 1403
ERROR_INVALID_ACCESS = 12
ERROR_INVALID_ACCOUNT_NAME = 1315
ERROR_INVALID_ACL = 1336
ERROR_INVALID_ADDRESS = 487
ERROR_INVALID_AT_INTERRUPT_TIME = 104
ERROR_INVALID_BLOCK = 9
ERROR_INVALID_BLOCK_LENGTH = 1106
ERROR_INVALID_CATEGORY = 117
ERROR_INVALID_CLEANER = 4310
ERROR_INVALID_CLUSTER_IPV6_ADDRESS = 5911
ERROR_INVALID_CMM = 2300
ERROR_INVALID_COLORINDEX = 2022
ERROR_INVALID_COLORSPACE = 2307
ERROR_INVALID_COMBOBOX_MESSAGE = 1422
ERROR_INVALID_COMMAND_LINE = 1639
ERROR_INVALID_COMPUTERNAME = 1210
ERROR_INVALID_CURSOR_HANDLE = 1402
ERROR_INVALID_DATA = 13
```

```
ERROR_INVALID_DATATYPE = 1804
ERROR_INVALID_DEVICE_OBJECT_PARAMETER = 650
ERROR_INVALID_DLL = 1154
ERROR_INVALID_DOMAINNAME = 1212
ERROR_INVALID_DOMAIN_ROLE = 1354
ERROR_INVALID_DOMAIN_STATE = 1353
ERROR_INVALID_DRIVE = 15
ERROR_INVALID_DRIVE_OBJECT = 4321
ERROR_INVALID_DWP_HANDLE = 1405
ERROR_INVALID_EA_HANDLE = 278
ERROR_INVALID_EA_NAME = 254
ERROR_INVALID_EDIT_HEIGHT = 1424
ERROR_INVALID_ENVIRONMENT = 1805
ERROR_INVALID_EVENTNAME = 1211
ERROR_INVALID_EVENT_COUNT = 151
ERROR_INVALID_EXE_SIGNATURE = 191
ERROR_INVALID_FIELD = 1616
ERROR_INVALID_FILTER_PROC = 1427
ERROR_INVALID_FLAGS = 1004
ERROR_INVALID_FLAG_NUMBER = 186
ERROR_INVALID_FORM_NAME = 1902
ERROR_INVALID_FORM_SIZE = 1903
ERROR_INVALID_FUNCTION = 1
ERROR_INVALID_GROUPNAME = 1209
ERROR_INVALID_GROUP_ATTRIBUTES = 1345
ERROR_INVALID_GW_COMMAND = 1443
ERROR_INVALID_HANDLE = 6
ERROR_INVALID_HANDLE_STATE = 1609
ERROR_INVALID_HOOK_FILTER = 1426
ERROR_INVALID_HOOK_HANDLE = 1404
ERROR_INVALID_HW_PROFILE = 619
ERROR_INVALID_ICON_HANDLE = 1414
ERROR_INVALID_ID_AUTHORITY = 1343
ERROR_INVALID_IMAGE_HASH = 577
ERROR_INVALID_INDEX = 1413
ERROR_INVALID_KEYBOARD_HANDLE = 1457
```

```
ERROR_INVALID_LB_MESSAGE = 1432
ERROR_INVALID_LDT_DESCRIPTOR = 564
ERROR_INVALID_LDT_OFFSET = 563
ERROR_INVALID_LDT_SIZE = 561
ERROR_INVALID_LEVEL = 124
ERROR_INVALID_LIBRARY = 4301
ERROR_INVALID_LIST_FORMAT = 153
ERROR_INVALID_LOGON_HOURS = 1328
ERROR_INVALID_LOGON_TYPE = 1367
ERROR_INVALID_MEDIA = 4300
ERROR_INVALID_MEDIA_POOL = 4302
ERROR_INVALID_MEMBER = 1388
ERROR_INVALID_MENU_HANDLE = 1401
ERROR_INVALID_MESSAGE = 1002
ERROR_INVALID_MESSAGEDEST = 1218
ERROR_INVALID_MESSAGE_NAME = 1217
ERROR_INVALID_MINALLOC_SIZE = 195
ERROR_INVALID_MODULETYPE = 190
ERROR_INVALID_MONITOR_HANDLE = 1461
ERROR_INVALID_MSGBOX_STYLE = 1438
ERROR_INVALID_NAME = 123
ERROR_INVALID_NETNAME = 1214
ERROR_INVALID_OPERATION = 4317
ERROR_INVALID_OPERATION_ON_QUORUM = 5068
ERROR_INVALID_OPLOCK_PROTOCOL = 301
ERROR_INVALID_ORDINAL = 182
ERROR_INVALID_OWNER = 1307
ERROR_INVALID_PARAMETER = 87
ERROR_INVALID_PASSWORD = 86
ERROR_INVALID_PASSWORDNAME = 1216
ERROR_INVALID_PATCH_XML = 1650
ERROR_INVALID_PIXEL_FORMAT = 2000
ERROR_INVALID_PLUGPLAY_DEVICE_PATH = 620
ERROR_INVALID_PORT_ATTRIBUTES = 545
ERROR_INVALID_PRIMARY_GROUP = 1308
ERROR_INVALID_PRINTER_COMMAND = 1803
```

```
ERROR_INVALID_PRINTER_NAME = 1801
ERROR_INVALID_PRINTER_STATE = 1906
ERROR_INVALID_PRINT_MONITOR = 3007
ERROR_INVALID_PRIORITY = 1800
ERROR_INVALID_PROFILE = 2301
ERROR_INVALID_QUOTA_LOWER = 547
ERROR_INVALID_REPARSE_DATA = 4392
ERROR_INVALID_SCROLLBAR_RANGE = 1448
ERROR_INVALID_SECURITY_DESCR = 1338
ERROR_INVALID_SEGDPL = 198
ERROR_INVALID_SEGMENT_NUMBER = 180
ERROR_INVALID_SEPARATOR_FILE = 1799
ERROR_INVALID_SERVER_STATE = 1352
ERROR_INVALID_SERVICENAME = 1213
ERROR_INVALID_SERVICE_ACCOUNT = 1057
ERROR_INVALID_SERVICE_CONTROL = 1052
ERROR_INVALID_SERVICE_LOCK = 1071
ERROR_INVALID_SHARENAME = 1215
ERROR_INVALID_SHOWWIN_COMMAND = 1449
ERROR_INVALID_SID = 1337
ERROR_INVALID_SIGNAL_NUMBER = 209
ERROR_INVALID_SPI_VALUE = 1439
ERROR_INVALID_STACKSEG = 189
ERROR_INVALID_STARTING_CODESEG = 188
ERROR_INVALID_STATE = 5023
ERROR_INVALID_SUB_AUTHORITY = 1335
ERROR_INVALID_TABLE = 1628
ERROR_INVALID_TARGET_HANDLE = 114
ERROR_INVALID_THREAD_ID = 1444
ERROR_INVALID_TIME = 1901
ERROR_INVALID_TRANSACTION = 6700
ERROR_INVALID_TRANSFORM = 2310
ERROR_INVALID_UNWIND_TARGET = 544
ERROR_INVALID_USER_BUFFER = 1784
ERROR_INVALID_VARIANT = 604
ERROR_INVALID_VERIFY_SWITCH = 118
```

```
ERROR_INVALID_WINDOW_HANDLE = 1400
ERROR_INVALID_WINDOW_STYLE = 2002
ERROR_INVALID_WORKSTATION = 1329
ERROR_IOPL_NOT_ENABLED = 197
ERROR_IO_DEVICE = 1117
ERROR_IO_INCOMPLETE = 996
ERROR_IO_PENDING = 997
ERROR_IO_PRIVILEGE_FAILED = 571
ERROR_IO_REISSUE_AS_CACHED = 3950
ERROR_IP_ADDRESS_CONFLICT1 = 611
ERROR_IP_ADDRESS_CONFLICT2 = 612
ERROR_IRQ_BUSY = 1119
ERROR_IS_JOINED = 134
ERROR_IS_JOIN_PATH = 147
ERROR_IS_JOIN_TARGET = 133
ERROR_IS_SUBSTED = 135
ERROR_IS_SUBST_PATH = 146
ERROR_IS_SUBST_TARGET = 149
ERROR_ITERATED_DATA_EXCEEDS_64k = 194
ERROR_JOIN_TO_JOIN = 138
ERROR_JOIN_TO_SUBST = 140
ERROR_JOURNAL_HOOK_SET = 1430
ERROR_KERNEL_APC = 738
ERROR_KEY_DELETED = 1018
ERROR_KEY_HAS_CHILDREN = 1020
ERROR_KM_DRIVER_BLOCKED = 1930
ERROR_LABEL_TOO_LONG = 154
ERROR_LAST_ADMIN = 1322
ERROR_LB_WITHOUT_TABSTOPS = 1434
ERROR_LIBRARY_FULL = 4322
ERROR_LIBRARY_OFFLINE = 4305
ERROR_LICENSE_QUOTA_EXCEEDED = 1395
ERROR_LISTBOX_ID_NOT_FOUND = 1416
ERROR_LM_CROSS_ENCRYPTION_REQUIRED = 1390
ERROR_LOCAL_USER_SESSION_KEY = 1303
ERROR_LOCKED = 212
```



```
ERROR_LOCK_FAILED = 167
ERROR_LOCK_VIOLATION = 33
ERROR_LOGIN_TIME_RESTRICTION = 1239
ERROR_LOGIN_WKSTA_RESTRICTION = 1240
ERROR_LOGON_FAILURE = 1326
ERROR_LOGON_NOT_GRANTED = 1380
ERROR_LOGON_SERVER_CONFLICT = 568
ERROR_LOGON_SESSION_COLLISION = 1366
ERROR_LOGON_SESSION_EXISTS = 1363
ERROR_LOGON_TYPE_NOT_GRANTED = 1385
ERROR_LOG_APPENDED_FLUSH_FAILED = 6647
ERROR_LOG_ARCHIVE_IN_PROGRESS = 6633
ERROR_LOG_ARCHIVE_NOT_IN_PROGRESS = 6632
ERROR_LOG_BLOCKS_EXHAUSTED = 6605
ERROR_LOG_BLOCK_INCOMPLETE = 6603
ERROR_LOG_BLOCK_INVALID = 6609
ERROR_LOG_BLOCK_VERSION = 6608
ERROR_LOG_CANT_DELETE = 6616
ERROR_LOG_CLIENT_ALREADY_REGISTERED = 6636
ERROR_LOG_CLIENT_NOT_REGISTERED = 6637
ERROR_LOG_CONTAINER_LIMIT_EXCEEDED = 6617
ERROR_LOG_CONTAINER_OPEN_FAILED = 6641
ERROR_LOG_CONTAINER_READ_FAILED = 6639
ERROR_LOG_CONTAINER_STATE_INVALID = 6642
ERROR_LOG_CONTAINER_WRITE_FAILED = 6640
ERROR_LOG_CORRUPTION_DETECTED = 6817
ERROR_LOG_DEDICATED = 6631
ERROR_LOG_EPHEMERAL = 6634
ERROR_LOG_FILE_FULL = 1502
ERROR_LOG_FULL = 6628
ERROR_LOG_FULL_HANDLER_IN_PROGRESS = 6638
ERROR_LOG_GROWTH_FAILED = 6833
ERROR_LOG_HARD_ERROR = 718
ERROR_LOG_INCONSISTENT_SECURITY = 6646
ERROR_LOG_INVALID_RANGE = 6604
ERROR_LOG_METADATA_CORRUPT = 6612
```

ERROR_LOG_METADATA_FLUSH_FAILED = 6645
ERROR_LOG_METADATA_INCONSISTENT = 6614
ERROR_LOG_METADATA_INVALID = 6613
ERROR_LOG_MULTIPLEXED = 6630
ERROR_LOG_NOT_ENOUGH_CONTAINERS = 6635
ERROR_LOG_NO_RESTART = 6611
ERROR_LOG_PINNED = 6644
ERROR_LOG_PINNED_ARCHIVE_TAIL = 6623
ERROR_LOG_PINNED_RESERVATION = 6648
ERROR_LOG_POLICY_ALREADY_INSTALLED = 6619
ERROR_LOG_POLICY_CONFLICT = 6622
ERROR_LOG_POLICY_INVALID = 6621
ERROR_LOG_POLICY_NOT_INSTALLED = 6620
ERROR_LOG_READ_CONTEXT_INVALID = 6606
ERROR_LOG_READ_MODE_INVALID = 6610
ERROR_LOG_RECORDS_RESERVED_INVALID = 6625
ERROR_LOG_RECORD_NONEXISTENT = 6624
ERROR_LOG_RESERVATION_INVALID = 6615
ERROR_LOG_RESIZE_INVALID_SIZE = 6806
ERROR_LOG_RESTART_INVALID = 6607
ERROR_LOG_SECTOR_INVALID = 6600
ERROR_LOG_SECTOR_PARITY_INVALID = 6601
ERROR_LOG_SECTOR_REMAPPED = 6602
ERROR_LOG_SPACE_RESERVED_INVALID = 6626
ERROR_LOG_START_OF_LOG = 6618
ERROR_LOG_STATE_INVALID = 6643
ERROR_LOG_TAIL_INVALID = 6627
ERROR_LONGJUMP = 682
ERROR_LOST_WRITEBEHIND_DATA = 596
ERROR_LOST_WRITEBEHIND_DATA_LOCAL_DISK_ERROR = 790
ERROR_LOST_WRITEBEHIND_DATA_NETWORK_DISCONNECTED = 788
ERROR_LOST_WRITEBEHIND_DATA_NETWORK_SERVER_ERROR = 789
ERROR_LUIDS_EXHAUSTED = 1334
ERROR_MAGAZINE_NOT_PRESENT = 1163
ERROR_MAPPED_ALIGNMENT = 1132
ERROR_MARSHALL_OVERFLOW = 603

```
ERROR_MAX_SESSIONS_REACHED = 353
ERROR_MAX_THRDS_REACHED = 164
ERROR_MCA_EXCEPTION = 784
ERROR_MCA_OCCURED = 651
ERROR_MEDIA_CHANGED = 1110
ERROR_MEDIA_CHECK = 679
ERROR_MEDIA_INCOMPATIBLE = 4315
ERROR_MEDIA_NOT_AVAILABLE = 4318
ERROR_MEDIA_OFFLINE = 4304
ERROR_MEDIA_UNAVAILABLE = 4308
ERROR_MEDIUM_NOT_ACCESSIBLE = 4323
ERROR_MEMBERS_PRIMARY_GROUP = 1374
ERROR_MEMBER_IN_ALIAS = 1378
ERROR_MEMBER_IN_GROUP = 1320
ERROR_MEMBER_NOT_IN_ALIAS = 1377
ERROR_MEMBER_NOT_IN_GROUP = 1321
ERROR_MEMORY_HARDWARE = 779
ERROR_MENU_ITEM_NOT_FOUND = 1456
ERROR_MESSAGE_EXCEEDS_MAX_SIZE = 4336
ERROR_MESSAGE_SYNC_ONLY = 1159
ERROR_METAFILE_NOT_SUPPORTED = 2003
ERROR_META_EXPANSION_TOO_LONG = 208
ERROR_MINIVERSION_INACCESSIBLE_FROM_SPECIFIED_TRANSACTION = 6810
ERROR_MISSING_SYSTEMFILE = 573
ERROR_MOD_NOT_FOUND = 126
ERROR_MORE_DATA = 234
ERROR_MORE_WRITES = 1120
ERROR_MOUNT_POINT_NOT_RESOLVED = 649
ERROR_MP_PROCESSOR_MISMATCH = 725
ERROR_MR_MID_NOT_FOUND = 317
ERROR_MULTIPLE_FAULT_VIOLATION = 640
ERROR_MUTANT_LIMIT_EXCEEDED = 587
ERROR_NEGATIVE_SEEK = 131
ERROR_NESTING_NOT_ALLOWED = 215
ERROR_NETLOGON_NOT_STARTED = 1792
ERROR_NETNAME_DELETED = 64
```

```
ERROR_NETWORK_ACCESS_DENIED = 65
ERROR_NETWORK_BUSY = 54
ERROR_NETWORK_NOT_AVAILABLE = 5035
ERROR_NETWORK_UNREACHABLE = 1231
ERROR_NET_OPEN_FAILED = 570
ERROR_NET_WRITE_FAULT = 88
ERROR_NOACCESS = 998
ERROR_NODE_CANNOT_BE_CLUSTERED = 5898
ERROR_NODE_CANT_HOST_RESOURCE = 5071
ERROR_NODE_NOT_AVAILABLE = 5036
ERROR_NOINTERFACE = 632
ERROR_NOLOGON_INTERDOMAIN_TRUST_ACCOUNT = 1807
ERROR_NOLOGON_SERVER_TRUST_ACCOUNT = 1809
ERROR_NOLOGON_WORKSTATION_TRUST_ACCOUNT = 1808
ERROR_NONE_MAPPED = 1332
ERROR_NONPAGED_SYSTEM_RESOURCES = 1451
ERROR_NON_MDICHILD_WINDOW = 1445
ERROR_NOTHING_TO_TERMINATE = 758
ERROR_NOTIFY_CLEANUP = 745
ERROR_NOTIFY_ENUM_DIR = 1022
ERROR_NOT_ALL_ASSIGNED = 1300
ERROR_NOT_AUTHENTICATED = 1244
ERROR_NOT_A_REPARSE_POINT = 4390
ERROR_NOT_CAPABLE = 775
ERROR_NOT_CHILD_WINDOW = 1442
ERROR_NOT_CONNECTED = 2250
ERROR_NOT_CONTAINER = 1207
ERROR_NOT_DOS_DISK = 26
ERROR_NOT_EMPTY = 4307
ERROR_NOT_ENOUGH_MEMORY = 8
ERROR_NOT_ENOUGH_QUOTA = 1816
ERROR_NOT_ENOUGH_SERVER_MEMORY = 1130
ERROR_NOT_EXPORT_FORMAT = 6008
ERROR_NOT_FOUND = 1168
ERROR_NOT_JOINED = 136
ERROR_NOT_LOCKED = 158
```

```
ERROR_NOT_LOGGED_ON = 1245
ERROR_NOT_LOGON_PROCESS = 1362
ERROR_NOT_OWNER = 288
ERROR_NOT_QUORUM_CAPABLE = 5021
ERROR_NOT_QUORUM_CLASS = 5025
ERROR_NOT_READY = 21
ERROR_NOT_REGISTRY_FILE = 1017
ERROR_NOT_SAFEBOOT_SERVICE = 1084
ERROR_NOT_SAFE_MODE_DRIVER = 646
ERROR_NOT_SAME_DEVICE = 17
ERROR_NOT_SNAPSHOT_VOLUME = 6841
ERROR_NOT_SUBSTED = 137
ERROR_NOT_SUPPORTED = 50
ERROR_NOT_SUPPORTED_ON_STANDARD_SERVER = 8584
ERROR_NOT_TINY_STREAM = 598
ERROR_NO_ASSOCIATION = 1155
ERROR_NO_BROWSER_SERVERS_FOUND = 6118
ERROR_NO_CALLBACK_ACTIVE = 614
ERROR_NO_DATA = 232
ERROR_NO_DATA_DETECTED = 1104
ERROR_NO_EFS = 6004
ERROR_NO_EVENT_PAIR = 580
ERROR_NO_GUID_TRANSLATION = 560
ERROR_NO_IMPERSONATION_TOKEN = 1309
ERROR_NO_INHERITANCE = 1391
ERROR_NO_LINK_TRACKING_IN_TRANSACTION = 6852
ERROR_NO_LOGON_SERVERS = 1311
ERROR_NO_LOG_SPACE = 1019
ERROR_NO_MATCH = 1169
ERROR_NO_MEDIA_IN_DRIVE = 1112
ERROR_NO_MORE_DEVICES = 1248
ERROR_NO_MORE_FILES = 18
ERROR_NO_MORE_ITEMS = 259
ERROR_NO_MORE_MATCHES = 626
ERROR_NO_MORE_SEARCH_HANDLES = 113
ERROR_NO_MORE_USER_HANDLES = 1158
```

```
ERROR_NO_NETWORK = 1222
ERROR_NO_NET_OR_BAD_PATH = 1203
ERROR_NO_PAGEFILE = 578
ERROR_NO_PROC_SLOTS = 89
ERROR_NO_PROMOTION_ACTIVE = 8222
ERROR_NO_QUOTAS_FOR_ACCOUNT = 1302
ERROR_NO_RECOVERY_POLICY = 6003
ERROR_NO_RECOVERY_PROGRAM = 1082
ERROR_NO_SAVEPOINT_WITH_OPEN_FILES = 6842
ERROR_NO_SCROLLBARS = 1447
ERROR_NO_SECRETS = 8620
ERROR_NO_SECURITY_ON_OBJECT = 1350
ERROR_NO_SHUTDOWN_IN_PROGRESS = 1116
ERROR_NO_SIGNAL_SENT = 205
ERROR_NO_SITE_NAME = 1919
ERROR_NO_SITE_SETTINGS_OBJECT = 8619
ERROR_NO_SPOOL_SPACE = 62
ERROR_NO_SUCH_ALIAS = 1376
ERROR_NO_SUCH_DOMAIN = 1355
ERROR_NO_SUCH_GROUP = 1319
ERROR_NO_SUCH_LOGON_SESSION = 1312
ERROR_NO_SUCH_MEMBER = 1387
ERROR_NO_SUCH_PACKAGE = 1364
ERROR_NO_SUCH_PRIVILEGE = 1313
ERROR_NO_SUCH_SITE = 1249
ERROR_NO_SUCH_USER = 1317
ERROR_NO_SUPPORTING_DRIVES = 4339
ERROR_NO_SYSTEM_MENU = 1437
ERROR_NO_SYSTEM_RESOURCES = 1450
ERROR_NO_TOKEN = 1008
ERROR_NO_TRACKING_SERVICE = 1172
ERROR_NO_TRUST_LSA_SECRET = 1786
ERROR_NO_TRUST_SAM_ACCOUNT = 1787
ERROR_NO_TXF_METADATA = 6816
ERROR_NO_UNICODE_TRANSLATION = 1113
ERROR_NO_USER_KEYS = 6006
```

`ERROR_NO_USER_SESSION_KEY = 1394`
`ERROR_NO_VOLUME_ID = 1173`
`ERROR_NO_VOLUME_LABEL = 125`
`ERROR_NO_WILDCARD_CHARACTERS = 1417`
`ERROR_NO_WRITABLE_DC_FOUND = 8621`
`ERROR_NO_YIELD_PERFORMED = 721`
`ERROR_NTLM_BLOCKED = 1937`
`ERROR_NT_CROSS_ENCRYPTION_REQUIRED = 1386`
`ERROR_NULL_LM_PASSWORD = 1304`
`ERROR_OBJECT_ALREADY_EXISTS = 5010`
`ERROR_OBJECT_IN_LIST = 5011`
`ERROR_OBJECT_NAME_EXISTS = 698`
`ERROR_OBJECT_NOT_FOUND = 4312`
`ERROR_OBJECT_NO_LONGER_EXISTS = 6807`
`ERROR_OLD_WIN_VERSION = 1150`
`ERROR_OPEN_FAILED = 110`
`ERROR_OPEN_FILES = 2401`
`ERROR_OPERATION_ABORTED = 995`
`ERROR_OPERATION_NOT_SUPPORTED_IN_TRANSACTION = 6853`
`ERROR_OPLOCK_BREAK_IN_PROGRESS = 742`
`ERROR_OPLOCK_NOT_GRANTED = 300`
`ERROR_OUTOFMEMORY = 14`
`ERROR_OUT_OF_PAPER = 28`
`ERROR_OUT_OF_STRUCTURES = 84`
`ERROR_PAGED_SYSTEM_RESOURCES = 1452`
`ERROR_PAGEFILE_CREATE_FAILED = 576`
`ERROR_PAGEFILE_QUOTA = 1454`
`ERROR_PAGEFILE_QUOTA_EXCEEDED = 567`
`ERROR_PAGE_FAULT_COPY_ON_WRITE = 749`
`ERROR_PAGE_FAULT_DEMAND_ZERO = 748`
`ERROR_PAGE_FAULT_GUARD_PAGE = 750`
`ERROR_PAGE_FAULT_PAGING_FILE = 751`
`ERROR_PAGE_FAULT_TRANSITION = 747`
`ERROR_PARTIAL_COPY = 299`
`ERROR_PARTITION_FAILURE = 1105`
`ERROR_PASSWORD_EXPIRED = 1330`

ERROR_PASSWORD_MUST_CHANGE = 1907
ERROR_PASSWORD_RESTRICTION = 1325
ERROR_PATCH_MANAGED_ADVERTISED_PRODUCT = 1651
ERROR_PATCH_NO_SEQUENCE = 1648
ERROR_PATCH_PACKAGE_INVALID = 1636
ERROR_PATCH_PACKAGE_OPEN_FAILED = 1635
ERROR_PATCH_PACKAGE_REJECTED = 1643
ERROR_PATCH_PACKAGE_UNSUPPORTED = 1637
ERROR_PATCH_REMOVAL_DISALLOWED = 1649
ERROR_PATCH_REMOVAL_UNSUPPORTED = 1646
ERROR_PATCH_TARGET_NOT_FOUND = 1642
ERROR_PATH_BUSY = 148
ERROR_PATH_NOT_FOUND = 3
ERROR_PER_USER_TRUST_QUOTA_EXCEEDED = 1932
ERROR_PIPE_BUSY = 231
ERROR_PIPE_CONNECTED = 535
ERROR_PIPE_LISTENING = 536
ERROR_PIPE_LOCAL = 229
ERROR_PIPE_NOT_CONNECTED = 233
ERROR_PLUGPLAY_QUERY_VETOED = 683
ERROR_PNP_BAD_MPS_TABLE = 671
ERROR_PNP_INVALID_ID = 674
ERROR_PNP_IRQ_TRANSLATION_FAILED = 673
ERROR_PNP_REBOOT_REQUIRED = 638
ERROR_PNP_RESTART_ENUMERATION = 636
ERROR_PNP_TRANSLATION_FAILED = 672
ERROR_POINT_NOT_FOUND = 1171
ERROR_POLICY_OBJECT_NOT_FOUND = 8219
ERROR_POLICY_ONLY_IN_DS = 8220
ERROR_POPUP_ALREADY_ACTIVE = 1446
ERROR_PORT_MESSAGE_TOO_LONG = 546
ERROR_PORT_NOT_SET = 642
ERROR_PORT_UNREACHABLE = 1234
ERROR_POSSIBLE_DEADLOCK = 1131
ERROR_PREDEFINED_HANDLE = 714
ERROR_PRIMARY_TRANSPORT_CONNECT_FAILED = 746

ERROR_PRINTER_ALREADY_EXISTS = 1802
ERROR_PRINTER_DELETED = 1905
ERROR_PRINTER_DRIVER_ALREADY_INSTALLED = 1795
ERROR_PRINTER_DRIVER_BLOCKED = 3014
ERROR_PRINTER_DRIVER_DOWNLOAD_NEEDED = 3019
ERROR_PRINTER_DRIVER_IN_USE = 3001
ERROR_PRINTER_DRIVER_PACKAGE_IN_USE = 3015
ERROR_PRINTER_DRIVER_WARNED = 3013
ERROR_PRINTER_HAS_JOBS_QUEUED = 3009
ERROR_PRINTER_NOT_FOUND = 3012
ERROR_PRINTQ_FULL = 61
ERROR_PRINT_CANCELLED = 63
ERROR_PRINT_JOB_RESTART_REQUIRED = 3020
ERROR_PRINT_MONITOR_ALREADY_INSTALLED = 3006
ERROR_PRINT_MONITOR_IN_USE = 3008
ERROR_PRINT_PROCESSOR_ALREADY_INSTALLED = 3005
ERROR_PRIVATE_DIALOG_INDEX = 1415
ERROR_PRIVILEGE_NOT_HELD = 1314
ERROR_PROCESS_ABORTED = 1067
ERROR_PROCESS_IN_JOB = 760
ERROR_PROCESS_MODE_ALREADY_BACKGROUND = 402
ERROR_PROCESS_MODE_NOT_BACKGROUND = 403
ERROR_PROCESS_NOT_IN_JOB = 759
ERROR_PROC_NOT_FOUND = 127
ERROR_PRODUCT_UNINSTALLED = 1614
ERROR_PRODUCT_VERSION = 1638
ERROR_PROFILE_DOES_NOT_MATCH_DEVICE = 2023
ERROR_PROFILE_NOT_ASSOCIATED_WITH_DEVICE = 2305
ERROR_PROFILE_NOT_FOUND = 2306
ERROR_PROFILING_AT_LIMIT = 553
ERROR_PROFILING_NOT_STARTED = 550
ERROR_PROFILING_NOT_STOPPED = 551
ERROR_PROMOTION_ACTIVE = 8221
ERROR_PROTOCOL_UNREACHABLE = 1233
ERROR_PWD_HISTORY_CONFLICT = 617
ERROR_PWD_TOO_RECENT = 616

```
ERROR_PWD_TOO_SHORT = 615
ERROR_QUORUMLOG_OPEN_FAILED = 5028
ERROR_QUORUM_DISK_NOT_FOUND = 5086
ERROR_QUORUM_NOT_ALLOWED_IN_THIS_GROUP = 5928
ERROR_QUORUM_OWNER_ALIVE = 5034
ERROR_QUORUM_RESOURCE = 5020
ERROR_QUORUM_RESOURCE_ONLINE_FAILED = 5027
ERROR_QUOTA_LIST_INCONSISTENT = 621
ERROR_RANGE_LIST_CONFLICT = 627
ERROR_RANGE_NOT_FOUND = 644
ERROR_RDP_PROTOCOL_ERROR = 7065
ERROR_READ_FAULT = 30
ERROR_RECEIVE_EXPEDITED = 708
ERROR_RECEIVE_PARTIAL = 707
ERROR_RECEIVE_PARTIAL_EXPEDITED = 709
ERROR_RECOVERY_NOT_NEEDED = 6821
ERROR_REC_NON_EXISTENT = 4005
ERROR_REDIRECTOR_HAS_OPEN_HANDLES = 1794
ERROR_REDIR_PAUSED = 72
ERROR_REGISTRY_CORRUPT = 1015
ERROR_REGISTRY_HIVE_RECOVERED = 685
ERROR_REGISTRY_IO_FAILED = 1016
ERROR_REGISTRY_QUOTA_LIMIT = 613
ERROR_REGISTRY_RECOVERED = 1014
ERROR_RELOC_CHAIN_XEEDS_SEGLIM = 201
ERROR_REMOTE_FILE_VERSION_MISMATCH = 6814
ERROR_REMOTE_PRINT_CONNECTIONS_BLOCKED = 1936
ERROR_REMOTE_SESSION_LIMIT_EXCEEDED = 1220
ERROR_REMOTE_STORAGE_MEDIA_ERROR = 4352
ERROR_REMOTE_STORAGE_NOT_ACTIVE = 4351
ERROR_REM_NOT_LIST = 51
ERROR_REPARSE = 741
ERROR_REPARSE_ATTRIBUTE_CONFLICT = 4391
ERROR_REPARSE_OBJECT = 755
ERROR_REPARSE_TAG_INVALID = 4393
ERROR_REPARSE_TAG_MISMATCH = 4394
```

```
ERROR_REPLY_MESSAGE_MISMATCH = 595
ERROR_REQUEST_ABORTED = 1235
ERROR_REQUEST_OUT_OF_SEQUENCE = 776
ERROR_REQUEST_REFUSED = 4320
ERROR_REQUIRES_INTERACTIVE_WINDOWSTATION = 1459
ERROR_REQ_NOT_ACCEP = 71
ERROR_RESMON_CREATE_FAILED = 5017
ERROR_RESMON_INVALID_STATE = 5084
ERROR_RESMON_ONLINE_FAILED = 5018
ERROR_RESOURCEMANAGER_NOT_FOUND = 6716
ERROR_RESOURCEMANAGER_READ_ONLY = 6707
ERROR_RESOURCE_CALL_TIMED_OUT = 5910
ERROR_RESOURCE_DATA_NOT_FOUND = 1812
ERROR_RESOURCE_DISABLED = 4309
ERROR_RESOURCE_FAILED = 5038
ERROR_RESOURCE_LANG_NOT_FOUND = 1815
ERROR_RESOURCE_NAME_NOT_FOUND = 1814
ERROR_RESOURCE_NOT_AVAILABLE = 5006
ERROR_RESOURCE_NOT_FOUND = 5007
ERROR_RESOURCE_NOT_ONLINE = 5004
ERROR_RESOURCE_NOT_PRESENT = 4316
ERROR_RESOURCE_ONLINE = 5019
ERROR_RESOURCE_PROPERTIES_STORED = 5024
ERROR_RESOURCE_PROPERTY_UNCHANGEABLE = 5089
ERROR_RESOURCE_REQUIREMENTS_CHANGED = 756
ERROR_RESOURCE_TYPE_NOT_FOUND = 1813
ERROR_RESTART_APPLICATION = 1467
ERROR_RESUME_HIBERNATION = 727
ERROR_RETRY = 1237
ERROR_REVISION_MISMATCH = 1306
ERROR_RING2SEG_MUST_BE_MOVABLE = 200
ERROR_RING2_STACK_IN_USE = 207
ERROR_RMODE_APP = 1153
ERROR_RM_ALREADY_STARTED = 6822
ERROR_RM_DISCONNECTED = 6819
ERROR_RM_METADATA_CORRUPT = 6802
```

```
ERROR_RM_NOT_ACTIVE = 6801
ERROR_ROLLBACK_TIMER_EXPIRED = 6829
ERROR_ROWSNOTRELEASED = 772
ERROR_RPL_NOT_ALLOWED = 4006
ERROR_RXACT_COMMITTED = 744
ERROR_RXACT_COMMIT_FAILURE = 1370
ERROR_RXACT_COMMIT_NECESSARY = 678
ERROR_RXACT_INVALID_STATE = 1369
ERROR_RXACT_STATE_CREATED = 701
ERROR_SAME_DRIVE = 143
ERROR_SAM_INIT_FAILURE = 8541
ERROR_SCOPE_NOT_FOUND = 318
ERROR_SCREEN_ALREADY_LOCKED = 1440
ERROR_SECRET_TOO_LONG = 1382
ERROR_SECTOR_NOT_FOUND = 27
ERROR_SEEK = 25
ERROR_SEEK_ON_DEVICE = 132
ERROR_SEGMENT_NOTIFICATION = 702
ERROR_SEM_IS_SET = 102
ERROR_SEM_NOT_FOUND = 187
ERROR_SEM_OWNER_DIED = 105
ERROR_SEM_TIMEOUT = 121
ERROR_SEM_USER_LIMIT = 106
ERROR_SERIAL_NO_DEVICE = 1118
ERROR_SERVER_DISABLED = 1341
ERROR_SERVER_HAS_OPEN_HANDLES = 1811
ERROR_SERVER_NOT_DISABLED = 1342
ERROR_SERVER_SID_MISMATCH = 628
ERROR_SERVICE_ALREADY_RUNNING = 1056
ERROR_SERVICE_CANNOT_ACCEPT_CTRL = 1061
ERROR_SERVICE_DATABASE_LOCKED = 1055
ERROR_SERVICE_DEPENDENCY_DELETED = 1075
ERROR_SERVICE_DEPENDENCY_FAIL = 1068
ERROR_SERVICE_DISABLED = 1058
ERROR_SERVICE_DOES_NOT_EXIST = 1060
ERROR_SERVICE_EXISTS = 1073
```

ERROR_SERVICE_LOGON_FAILED = 1069
ERROR_SERVICE_MARKED_FOR_DELETE = 1072
ERROR_SERVICE_NEVER_STARTED = 1077
ERROR_SERVICE_NOTIFICATION = 716
ERROR_SERVICE_NOT_ACTIVE = 1062
ERROR_SERVICE_NOT_FOUND = 1243
ERROR_SERVICE_NOT_IN_EXE = 1083
ERROR_SERVICE_NO_THREAD = 1054
ERROR_SERVICE_REQUEST_TIMEOUT = 1053
ERROR_SERVICE_SPECIFIC_ERROR = 1066
ERROR_SERVICE_START_HANG = 1070
ERROR_SESSION_CREDENTIAL_CONFLICT = 1219
ERROR_SETCOUNT_ON_BAD_LB = 1433
ERROR_SETMARK_DETECTED = 1103
ERROR_SET_NOT_FOUND = 1170
ERROR_SET_POWER_STATE_FAILED = 1141
ERROR_SET_POWER_STATE_VETOED = 1140
ERROR_SHARED_POLICY = 8218
ERROR_SHARING_BUFFER_EXCEEDED = 36
ERROR_SHARING_PAUSED = 70
ERROR_SHARING_VIOLATION = 32
ERROR_SHUTDOWN_CLUSTER = 5008
ERROR_SHUTDOWN_IN_PROGRESS = 1115
ERROR_SIGNAL_PENDING = 162
ERROR_SIGNAL_REFUSED = 156
ERROR_SINGLE_INSTANCE_APP = 1152
ERROR_SOME_NOT_MAPPED = 1301
ERROR_SOURCE_ELEMENT_EMPTY = 1160
ERROR_SPARSE_NOT_ALLOWED_IN_TRANSACTION = 6844
ERROR_SPECIAL_ACCOUNT = 1371
ERROR_SPECIAL_GROUP = 1372
ERROR_SPECIAL_USER = 1373
ERROR_SPL_NO_ADDJOB = 3004
ERROR_SPL_NO_STARTDOC = 3003
ERROR_SPOOL_FILE_NOT_FOUND = 3002
ERROR_STACK_OVERFLOW = 1001

```
ERROR_STACK_OVERFLOW_READ = 599
ERROR_STATIC_INIT = 4002
ERROR_STOPPED_ON_SYMLINK = 681
ERROR_STREAM_MINIVERSION_NOT_FOUND = 6808
ERROR_STREAM_MINIVERSION_NOT_VALID = 6809
ERROR_SUBST_TO_JOIN = 141
ERROR_SUBST_TO_SUBST = 139
ERROR_SUCCESS = 0
ERROR_SUCCESS_REBOOT_INITIATED = 1641
ERROR_SUCCESS_REBOOT_REQUIRED = 3010
ERROR_SUCCESS_RESTART_REQUIRED = 3011
ERROR_SWAPERROR = 999
ERROR_SYMLINK_CLASS_DISABLED = 1463
ERROR_SYMLINK_NOT_SUPPORTED = 1464
ERROR_SYNCHRONIZATION_REQUIRED = 569
ERROR_SYSTEM_HIVE_TOO_LARGE = 653
ERROR_SYSTEM_IMAGE_BAD_SIGNATURE = 637
ERROR_SYSTEM_POWERSTATE_COMPLEX_TRANSITION = 783
ERROR_SYSTEM_POWERSTATE_TRANSITION = 782
ERROR_SYSTEM_PROCESS_TERMINATED = 591
ERROR_SYSTEM_SHUTDOWN = 641
ERROR_SYSTEM_TRACE = 150
ERROR_TAG_NOT_FOUND = 2302
ERROR_TAG_NOT_PRESENT = 2303
ERROR_THREAD_1_INACTIVE = 210
ERROR_THREAD_MODE_ALREADY_BACKGROUND = 400
ERROR_THREAD_MODE_NOT_BACKGROUND = 401
ERROR_THREAD_NOT_IN_PROCESS = 566
ERROR_THREAD_WAS_SUSPENDED = 699
ERROR_TIMEOUT = 1460
ERROR_TIMER_NOT_CANCELED = 541
ERROR_TIMER_RESOLUTION_NOT_SET = 607
ERROR_TIMER_RESUME_IGNORED = 722
ERROR_TLW_WITH_WSCHILD = 1406
ERROR_TM_IDENTITY_MISMATCH = 6845
ERROR_TM_INITIALIZATION_FAILED = 6706
```

```
ERROR_TM_VOLATILE = 6828
ERROR_TOKEN_ALREADY_IN_USE = 1375
ERROR_TOO_MANY_CMDS = 56
ERROR_TOO_MANY_CONTEXT_IDS = 1384
ERROR_TOO_MANY_LINKS = 1142
ERROR_TOO_MANY_LUIDS_REQUESTED = 1333
ERROR_TOO_MANY_MODULES = 214
ERROR_TOO_MANY_MUXWAITERS = 152
ERROR_TOO_MANY_NAMES = 68
ERROR_TOO_MANY_OPEN_FILES = 4
ERROR_TOO_MANY_POSTS = 298
ERROR_TOO_MANY_SECRETS = 1381
ERROR_TOO_MANY_SEMAPHORES = 100
ERROR_TOO_MANY_SEM_REQUESTS = 103
ERROR_TOO_MANY_SESS = 69
ERROR_TOO_MANY_SIDS = 1389
ERROR_TOO_MANY_TCBS = 155
ERROR_TOO_MANY_THREADS = 565
ERROR_TRANSACTION_MAPPING_UNSUPPORTED_REMOTE = 6834
ERROR_TRANSACTIONAL_CONFLICT = 6800
ERROR_TRANSACTIONAL_OPEN_NOT_ALLOWED = 6832
ERROR_TRANSACTIONMANAGER_NOT_FOUND = 6718
ERROR_TRANSACTIONMANAGER_NOT_ONLINE = 6719
ERROR_TRANSACTIONMANAGER_RECOVERY_NAME_COLLISION = 6720
ERROR_TRANSACTIONS_NOT_FROZEN = 6839
ERROR_TRANSACTIONS_UNSUPPORTED_REMOTE = 6805
ERROR_TRANSACTION_ALREADY_ABORTED = 6704
ERROR_TRANSACTION_ALREADY_COMMITTED = 6705
ERROR_TRANSACTION_FREEZE_IN_PROGRESS = 6840
ERROR_TRANSACTION_INTEGRITY_VIOLATED = 6726
ERROR_TRANSACTION_INVALID_MARSHALL_BUFFER = 6713
ERROR_TRANSACTION_NOT_ACTIVE = 6701
ERROR_TRANSACTION_NOT_FOUND = 6715
ERROR_TRANSACTION_NOT_JOINED = 6708
ERROR_TRANSACTION_NOT_REQUESTED = 6703
ERROR_TRANSACTION_NOT_ROOT = 6721
```

```
ERROR_TRANSACTION_OBJECT_EXPIRED = 6722
ERROR_TRANSACTION_PROPAGATION_FAILED = 6711
ERROR_TRANSACTION_RECORD_TOO_LONG = 6724
ERROR_TRANSACTION_REQUEST_NOT_VALID = 6702
ERROR_TRANSACTION_REQUIRED_PROMOTION = 6837
ERROR_TRANSACTION_RESPONSE_NOT_ENLISTED = 6723
ERROR_TRANSACTION_SCOPE_CALLBACKS_NOT_SET = 6836
ERROR_TRANSACTION_SUPERIOR_EXISTS = 6709
ERROR_TRANSFORM_NOT_SUPPORTED = 2004
ERROR_TRANSLATION_COMPLETE = 757
ERROR_TRANSPORT_FULL = 4328
ERROR_TRUSTED_DOMAIN_FAILURE = 1788
ERROR_TRUSTED_RELATIONSHIP_FAILURE = 1789
ERROR_TRUST_FAILURE = 1790
ERROR_TS_INCOMPATIBLE_SESSIONS = 7069
ERROR_TXF_ATTRIBUTE_CORRUPT = 6830
ERROR_TXF_DIR_NOT_EMPTY = 6826
ERROR_TXF_METADATA_ALREADY_PRESENT = 6835
ERROR_UNABLE_TO_CLEAN = 4311
ERROR_UNABLE_TO_EJECT_MOUNTED_MEDIA = 4330
ERROR_UNABLE_TO_INVENTORY_DRIVE = 4325
ERROR_UNABLE_TO_INVENTORY_SLOT = 4326
ERROR_UNABLE_TO_INVENTORY_TRANSPORT = 4327
ERROR_UNABLE_TO_LOAD_MEDIUM = 4324
ERROR_UNABLE_TO_LOCK_MEDIA = 1108
ERROR_UNABLE_TO_UNLOAD_MEDIA = 1109
ERROR_UNDEFINED_CHARACTER = 583
ERROR_UNEXPECTED_MM_CREATE_ERR = 556
ERROR_UNEXPECTED_MM_EXTEND_ERR = 558
ERROR_UNEXPECTED_MM_MAP_ERROR = 557
ERROR_UNEXPECTED_OMID = 4334
ERROR_UNEXP_NET_ERR = 59
ERROR_UNHANDLED_EXCEPTION = 574
ERROR_UNKNOWN_COMPONENT = 1607
ERROR_UNKNOWN_FEATURE = 1606
ERROR_UNKNOWN_PATCH = 1647
```



```
ERROR_UNKNOWN_PORT = 1796
ERROR_UNKNOWN_PRINTER_DRIVER = 1797
ERROR_UNKNOWN_PRINTPROCESSOR = 1798
ERROR_UNKNOWN_PRINT_MONITOR = 3000
ERROR_UNKNOWN_PRODUCT = 1605
ERROR_UNKNOWN_PROPERTY = 1608
ERROR_UNKNOWN_REVISION = 1305
ERROR_UNRECOGNIZED_MEDIA = 1785
ERROR_UNRECOGNIZED_VOLUME = 1005
ERROR_UNSUPPORTED_COMPRESSION = 618
ERROR_UNSUPPORTED_TYPE = 1630
ERROR_UNWIND = 542
ERROR_UNWIND_CONSOLIDATE = 684
ERROR_USER_APC = 737
ERROR_USER_DELETE_TRUST_QUOTA_EXCEEDED = 1934
ERROR_USER_EXISTS = 1316
ERROR_USER_MAPPED_FILE = 1224
ERROR_USER_PROFILE_LOAD = 500
ERROR_VALIDATE_CONTINUE = 625
ERROR_VC_DISCONNECTED = 240
ERROR_VDM_HARD_ERROR = 593
ERROR_VERIFIER_STOP = 537
ERROR_VERSION_PARSE_ERROR = 777
ERROR_VIRUS_DELETED = 226
ERROR_VIRUS_INFECTED = 225
ERROR_VOLSNAP_HIBERNATE_READY = 761
ERROR_VOLSNAP_PREPARE_HIBERNATE = 655
ERROR_VOLUME_CONTAINS_SYS_FILES = 4337
ERROR_VOLUME_DIRTY = 6851
ERROR_VOLUME_MOUNTED = 743
ERROR_VOLUME_NOT_SIS_ENABLED = 4500
ERROR_VOLUME_NOT_SUPPORT_EFS = 6014
ERROR_WAIT_1 = 731
ERROR_WAIT_2 = 732
ERROR_WAIT_3 = 733
ERROR_WAIT_63 = 734
```

```
ERROR_WAIT_FOR_OPLOCK = 765
ERROR_WAIT_NO_CHILDREN = 128
ERROR_WAKE_SYSTEM = 730
ERROR_WAKE_SYSTEM_DEBUGGER = 675
ERROR_WAS_LOCKED = 717
ERROR_WAS_UNLOCKED = 715
ERROR_WINDOW_NOT_COMBOBOX = 1423
ERROR_WINDOW_NOT_DIALOG = 1420
ERROR_WINDOW_OF_OTHER_THREAD = 1408
ERROR_WINS_INTERNAL = 4000
ERROR_WMI_ALREADY_DISABLED = 4212
ERROR_WMI_ALREADY_ENABLED = 4206
ERROR_WMI_DP_FAILED = 4209
ERROR_WMI_DP_NOT_FOUND = 4204
ERROR_WMI_GUID_DISCONNECTED = 4207
ERROR_WMI_GUID_NOT_FOUND = 4200
ERROR_WMI_INSTANCE_NOT_FOUND = 4201
ERROR_WMI_INVALID_MOF = 4210
ERROR_WMI_INVALID_REGINFO = 4211
ERROR_WMI_ITEMID_NOT_FOUND = 4202
ERROR_WMI_READ_ONLY = 4213
ERROR_WMI_SERVER_UNAVAILABLE = 4208
ERROR_WMI_SET_FAILURE = 4214
ERROR_WMI_TRY_AGAIN = 4203
ERROR_WMI_UNRESOLVED_INSTANCE_REF = 4205
ERROR_WORKING_SET_QUOTA = 1453
ERROR_WOW_ASSERTION = 670
ERROR_WRITE_FAULT = 29
ERROR_WRITE_PROTECT = 19
ERROR_WRONG_COMPARTMENT = 1468
ERROR_WRONG_DISK = 34
ERROR_WRONG_EFS = 6005
ERROR_WRONG_PASSWORD = 1323
ERROR_WX86_ERROR = 540
ERROR_WX86_WARNING = 539
ERROR_XMLDSIG_ERROR = 1466
```

```
ERROR_XML_PARSE_ERROR = 1465
FRS_ERR_AUTHENTICATION = 8008
FRS_ERR_CHILD_TO_PARENT_COMM = 8011
FRS_ERR_INSUFFICIENT_PRIV = 8007
FRS_ERR_INTERNAL = 8005
FRS_ERR_INTERNAL_API = 8004
FRS_ERR_INVALID_API_SEQUENCE = 8001
FRS_ERR_INVALID_SERVICE_PARAMETER = 8017
FRS_ERR_PARENT_AUTHENTICATION = 8010
FRS_ERR_PARENT_INSUFFICIENT_PRIV = 8009
FRS_ERR_PARENT_TO_CHILD_COMM = 8012
FRS_ERR_SERVICE_COMM = 8006
FRS_ERR_STARTING_SERVICE = 8002
FRS_ERR_STOPPING_SERVICE = 8003
FRS_ERR_SYSVOL_DEMOTE = 8016
FRS_ERR_SYSVOL_IS_BUSY = 8015
FRS_ERR_SYSVOL_POPULATE = 8013
FRS_ERR_SYSVOL_POPULATE_TIMEOUT = 8014
OR_INVALID_OID = 1911
OR_INVALID_OXID = 1910
OR_INVALID_SET = 1912
RPC_S_ADDRESS_ERROR = 1768
RPC_S_ALREADY_LISTENING = 1713
RPC_S_ALREADY_REGISTERED = 1711
RPC_S_BINDING_HAS_NO_AUTH = 1746
RPC_S_BINDING_INCOMPLETE = 1819
RPC_S_CALL_CANCELLED = 1818
RPC_S_CALL_FAILED = 1726
RPC_S_CALL_FAILED_DNE = 1727
RPC_S_CALL_IN_PROGRESS = 1791
RPC_S_CANNOT_SUPPORT = 1764
RPC_S_CANT_CREATE_ENDPOINT = 1720
RPC_S_COMM_FAILURE = 1820
RPC_S_DUPLICATE_ENDPOINT = 1740
RPC_S_ENTRY_ALREADY_EXISTS = 1760
RPC_S_ENTRY_NOT_FOUND = 1761
```

```
RPC_S_ENTRY_TYPE_MISMATCH = 1922
RPC_S_FP_DIV_ZERO = 1769
RPC_S_FP_OVERFLOW = 1771
RPC_S_FP_UNDERFLOW = 1770
RPC_S_GROUP_MEMBER_NOT_FOUND = 1898
RPC_S_GRP_ELT_NOT_ADDED = 1928
RPC_S_GRP_ELT_NOT_REMOVED = 1929
RPC_S_INCOMPLETE_NAME = 1755
RPC_S_INTERFACE_NOT_EXPORTED = 1924
RPC_S_INTERFACE_NOT_FOUND = 1759
RPC_S_INTERNAL_ERROR = 1766
RPC_S_INVALID_ASYNC_CALL = 1915
RPC_S_INVALID_ASYNC_HANDLE = 1914
RPC_S_INVALID_AUTH_IDENTITY = 1749
RPC_S_INVALID_BINDING = 1702
RPC_S_INVALID_BOUND = 1734
RPC_S_INVALID_ENDPOINT_FORMAT = 1706
RPC_S_INVALID_NAF_ID = 1763
RPC_S_INVALID_NAME_SYNTAX = 1736
RPC_S_INVALID_NETWORK_OPTIONS = 1724
RPC_S_INVALID_NET_ADDR = 1707
RPC_S_INVALID_OBJECT = 1900
RPC_S_INVALID_RPC_PROTSEQ = 1704
RPC_S_INVALID_STRING_BINDING = 1700
RPC_S_INVALID_STRING_UUID = 1705
RPC_S_INVALID_TAG = 1733
RPC_S_INVALID_TIMEOUT = 1709
RPC_S_INVALID_VERS_OPTION = 1756
RPC_S_MAX_CALLS_TOO_SMALL = 1742
RPC_S_NAME_SERVICE_UNAVAILABLE = 1762
RPC_S_NOTHING_TO_EXPORT = 1754
RPC_S_NOT_ALL_OBJS_EXPORTED = 1923
RPC_S_NOT_ALL_OBJS_UNEXPORTED = 1758
RPC_S_NOT_CANCELLED = 1826
RPC_S_NOT_LISTENING = 1715
RPC_S_NOT_RPC_ERROR = 1823
```

```
RPC_S_NO_BINDINGS = 1718
RPC_S_NO_CALL_ACTIVE = 1725
RPC_S_NO_CONTEXT_AVAILABLE = 1765
RPC_S_NO_ENDPOINT_FOUND = 1708
RPC_S_NO_ENTRY_NAME = 1735
RPC_S_NO_INTERFACES = 1817
RPC_S_NO_MORE_BINDINGS = 1806
RPC_S_NO_MORE_MEMBERS = 1757
RPC_S_NO_PRINC_NAME = 1822
RPC_S_NO_PROTSEQS = 1719
RPC_S_NO_PROTSEQS_REGISTERED = 1714
RPC_S_OBJECT_NOT_FOUND = 1710
RPC_S_OUT_OF_RESOURCES = 1721
RPC_S_PRF_ELT_NOT_ADDED = 1926
RPC_S_PRF_ELT_NOT_REMOVED = 1927
RPC_S_PROCNUM_OUT_OF_RANGE = 1745
RPC_S_PROFILE_NOT_ADDED = 1925
RPC_S_PROTOCOL_ERROR = 1728
RPC_S_PROTSEQ_NOT_FOUND = 1744
RPC_S_PROTSEQ_NOT_SUPPORTED = 1703
RPC_S_PROXY_ACCESS_DENIED = 1729
RPC_S_SEC_PKG_ERROR = 1825
RPC_S_SEND_INCOMPLETE = 1913
RPC_S_SERVER_TOO_BUSY = 1723
RPC_S_SERVER_UNAVAILABLE = 1722
RPC_S_STRING_TOO_LONG = 1743
RPC_S_TYPE_ALREADY_REGISTERED = 1712
RPC_S_UNKNOWN_AUTHN_LEVEL = 1748
RPC_S_UNKNOWN_AUTHN_SERVICE = 1747
RPC_S_UNKNOWN_AUTHN_TYPE = 1741
RPC_S_UNKNOWN_AUTHZ_SERVICE = 1750
RPC_S_UNKNOWN_IF = 1717
RPC_S_UNKNOWN_MGR_TYPE = 1716
RPC_S_UNSUPPORTED_AUTHN_LEVEL = 1821
RPC_S_UNSUPPORTED_NAME_SYNTAX = 1737
RPC_S_UNSUPPORTED_TRANS_SYN = 1730
```

```
RPC_S_UNSUPPORTED_TYPE = 1732
RPC_S_UUID_LOCAL_ONLY = 1824
RPC_S_UUID_NO_ADDRESS = 1739
RPC_S_WRONG_KIND_OF_BINDING = 1701
RPC_S_ZERO_DIVIDE = 1767
RPC_X_BAD_STUB_DATA = 1783
RPC_X_BYTE_COUNT_TOO_SMALL = 1782
RPC_X_ENUM_VALUE_OUT_OF_RANGE = 1781
RPC_X_INVALID_ES_ACTION = 1827
RPC_X_INVALID_PIPE_OBJECT = 1830
RPC_X_NO_MORE_ENTRIES = 1772
RPC_X_NULL_REF_POINTER = 1780
RPC_X_PIPE_CLOSED = 1916
RPC_X_PIPE_DISCIPLINE_ERROR = 1917
RPC_X_PIPE_EMPTY = 1918
RPC_X_SS_CANNOT_GET_CALL_HANDLE = 1779
RPC_X_SS_CHAR_TRANS_OPEN_FAIL = 1773
RPC_X_SS_CHAR_TRANS_SHORT_FILE = 1774
RPC_X_SS_CONTEXT_DAMAGED = 1777
RPC_X_SS_HANDLES_MISMATCH = 1778
RPC_X_SS_IN_NULL_CONTEXT = 1775
RPC_X_WRONG_ES_VERSION = 1828
RPC_X_WRONG_PIPE_ORDER = 1831
RPC_X_WRONG_PIPE_VERSION = 1832
RPC_X_WRONG_STUB_VERSION = 1829
WAIT_TIMEOUT = 258
```

`decode_hresult` (*hresult*)

Look up a Win32 error code based on the error code in a HRESULT.

5.8.12 Module contents

5.9 Exceptions

If an error occurs, the API attempts to roll the error into an appropriate Exception class.

5.9.1 Exception Classes

exception `ApiError` (*message=None, original_exception=None*)

Base class for all CBC SDK errors; also raised for generic internal errors.

Initialize the `ApiError`.

Parameters

- **`message`** (*str*) – The actual error message.
- **`original_exception`** (*Exception*) – The exception that caused this one to be raised.

exception `CredentialError` (*message=None, original_exception=None*)

The credentials had an unspecified error.

Initialize the `ApiError`.

Parameters

- **`message`** (*str*) – The actual error message.
- **`original_exception`** (*Exception*) – The exception that caused this one to be raised.

exception `ServerError` (*error_code, message, result=None, original_exception=None*)

A `ServerError` is raised when an HTTP 5xx error code is returned from the Carbon Black server.

Initialize the `ServerError`.

Parameters

- **`error_code`** (*int*) – The error code that was received from the server.
- **`message`** (*str*) – The actual error message.
- **`result`** (*object*) – The result of the operation from the server.
- **`original_exception`** (*Exception*) – The exception that caused this one to be raised.

exception `ObjectNotFoundError` (*uri, message=None, original_exception=None*)

The requested object could not be found in the Carbon Black datastore.

Initialize the `ObjectNotFoundError`.

Parameters

- **`uri`** (*str*) – The URI of the action that failed.
- **`message`** (*str*) – The error message.
- **`original_exception`** (*Exception*) – The exception that caused this one to be raised.

exception `MoreThanOneResultError` (*message=None, original_exception=None, results=None*)

Only one object was requested, but multiple matches were found in the Carbon Black datastore.

Initialize the `MoreThanOneResultError`.

Parameters

- **`message`** (*str*) – The actual error message.
- **`original_exception`** (*Exception*) – The exception that caused this one to be raised.
- **`results`** (*list*) – List of results returned

exception `InvalidObjectError` (*message=None, original_exception=None*)

An invalid object was received by the server.

Initialize the `ApiError`.

Parameters

- **message** (*str*) – The actual error message.
- **original_exception** (*Exception*) – The exception that caused this one to be raised.

exception TimeoutError (*uri=None, error_code=None, message=None, original_exception=None*)

A requested operation timed out.

Initialize the TimeoutError.

Parameters

- **uri** (*str*) – The URI of the action that timed out.
- **error_code** (*int*) – The error code that was received from the server.
- **message** (*str*) – The error message.
- **original_exception** (*Exception*) – The exception that caused this one to be raised.

CHAPTER 6

Indices and tables

- `genindex`
- `modindex`
- `search`

C

`cbc_sdk`, 338
`cbc_sdk.audit_remediation`, 69
`cbc_sdk.audit_remediation.base`, 49
`cbc_sdk.audit_remediation.differential`, 66
`cbc_sdk.base`, 215
`cbc_sdk.cache`, 215
`cbc_sdk.cache.lru`, 213
`cbc_sdk.connection`, 228
`cbc_sdk.credential_providers`, 72
`cbc_sdk.credential_providers.default`, 69
`cbc_sdk.credential_providers.environ_credential_provider`, 70
`cbc_sdk.credential_providers.file_credential_provider`, 70
`cbc_sdk.credential_providers.keychain_credential_provider`, 71
`cbc_sdk.credential_providers.registry_credential_provider`, 71
`cbc_sdk.credentials`, 234
`cbc_sdk.endpoint_standard`, 91
`cbc_sdk.endpoint_standard.base`, 75
`cbc_sdk.endpoint_standard.recommendation`, 78
`cbc_sdk.endpoint_standard.usb_device_control`, 83
`cbc_sdk.enterprise_edr`, 115
`cbc_sdk.enterprise_edr.auth_events`, 91
`cbc_sdk.enterprise_edr.threat_intelligence`, 97
`cbc_sdk.enterprise_edr.ubs`, 112
`cbc_sdk.errors`, 236
`cbc_sdk.helpers`, 239
`cbc_sdk.live_response_api`, 240
`cbc_sdk.platform`, 194
`cbc_sdk.platform.alerts`, 115
`cbc_sdk.platform.base`, 131
`cbc_sdk.platform.devices`, 132
`cbc_sdk.platform.events`, 144
`cbc_sdk.platform.grants`, 146
`cbc_sdk.platform.jobs`, 151
`cbc_sdk.platform.network_threat_metadata`, 153
`cbc_sdk.platform.observations`, 154
`cbc_sdk.platform.policies`, 159
`cbc_sdk.platform.policy_ruleconfigs`, 169
`cbc_sdk.platform.processes`, 171
`cbc_sdk.platform.reputation`, 178
`cbc_sdk.platform.users`, 181
`cbc_sdk.platform.vulnerability_assessment`, 186
`cbc_sdk.rest_api`, 251
`cbc_sdk.utils`, 255
`cbc_sdk.winerror`, 256
`cbc_sdk.workload`, 213
`cbc_sdk.workload.nsx_remediation`, 194
`cbc_sdk.workload.sensor_lifecycle`, 195
`cbc_sdk.workload.vc_workloads_search`, 197

A

- `accept()` (*Recommendation method*), 81
- `access` (*Feed attribute*), 98
- `action` (*PolicyRule attribute*), 168
- `action` (*Recommendation.RecommendationNewRule attribute*), 80
- `activation_code` (*Device attribute*), 134
- `activation_code_expiry_time` (*Device attribute*), 134
- `active_org_devices` (*Run attribute*), 58
- `active_org_devices` (*Template attribute*), 64
- `ad_group_id` (*Device attribute*), 134
- `add_criteria()` (*CriteriaBuilderSupportMixin method*), 216
- `add_criteria()` (*VulnerabilityQuery method*), 189
- `add_descriptions()` (*PolicyQuery method*), 167
- `add_directory_action_rule()` (*Policy.PolicyBuilder method*), 160
- `add_exclusions()` (*FacetQuery method*), 217
- `add_exclusions()` (*Query method*), 224
- `add_facet_field()` (*FacetQuery method*), 217
- `add_grant_profile()` (*User.UserBuilder method*), 182
- `add_ioc()` (*Report.ReportBuilder method*), 104
- `add_names()` (*PolicyQuery method*), 167
- `add_org()` (*Grant.ProfileBuilder method*), 148
- `add_policy_ids()` (*PolicyQuery method*), 167
- `add_principal()` (*GrantQuery method*), 150
- `add_priorities()` (*PolicyQuery method*), 167
- `add_profiles()` (*User method*), 183
- `add_range()` (*FacetQuery method*), 218
- `add_report_ids()` (*Watchlist method*), 110
- `add_report_ids()` (*Watchlist.WatchlistBuilder method*), 109
- `add_reports()` (*Feed.FeedBuilder method*), 97
- `add_reports()` (*Watchlist method*), 110
- `add_reports()` (*Watchlist.WatchlistBuilder method*), 109
- `add_role()` (*Grant.GrantBuilder method*), 146
- `add_role()` (*Grant.ProfileBuilder method*), 148
- `add_rule()` (*Policy method*), 164
- `add_rule()` (*Policy.PolicyBuilder method*), 161
- `add_rule_config()` (*Policy.PolicyBuilder method*), 161
- `add_rule_config_copy()` (*Policy.PolicyBuilder method*), 161
- `add_rule_copy()` (*Policy.PolicyBuilder method*), 161
- `add_sensor_kit_type()` (*SensorKitQuery method*), 196
- `add_sensor_setting()` (*Policy.PolicyBuilder method*), 161
- `add_tag()` (*Report.ReportBuilder method*), 104
- `admin_login_version` (*User attribute*), 183
- `affected_assets` (*Vulnerability attribute*), 187
- `AffectedAssetQuery` (*class in cbc_sdk.platform.vulnerability_assessment*), 186
- `aggregation()` (*EnrichedEventQuery method*), 77
- `alert_category` (*Observation attribute*), 154
- `alert_id` (*Observation attribute*), 154
- `alert_search_suggestions()` (*CBCCloudAPI method*), 251
- `alerts_enabled` (*Watchlist attribute*), 110
- `all()` (*IterableQueryMixin method*), 220
- `allowed_orgs` (*Grant.Profile attribute*), 147
- `and_()` (*QueryBuilder method*), 225
- `and_()` (*QueryBuilderSupportMixin method*), 226
- `and_()` (*SimpleQuery method*), 227
- `api_json_request()` (*BaseAPI method*), 229
- `api_request_iterate()` (*BaseAPI method*), 229
- `api_request_stream()` (*BaseAPI method*), 229
- `ApiError`, 236, 339
- `append_iocs()` (*Report method*), 105
- `append_reports()` (*Feed method*), 98
- `append_reports_rawdata()` (*Feed method*), 98
- `application` (*PolicyRule attribute*), 168
- `application` (*Recommendation.RecommendationNewRule attribute*),

- 80
- application_ (Recommendation.RecommendationNewRule attribute), 80
- approval_name (USBDeviceApproval attribute), 86
- approve () (USBDevice method), 84
- approve_process_sha256 () (EnrichedEvent method), 75
- approve_process_sha256 () (Process method), 174
- architecture (Binary attribute), 113
- archive_time (Template attribute), 64
- archived_by (Template attribute), 64
- ArrayFieldDescriptor (class in cbc_sdk.base), 215
- async_await_result () (NSXRemediationJob method), 194
- async_export () (DifferentialQuery method), 67
- async_export () (ResultQuery method), 54
- ASYNC_RATE_LIMIT (in module cbc_sdk.audit_remediation.differential), 66
- AsyncProcessQuery (class in cbc_sdk.platform.processes), 171
- AsyncQueryMixin (class in cbc_sdk.base), 215
- audit_remediation () (CBCloudAPI method), 251
- audit_remediation_history () (CBCloudAPI method), 252
- auth_domain_name (AuthEvent attribute), 91
- auth_event_action (AuthEvent attribute), 91
- auth_method (User attribute), 183
- auth_remote_device (AuthEvent attribute), 91
- auth_remote_port (AuthEvent attribute), 91
- auth_username (AuthEvent attribute), 91
- AuthEvent (class in cbc_sdk.enterprise_edr.auth_events), 91
- AuthEventFacet (class in cbc_sdk.enterprise_edr.auth_events), 94
- AuthEventFacet.Ranges (class in cbc_sdk.enterprise_edr.auth_events), 94
- AuthEventFacet.Terms (class in cbc_sdk.enterprise_edr.auth_events), 94
- AuthEventGroup (class in cbc_sdk.enterprise_edr.auth_events), 95
- AuthEventQuery (class in cbc_sdk.enterprise_edr.auth_events), 95
- author (BaseAlert.Note attribute), 116
- auto_delete_known_bad_hashes_delay (Policy attribute), 164
- auto_deregister_inactive_vdi_interval_ms (Policy attribute), 164
- av_ave_version (Device attribute), 134
- av_engine (Device attribute), 134
- av_last_scan_time (Device attribute), 134
- av_master (Device attribute), 134
- av_pack_version (Device attribute), 134
- av_product_version (Device attribute), 134
- av_settings (Policy attribute), 164
- av_status (Device attribute), 134
- av_update_servers (Device attribute), 134
- av_vdf_version (Device attribute), 134
- available_file_size (Binary attribute), 113
- await_completion () (Job method), 151
- await_result () (NSXRemediationJob method), 194
- AWSComputeResource (class in cbc_sdk.workload.vm_workloads_search), 197
- AWSComputeResourceQuery (class in cbc_sdk.workload.vm_workloads_search), 198
- ## B
- backend_timestamp (AuthEvent attribute), 91
- backend_timestamp (Observation attribute), 154
- background_scan () (Device method), 134
- background_scan () (DeviceSearchQuery method), 139
- ban_process_sha256 () (EnrichedEvent method), 75
- ban_process_sha256 () (Process method), 174
- BaseAlert (class in cbc_sdk.platform.alerts), 115
- BaseAlert.Note (class in cbc_sdk.platform.alerts), 116
- BaseAlertSearchQuery (class in cbc_sdk.platform.alerts), 118
- BaseAPI (class in cbc_sdk.connection), 228
- BaseComputeResource (class in cbc_sdk.workload.vm_workloads_search), 202
- BaseComputeResourceQuery (class in cbc_sdk.workload.vm_workloads_search), 203
- BaseQuery (class in cbc_sdk.base), 215
- batch_size () (PaginatedQuery method), 223
- Binary (class in cbc_sdk.enterprise_edr.ubs), 112
- Binary.Summary (class in cbc_sdk.enterprise_edr.ubs), 113
- BinaryFieldDescriptor (class in cbc_sdk.base), 216
- build () (Feed.FeedBuilder method), 98
- build () (Grant.GrantBuilder method), 146
- build () (Grant.ProfileBuilder method), 148
- build () (Policy.PolicyBuilder method), 161
- build () (Report.ReportBuilder method), 104
- build () (User.UserBuilder method), 182
- build () (Watchlist.WatchlistBuilder method), 109
- build_cli_parser () (in module cbc_sdk.helpers), 239

[bulk_add_profiles\(\)](#) ([cbc_sdk.platform.users.User](#) class method), 183
[bulk_create\(\)](#) ([cbc_sdk.endpoint_standard.usb_device_control.USBDeviceAppControl](#) class method), 86
[bulk_create\(\)](#) ([cbc_sdk.endpoint_standard.usb_device_control.USBDeviceBlockControl](#) class method), 88
[bulk_create\(\)](#) ([cbc_sdk.platform.users.User](#) class method), 183
[bulk_create_csv\(\)](#) ([cbc_sdk.endpoint_standard.usb_device_control.USBDeviceAppControl](#) class method), 86
[bulk_delete\(\)](#) ([cbc_sdk.platform.reputation.ReputationOverride](#) class method), 179
[bulk_delete\(\)](#) ([cbc_sdk.platform.users.User](#) class method), 183
[bulk_disable_all_access\(\)](#) ([cbc_sdk.platform.users.User](#) class method), 183
[bulk_disable_profiles\(\)](#) ([cbc_sdk.platform.users.User](#) class method), 183
[bulk_get_details\(\)](#) ([AuthEvent](#) static method), 91
[bulk_get_details\(\)](#) ([Observation](#) static method), 154
[bulk_install\(\)](#) ([cbc_sdk.workload.vm_workloads_search.AWSCloudComputeResource](#) class method), 197
[bulk_install\(\)](#) ([cbc_sdk.workload.vm_workloads_search.BaseCloudComputeResource](#) class method), 202
[bulk_install\(\)](#) ([cbc_sdk.workload.vm_workloads_search.VCenterComputeResource](#) class method), 205
[bulk_install_by_id\(\)](#) ([cbc_sdk.workload.vm_workloads_search.AWSCloudComputeResource](#) class method), 197
[bulk_install_by_id\(\)](#) ([cbc_sdk.workload.vm_workloads_search.BaseCloudComputeResource](#) class method), 202
[bulk_install_by_id\(\)](#) ([cbc_sdk.workload.vm_workloads_search.VCenterComputeResource](#) class method), 206
[bulk_threat_dismiss\(\)](#) ([CBCloudAPI](#) method), 252
[bulk_threat_update\(\)](#) ([CBCloudAPI](#) method), 252
[bypass\(\)](#) ([Device](#) method), 134
[bypass\(\)](#) ([DeviceSearchQuery](#) method), 139

C

[CACHE_E_FIRST](#) ([RawErrorCode](#) attribute), 270
[CACHE_E_LAST](#) ([RawErrorCode](#) attribute), 270
[CACHE_E_NOCACHE_UPDATED](#) ([RawErrorCode](#) attribute), 270
[CACHE_S_FIRST](#) ([RawErrorCode](#) attribute), 270
[CACHE_S_LAST](#) ([RawErrorCode](#) attribute), 270
[can_manage](#) ([Grant](#) attribute), 149
[can_manage](#) ([Grant.Profile](#) attribute), 148
[cancel_command\(\)](#) ([CbLRSessionBase](#) method), 241
[cancel_usb_device_app_control\(\)](#) ([Run](#) attribute), 58
[cancellation_time](#) ([Template](#) attribute), 64
[cancelled_by](#) ([Template](#) attribute), 64
[cancelled_count](#) ([Run](#) attribute), 58
[cancelled_count](#) ([Template](#) attribute), 64
[CAT_E_CATIDNOEXIST](#) ([RawErrorCode](#) attribute), 270
[CAT_E_APPROVAL](#) ([RawErrorCode](#) attribute), 270
[CAT_E_FIRST](#) ([RawErrorCode](#) attribute), 270
[CAT_E_LAST](#) ([RawErrorCode](#) attribute), 270
[CAT_E_NODescription](#) ([RawErrorCode](#) attribute), 270
[category](#) ([BaseAlert](#) attribute), 116
[category](#) ([CorePreventionRuleConfig](#) attribute), 169
[category](#) ([Feed](#) attribute), 98
[category](#) ([PolicyRuleConfig](#) attribute), 171
[category](#) ([Vulnerability](#) attribute), 187
[CBAnalyticsAlert](#) (class in [cbc_sdk.platform.alerts](#)), 123
[CBAnalyticsAlertSearchQuery](#) (class in [cbc_sdk.platform.alerts](#)), 123
[cbc_sdk](#) (module), 338
[cbc.AWSCloudComputeResource](#) (module), 69
[cbc_sdk.audit_remediation.base](#) (module), 169
[cbc_sdk.audit_remediation.differential](#) (module), 169
[cbc_sdk.base](#) (module), 215
[cbc_sdk.cache](#) (module), 215
[cbc_sdk.cache.lru](#) (module), 213
[cbc_sdk.connection](#) (module), 228
[cbc_sdk.credential_providers](#) (module), 72
[cbc_sdk.credential_providers.default](#) (module), 69
[cbc_sdk.credential_providers.environ_credential_provider](#) (module), 70
[cbc_sdk.credential_providers.file_credential_provider](#) (module), 70
[cbc_sdk.credential_providers.keychain_credential_provider](#) (module), 71
[cbc_sdk.credential_providers.registry_credential_provider](#) (module), 71
[cbc_sdk.credentials](#) (module), 234
[cbc_sdk.endpoint_standard](#) (module), 91
[cbc_sdk.endpoint_standard.base](#) (module), 75
[cbc_sdk.endpoint_standard.recommendation](#) (module), 78
[cbc_sdk.endpoint_standard.usb_device_control](#) (module), 83
[cbc_sdk.enterprise_edr](#) (module), 115

`cbc_sdk.enterprise_edr.auth_events` (module), 91
`cbc_sdk.enterprise_edr.threat_intelligence` (module), 97
`cbc_sdk.enterprise_edr.ubs` (module), 112
`cbc_sdk.errors` (module), 236
`cbc_sdk.helpers` (module), 239
`cbc_sdk.live_response_api` (module), 240
`cbc_sdk.platform` (module), 194
`cbc_sdk.platform.alerts` (module), 115
`cbc_sdk.platform.base` (module), 131
`cbc_sdk.platform.devices` (module), 132
`cbc_sdk.platform.events` (module), 144
`cbc_sdk.platform.grants` (module), 146
`cbc_sdk.platform.jobs` (module), 151
`cbc_sdk.platform.network_threat_metadata` (module), 153
`cbc_sdk.platform.observations` (module), 154
`cbc_sdk.platform.policies` (module), 159
`cbc_sdk.platform.policy_ruleconfigs` (module), 169
`cbc_sdk.platform.processes` (module), 171
`cbc_sdk.platform.reputation` (module), 178
`cbc_sdk.platform.users` (module), 181
`cbc_sdk.platform.vulnerability_assessment` (module), 186
`cbc_sdk.rest_api` (module), 251
`cbc_sdk.utils` (module), 255
`cbc_sdk.winerror` (module), 256
`cbc_sdk.workload` (module), 213
`cbc_sdk.workload.nsx_remediation` (module), 194
`cbc_sdk.workload.sensor_lifecycle` (module), 195
`cbc_sdk.workload.vm_workloads_search` (module), 197
`CBCcloudAPI` (class in `cbc_sdk.rest_api`), 251
`CBCSDKSessionAdapter` (class in `cbc_sdk.connection`), 231
`cblr_base` (*CbLRManagerBase* attribute), 240
`cblr_base` (*LiveResponseSessionManager* attribute), 249
`cblr_session_cls` (*CbLRManagerBase* attribute), 240
`cblr_session_cls` (*LiveResponseSessionManager* attribute), 250
`CbLRManagerBase` (class in `cbc_sdk.live_response_api`), 240
`CbLRSessionBase` (class in `cbc_sdk.live_response_api`), 241
`CbMetaModel` (class in `cbc_sdk.base`), 216
`CCERR_CHOOSECOLORCODES` (*CommDlgError* attribute), 256
`CDERR_DIALOGFAILURE` (*CommDlgError* attribute), 256
`CDERR_FINDRESFAILURE` (*CommDlgError* attribute), 256
`CDERR_GENERALCODES` (*CommDlgError* attribute), 256
`CDERR_INITIALIZATION` (*CommDlgError* attribute), 256
`CDERR_LOADRESFAILURE` (*CommDlgError* attribute), 256
`CDERR_LOADSTRFAILURE` (*CommDlgError* attribute), 256
`CDERR_LOCKRESFAILURE` (*CommDlgError* attribute), 256
`CDERR_MEMALLOCFAILURE` (*CommDlgError* attribute), 256
`CDERR_MEMLOCKFAILURE` (*CommDlgError* attribute), 256
`CDERR_NOINSTANCE` (*CommDlgError* attribute), 256
`CDERR_NOHOOK` (*CommDlgError* attribute), 256
`CDERR_NOTEMPLATE` (*CommDlgError* attribute), 256
`CDERR_REGISTERMSGFAIL` (*CommDlgError* attribute), 256
`CDERR_STRUCTSIZE` (*CommDlgError* attribute), 256
`CERT_E_CHAINING` (*RawErrorCode* attribute), 270
`CERT_E_CN_NO_MATCH` (*RawErrorCode* attribute), 270
`CERT_E_CRITICAL` (*RawErrorCode* attribute), 270
`CERT_E_EXPIRED` (*RawErrorCode* attribute), 270
`CERT_E_ISSUERCHAINING` (*RawErrorCode* attribute), 270
`CERT_E_MALFORMED` (*RawErrorCode* attribute), 270
`CERT_E_PATHLENCONST` (*RawErrorCode* attribute), 270
`CERT_E_PURPOSE` (*RawErrorCode* attribute), 270
`CERT_E_REVOCATION_FAILURE` (*RawErrorCode* attribute), 270
`CERT_E_REVOKED` (*RawErrorCode* attribute), 270
`CERT_E_ROLE` (*RawErrorCode* attribute), 270
`CERT_E_UNTRUSTEDROOT` (*RawErrorCode* attribute), 270
`CERT_E_UNTRUSTEDTESTROOT` (*RawErrorCode* attribute), 270
`CERT_E_VALIDITYPERIODNESTING` (*RawErrorCode* attribute), 270
`CERT_E_WRONG_USAGE` (*RawErrorCode* attribute), 270
`CERTDB_E_JET_ERROR` (*RawErrorCode* attribute), 270
`certificate_authority` (*Recommendation.RecommendationNewRule* attribute), 80
`certificate_authority` (*ReputationOverride* attribute), 179

- CERTSRV_E_BAD_REQUESTSTATUS (*RawErrorCode attribute*), 270
- CERTSRV_E_BAD_REQUESTSUBJECT (*RawErrorCode attribute*), 270
- CERTSRV_E_NO_REQUEST (*RawErrorCode attribute*), 270
- CERTSRV_E_PROPERTY_EMPTY (*RawErrorCode attribute*), 270
- CFERR_CHOOSEFONTCODES (*CommDlgError attribute*), 256
- CFERR_MAXLESSTHANMIN (*CommDlgError attribute*), 256
- CFERR_NOFONTS (*CommDlgError attribute*), 256
- change_role() (*User method*), 183
- changed_by (*Recommendation attribute*), 81
- changed_by (*Recommendation.RecommendationWorkflow attribute*), 81
- changed_by (*Workflow attribute*), 130
- charset_id (*Binary attribute*), 113
- check_python_tls_compatibility() (*in module cbc_sdk.connection*), 234
- childproc_count (*AuthEvent attribute*), 92
- children (*Process attribute*), 174
- CLASS_E_CLASSNOTAVAILABLE (*RawErrorCode attribute*), 270
- CLASS_E_NOAGGREGATION (*RawErrorCode attribute*), 271
- CLASS_E_NOTLICENSED (*RawErrorCode attribute*), 271
- CLASSFACTORY_E_FIRST (*RawErrorCode attribute*), 270
- CLASSFACTORY_E_LAST (*RawErrorCode attribute*), 270
- CLASSFACTORY_S_FIRST (*RawErrorCode attribute*), 270
- CLASSFACTORY_S_LAST (*RawErrorCode attribute*), 270
- classifier (*Watchlist attribute*), 110
- classifier_ (*Watchlist attribute*), 110
- cleanup() (*LRUCacheDict method*), 214
- clear() (*LRUCacheDict method*), 214
- ClientError, 236
- CLIENTSITE_E_FIRST (*RawErrorCode attribute*), 271
- CLIENTSITE_E_LAST (*RawErrorCode attribute*), 271
- CLIENTSITE_S_FIRST (*RawErrorCode attribute*), 271
- CLIENTSITE_S_LAST (*RawErrorCode attribute*), 271
- CLIPBRD_E_BAD_DATA (*RawErrorCode attribute*), 271
- CLIPBRD_E_CANT_CLOSE (*RawErrorCode attribute*), 271
- CLIPBRD_E_CANT_EMPTY (*RawErrorCode attribute*), 271
- CLIPBRD_E_CANT_OPEN (*RawErrorCode attribute*), 271
- CLIPBRD_E_CANT_SET (*RawErrorCode attribute*), 271
- CLIPBRD_E_FIRST (*RawErrorCode attribute*), 271
- CLIPBRD_E_LAST (*RawErrorCode attribute*), 271
- CLIPBRD_S_FIRST (*RawErrorCode attribute*), 271
- CLIPBRD_S_LAST (*RawErrorCode attribute*), 271
- close() (*CbLRSessionBase method*), 241
- close_session() (*CbLRManagerBase method*), 240
- CO_E_ACCESSCHECKFAILED (*RawErrorCode attribute*), 271
- CO_E_ACESINWRONGORDER (*RawErrorCode attribute*), 271
- CO_E_ACNOTINITIALIZED (*RawErrorCode attribute*), 271
- CO_E_ALREADYINITIALIZED (*RawErrorCode attribute*), 271
- CO_E_APPDIDNTREG (*RawErrorCode attribute*), 271
- CO_E_APPNOTFOUND (*RawErrorCode attribute*), 271
- CO_E_APPSINGLEUSE (*RawErrorCode attribute*), 271
- CO_E_BAD_PATH (*RawErrorCode attribute*), 271
- CO_E_BAD_SERVER_NAME (*RawErrorCode attribute*), 271
- CO_E_CANT_REMOTE (*RawErrorCode attribute*), 272
- CO_E_CANTDETERMINECLASS (*RawErrorCode attribute*), 271
- CO_E_CLASS_CREATE_FAILED (*RawErrorCode attribute*), 272
- CO_E_CLASSSTRING (*RawErrorCode attribute*), 272
- CO_E_CLSREG_INCONSISTENT (*RawErrorCode attribute*), 272
- CO_E_CONVERSIONFAILED (*RawErrorCode attribute*), 272
- CO_E_CREATEPROCESS_FAILURE (*RawErrorCode attribute*), 272
- CO_E_DECODEFAILED (*RawErrorCode attribute*), 272
- CO_E_DLLNOTFOUND (*RawErrorCode attribute*), 272
- CO_E_ERRORINAPP (*RawErrorCode attribute*), 272
- CO_E_ERRORINDLL (*RawErrorCode attribute*), 272
- CO_E_EXCEEDSYSACLLIMIT (*RawErrorCode attribute*), 272
- CO_E_FAILEDTOCLOSEHANDLE (*RawErrorCode attribute*), 272
- CO_E_FAILEDTOCREATEFILE (*RawErrorCode attribute*), 272
- CO_E_FAILEDTOGENUUID (*RawErrorCode attribute*), 272
- CO_E_FAILEDTOGETSECCTX (*RawErrorCode attribute*), 272
- CO_E_FAILEDTOGETTOKENINFO (*RawErrorCode attribute*), 272
- CO_E_FAILEDTOGETWINDIR (*RawErrorCode attribute*), 272

tribute), 272

CO_E_FAILEDTOIMPERSONATE (*RawErrorCode attribute*), 272

CO_E_FAILEDTOOPENPROCESSTOKEN (*RawErrorCode attribute*), 272

CO_E_FAILEDTOOPENTHREADTOKEN (*RawErrorCode attribute*), 272

CO_E_FAILEDTOQUERYCLIENTBLANKET (*RawErrorCode attribute*), 272

CO_E_FAILEDTOSETDACL (*RawErrorCode attribute*), 272

CO_E_FIRST (*RawErrorCode attribute*), 272

CO_E_IIDREG_INCONSISTENT (*RawErrorCode attribute*), 272

CO_E_IIDSTRING (*RawErrorCode attribute*), 272

CO_E_INCOMPATIBLESTREAMVERSION (*RawErrorCode attribute*), 272

CO_E_INIT_CLASS_CACHE (*RawErrorCode attribute*), 272

CO_E_INIT_MEMORY_ALLOCATOR (*RawErrorCode attribute*), 272

CO_E_INIT_ONLY_SINGLE_THREADED (*RawErrorCode attribute*), 272

CO_E_INIT_RPC_CHANNEL (*RawErrorCode attribute*), 272

CO_E_INIT_SCM_EXEC_FAILURE (*RawErrorCode attribute*), 272

CO_E_INIT_SCM_FILE_MAPPING_EXISTS (*RawErrorCode attribute*), 272

CO_E_INIT_SCM_MAP_VIEW_OF_FILE (*RawErrorCode attribute*), 272

CO_E_INIT_SCM_MUTEX_EXISTS (*RawErrorCode attribute*), 272

CO_E_INIT_SHARED_ALLOCATOR (*RawErrorCode attribute*), 272

CO_E_INIT_TLS (*RawErrorCode attribute*), 272

CO_E_INIT_TLS_CHANNEL_CONTROL (*RawErrorCode attribute*), 273

CO_E_INIT_TLS_SET_CHANNEL_CONTROL (*RawErrorCode attribute*), 273

CO_E_INIT_UNACCEPTED_USER_ALLOCATOR (*RawErrorCode attribute*), 273

CO_E_INVALIDSID (*RawErrorCode attribute*), 273

CO_E_LAST (*RawErrorCode attribute*), 273

CO_E_LAUNCH_PERMSSION_DENIED (*RawErrorCode attribute*), 273

CO_E_LOOKUPACCFNAMEFAILED (*RawErrorCode attribute*), 273

CO_E_LOOKUPACCSIDFAILED (*RawErrorCode attribute*), 273

CO_E_MSI_ERROR (*RawErrorCode attribute*), 273

CO_E_NETACCESSAPIFAILED (*RawErrorCode attribute*), 273

CO_E_NOMATCHINGNAMEFOUND (*RawErrorCode attribute*), 273

CO_E_NOMATCHINGSIDFOUND (*RawErrorCode attribute*), 273

CO_E_NOT_SUPPORTED (*RawErrorCode attribute*), 273

CO_E_NOTINITIALIZED (*RawErrorCode attribute*), 273

CO_E_OBJISREG (*RawErrorCode attribute*), 273

CO_E_OBJNOTCONNECTED (*RawErrorCode attribute*), 273

CO_E_OBJNOTREG (*RawErrorCode attribute*), 273

CO_E_OBJSRV_RPC_FAILURE (*RawErrorCode attribute*), 273

CO_E_OLE1DDE_DISABLED (*RawErrorCode attribute*), 273

CO_E_PATHTOOLONG (*RawErrorCode attribute*), 273

CO_E_RELEASED (*RawErrorCode attribute*), 273

CO_E_RELOAD_DLL (*RawErrorCode attribute*), 273

CO_E_REMOTE_COMMUNICATION_FAILURE (*RawErrorCode attribute*), 273

CO_E_RUNAS_CREATEPROCESS_FAILURE (*RawErrorCode attribute*), 273

CO_E_RUNAS_LOGON_FAILURE (*RawErrorCode attribute*), 273

CO_E_RUNAS_SYNTAX (*RawErrorCode attribute*), 273

CO_E_SCM_ERROR (*RawErrorCode attribute*), 273

CO_E_SCM_RPC_FAILURE (*RawErrorCode attribute*), 273

CO_E_SERVER_EXEC_FAILURE (*RawErrorCode attribute*), 273

CO_E_SERVER_START_TIMEOUT (*RawErrorCode attribute*), 273

CO_E_SERVER_STOPPING (*RawErrorCode attribute*), 273

CO_E_SETSERLHNDLFAILED (*RawErrorCode attribute*), 273

CO_E_START_SERVICE_FAILURE (*RawErrorCode attribute*), 273

CO_E_TRUSTEEDOESNTMATCHCLIENT (*RawErrorCode attribute*), 273

CO_E_WRONG_SERVER_IDENTITY (*RawErrorCode attribute*), 274

CO_E_WRONGOSFORAPP (*RawErrorCode attribute*), 273

CO_E_WRONGTRUSTEENAMESYNTAX (*RawErrorCode attribute*), 273

CO_S_FIRST (*RawErrorCode attribute*), 274

CO_S_LAST (*RawErrorCode attribute*), 274

CO_S_NOTALLINTERFACES (*RawErrorCode attribute*), 274

command_status() (*CbLRSessionBase method*), 241

CommDlgError (*class in cbc_sdk.winerror*), 256

comment (*Recommendation attribute*), 81

comment (*Recommendation.RecommendationWorkflow*

- attribute*), 81
- comment (Workflow attribute)*, 130
- comments (Binary attribute)*, 113
- company_name (Binary attribute)*, 113
- completed (AuthEventFacet attribute)*, 94
- completed (EnrichedEventFacet attribute)*, 77
- completed (ObservationFacet attribute)*, 157
- completed (ProcessFacet attribute)*, 177
- CompletionNotification (class in cbc_sdk.live_response_api)*, 248
- COMPUTE_RESOURCE_MAP (SensorKit attribute)*, 195
- ComputeResourceFacet (class in cbc_sdk.workload.vm_workloads_search)*, 205
- ComputeResourceFacet.ComputeResourceFacetValue (class in cbc_sdk.workload.vm_workloads_search)*, 205
- conditions (Grant.Profile attribute)*, 148
- config_params() (SensorKitQuery method)*, 196
- Connection (class in cbc_sdk.connection)*, 232
- ConnectionError*, 236
- connector_id (Job attribute)*, 152
- construct_include() (in module cbc_sdk.base)*, 228
- contact_id (User attribute)*, 184
- contact_version (User attribute)*, 184
- contacted (AuthEventFacet attribute)*, 94
- contacted (EnrichedEventFacet attribute)*, 77
- contacted (ObservationFacet attribute)*, 157
- contacted (ProcessFacet attribute)*, 177
- ContainerRuntimeAlert (class in cbc_sdk.platform.alerts)*, 125
- ContainerRuntimeAlertSearchQuery (class in cbc_sdk.platform.alerts)*, 125
- CONVERT10_E_FIRST (RawErrorCode attribute)*, 271
- CONVERT10_E_LAST (RawErrorCode attribute)*, 271
- CONVERT10_E_OLESTREAM_BITMAP_TO_DIB (RawErrorCode attribute)*, 271
- CONVERT10_E_OLESTREAM_FMT (RawErrorCode attribute)*, 271
- CONVERT10_E_OLESTREAM_GET (RawErrorCode attribute)*, 271
- CONVERT10_E_OLESTREAM_PUT (RawErrorCode attribute)*, 271
- CONVERT10_E_STG_DIB_TO_BITMAP (RawError-Code attribute)*, 271
- CONVERT10_E_STG_FMT (RawErrorCode attribute)*, 271
- CONVERT10_E_STG_NO_STD_STREAM (RawError-Code attribute)*, 271
- CONVERT10_S_FIRST (RawErrorCode attribute)*, 271
- CONVERT10_S_LAST (RawErrorCode attribute)*, 271
- convert_feed_query() (CBCloudAPI method)*, 252
- convert_from_cb() (in module cbc_sdk.utils)*, 255
- convert_to_cb() (in module cbc_sdk.utils)*, 255
- core_prevention_rule_configs (Policy attribute)*, 164
- core_prevention_rule_configs_list (Policy attribute)*, 165
- CorePreventionRuleConfig (class in cbc_sdk.platform.policy_ruleconfigs)*, 169
- count_only() (DifferentialQuery method)*, 68
- CreatableModelMixin (class in cbc_sdk.base)*, 216
- create() (BaseAPI method)*, 229
- create() (cbc_sdk.endpoint_standard.usb_device_control.USBDeviceBl class method)*, 88
- create() (cbc_sdk.enterprise_edr.threat_intelligence.Feed class method)*, 98
- create() (cbc_sdk.enterprise_edr.threat_intelligence.Report class method)*, 105
- create() (cbc_sdk.enterprise_edr.threat_intelligence.Watchlist class method)*, 110
- create() (cbc_sdk.platform.grants.Grant class method)*, 149
- create() (cbc_sdk.platform.policies.Policy class method)*, 165
- create() (cbc_sdk.platform.reputation.ReputationOverride class method)*, 179
- create() (cbc_sdk.platform.users.User class method)*, 184
- create() (CBCloudAPI method)*, 252
- create_directory() (CbLRSessionBase method)*, 241
- create_equality() (cbc_sdk.enterprise_edr.threat_intelligence.IOC_V2 class method)*, 102
- create_from_feed() (cbc_sdk.enterprise_edr.threat_intelligence.Watchlist class method)*, 110
- create_from_usb_device() (cbc_sdk.endpoint_standard.usb_device_control.USBDeviceAppr class method)*, 86
- create_note() (BaseAlert method)*, 116
- create_process() (CbLRSessionBase method)*, 241
- create_profile() (Grant method)*, 149
- create_profile() (Grant.GrantBuilder method)*, 147
- create_query() (cbc_sdk.enterprise_edr.threat_intelligence.IOC_V2 class method)*, 102
- create_regex() (cbc_sdk.enterprise_edr.threat_intelligence.IOC_V2 class method)*, 102
- create_registry_key() (CbLRSessionBase method)*, 242
- create_time (BaseAlert attribute)*, 116
- create_time (BaseAlert.Note attribute)*, 116
- create_time (Grant attribute)*, 150
- create_time (Job attribute)*, 152

`create_time` (*Recommendation attribute*), 81
`create_time` (*Recommendation.RecommendationWorkflow attribute*), 81
`create_time` (*ReputationOverride attribute*), 180
`create_time` (*Run attribute*), 58
`create_time` (*Template attribute*), 64
`create_timestamp` (*Watchlist attribute*), 110
`created_at` (*USBDevice attribute*), 84
`created_at` (*USBDeviceApproval attribute*), 87
`created_at` (*USBDeviceBlock attribute*), 88
`created_by` (*Grant attribute*), 150
`created_by` (*ReputationOverride attribute*), 180
`created_by` (*Run attribute*), 58
`created_by` (*Template attribute*), 64
`CredentialError`, 236, 339
`CredentialProvider` (class in `cbc_sdk.credentials`), 234
`Credentials` (class in `cbc_sdk.credentials`), 235
`CredentialValue` (class in `cbc_sdk.credentials`), 234
`CriteriaBuilderSupportMixin` (class in `cbc_sdk.base`), 216
`crossproc_count` (*AuthEvent attribute*), 92
`CRYPT_E_ALREADY_DECRYPTED` (*RawErrorCode attribute*), 274
`CRYPT_E_ATTRIBUTES_MISSING` (*RawErrorCode attribute*), 274
`CRYPT_E_AUTH_ATTR_MISSING` (*RawErrorCode attribute*), 274
`CRYPT_E_BAD_ENCODE` (*RawErrorCode attribute*), 274
`CRYPT_E_BAD_LEN` (*RawErrorCode attribute*), 274
`CRYPT_E_BAD_MSG` (*RawErrorCode attribute*), 274
`CRYPT_E_CONTROL_TYPE` (*RawErrorCode attribute*), 274
`CRYPT_E_DELETED_PREV` (*RawErrorCode attribute*), 274
`CRYPT_E_EXISTS` (*RawErrorCode attribute*), 274
`CRYPT_E_FILE_ERROR` (*RawErrorCode attribute*), 274
`CRYPT_E_FILERESIZED` (*RawErrorCode attribute*), 274
`CRYPT_E_HASH_VALUE` (*RawErrorCode attribute*), 274
`CRYPT_E_INVALID_IA5_STRING` (*RawErrorCode attribute*), 274
`CRYPT_E_INVALID_INDEX` (*RawErrorCode attribute*), 274
`CRYPT_E_INVALID_MSG_TYPE` (*RawErrorCode attribute*), 274
`CRYPT_E_INVALID_NUMERIC_STRING` (*RawErrorCode attribute*), 274
`CRYPT_E_INVALID_PRINTABLE_STRING` (*RawErrorCode attribute*), 274
`CRYPT_E_INVALID_X500_STRING` (*RawErrorCode attribute*), 274
`CRYPT_E_ISSUER_SERIALNUMBER` (*RawErrorCode attribute*), 274
`CRYPT_E_MSG_ERROR` (*RawErrorCode attribute*), 274
`CRYPT_E_NO_DECRYPT_CERT` (*RawErrorCode attribute*), 274
`CRYPT_E_NO_KEY_PROPERTY` (*RawErrorCode attribute*), 274
`CRYPT_E_NO_MATCH` (*RawErrorCode attribute*), 274
`CRYPT_E_NO_PROVIDER` (*RawErrorCode attribute*), 274
`CRYPT_E_NO_REVOCATION_CHECK` (*RawErrorCode attribute*), 274
`CRYPT_E_NO_REVOCATION_DLL` (*RawErrorCode attribute*), 274
`CRYPT_E_NO_SIGNER` (*RawErrorCode attribute*), 274
`CRYPT_E_NO_TRUSTED_SIGNER` (*RawErrorCode attribute*), 275
`CRYPT_E_NO_VERIFY_USAGE_CHECK` (*RawErrorCode attribute*), 275
`CRYPT_E_NO_VERIFY_USAGE_DLL` (*RawErrorCode attribute*), 275
`CRYPT_E_NOT_CHAR_STRING` (*RawErrorCode attribute*), 274
`CRYPT_E_NOT_DECRYPTED` (*RawErrorCode attribute*), 274
`CRYPT_E_NOT_FOUND` (*RawErrorCode attribute*), 274
`CRYPT_E_NOT_IN_CTL` (*RawErrorCode attribute*), 274
`CRYPT_E_NOT_IN_REVOCATION_DATABASE` (*RawErrorCode attribute*), 274
`CRYPT_E_OID_FORMAT` (*RawErrorCode attribute*), 275
`CRYPT_E_OSS_ERROR` (*RawErrorCode attribute*), 275
`CRYPT_E_PENDING_CLOSE` (*RawErrorCode attribute*), 275
`CRYPT_E_RECIPIENT_NOT_FOUND` (*RawErrorCode attribute*), 275
`CRYPT_E_REVOCATION_OFFLINE` (*RawErrorCode attribute*), 275
`CRYPT_E_REVOKED` (*RawErrorCode attribute*), 275
`CRYPT_E_SECURITY_SETTINGS` (*RawErrorCode attribute*), 275
`CRYPT_E_SELF_SIGNED` (*RawErrorCode attribute*), 275
`CRYPT_E_SIGNER_NOT_FOUND` (*RawErrorCode attribute*), 275
`CRYPT_E_STREAM_INSUFFICIENT_DATA` (*RawErrorCode attribute*), 275
`CRYPT_E_STREAM_MSG_NOT_READY` (*RawErrorCode attribute*), 275
`CRYPT_E_UNEXPECTED_ENCODING` (*RawErrorCode attribute*), 275

- CRYPT_E_UNEXPECTED_MSG_TYPE (*RawErrorCode attribute*), 275
- CRYPT_E_UNKNOWN_ALGO (*RawErrorCode attribute*), 275
- CRYPT_E_VERIFY_USAGE_OFFLINE (*RawErrorCode attribute*), 275
- CS_E_CLASS_NOTFOUND (*RawErrorCode attribute*), 275
- CS_E_FIRST (*RawErrorCode attribute*), 275
- CS_E_INVALID_VERSION (*RawErrorCode attribute*), 275
- CS_E_LAST (*RawErrorCode attribute*), 275
- CS_E_NO_CLASSSTORE (*RawErrorCode attribute*), 275
- CS_E_NOT_DELETABLE (*RawErrorCode attribute*), 275
- CS_E_PACKAGE_NOTFOUND (*RawErrorCode attribute*), 275
- CSP_API_TOKEN (*CredentialValue attribute*), 234
- CSP_OAUTH_APP_ID (*CredentialValue attribute*), 234
- CSP_OAUTH_APP_SECRET (*CredentialValue attribute*), 234
- CSP_URL_OVERRIDE (*CredentialValue attribute*), 235
- current_sensor_policy_name (*Device attribute*), 134
- custom_severities (*CBCloudAPI attribute*), 253
- custom_severity (*Report attribute*), 105
- ## D
- daemon (*LiveResponseJobScheduler attribute*), 249
- daemon (*LRUCacheDict.EmptyCacheThread attribute*), 214
- DATA_E_FIRST (*RawErrorCode attribute*), 275
- DATA_E_LAST (*RawErrorCode attribute*), 275
- DATA_S_FIRST (*RawErrorCode attribute*), 275
- DATA_S_LAST (*RawErrorCode attribute*), 275
- decode_hresult() (in module *cbc_sdk.winerror*), 338
- default_credential_provider() (in module *cbc_sdk.credential_providers.default*), 70
- DEFAULT_FACET_ROWS (*BaseComputeResourceQuery attribute*), 203
- default_sort (*EnrichedEvent attribute*), 75
- default_sort (*Event attribute*), 144
- default_sort (*Process attribute*), 174
- default_sort (*Process.Summary attribute*), 173
- default_sort (*Process.Tree attribute*), 173
- DefaultProvider (class in *cbc_sdk.credential_providers.default*), 69
- delete() (*BaseAlert.Note method*), 116
- delete() (*Connection method*), 232
- delete() (*Feed method*), 99
- delete() (*LiveResponseMemdump method*), 249
- delete() (*MutableBaseModel method*), 221
- delete() (*Report method*), 106
- delete() (*ReputationOverride method*), 180
- delete() (*Run method*), 58
- delete() (*USBDeviceBlock method*), 88
- delete() (*User method*), 184
- delete() (*Watchlist method*), 110
- delete_file() (*CbLRSessionBase method*), 242
- delete_object() (*BaseAPI method*), 230
- delete_registry_key() (*CbLRSessionBase method*), 242
- delete_registry_value() (*CbLRSessionBase method*), 243
- delete_rule() (*Policy method*), 165
- delete_rule_config() (*Policy method*), 165
- delete_sensor() (*Device method*), 134
- delete_sensor() (*DeviceSearchQuery method*), 139
- deployment_type (*Device attribute*), 135
- deregistered_time (*Device attribute*), 135
- description (*CorePreventionRuleConfig attribute*), 170
- description (*Policy attribute*), 165
- description (*PolicyRuleConfig attribute*), 171
- description (*Report attribute*), 106
- description (*ReputationOverride attribute*), 180
- description (*Watchlist attribute*), 111
- detector_abstract (*NetworkThreatMetadata attribute*), 153
- detector_goal (*NetworkThreatMetadata attribute*), 154
- Device (class in *cbc_sdk.platform.devices*), 132
- device (*DeviceSummary attribute*), 50
- device (*Result attribute*), 53
- device_ (*Result attribute*), 53
- device_background_scan() (*CBCloudAPI method*), 253
- device_bypass() (*CBCloudAPI method*), 253
- device_count (*Vulnerability attribute*), 187
- device_delete_sensor() (*CBCloudAPI method*), 253
- device_filter (*Run attribute*), 58
- device_filter (*Template attribute*), 64
- device_friendly_name (*USBDevice attribute*), 84
- device_group_id (*AuthEvent attribute*), 92
- device_group_id (*Observation attribute*), 154
- device_id (*AuthEvent attribute*), 92
- device_id (*BaseAlert attribute*), 116
- device_id (*Device attribute*), 135
- device_id (*Observation attribute*), 154
- device_ids() (*RunQuery method*), 61
- device_message (*DeviceSummary attribute*), 50
- device_message (*Result attribute*), 53
- device_meta_data_item_list (*Device attribute*), 135
- device_name (*AuthEvent attribute*), 92

- `device_name` (*BaseAlert attribute*), 116
- `device_name` (*Observation attribute*), 155
- `device_name` (*USBDevice attribute*), 84
- `device_os` (*BaseAlert attribute*), 117
- `device_os_version` (*BaseAlert attribute*), 117
- `device_owner_id` (*Device attribute*), 135
- `device_policy` (*Observation attribute*), 155
- `device_policy_id` (*AuthEvent attribute*), 92
- `device_policy_id` (*Observation attribute*), 155
- `device_quarantine()` (*CBCloudAPI method*), 253
- `device_timestamp` (*AuthEvent attribute*), 92
- `device_timestamp` (*Observation attribute*), 155
- `device_type` (*USBDevice attribute*), 84
- `device_types()` (*RunQuery method*), 61
- `device_uninstall_sensor()` (*CBCloudAPI method*), 253
- `device_update_policy()` (*CBCloudAPI method*), 254
- `device_update_sensor_version()` (*CB-CloudAPI method*), 254
- `device_username` (*BaseAlert attribute*), 117
- `DeviceControlAlert` (class in *cbc_sdk.platform.alerts*), 127
- `DeviceControlAlertSearchQuery` (class in *cbc_sdk.platform.alerts*), 128
- `DeviceFacet` (class in *cbc_sdk.platform.devices*), 138
- `DeviceFacet.DeviceFacetValue` (class in *cbc_sdk.platform.devices*), 138
- `deviceId` (*Device attribute*), 135
- `DeviceSearchQuery` (class in *cbc_sdk.platform.devices*), 139
- `DeviceSummary` (class in *cbc_sdk.audit_remediation.base*), 49
- `DeviceSummary.Metrics` (class in *cbc_sdk.audit_remediation.base*), 50
- `DeviceSummaryFacet` (class in *cbc_sdk.audit_remediation.base*), 50
- `diff_processed_time` (*Differential attribute*), 67
- `diff_results` (*Differential attribute*), 67
- `Differential` (class in *cbc_sdk.audit_remediation.differential*), 66
- `DifferentialQuery` (class in *cbc_sdk.audit_remediation.differential*), 67
- `DIGSIG_E_CRYPTO` (*RawErrorCode attribute*), 275
- `DIGSIG_E_DECODE` (*RawErrorCode attribute*), 275
- `DIGSIG_E_ENCODE` (*RawErrorCode attribute*), 275
- `DIGSIG_E_EXTENSIBILITY` (*RawErrorCode attribute*), 275
- `directory_action_rules` (*Policy attribute*), 165
- `DirectoryStorageError` (class in *cbc_sdk.winerror*), 257
- `disable_alerts()` (*Watchlist method*), 111
- `disable_all_access()` (*User method*), 184
- `disable_insecure_warnings()` (in module *cbc_sdk.helpers*), 239
- `disable_profiles()` (*User method*), 184
- `disable_tags()` (*Watchlist method*), 111
- `dismiss()` (*BaseAlert method*), 117
- `dismiss()` (*BaseAlertSearchQuery method*), 118
- `dismiss_threat()` (*BaseAlert method*), 117
- `DISP_E_ARRAYISLOCKED` (*RawErrorCode attribute*), 275
- `DISP_E_BADCALLEE` (*RawErrorCode attribute*), 275
- `DISP_E_BADINDEX` (*RawErrorCode attribute*), 275
- `DISP_E_BADPARAMCOUNT` (*RawErrorCode attribute*), 276
- `DISP_E_BADVARTYPE` (*RawErrorCode attribute*), 276
- `DISP_E_DIVBYZERO` (*RawErrorCode attribute*), 276
- `DISP_E_EXCEPTION` (*RawErrorCode attribute*), 276
- `DISP_E_MEMBERNOTFOUND` (*RawErrorCode attribute*), 276
- `DISP_E_NONAMEDARGS` (*RawErrorCode attribute*), 276
- `DISP_E_NOTACOLLECTION` (*RawErrorCode attribute*), 276
- `DISP_E_OVERFLOW` (*RawErrorCode attribute*), 276
- `DISP_E_PARAMNOTFOUND` (*RawErrorCode attribute*), 276
- `DISP_E_PARAMNOTOPTIONAL` (*RawErrorCode attribute*), 276
- `DISP_E_TYPEMISMATCH` (*RawErrorCode attribute*), 276
- `DISP_E_UNKNOWNINTERFACE` (*RawErrorCode attribute*), 276
- `DISP_E_UNKNOWNLCID` (*RawErrorCode attribute*), 276
- `DISP_E_UNKNOWNNAME` (*RawErrorCode attribute*), 276
- `dns` (*IOC attribute*), 101
- `download()` (*BaseComputeResourceQuery method*), 203
- `download()` (*DeviceSearchQuery method*), 140
- `download_url()` (*Binary method*), 113
- `Downloads` (class in *cbc_sdk.enterprise_edr.ubs*), 114
- `Downloads.FoundItem` (class in *cbc_sdk.enterprise_edr.ubs*), 114
- `DROPTROP_E_ALREADYREGISTERED` (*RawErrorCode attribute*), 276
- `DROPTROP_E_FIRST` (*RawErrorCode attribute*), 276
- `DROPTROP_E_INVALIDHWND` (*RawErrorCode attribute*), 276
- `DROPTROP_E_LAST` (*RawErrorCode attribute*), 276
- `DROPTROP_E_NOTREGISTERED` (*RawErrorCode attribute*), 276
- `DROPTROP_S_FIRST` (*RawErrorCode attribute*), 276
- `DROPTROP_S_LAST` (*RawErrorCode attribute*), 276
- `DS_S_SUCCESS` (*Win32Error attribute*), 286

DV_E_CLIPFORMAT (*RawErrorCode attribute*), 276
 DV_E_DVASPECT (*RawErrorCode attribute*), 276
 DV_E_DVTARGETDEVICE (*RawErrorCode attribute*), 276
 DV_E_DVTARGETDEVICE_SIZE (*RawErrorCode attribute*), 276
 DV_E_FORMATETC (*RawErrorCode attribute*), 276
 DV_E_LINDEX (*RawErrorCode attribute*), 276
 DV_E_NOIVIEWOBJECT (*RawErrorCode attribute*), 276
 DV_E_STATDATA (*RawErrorCode attribute*), 276
 DV_E_STGMEDIUM (*RawErrorCode attribute*), 276
 DV_E_TYMED (*RawErrorCode attribute*), 276

E

E_ABORT (*RawErrorCode attribute*), 276
 E_ACCESSDENIED (*RawErrorCode attribute*), 277
 E_FAIL (*RawErrorCode attribute*), 277
 E_HANDLE (*RawErrorCode attribute*), 277
 E_INVALIDARG (*RawErrorCode attribute*), 277
 E_NOINTERFACE (*RawErrorCode attribute*), 277
 E_NOTIMPL (*RawErrorCode attribute*), 277
 E_OUTOFMEMORY (*RawErrorCode attribute*), 277
 E_PENDING (*RawErrorCode attribute*), 277
 E_POINTER (*RawErrorCode attribute*), 277
 E_UNEXPECTED (*RawErrorCode attribute*), 277
 email (*Device attribute*), 135
 email (*User attribute*), 184
 email_addresses() (*UserQuery method*), 185
 enable_alerts() (*Watchlist method*), 111
 enable_tags() (*Watchlist method*), 111
 encoded_activation_code (*Device attribute*), 135
 endpoint_count (*USBDevice attribute*), 84
 enriched (*Observation attribute*), 155
 enriched_event_type (*Observation attribute*), 155
 EnrichedEvent (class in *cbc_sdk.endpoint_standard.base*), 75
 EnrichedEventFacet (class in *cbc_sdk.endpoint_standard.base*), 76
 EnrichedEventFacet.Ranges (class in *cbc_sdk.endpoint_standard.base*), 76
 EnrichedEventFacet.Terms (class in *cbc_sdk.endpoint_standard.base*), 76
 EnrichedEventQuery (class in *cbc_sdk.endpoint_standard.base*), 77
 ENUM_E_FIRST (*RawErrorCode attribute*), 276
 ENUM_E_LAST (*RawErrorCode attribute*), 276
 ENUM_S_FIRST (*RawErrorCode attribute*), 276
 ENUM_S_LAST (*RawErrorCode attribute*), 276
 EnvironCredentialProvider (class in *cbc_sdk.credential_providers.environ_credential_provider*), 70
 EpochDateTimeFieldDescriptor (class in *cbc_sdk.base*), 217
 eprint() (*in module cbc_sdk.helpers*), 239
 EPT_S_CANT_CREATE (*Win32Error attribute*), 286
 EPT_S_CANT_PERFORM_OP (*Win32Error attribute*), 286
 EPT_S_INVALID_ENTRY (*Win32Error attribute*), 286
 EPT_S_NOT_REGISTERED (*Win32Error attribute*), 286
 ERROR_ABANDON_HIBERFILE (*Win32Error attribute*), 286
 ERROR_ABANDONED_WAIT_0 (*Win32Error attribute*), 286
 ERROR_ABANDONED_WAIT_63 (*Win32Error attribute*), 286
 ERROR_ABIOS_ERROR (*Win32Error attribute*), 286
 ERROR_ACCESS_AUDIT_BY_POLICY (*Win32Error attribute*), 286
 ERROR_ACCESS_DENIED (*Win32Error attribute*), 286
 ERROR_ACCESS_DISABLED_NO_SAFER_UI_BY_POLICY (*Win32Error attribute*), 286
 ERROR_ACCOUNT_DISABLED (*Win32Error attribute*), 286
 ERROR_ACCOUNT_EXPIRED (*Win32Error attribute*), 286
 ERROR_ACCOUNT_LOCKED_OUT (*Win32Error attribute*), 286
 ERROR_ACCOUNT_RESTRICTION (*Win32Error attribute*), 286
 ERROR_ACPI_ERROR (*Win32Error attribute*), 286
 ERROR_ACTIVATION_COUNT_EXCEEDED (*Win32Error attribute*), 286
 ERROR_ACTIVE_CONNECTIONS (*Win32Error attribute*), 286
 ERROR_ADAP_HDW_ERR (*Win32Error attribute*), 286
 ERROR_ADDRESS_ALREADY_ASSOCIATED (*Win32Error attribute*), 286
 ERROR_ADDRESS_NOT_ASSOCIATED (*Win32Error attribute*), 286
 ERROR_ALERTED (*Win32Error attribute*), 286
 ERROR_ALIAS_EXISTS (*Win32Error attribute*), 286
 ERROR_ALL_NODES_NOT_AVAILABLE (*Win32Error attribute*), 287
 ERROR_ALL_USER_TRUST_QUOTA_EXCEEDED (*Win32Error attribute*), 287
 ERROR_ALLOCATE_BUCKET (*Win32Error attribute*), 286
 ERROR_ALLOTTED_SPACE_EXCEEDED (*Win32Error attribute*), 287
 ERROR_ALREADY_ASSIGNED (*Win32Error attribute*), 287
 ERROR_ALREADY_EXISTS (*Win32Error attribute*), 287
 ERROR_ALREADY_INITIALIZED (*Win32Error attribute*), 287
 ERROR_ALREADY_REGISTERED (*Win32Error attribute*), 287

- tribute*), 287
- ERROR_ALREADY_RUNNING_LKG (*Win32Error attribute*), 287
- ERROR_ALREADY_WAITING (*Win32Error attribute*), 287
- ERROR_ALREADY_WIN32 (*Win32Error attribute*), 287
- ERROR_APP_INIT_FAILURE (*Win32Error attribute*), 287
- ERROR_APP_WRONG_OS (*Win32Error attribute*), 287
- ERROR_ARBITRATION_UNHANDLED (*Win32Error attribute*), 287
- ERROR_ARENA_TRASHED (*Win32Error attribute*), 287
- ERROR_ARITHMETIC_OVERFLOW (*Win32Error attribute*), 287
- ERROR_ASSERTION_FAILURE (*Win32Error attribute*), 287
- ERROR_ATOMIC_LOCKS_NOT_SUPPORTED (*Win32Error attribute*), 287
- ERROR_AUDIT_FAILED (*Win32Error attribute*), 287
- ERROR_AUTHENTICATION_FIREWALL_FAILED (*Win32Error attribute*), 287
- ERROR_AUTHIP_FAILURE (*Win32Error attribute*), 287
- ERROR_AUTODATASEG_EXCEEDS_64k (*Win32Error attribute*), 287
- ERROR_BACKUP_CONTROLLER (*Win32Error attribute*), 287
- ERROR_BAD_ACCESSOR_FLAGS (*Win32Error attribute*), 287
- ERROR_BAD_ARGUMENTS (*Win32Error attribute*), 287
- ERROR_BAD_CLUSTERS (*Win32Error attribute*), 287
- ERROR_BAD_COMMAND (*Win32Error attribute*), 287
- ERROR_BAD_COMPRESSION_BUFFER (*Win32Error attribute*), 287
- ERROR_BAD_CONFIGURATION (*Win32Error attribute*), 287
- ERROR_BAD_CURRENT_DIRECTORY (*Win32Error attribute*), 287
- ERROR_BAD_DATABASE_VERSION (*Win32Error attribute*), 287
- ERROR_BAD_DESCRIPTOR_FORMAT (*Win32Error attribute*), 287
- ERROR_BAD_DEV_TYPE (*Win32Error attribute*), 287
- ERROR_BAD_DEVICE (*Win32Error attribute*), 287
- ERROR_BAD_DLL_ENTRYPOINT (*Win32Error attribute*), 288
- ERROR_BAD_DRIVER (*Win32Error attribute*), 288
- ERROR_BAD_DRIVER_LEVEL (*Win32Error attribute*), 288
- ERROR_BAD_ENVIRONMENT (*Win32Error attribute*), 288
- ERROR_BAD_EXE_FORMAT (*Win32Error attribute*), 288
- ERROR_BAD_FILE_TYPE (*Win32Error attribute*), 288
- ERROR_BAD_FORMAT (*Win32Error attribute*), 288
- ERROR_BAD_FUNCTION_TABLE (*Win32Error attribute*), 288
- ERROR_BAD_IMPERSONATION_LEVEL (*Win32Error attribute*), 288
- ERROR_BAD_INHERITANCE_ACL (*Win32Error attribute*), 288
- ERROR_BAD_LENGTH (*Win32Error attribute*), 288
- ERROR_BAD_LOGON_SESSION_STATE (*Win32Error attribute*), 288
- ERROR_BAD_MCFG_TABLE (*Win32Error attribute*), 288
- ERROR_BAD_NET_NAME (*Win32Error attribute*), 288
- ERROR_BAD_NET_RESP (*Win32Error attribute*), 288
- ERROR_BAD_NETPATH (*Win32Error attribute*), 288
- ERROR_BAD_PATHNAME (*Win32Error attribute*), 288
- ERROR_BAD_PIPE (*Win32Error attribute*), 288
- ERROR_BAD_PROFILE (*Win32Error attribute*), 288
- ERROR_BAD_PROVIDER (*Win32Error attribute*), 288
- ERROR_BAD_QUERY_SYNTAX (*Win32Error attribute*), 288
- ERROR_BAD_RECOVERY_POLICY (*Win32Error attribute*), 288
- ERROR_BAD_REM_ADAP (*Win32Error attribute*), 288
- ERROR_BAD_SERVICE_ENTRYPOINT (*Win32Error attribute*), 288
- ERROR_BAD_STACK (*Win32Error attribute*), 288
- ERROR_BAD_THREADID_ADDR (*Win32Error attribute*), 288
- ERROR_BAD_TOKEN_TYPE (*Win32Error attribute*), 288
- ERROR_BAD_UNIT (*Win32Error attribute*), 288
- ERROR_BAD_USERNAME (*Win32Error attribute*), 288
- ERROR_BAD_VALIDATION_CLASS (*Win32Error attribute*), 288
- ERROR_BADDB (*Win32Error attribute*), 287
- ERROR_BADKEY (*Win32Error attribute*), 287
- ERROR_BADSTARTPOSITION (*Win32Error attribute*), 287
- ERROR_BEGINNING_OF_MEDIA (*Win32Error attribute*), 288
- ERROR_BIOS_FAILED_TO_CONNECT_INTERRUPT (*Win32Error attribute*), 288
- ERROR_BOOT_ALREADY_ACCEPTED (*Win32Error attribute*), 288
- ERROR_BROKEN_PIPE (*Win32Error attribute*), 288
- ERROR_BUFFER_ALL_ZEROS (*Win32Error attribute*), 288
- ERROR_BUFFER_OVERFLOW (*Win32Error attribute*), 288
- ERROR_BUS_RESET (*Win32Error attribute*), 289
- ERROR_BUSY (*Win32Error attribute*), 289
- ERROR_BUSY_DRIVE (*Win32Error attribute*), 289

ERROR_CACHE_PAGE_LOCKED (*Win32Error attribute*), 289
 ERROR_CALL_NOT_IMPLEMENTED (*Win32Error attribute*), 289
 ERROR_CALLBACK_POP_STACK (*Win32Error attribute*), 289
 ERROR_CAN_NOT_COMPLETE (*Win32Error attribute*), 290
 ERROR_CAN_NOT_DEL_LOCAL_WINS (*Win32Error attribute*), 290
 ERROR_CANCEL_VIOLATION (*Win32Error attribute*), 289
 ERROR_CANCELLED (*Win32Error attribute*), 289
 ERROR_CANNOT_ABORT_TRANSACTIONS (*Win32Error attribute*), 289
 ERROR_CANNOT_ACCEPT_TRANSACTED_WORK (*Win32Error attribute*), 289
 ERROR_CANNOT_COPY (*Win32Error attribute*), 289
 ERROR_CANNOT_DETECT_DRIVER_FAILURE (*Win32Error attribute*), 289
 ERROR_CANNOT_DETECT_PROCESS_ABORT (*Win32Error attribute*), 289
 ERROR_CANNOT_EXECUTE_FILE_IN_TRANSACTION (*Win32Error attribute*), 289
 ERROR_CANNOT_FIND_WND_CLASS (*Win32Error attribute*), 289
 ERROR_CANNOT_IMPERSONATE (*Win32Error attribute*), 289
 ERROR_CANNOT_LOAD_REGISTRY_FILE (*Win32Error attribute*), 289
 ERROR_CANNOT_MAKE (*Win32Error attribute*), 289
 ERROR_CANNOT_OPEN_PROFILE (*Win32Error attribute*), 289
 ERROR_CANT_ACCESS_DOMAIN_INFO (*Win32Error attribute*), 289
 ERROR_CANT_ACCESS_FILE (*Win32Error attribute*), 289
 ERROR_CANT_BREAK_TRANSACTIONAL_DEPENDENCY (*Win32Error attribute*), 289
 ERROR_CANT_CREATE_MORE_STREAM_MINIVERSIONS (*Win32Error attribute*), 289
 ERROR_CANT_CROSS_RM_BOUNDARY (*Win32Error attribute*), 289
 ERROR_CANT_DELETE_LAST_ITEM (*Win32Error attribute*), 289
 ERROR_CANT_DISABLE_MANDATORY (*Win32Error attribute*), 289
 ERROR_CANT_ENABLE_DENY_ONLY (*Win32Error attribute*), 289
 ERROR_CANT_EVICT_ACTIVE_NODE (*Win32Error attribute*), 289
 ERROR_CANT_OPEN_ANONYMOUS (*Win32Error attribute*), 289
 ERROR_CANT_OPEN_MINIVERSION_WITH_MODIFY_INTENT (*Win32Error attribute*), 289
 ERROR_CANT_RECOVER_WITH_HANDLE_OPEN (*Win32Error attribute*), 289
 ERROR_CANT_RESOLVE_FILENAME (*Win32Error attribute*), 290
 ERROR_CANT_TERMINATE_SELF (*Win32Error attribute*), 290
 ERROR_CANT_WAIT (*Win32Error attribute*), 290
 ERROR_CANTFETCHBACKWARDS (*Win32Error attribute*), 289
 ERROR_CANTOPEN (*Win32Error attribute*), 289
 ERROR_CANTREAD (*Win32Error attribute*), 289
 ERROR_CANTSCROLLBACKWARDS (*Win32Error attribute*), 289
 ERROR_CANTWRITE (*Win32Error attribute*), 289
 ERROR_CARDBUS_NOT_SUPPORTED (*Win32Error attribute*), 290
 ERROR_CHECKING_FILE_SYSTEM (*Win32Error attribute*), 290
 ERROR_CHECKOUT_REQUIRED (*Win32Error attribute*), 290
 ERROR_CHILD_MUST_BE_VOLATILE (*Win32Error attribute*), 290
 ERROR_CHILD_NOT_COMPLETE (*Win32Error attribute*), 290
 ERROR_CHILD_WINDOW_MENU (*Win32Error attribute*), 290
 ERROR_CIRCULAR_DEPENDENCY (*Win32Error attribute*), 290
 ERROR_CLASS_ALREADY_EXISTS (*Win32Error attribute*), 290
 ERROR_CLASS_DOES_NOT_EXIST (*Win32Error attribute*), 290
 ERROR_CLASS_HAS_WINDOWS (*Win32Error attribute*), 290
 ERROR_CLEANER_CARTRIDGE_INSTALLED (*Win32Error attribute*), 290
 ERROR_CLEANER_CARTRIDGE_SPENT (*Win32Error attribute*), 290
 ERROR_CLEANER_SLOT_NOT_SET (*Win32Error attribute*), 290
 ERROR_CLEANER_SLOT_SET (*Win32Error attribute*), 290
 ERROR_CLIENT_SERVER_PARAMETERS_INVALID (*Win32Error attribute*), 290
 ERROR_CLIPBOARD_NOT_OPEN (*Win32Error attribute*), 290
 ERROR_CLIPPING_NOT_SUPPORTED (*Win32Error attribute*), 290
 ERROR_CLUSCFG_ALREADY_COMMITTED (*Win32Error attribute*), 290
 ERROR_CLUSCFG_ROLLBACK_FAILED (*Win32Error attribute*), 290
 ERROR_CLUSCFG_SYSTEM_DISK_DRIVE_LETTER_CONFLICT (*Win32Error attribute*), 290

(*Win32Error attribute*), 290

ERROR_CLUSTER_CANT_CREATE_DUP_CLUSTER_NAME (*Win32Error attribute*), 290

ERROR_CLUSTER_CANT_DESERIALIZE_DATA (*Win32Error attribute*), 290

ERROR_CLUSTER_DATABASE_SEQMISMATCH (*Win32Error attribute*), 290

ERROR_CLUSTER_DATABASE_TRANSACTION_IN_PROGRESS (*Win32Error attribute*), 290

ERROR_CLUSTER_DATABASE_TRANSACTION_NOT_IN_PROGRESS (*Win32Error attribute*), 290

ERROR_CLUSTER_EVICT_WITHOUT_CLEANUP (*Win32Error attribute*), 290

ERROR_CLUSTER_GROUP_MOVING (*Win32Error attribute*), 291

ERROR_CLUSTER_GUM_NOT_LOCKER (*Win32Error attribute*), 291

ERROR_CLUSTER_INCOMPATIBLE_VERSIONS (*Win32Error attribute*), 291

ERROR_CLUSTER_INSTANCE_ID_MISMATCH (*Win32Error attribute*), 291

ERROR_CLUSTER_INTERNAL_INVALID_FUNCTION (*Win32Error attribute*), 291

ERROR_CLUSTER_INVALID_IPV6_NETWORK (*Win32Error attribute*), 291

ERROR_CLUSTER_INVALID_IPV6_TUNNEL_NETWORK (*Win32Error attribute*), 291

ERROR_CLUSTER_INVALID_NETWORK (*Win32Error attribute*), 291

ERROR_CLUSTER_INVALID_NETWORK_PROVIDER (*Win32Error attribute*), 291

ERROR_CLUSTER_INVALID_NODE (*Win32Error attribute*), 291

ERROR_CLUSTER_INVALID_REQUEST (*Win32Error attribute*), 291

ERROR_CLUSTER_INVALID_STRING_FORMAT (*Win32Error attribute*), 291

ERROR_CLUSTER_INVALID_STRING_TERMINATION (*Win32Error attribute*), 291

ERROR_CLUSTER_IPADDR_IN_USE (*Win32Error attribute*), 291

ERROR_CLUSTER_JOIN_ABORTED (*Win32Error attribute*), 291

ERROR_CLUSTER_JOIN_IN_PROGRESS (*Win32Error attribute*), 291

ERROR_CLUSTER_JOIN_NOT_IN_PROGRESS (*Win32Error attribute*), 291

ERROR_CLUSTER_LAST_INTERNAL_NETWORK (*Win32Error attribute*), 291

ERROR_CLUSTER_LOCAL_NODE_NOT_FOUND (*Win32Error attribute*), 291

ERROR_CLUSTER_MAXNUM_OF_RESOURCES_EXCEEDED (*Win32Error attribute*), 291

ERROR_CLUSTER_MEMBERSHIP_HALT (*Win32Error attribute*), 291

ERROR_CLUSTER_MEMBERSHIP_INVALID_STATE (*Win32Error attribute*), 291

ERROR_CLUSTER_MISMATCHED_COMPUTER_ACCT_NAME (*Win32Error attribute*), 291

ERROR_CLUSTER_NETINTERFACE_EXISTS (*Win32Error attribute*), 291

ERROR_CLUSTER_NETINTERFACE_NOT_FOUND (*Win32Error attribute*), 291

ERROR_CLUSTER_NETWORK_ALREADY_OFFLINE (*Win32Error attribute*), 291

ERROR_CLUSTER_NETWORK_ALREADY_ONLINE (*Win32Error attribute*), 291

ERROR_CLUSTER_NETWORK_EXISTS (*Win32Error attribute*), 291

ERROR_CLUSTER_NETWORK_HAS_DEPENDENTS (*Win32Error attribute*), 291

ERROR_CLUSTER_NETWORK_NOT_FOUND (*Win32Error attribute*), 291

ERROR_CLUSTER_NETWORK_NOT_FOUND_FOR_IP (*Win32Error attribute*), 291

ERROR_CLUSTER_NETWORK_NOT_INTERNAL (*Win32Error attribute*), 291

ERROR_CLUSTER_NO_NET_ADAPTERS (*Win32Error attribute*), 292

ERROR_CLUSTER_NO_QUORUM (*Win32Error attribute*), 292

ERROR_CLUSTER_NO_RPC_PACKAGES_REGISTERED (*Win32Error attribute*), 292

ERROR_CLUSTER_NO_SECURITY_CONTEXT (*Win32Error attribute*), 292

ERROR_CLUSTER_NODE_ALREADY_DOWN (*Win32Error attribute*), 291

ERROR_CLUSTER_NODE_ALREADY_HAS_DFS_ROOT (*Win32Error attribute*), 291

ERROR_CLUSTER_NODE_ALREADY_MEMBER (*Win32Error attribute*), 291

ERROR_CLUSTER_NODE_ALREADY_UP (*Win32Error attribute*), 291

ERROR_CLUSTER_NODE_DOWN (*Win32Error attribute*), 292

ERROR_CLUSTER_NODE_EXISTS (*Win32Error attribute*), 292

ERROR_CLUSTER_NODE_NOT_FOUND (*Win32Error attribute*), 292

ERROR_CLUSTER_NODE_NOT_MEMBER (*Win32Error attribute*), 292

ERROR_CLUSTER_NODE_NOT_PAUSED (*Win32Error attribute*), 292

ERROR_CLUSTER_NODE_NOT_READY (*Win32Error attribute*), 292

ERROR_CLUSTER_NODE_PAUSED (*Win32Error attribute*), 292

ERROR_CLUSTER_NODE_SHUTTING_DOWN

(Win32Error attribute), 292
 ERROR_CLUSTER_NODE_UNREACHABLE (Win32Error attribute), 292
 ERROR_CLUSTER_NODE_UP (Win32Error attribute), 292
 ERROR_CLUSTER_NOT_INSTALLED (Win32Error attribute), 292
 ERROR_CLUSTER_NULL_DATA (Win32Error attribute), 292
 ERROR_CLUSTER_OLD_VERSION (Win32Error attribute), 292
 ERROR_CLUSTER_OWNER_NOT_IN_PREFLIST (Win32Error attribute), 292
 ERROR_CLUSTER_PARAMETER_MISMATCH (Win32Error attribute), 292
 ERROR_CLUSTER_PARAMETER_OUT_OF_BOUNDS (Win32Error attribute), 292
 ERROR_CLUSTER_PARTIAL_READ (Win32Error attribute), 292
 ERROR_CLUSTER_PARTIAL_SEND (Win32Error attribute), 292
 ERROR_CLUSTER_PARTIAL_WRITE (Win32Error attribute), 292
 ERROR_CLUSTER_POISONED (Win32Error attribute), 292
 ERROR_CLUSTER_PROPERTY_DATA_TYPE_MISMATCH (Win32Error attribute), 292
 ERROR_CLUSTER_QUORUMLOG_NOT_FOUND (Win32Error attribute), 292
 ERROR_CLUSTER_REGISTRY_INVALID_FUNCTION (Win32Error attribute), 292
 ERROR_CLUSTER_RESNAME_NOT_FOUND (Win32Error attribute), 292
 ERROR_CLUSTER_RESOURCE_TYPE_BUSY (Win32Error attribute), 292
 ERROR_CLUSTER_RESOURCE_TYPE_NOT_FOUND (Win32Error attribute), 292
 ERROR_CLUSTER_RESOURCES_MUST_BE_ONLINE_OR_OFFLINE (Win32Error attribute), 292
 ERROR_CLUSTER_RESTYPE_NOT_SUPPORTED (Win32Error attribute), 292
 ERROR_CLUSTER_RHS_FAILED_INITIALIZATION (Win32Error attribute), 292
 ERROR_CLUSTER_SHUTTING_DOWN (Win32Error attribute), 292
 ERROR_CLUSTER_SYSTEM_CONFIG_CHANGED (Win32Error attribute), 292
 ERROR_CLUSTER_WRONG_OS_VERSION (Win32Error attribute), 292
 ERROR_CLUSTERLOG_CHKPOINT_NOT_FOUND (Win32Error attribute), 290
 ERROR_CLUSTERLOG_CORRUPT (Win32Error attribute), 290
 ERROR_CLUSTERLOG_EXCEEDS_MAXSIZE (Win32Error attribute), 290
 ERROR_CLUSTERLOG_NOT_ENOUGH_SPACE (Win32Error attribute), 290
 ERROR_CLUSTERLOG_RECORD_EXCEEDS_MAXSIZE (Win32Error attribute), 290
 error_code (SensorKit attribute), 195
 ERROR_COLORSPACE_MISMATCH (Win32Error attribute), 293
 ERROR_COMMITMENT_LIMIT (Win32Error attribute), 293
 ERROR_COMMITMENT_MINIMUM (Win32Error attribute), 293
 ERROR_COMPRESSION_DISABLED (Win32Error attribute), 293
 ERROR_COMPRESSION_NOT_ALLOWED_IN_TRANSACTION (Win32Error attribute), 293
 ERROR_CONNECTED_OTHER_PASSWORD (Win32Error attribute), 293
 ERROR_CONNECTED_OTHER_PASSWORD_DEFAULT (Win32Error attribute), 293
 ERROR_CONNECTION_ABORTED (Win32Error attribute), 293
 ERROR_CONNECTION_ACTIVE (Win32Error attribute), 293
 ERROR_CONNECTION_COUNT_LIMIT (Win32Error attribute), 293
 ERROR_CONNECTION_INVALID (Win32Error attribute), 293
 ERROR_CONNECTION_REFUSED (Win32Error attribute), 293
 ERROR_CONNECTION_UNAVAIL (Win32Error attribute), 293
 ERROR_CONTEXT_EXPIRED (Win32Error attribute), 293
 ERROR_CONTINUE (Win32Error attribute), 293
 ERROR_CONTROL_C_EXIT (Win32Error attribute), 293
 ERROR_CRYPT_KEY_NOT_FOUND (Win32Error attribute), 293
 ERROR_CONTROLLER_NOT_FOUND (Win32Error attribute), 293
 ERROR_CONTROLLING_IEPORT (Win32Error attribute), 293
 ERROR_CONVERT_TO_LARGE (Win32Error attribute), 293
 ERROR_CORE_DRIVER_PACKAGE_NOT_FOUND (Win32Error attribute), 293
 ERROR_CORE_RESOURCE (Win32Error attribute), 293
 ERROR_CORRUPT_SYSTEM_FILE (Win32Error attribute), 293
 ERROR_COULD_NOT_INTERPRET (Win32Error attribute), 293
 ERROR_COULD_NOT_RESIZE_LOG (Win32Error attribute), 293
 error_count (Run attribute), 58
 error_count (Template attribute), 64

`ERROR_COUNTER_TIMEOUT` (*Win32Error attribute*), 293

`ERROR_CRASH_DUMP` (*Win32Error attribute*), 293

`ERROR_CRC` (*Win32Error attribute*), 293

`ERROR_CREATE_FAILED` (*Win32Error attribute*), 293

`ERROR_CRM_PROTOCOL_ALREADY_EXISTS` (*Win32Error attribute*), 293

`ERROR_CRM_PROTOCOL_NOT_FOUND` (*Win32Error attribute*), 293

`ERROR_CS_ENCRYPTION_EXISTING_ENCRYPTED_FILE` (*Win32Error attribute*), 293

`ERROR_CS_ENCRYPTION_FILE_NOT_CSE` (*Win32Error attribute*), 293

`ERROR_CS_ENCRYPTION_INVALID_SERVER_RESPONSE` (*Win32Error attribute*), 293

`ERROR_CS_ENCRYPTION_NEW_ENCRYPTED_FILE` (*Win32Error attribute*), 293

`ERROR_CS_ENCRYPTION_UNSUPPORTED_SERVER` (*Win32Error attribute*), 293

`ERROR_CTX_ACCOUNT_RESTRICTION` (*Win32Error attribute*), 293

`ERROR_CTX_BAD_VIDEO_MODE` (*Win32Error attribute*), 294

`ERROR_CTX_CANNOT_MAKE_EVENTLOG_ENTRY` (*Win32Error attribute*), 294

`ERROR_CTX_CDM_CONNECT` (*Win32Error attribute*), 294

`ERROR_CTX_CDM_DISCONNECT` (*Win32Error attribute*), 294

`ERROR_CTX_CLIENT_LICENSE_IN_USE` (*Win32Error attribute*), 294

`ERROR_CTX_CLIENT_LICENSE_NOT_SET` (*Win32Error attribute*), 294

`ERROR_CTX_CLIENT_QUERY_TIMEOUT` (*Win32Error attribute*), 294

`ERROR_CTX_CLOSE_PENDING` (*Win32Error attribute*), 294

`ERROR_CTX_CONSOLE_CONNECT` (*Win32Error attribute*), 294

`ERROR_CTX_CONSOLE_DISCONNECT` (*Win32Error attribute*), 294

`ERROR_CTX_ENCRYPTION_LEVEL_REQUIRED` (*Win32Error attribute*), 294

`ERROR_CTX_GRAPHICS_INVALID` (*Win32Error attribute*), 294

`ERROR_CTX_INVALID_MODEMNAME` (*Win32Error attribute*), 294

`ERROR_CTX_INVALID_PD` (*Win32Error attribute*), 294

`ERROR_CTX_INVALID_WD` (*Win32Error attribute*), 294

`ERROR_CTX_LICENSE_CLIENT_INVALID` (*Win32Error attribute*), 294

`ERROR_CTX_LICENSE_EXPIRED` (*Win32Error attribute*), 294

`ERROR_CTX_LICENSE_NOT_AVAILABLE` (*Win32Error attribute*), 294

`ERROR_CTX_LOGON_DISABLED` (*Win32Error attribute*), 294

`ERROR_CTX_MODEM_INF_NOT_FOUND` (*Win32Error attribute*), 294

`ERROR_CTX_MODEM_RESPONSE_BUSY` (*Win32Error attribute*), 294

`ERROR_CTX_MODEM_RESPONSE_ERROR` (*Win32Error attribute*), 294

`ERROR_CTX_MODEM_RESPONSE_NO_CARRIER` (*Win32Error attribute*), 294

`ERROR_CTX_MODEM_RESPONSE_NO_DIALTONE` (*Win32Error attribute*), 294

`ERROR_CTX_MODEM_RESPONSE_TIMEOUT` (*Win32Error attribute*), 294

`ERROR_CTX_MODEM_RESPONSE_VOICE` (*Win32Error attribute*), 294

`ERROR_CTX_NO_FORCE_LOGOFF` (*Win32Error attribute*), 294

`ERROR_CTX_NO_OUTBUF` (*Win32Error attribute*), 294

`ERROR_CTX_NOT_CONSOLE` (*Win32Error attribute*), 294

`ERROR_CTX_PD_NOT_FOUND` (*Win32Error attribute*), 294

`ERROR_CTX_SECURITY_LAYER_ERROR` (*Win32Error attribute*), 294

`ERROR_CTX_SERVICE_NAME_COLLISION` (*Win32Error attribute*), 294

`ERROR_CTX_SESSION_IN_USE` (*Win32Error attribute*), 294

`ERROR_CTX_SHADOW_DENIED` (*Win32Error attribute*), 294

`ERROR_CTX_SHADOW_DISABLED` (*Win32Error attribute*), 294

`ERROR_CTX_SHADOW_ENDED_BY_MODE_CHANGE` (*Win32Error attribute*), 294

`ERROR_CTX_SHADOW_INVALID` (*Win32Error attribute*), 295

`ERROR_CTX_SHADOW_NOT_RUNNING` (*Win32Error attribute*), 295

`ERROR_CTX_TD_ERROR` (*Win32Error attribute*), 295

`ERROR_CTX_WD_NOT_FOUND` (*Win32Error attribute*), 295

`ERROR_CTX_WINSTATION_ACCESS_DENIED` (*Win32Error attribute*), 295

`ERROR_CTX_WINSTATION_ALREADY_EXISTS` (*Win32Error attribute*), 295

`ERROR_CTX_WINSTATION_BUSY` (*Win32Error attribute*), 295

`ERROR_CTX_WINSTATION_NAME_INVALID` (*Win32Error attribute*), 295

`ERROR_CTX_WINSTATION_NOT_FOUND` (*Win32Error attribute*), 295

(Win32Error attribute), 295
 ERROR_CTX_WINSTATIONS_DISABLED (Win32Error attribute), 295
 ERROR_CURRENT_DIRECTORY (Win32Error attribute), 295
 ERROR_CURRENT_TRANSACTION_NOT_VALID (Win32Error attribute), 295
 ERROR_DATA_LOST_REPAIR (Win32Error attribute), 295
 ERROR_DATA_NOT_ACCEPTED (Win32Error attribute), 295
 ERROR_DATABASE_BACKUP_CORRUPT (Win32Error attribute), 295
 ERROR_DATABASE_DOES_NOT_EXIST (Win32Error attribute), 295
 ERROR_DATABASE_FAILURE (Win32Error attribute), 295
 ERROR_DATABASE_FULL (Win32Error attribute), 295
 ERROR_DATATYPE_MISMATCH (Win32Error attribute), 295
 ERROR_DBG_COMMAND_EXCEPTION (Win32Error attribute), 295
 ERROR_DBG_CONTINUE (Win32Error attribute), 295
 ERROR_DBG_CONTROL_BREAK (Win32Error attribute), 295
 ERROR_DBG_CONTROL_C (Win32Error attribute), 295
 ERROR_DBG_EXCEPTION_HANDLED (Win32Error attribute), 295
 ERROR_DBG_EXCEPTION_NOT_HANDLED (Win32Error attribute), 295
 ERROR_DBG_PRINTEXCEPTION_C (Win32Error attribute), 295
 ERROR_DBG_REPLY_LATER (Win32Error attribute), 295
 ERROR_DBG_RIPEXCEPTION (Win32Error attribute), 295
 ERROR_DBG_TERMINATE_PROCESS (Win32Error attribute), 295
 ERROR_DBG_TERMINATE_THREAD (Win32Error attribute), 295
 ERROR_DBG_UNABLE_TO_PROVIDE_HANDLE (Win32Error attribute), 295
 ERROR_DC_NOT_FOUND (Win32Error attribute), 295
 ERROR_DDE_FAIL (Win32Error attribute), 295
 ERROR_DEBUG_ATTACH_FAILED (Win32Error attribute), 295
 ERROR_DECRYPTION_FAILED (Win32Error attribute), 295
 ERROR_DELETE_PENDING (Win32Error attribute), 295
 ERROR_DELETING_ICM_XFORM (Win32Error attribute), 296
 ERROR_DEPENDENCY_ALREADY_EXISTS (Win32Error attribute), 296
 ERROR_DEPENDENCY_NOT_ALLOWED (Win32Error attribute), 296
 ERROR_DEPENDENCY_NOT_FOUND (Win32Error attribute), 296
 ERROR_DEPENDENCY_TREE_TOO_COMPLEX (Win32Error attribute), 296
 ERROR_DEPENDENT_RESOURCE_EXISTS (Win32Error attribute), 296
 ERROR_DEPENDENT_RESOURCE_PROPERTY_CONFLICT (Win32Error attribute), 296
 ERROR_DEPENDENT_SERVICES_RUNNING (Win32Error attribute), 296
 ERROR_DESTINATION_ELEMENT_FULL (Win32Error attribute), 296
 ERROR_DESTROY_OBJECT_OF_OTHER_THREAD (Win32Error attribute), 296
 ERROR_DEV_NOT_EXIST (Win32Error attribute), 296
 ERROR_DEVICE_ALREADY_ATTACHED (Win32Error attribute), 296
 ERROR_DEVICE_ALREADY_REMEMBERED (Win32Error attribute), 296
 ERROR_DEVICE_DOOR_OPEN (Win32Error attribute), 296
 ERROR_DEVICE_ENUMERATION_ERROR (Win32Error attribute), 296
 ERROR_DEVICE_IN_USE (Win32Error attribute), 296
 ERROR_DEVICE_NOT_AVAILABLE (Win32Error attribute), 296
 ERROR_DEVICE_NOT_CONNECTED (Win32Error attribute), 296
 ERROR_DEVICE_NOT_PARTITIONED (Win32Error attribute), 296
 ERROR_DEVICE_REINITIALIZATION_NEEDED (Win32Error attribute), 296
 ERROR_DEVICE_REMOVED (Win32Error attribute), 296
 ERROR_DEVICE_REQUIRES_CLEANING (Win32Error attribute), 296
 ERROR_DHCP_ADDRESS_CONFLICT (Win32Error attribute), 296
 ERROR_DIFFERENT_SERVICE_ACCOUNT (Win32Error attribute), 296
 ERROR_DIR_EFS_DISALLOWED (Win32Error attribute), 296
 ERROR_DIR_NOT_EMPTY (Win32Error attribute), 296
 ERROR_DIR_NOT_ROOT (Win32Error attribute), 296
 ERROR_DIRECT_ACCESS_HANDLE (Win32Error attribute), 296
 ERROR_DIRECTORY (Win32Error attribute), 296
 ERROR_DIRECTORY_NOT_RM (Win32Error attribute), 296
 ERROR_DISCARDED (Win32Error attribute), 296
 ERROR_DISK_CHANGE (Win32Error attribute), 296
 ERROR_DISK_CORRUPT (Win32Error attribute), 296

`ERROR_DISK_FULL` (*Win32Error attribute*), 296
`ERROR_DISK_OPERATION_FAILED` (*Win32Error attribute*), 296
`ERROR_DISK_RECALIBRATE_FAILED` (*Win32Error attribute*), 296
`ERROR_DISK_REPAIR_DISABLED` (*Win32Error attribute*), 297
`ERROR_DISK_RESET_FAILED` (*Win32Error attribute*), 297
`ERROR_DISK_TOO_FRAGMENTED` (*Win32Error attribute*), 297
`ERROR_DLL_INIT_FAILED` (*Win32Error attribute*), 297
`ERROR_DLL_INIT_FAILED_LOGOFF` (*Win32Error attribute*), 297
`ERROR_DLL_MIGHT_BE_INCOMPATIBLE` (*Win32Error attribute*), 297
`ERROR_DLL_MIGHT_BE_INSECURE` (*Win32Error attribute*), 297
`ERROR_DLL_NOT_FOUND` (*Win32Error attribute*), 297
`ERROR_DOMAIN_CONTROLLER_EXISTS` (*Win32Error attribute*), 297
`ERROR_DOMAIN_CONTROLLER_NOT_FOUND` (*Win32Error attribute*), 297
`ERROR_DOMAIN_CTRLR_CONFIG_ERROR` (*Win32Error attribute*), 297
`ERROR_DOMAIN_EXISTS` (*Win32Error attribute*), 297
`ERROR_DOMAIN_LIMIT_EXCEEDED` (*Win32Error attribute*), 297
`ERROR_DOMAIN_TRUST_INCONSISTENT` (*Win32Error attribute*), 297
`ERROR_DRIVE_LOCKED` (*Win32Error attribute*), 297
`ERROR_DRIVE_MEDIA_MISMATCH` (*Win32Error attribute*), 297
`ERROR_DRIVER_CANCEL_TIMEOUT` (*Win32Error attribute*), 297
`ERROR_DRIVER_DATABASE_ERROR` (*Win32Error attribute*), 297
`ERROR_DRIVER_FAILED_PRIOR_UNLOAD` (*Win32Error attribute*), 297
`ERROR_DRIVER_FAILED_SLEEP` (*Win32Error attribute*), 297
`ERROR_DRIVERS_LEAKING_LOCKED_PAGES` (*Win32Error attribute*), 297
`ERROR_DS_ADD_REPLICA_INHIBITED` (*DirectoryStorageError attribute*), 257
`ERROR_DS_ADD_REPLICA_INHIBITED` (*Win32Error attribute*), 297
`ERROR_DS_ADMIN_LIMIT_EXCEEDED` (*DirectoryStorageError attribute*), 257
`ERROR_DS_ADMIN_LIMIT_EXCEEDED` (*Win32Error attribute*), 297
`ERROR_DS_AFFECTS_MULTIPLE_DSAS` (*DirectoryStorageError attribute*), 257
`ERROR_DS_AFFECTS_MULTIPLE_DSAS` (*Win32Error attribute*), 297
`ERROR_DS_AG_CANT_HAVE_UNIVERSAL_MEMBER` (*DirectoryStorageError attribute*), 257
`ERROR_DS_AG_CANT_HAVE_UNIVERSAL_MEMBER` (*Win32Error attribute*), 297
`ERROR_DS_ALIAS_DEREF_PROBLEM` (*DirectoryStorageError attribute*), 257
`ERROR_DS_ALIAS_DEREF_PROBLEM` (*Win32Error attribute*), 297
`ERROR_DS_ALIAS_POINTS_TO_ALIAS` (*DirectoryStorageError attribute*), 257
`ERROR_DS_ALIAS_POINTS_TO_ALIAS` (*Win32Error attribute*), 297
`ERROR_DS_ALIAS_PROBLEM` (*DirectoryStorageError attribute*), 257
`ERROR_DS_ALIAS_PROBLEM` (*Win32Error attribute*), 297
`ERROR_DS_ALIASED_OBJ_MISSING` (*DirectoryStorageError attribute*), 257
`ERROR_DS_ALIASED_OBJ_MISSING` (*Win32Error attribute*), 297
`ERROR_DS_ATT_ALREADY_EXISTS` (*DirectoryStorageError attribute*), 257
`ERROR_DS_ATT_ALREADY_EXISTS` (*Win32Error attribute*), 297
`ERROR_DS_ATT_IS_NOT_ON_OBJ` (*DirectoryStorageError attribute*), 257
`ERROR_DS_ATT_IS_NOT_ON_OBJ` (*Win32Error attribute*), 297
`ERROR_DS_ATT_NOT_DEF_FOR_CLASS` (*DirectoryStorageError attribute*), 257
`ERROR_DS_ATT_NOT_DEF_FOR_CLASS` (*Win32Error attribute*), 297
`ERROR_DS_ATT_NOT_DEF_IN_SCHEMA` (*DirectoryStorageError attribute*), 257
`ERROR_DS_ATT_NOT_DEF_IN_SCHEMA` (*Win32Error attribute*), 297
`ERROR_DS_ATT_SCHEMA_REQ_ID` (*DirectoryStorageError attribute*), 257
`ERROR_DS_ATT_SCHEMA_REQ_ID` (*Win32Error attribute*), 298
`ERROR_DS_ATT_SCHEMA_REQ_SYNTAX` (*DirectoryStorageError attribute*), 257
`ERROR_DS_ATT_SCHEMA_REQ_SYNTAX` (*Win32Error attribute*), 298
`ERROR_DS_ATT_VAL_ALREADY_EXISTS` (*DirectoryStorageError attribute*), 257
`ERROR_DS_ATT_VAL_ALREADY_EXISTS` (*Win32Error attribute*), 298
`ERROR_DS_ATTRIBUTE_OR_VALUE_EXISTS` (*DirectoryStorageError attribute*), 257
`ERROR_DS_ATTRIBUTE_OR_VALUE_EXISTS` (*Win32Error attribute*), 297

ERROR_DS_ATTRIBUTE_OWNED_BY_SAM (*DirectoryStorageError* attribute), 257
 ERROR_DS_ATTRIBUTE_OWNED_BY_SAM (*Win32Error* attribute), 297
 ERROR_DS_ATTRIBUTE_TYPE_UNDEFINED (*DirectoryStorageError* attribute), 257
 ERROR_DS_ATTRIBUTE_TYPE_UNDEFINED (*Win32Error* attribute), 297
 ERROR_DS_AUDIT_FAILURE (*Win32Error* attribute), 298
 ERROR_DS_AUTH_METHOD_NOT_SUPPORTED (*DirectoryStorageError* attribute), 257
 ERROR_DS_AUTH_METHOD_NOT_SUPPORTED (*Win32Error* attribute), 298
 ERROR_DS_AUTH_UNKNOWN (*DirectoryStorageError* attribute), 257
 ERROR_DS_AUTH_UNKNOWN (*Win32Error* attribute), 298
 ERROR_DS_AUTHORIZATION_FAILED (*DirectoryStorageError* attribute), 257
 ERROR_DS_AUTHORIZATION_FAILED (*Win32Error* attribute), 298
 ERROR_DS_AUX_CLS_TEST_FAIL (*DirectoryStorageError* attribute), 257
 ERROR_DS_AUX_CLS_TEST_FAIL (*Win32Error* attribute), 298
 ERROR_DS_BACKLINK_WITHOUT_LINK (*DirectoryStorageError* attribute), 257
 ERROR_DS_BACKLINK_WITHOUT_LINK (*Win32Error* attribute), 298
 ERROR_DS_BAD_ATT_SCHEMA_SYNTAX (*DirectoryStorageError* attribute), 258
 ERROR_DS_BAD_ATT_SCHEMA_SYNTAX (*Win32Error* attribute), 298
 ERROR_DS_BAD_HIERARCHY_FILE (*DirectoryStorageError* attribute), 258
 ERROR_DS_BAD_HIERARCHY_FILE (*Win32Error* attribute), 298
 ERROR_DS_BAD_INSTANCE_TYPE (*DirectoryStorageError* attribute), 258
 ERROR_DS_BAD_INSTANCE_TYPE (*Win32Error* attribute), 298
 ERROR_DS_BAD_NAME_SYNTAX (*DirectoryStorageError* attribute), 258
 ERROR_DS_BAD_NAME_SYNTAX (*Win32Error* attribute), 298
 ERROR_DS_BAD_RDN_ATT_ID_SYNTAX (*DirectoryStorageError* attribute), 258
 ERROR_DS_BAD_RDN_ATT_ID_SYNTAX (*Win32Error* attribute), 298
 ERROR_DS_BUILD_HIERARCHY_TABLE_FAILED (*DirectoryStorageError* attribute), 258
 ERROR_DS_BUILD_HIERARCHY_TABLE_FAILED (*Win32Error* attribute), 298
 ERROR_DS_BUSY (*DirectoryStorageError* attribute), 258
 ERROR_DS_BUSY (*Win32Error* attribute), 298
 ERROR_DS_CANT_ACCESS_REMOTE_PART_OF_AD (*DirectoryStorageError* attribute), 258
 ERROR_DS_CANT_ACCESS_REMOTE_PART_OF_AD (*Win32Error* attribute), 298
 ERROR_DS_CANT_ADD_ATT_VALUES (*DirectoryStorageError* attribute), 258
 ERROR_DS_CANT_ADD_ATT_VALUES (*Win32Error* attribute), 298
 ERROR_DS_CANT_ADD_SYSTEM_ONLY (*DirectoryStorageError* attribute), 258
 ERROR_DS_CANT_ADD_SYSTEM_ONLY (*Win32Error* attribute), 298
 ERROR_DS_CANT_ADD_TO_GC (*DirectoryStorageError* attribute), 258
 ERROR_DS_CANT_ADD_TO_GC (*Win32Error* attribute), 298
 ERROR_DS_CANT_CACHE_ATT (*DirectoryStorageError* attribute), 258
 ERROR_DS_CANT_CACHE_ATT (*Win32Error* attribute), 298
 ERROR_DS_CANT_CACHE_CLASS (*DirectoryStorageError* attribute), 258
 ERROR_DS_CANT_CACHE_CLASS (*Win32Error* attribute), 298
 ERROR_DS_CANT_CREATE_IN_NONDOMAIN_NC (*DirectoryStorageError* attribute), 258
 ERROR_DS_CANT_CREATE_IN_NONDOMAIN_NC (*Win32Error* attribute), 298
 ERROR_DS_CANT_CREATE_UNDER_SCHEMA (*DirectoryStorageError* attribute), 258
 ERROR_DS_CANT_CREATE_UNDER_SCHEMA (*Win32Error* attribute), 298
 ERROR_DS_CANT_DEL_MASTER_CROSSREF (*DirectoryStorageError* attribute), 258
 ERROR_DS_CANT_DEL_MASTER_CROSSREF (*Win32Error* attribute), 298
 ERROR_DS_CANT_DELETE (*DirectoryStorageError* attribute), 258
 ERROR_DS_CANT_DELETE (*Win32Error* attribute), 298
 ERROR_DS_CANT_DELETE_DSA_OBJ (*DirectoryStorageError* attribute), 258
 ERROR_DS_CANT_DELETE_DSA_OBJ (*Win32Error* attribute), 298
 ERROR_DS_CANT_DEMOTE_WITH_WRITEABLE_NC (*DirectoryStorageError* attribute), 258
 ERROR_DS_CANT_DEMOTE_WITH_WRITEABLE_NC (*Win32Error* attribute), 298
 ERROR_DS_CANT_DEREF_ALIAS (*DirectoryStorageError* attribute), 258
 ERROR_DS_CANT_DEREF_ALIAS (*Win32Error* attribute), 298

tribute), 298

ERROR_DS_CANT_DERIVE_SPN_FOR_DELETED_DOMAIN (DirectoryStorageError attribute), 258

ERROR_DS_CANT_DERIVE_SPN_FOR_DELETED_DOMAIN (Win32Error attribute), 298

ERROR_DS_CANT_DERIVE_SPN_WITHOUT_SERVER_ERROR (DirectoryStorageError attribute), 258

ERROR_DS_CANT_DERIVE_SPN_WITHOUT_SERVER_ERROR (Win32Error attribute), 298

ERROR_DS_CANT_FIND_DC_FOR_SRC_DOMAIN (DirectoryStorageError attribute), 258

ERROR_DS_CANT_FIND_DC_FOR_SRC_DOMAIN (Win32Error attribute), 298

ERROR_DS_CANT_FIND_DSA_OBJ (DirectoryStorageError attribute), 258

ERROR_DS_CANT_FIND_DSA_OBJ (Win32Error attribute), 298

ERROR_DS_CANT_FIND_EXPECTED_NC (DirectoryStorageError attribute), 258

ERROR_DS_CANT_FIND_EXPECTED_NC (Win32Error attribute), 298

ERROR_DS_CANT_FIND_NC_IN_CACHE (DirectoryStorageError attribute), 258

ERROR_DS_CANT_FIND_NC_IN_CACHE (Win32Error attribute), 298

ERROR_DS_CANT_MIX_MASTER_AND_REPS (DirectoryStorageError attribute), 258

ERROR_DS_CANT_MIX_MASTER_AND_REPS (Win32Error attribute), 298

ERROR_DS_CANT_MOD_OBJ_CLASS (DirectoryStorageError attribute), 258

ERROR_DS_CANT_MOD_OBJ_CLASS (Win32Error attribute), 299

ERROR_DS_CANT_MOD_PRIMARYGROUPID (DirectoryStorageError attribute), 258

ERROR_DS_CANT_MOD_PRIMARYGROUPID (Win32Error attribute), 299

ERROR_DS_CANT_MOD_SYSTEM_ONLY (DirectoryStorageError attribute), 258

ERROR_DS_CANT_MOD_SYSTEM_ONLY (Win32Error attribute), 299

ERROR_DS_CANT_MOVE_ACCOUNT_GROUP (DirectoryStorageError attribute), 258

ERROR_DS_CANT_MOVE_ACCOUNT_GROUP (Win32Error attribute), 299

ERROR_DS_CANT_MOVE_APP_BASIC_GROUP (DirectoryStorageError attribute), 258

ERROR_DS_CANT_MOVE_APP_BASIC_GROUP (Win32Error attribute), 299

ERROR_DS_CANT_MOVE_APP_QUERY_GROUP (DirectoryStorageError attribute), 258

ERROR_DS_CANT_MOVE_APP_QUERY_GROUP (Win32Error attribute), 299

ERROR_DS_CANT_MOVE_DELETED_OBJECT (DirectoryStorageError attribute), 258

ERROR_DS_CANT_MOVE_DELETED_OBJECT (Win32Error attribute), 299

ERROR_DS_CANT_MOVE_RESOURCE_GROUP (DirectoryStorageError attribute), 258

ERROR_DS_CANT_MOVE_RESOURCE_GROUP (Win32Error attribute), 299

ERROR_DS_CANT_ON_NON_LEAF (DirectoryStorageError attribute), 258

ERROR_DS_CANT_ON_NON_LEAF (Win32Error attribute), 299

ERROR_DS_CANT_ON_RDN (DirectoryStorageError attribute), 259

ERROR_DS_CANT_ON_RDN (Win32Error attribute), 299

ERROR_DS_CANT_REM_MISSING_ATT (DirectoryStorageError attribute), 259

ERROR_DS_CANT_REM_MISSING_ATT (Win32Error attribute), 299

ERROR_DS_CANT_REM_MISSING_ATT_VAL (DirectoryStorageError attribute), 259

ERROR_DS_CANT_REM_MISSING_ATT_VAL (Win32Error attribute), 299

ERROR_DS_CANT_REMOVE_ATT_CACHE (DirectoryStorageError attribute), 259

ERROR_DS_CANT_REMOVE_ATT_CACHE (Win32Error attribute), 299

ERROR_DS_CANT_REMOVE_CLASS_CACHE (DirectoryStorageError attribute), 259

ERROR_DS_CANT_REMOVE_CLASS_CACHE (Win32Error attribute), 299

ERROR_DS_CANT_REPLACE_HIDDEN_REC (DirectoryStorageError attribute), 259

ERROR_DS_CANT_REPLACE_HIDDEN_REC (Win32Error attribute), 299

ERROR_DS_CANT_RETRIEVE_ATTS (DirectoryStorageError attribute), 259

ERROR_DS_CANT_RETRIEVE_ATTS (Win32Error attribute), 299

ERROR_DS_CANT_RETRIEVE_CHILD (DirectoryStorageError attribute), 259

ERROR_DS_CANT_RETRIEVE_CHILD (Win32Error attribute), 299

ERROR_DS_CANT_RETRIEVE_DN (DirectoryStorageError attribute), 259

ERROR_DS_CANT_RETRIEVE_DN (Win32Error attribute), 299

ERROR_DS_CANT_RETRIEVE_INSTANCE (DirectoryStorageError attribute), 259

ERROR_DS_CANT_RETRIEVE_INSTANCE (Win32Error attribute), 299

ERROR_DS_CANT_RETRIEVE_SD (DirectoryStorageError attribute), 259

ERROR_DS_CANT_RETRIEVE_SD (Win32Error attribute), 299

- tribute*), 299
- ERROR_DS_CANT_START (*DirectoryStorageError attribute*), 259
- ERROR_DS_CANT_START (*Win32Error attribute*), 299
- ERROR_DS_CANT_TREE_DELETE_CRITICAL_OBJ (*DirectoryStorageError attribute*), 259
- ERROR_DS_CANT_TREE_DELETE_CRITICAL_OBJ (*Win32Error attribute*), 299
- ERROR_DS_CANT_WITH_ACCT_GROUP_MEMBERSHPS (*DirectoryStorageError attribute*), 259
- ERROR_DS_CANT_WITH_ACCT_GROUP_MEMBERSHPS (*Win32Error attribute*), 299
- ERROR_DS_CHILDREN_EXIST (*DirectoryStorageError attribute*), 259
- ERROR_DS_CHILDREN_EXIST (*Win32Error attribute*), 299
- ERROR_DS_CLASS_MUST_BE_CONCRETE (*DirectoryStorageError attribute*), 259
- ERROR_DS_CLASS_MUST_BE_CONCRETE (*Win32Error attribute*), 299
- ERROR_DS_CLASS_NOT_DSA (*DirectoryStorageError attribute*), 259
- ERROR_DS_CLASS_NOT_DSA (*Win32Error attribute*), 299
- ERROR_DS_CLIENT_LOOP (*DirectoryStorageError attribute*), 259
- ERROR_DS_CLIENT_LOOP (*Win32Error attribute*), 299
- ERROR_DS_CODE_INCONSISTENCY (*DirectoryStorageError attribute*), 259
- ERROR_DS_CODE_INCONSISTENCY (*Win32Error attribute*), 299
- ERROR_DS_COMPARE_FALSE (*DirectoryStorageError attribute*), 259
- ERROR_DS_COMPARE_FALSE (*Win32Error attribute*), 299
- ERROR_DS_COMPARE_TRUE (*DirectoryStorageError attribute*), 259
- ERROR_DS_COMPARE_TRUE (*Win32Error attribute*), 299
- ERROR_DS_CONFIDENTIALITY_REQUIRED (*DirectoryStorageError attribute*), 259
- ERROR_DS_CONFIDENTIALITY_REQUIRED (*Win32Error attribute*), 299
- ERROR_DS_CONFIG_PARAM_MISSING (*DirectoryStorageError attribute*), 259
- ERROR_DS_CONFIG_PARAM_MISSING (*Win32Error attribute*), 299
- ERROR_DS_CONSTRAINT_VIOLATION (*DirectoryStorageError attribute*), 259
- ERROR_DS_CONSTRAINT_VIOLATION (*Win32Error attribute*), 299
- ERROR_DS_CONSTRUCTED_ATT_MOD (*DirectoryStorageError attribute*), 259
- ERROR_DS_CONSTRUCTED_ATT_MOD (*Win32Error attribute*), 299
- ERROR_DS_CONTROL_NOT_FOUND (*DirectoryStorageError attribute*), 259
- ERROR_DS_CONTROL_NOT_FOUND (*Win32Error attribute*), 299
- ERROR_DS_COULDNT_CONTACT_FSMO (*DirectoryStorageError attribute*), 259
- ERROR_DS_COULDNT_CONTACT_FSMO (*Win32Error attribute*), 299
- ERROR_DS_COULDNT_IDENTIFY_OBJECTS_FOR_TREE_DELETE (*DirectoryStorageError attribute*), 259
- ERROR_DS_COULDNT_IDENTIFY_OBJECTS_FOR_TREE_DELETE (*Win32Error attribute*), 300
- ERROR_DS_COULDNT_LOCK_TREE_FOR_DELETE (*DirectoryStorageError attribute*), 259
- ERROR_DS_COULDNT_LOCK_TREE_FOR_DELETE (*Win32Error attribute*), 300
- ERROR_DS_COULDNT_UPDATE_SPNS (*DirectoryStorageError attribute*), 259
- ERROR_DS_COULDNT_UPDATE_SPNS (*Win32Error attribute*), 300
- ERROR_DS_COUNTING_AB_INDICES_FAILED (*DirectoryStorageError attribute*), 259
- ERROR_DS_COUNTING_AB_INDICES_FAILED (*Win32Error attribute*), 300
- ERROR_DS_CR_IMPOSSIBLE_TO_VALIDATE (*DirectoryStorageError attribute*), 260
- ERROR_DS_CR_IMPOSSIBLE_TO_VALIDATE (*Win32Error attribute*), 300
- ERROR_DS_CR_IMPOSSIBLE_TO_VALIDATE_V2 (*DirectoryStorageError attribute*), 260
- ERROR_DS_CR_IMPOSSIBLE_TO_VALIDATE_V2 (*Win32Error attribute*), 300
- ERROR_DS_CROSS_DOM_MOVE_ERROR (*DirectoryStorageError attribute*), 259
- ERROR_DS_CROSS_DOM_MOVE_ERROR (*Win32Error attribute*), 300
- ERROR_DS_CROSS_DOMAIN_CLEANUP_REQD (*DirectoryStorageError attribute*), 259
- ERROR_DS_CROSS_DOMAIN_CLEANUP_REQD (*Win32Error attribute*), 300
- ERROR_DS_CROSS_NC_DN_RENAME (*DirectoryStorageError attribute*), 259
- ERROR_DS_CROSS_NC_DN_RENAME (*Win32Error attribute*), 300
- ERROR_DS_CROSS_REF_BUSY (*DirectoryStorageError attribute*), 259
- ERROR_DS_CROSS_REF_BUSY (*Win32Error attribute*), 300
- ERROR_DS_CROSS_REF_EXISTS (*DirectoryStorageError attribute*), 259
- ERROR_DS_CROSS_REF_EXISTS (*Win32Error attribute*), 300

`ERROR_DS_DATABASE_ERROR` (*DirectoryStorageError attribute*), 260

`ERROR_DS_DATABASE_ERROR` (*Win32Error attribute*), 300

`ERROR_DS_DECODING_ERROR` (*DirectoryStorageError attribute*), 260

`ERROR_DS_DECODING_ERROR` (*Win32Error attribute*), 300

`ERROR_DS_DESTINATION_AUDITING_NOT_ENABLED` (*DirectoryStorageError attribute*), 260

`ERROR_DS_DESTINATION_AUDITING_NOT_ENABLED` (*Win32Error attribute*), 300

`ERROR_DS_DESTINATION_DOMAIN_NOT_IN_FOREST` (*DirectoryStorageError attribute*), 260

`ERROR_DS_DESTINATION_DOMAIN_NOT_IN_FOREST` (*Win32Error attribute*), 300

`ERROR_DS_DIFFERENT_REPL_EPOCHS` (*DirectoryStorageError attribute*), 260

`ERROR_DS_DIFFERENT_REPL_EPOCHS` (*Win32Error attribute*), 300

`ERROR_DS_DISALLOWED_IN_SYSTEM_CONTAINER` (*DirectoryStorageError attribute*), 260

`ERROR_DS_DISALLOWED_IN_SYSTEM_CONTAINER` (*Win32Error attribute*), 300

`ERROR_DS_DNS_LOOKUP_FAILURE` (*DirectoryStorageError attribute*), 260

`ERROR_DS_DNS_LOOKUP_FAILURE` (*Win32Error attribute*), 300

`ERROR_DS_DOMAIN_RENAME_IN_PROGRESS` (*DirectoryStorageError attribute*), 260

`ERROR_DS_DOMAIN_RENAME_IN_PROGRESS` (*Win32Error attribute*), 300

`ERROR_DS_DOMAIN_VERSION_TOO_HIGH` (*DirectoryStorageError attribute*), 260

`ERROR_DS_DOMAIN_VERSION_TOO_HIGH` (*Win32Error attribute*), 300

`ERROR_DS_DOMAIN_VERSION_TOO_LOW` (*DirectoryStorageError attribute*), 260

`ERROR_DS_DOMAIN_VERSION_TOO_LOW` (*Win32Error attribute*), 300

`ERROR_DS_DRA_ABANDON_SYNC` (*DirectoryStorageError attribute*), 260

`ERROR_DS_DRA_ABANDON_SYNC` (*Win32Error attribute*), 300

`ERROR_DS_DRA_ACCESS_DENIED` (*DirectoryStorageError attribute*), 260

`ERROR_DS_DRA_ACCESS_DENIED` (*Win32Error attribute*), 300

`ERROR_DS_DRA_BAD_DN` (*DirectoryStorageError attribute*), 260

`ERROR_DS_DRA_BAD_DN` (*Win32Error attribute*), 300

`ERROR_DS_DRA_BAD_INSTANCE_TYPE` (*DirectoryStorageError attribute*), 260

`ERROR_DS_DRA_BAD_INSTANCE_TYPE` (*Win32Error attribute*), 300

`ERROR_DS_DRA_BAD_NC` (*DirectoryStorageError attribute*), 260

`ERROR_DS_DRA_BAD_NC` (*Win32Error attribute*), 300

`ERROR_DS_DRA_BUSY` (*DirectoryStorageError attribute*), 260

`ERROR_DS_DRA_BUSY` (*Win32Error attribute*), 300

`ERROR_DS_DRA_CONNECTION_FAILED` (*DirectoryStorageError attribute*), 260

`ERROR_DS_DRA_CONNECTION_FAILED` (*Win32Error attribute*), 300

`ERROR_DS_DRA_DB_ERROR` (*DirectoryStorageError attribute*), 260

`ERROR_DS_DRA_DB_ERROR` (*Win32Error attribute*), 300

`ERROR_DS_DRA_DN_EXISTS` (*DirectoryStorageError attribute*), 260

`ERROR_DS_DRA_DN_EXISTS` (*Win32Error attribute*), 300

`ERROR_DS_DRA_EARLIER_SCHEMA_CONFLICT` (*DirectoryStorageError attribute*), 260

`ERROR_DS_DRA_EARLIER_SCHEMA_CONFLICT` (*Win32Error attribute*), 300

`ERROR_DS_DRA_EXTN_CONNECTION_FAILED` (*DirectoryStorageError attribute*), 260

`ERROR_DS_DRA_EXTN_CONNECTION_FAILED` (*Win32Error attribute*), 300

`ERROR_DS_DRA_GENERIC` (*DirectoryStorageError attribute*), 260

`ERROR_DS_DRA_GENERIC` (*Win32Error attribute*), 300

`ERROR_DS_DRA_INCOMPATIBLE_PARTIAL_SET` (*DirectoryStorageError attribute*), 260

`ERROR_DS_DRA_INCOMPATIBLE_PARTIAL_SET` (*Win32Error attribute*), 300

`ERROR_DS_DRA_INCONSISTENT_DIT` (*DirectoryStorageError attribute*), 260

`ERROR_DS_DRA_INCONSISTENT_DIT` (*Win32Error attribute*), 300

`ERROR_DS_DRA_INTERNAL_ERROR` (*DirectoryStorageError attribute*), 260

`ERROR_DS_DRA_INTERNAL_ERROR` (*Win32Error attribute*), 300

`ERROR_DS_DRA_INVALID_PARAMETER` (*DirectoryStorageError attribute*), 260

`ERROR_DS_DRA_INVALID_PARAMETER` (*Win32Error attribute*), 301

`ERROR_DS_DRA_MAIL_PROBLEM` (*DirectoryStorageError attribute*), 260

`ERROR_DS_DRA_MAIL_PROBLEM` (*Win32Error attribute*), 301

`ERROR_DS_DRA_MISSING_PARENT` (*DirectoryStorageError attribute*), 260

`ERROR_DS_DRA_MISSING_PARENT` (*Win32Error attribute*), 300

tribute), 301

ERROR_DS_DRA_NAME_COLLISION (*DirectoryStorageError attribute*), 260

ERROR_DS_DRA_NAME_COLLISION (*Win32Error attribute*), 301

ERROR_DS_DRA_NO_REPLICA (*DirectoryStorageError attribute*), 260

ERROR_DS_DRA_NO_REPLICA (*Win32Error attribute*), 301

ERROR_DS_DRA_NOT_SUPPORTED (*DirectoryStorageError attribute*), 260

ERROR_DS_DRA_NOT_SUPPORTED (*Win32Error attribute*), 301

ERROR_DS_DRA_OBJ_IS_REP_SOURCE (*DirectoryStorageError attribute*), 260

ERROR_DS_DRA_OBJ_IS_REP_SOURCE (*Win32Error attribute*), 301

ERROR_DS_DRA_OBJ_NC_MISMATCH (*DirectoryStorageError attribute*), 260

ERROR_DS_DRA_OBJ_NC_MISMATCH (*Win32Error attribute*), 301

ERROR_DS_DRA_OUT_OF_MEM (*DirectoryStorageError attribute*), 260

ERROR_DS_DRA_OUT_OF_MEM (*Win32Error attribute*), 301

ERROR_DS_DRA_OUT_SCHEDULE_WINDOW (*DirectoryStorageError attribute*), 261

ERROR_DS_DRA_OUT_SCHEDULE_WINDOW (*Win32Error attribute*), 301

ERROR_DS_DRA_PREEMPTED (*DirectoryStorageError attribute*), 261

ERROR_DS_DRA_PREEMPTED (*Win32Error attribute*), 301

ERROR_DS_DRA_REF_ALREADY_EXISTS (*DirectoryStorageError attribute*), 261

ERROR_DS_DRA_REF_ALREADY_EXISTS (*Win32Error attribute*), 301

ERROR_DS_DRA_REF_NOT_FOUND (*DirectoryStorageError attribute*), 261

ERROR_DS_DRA_REF_NOT_FOUND (*Win32Error attribute*), 301

ERROR_DS_DRA_REPL_PENDING (*DirectoryStorageError attribute*), 261

ERROR_DS_DRA_REPL_PENDING (*Win32Error attribute*), 301

ERROR_DS_DRA_RPC_CANCELLED (*DirectoryStorageError attribute*), 261

ERROR_DS_DRA_RPC_CANCELLED (*Win32Error attribute*), 301

ERROR_DS_DRA_SCHEMA_CONFLICT (*DirectoryStorageError attribute*), 261

ERROR_DS_DRA_SCHEMA_CONFLICT (*Win32Error attribute*), 301

ERROR_DS_DRA_SCHEMA_INFO_SHIP (*DirectoryStorageError attribute*), 261

ERROR_DS_DRA_SCHEMA_INFO_SHIP (*Win32Error attribute*), 301

ERROR_DS_DRA_SCHEMA_MISMATCH (*DirectoryStorageError attribute*), 261

ERROR_DS_DRA_SCHEMA_MISMATCH (*Win32Error attribute*), 301

ERROR_DS_DRA_SHUTDOWN (*DirectoryStorageError attribute*), 261

ERROR_DS_DRA_SHUTDOWN (*Win32Error attribute*), 301

ERROR_DS_DRA_SINK_DISABLED (*DirectoryStorageError attribute*), 261

ERROR_DS_DRA_SINK_DISABLED (*Win32Error attribute*), 301

ERROR_DS_DRA_SOURCE_DISABLED (*DirectoryStorageError attribute*), 261

ERROR_DS_DRA_SOURCE_DISABLED (*Win32Error attribute*), 301

ERROR_DS_DRA_SOURCE_IS_PARTIAL_REPLICA (*DirectoryStorageError attribute*), 261

ERROR_DS_DRA_SOURCE_IS_PARTIAL_REPLICA (*Win32Error attribute*), 301

ERROR_DS_DRA_SOURCE_REINSTALLED (*DirectoryStorageError attribute*), 261

ERROR_DS_DRA_SOURCE_REINSTALLED (*Win32Error attribute*), 301

ERROR_DS_DRS_EXTENSIONS_CHANGED (*DirectoryStorageError attribute*), 261

ERROR_DS_DRS_EXTENSIONS_CHANGED (*Win32Error attribute*), 301

ERROR_DS_DS_REQUIRED (*DirectoryStorageError attribute*), 261

ERROR_DS_DS_REQUIRED (*Win32Error attribute*), 301

ERROR_DS_DSA_MUST_BE_INT_MASTER (*DirectoryStorageError attribute*), 261

ERROR_DS_DSA_MUST_BE_INT_MASTER (*Win32Error attribute*), 301

ERROR_DS_DST_DOMAIN_NOT_NATIVE (*DirectoryStorageError attribute*), 261

ERROR_DS_DST_DOMAIN_NOT_NATIVE (*Win32Error attribute*), 301

ERROR_DS_DST_NC_MISMATCH (*DirectoryStorageError attribute*), 261

ERROR_DS_DST_NC_MISMATCH (*Win32Error attribute*), 301

ERROR_DS_DUP_LDAP_DISPLAY_NAME (*DirectoryStorageError attribute*), 261

ERROR_DS_DUP_LDAP_DISPLAY_NAME (*Win32Error attribute*), 301

ERROR_DS_DUP_LINK_ID (*DirectoryStorageError attribute*), 261

ERROR_DS_DUP_LINK_ID (*Win32Error attribute*), 301

301
ERROR_DS_DUP_MAPI_ID (*DirectoryStorageError* attribute), 261
ERROR_DS_DUP_MAPI_ID (*Win32Error* attribute), 301
ERROR_DS_DUP_MSDS_INTID (*DirectoryStorageError* attribute), 261
ERROR_DS_DUP_MSDS_INTID (*Win32Error* attribute), 301
ERROR_DS_DUP_OID (*DirectoryStorageError* attribute), 261
ERROR_DS_DUP_OID (*Win32Error* attribute), 301
ERROR_DS_DUP_RDN (*DirectoryStorageError* attribute), 261
ERROR_DS_DUP_RDN (*Win32Error* attribute), 301
ERROR_DS_DUP_SCHEMA_ID_GUID (*DirectoryStorageError* attribute), 261
ERROR_DS_DUP_SCHEMA_ID_GUID (*Win32Error* attribute), 301
ERROR_DS_DUPLICATE_ID_FOUND (*DirectoryStorageError* attribute), 261
ERROR_DS_DUPLICATE_ID_FOUND (*Win32Error* attribute), 301
ERROR_DS_ENCODING_ERROR (*DirectoryStorageError* attribute), 261
ERROR_DS_ENCODING_ERROR (*Win32Error* attribute), 302
ERROR_DS_EPOCH_MISMATCH (*DirectoryStorageError* attribute), 261
ERROR_DS_EPOCH_MISMATCH (*Win32Error* attribute), 302
ERROR_DS_EXISTING_AD_CHILD_NC (*DirectoryStorageError* attribute), 261
ERROR_DS_EXISTING_AD_CHILD_NC (*Win32Error* attribute), 302
ERROR_DS_EXISTS_IN_AUX_CLS (*DirectoryStorageError* attribute), 261
ERROR_DS_EXISTS_IN_AUX_CLS (*Win32Error* attribute), 302
ERROR_DS_EXISTS_IN_MAY_HAVE (*DirectoryStorageError* attribute), 261
ERROR_DS_EXISTS_IN_MAY_HAVE (*Win32Error* attribute), 302
ERROR_DS_EXISTS_IN_MUST_HAVE (*DirectoryStorageError* attribute), 261
ERROR_DS_EXISTS_IN_MUST_HAVE (*Win32Error* attribute), 302
ERROR_DS_EXISTS_IN_POSS_SUP (*DirectoryStorageError* attribute), 261
ERROR_DS_EXISTS_IN_POSS_SUP (*Win32Error* attribute), 302
ERROR_DS_EXISTS_IN_RDNATTID (*DirectoryStorageError* attribute), 261
ERROR_DS_EXISTS_IN_RDNATTID (*Win32Error* attribute), 302
ERROR_DS_EXISTS_IN_SUB_CLS (*DirectoryStorageError* attribute), 261
ERROR_DS_EXISTS_IN_SUB_CLS (*Win32Error* attribute), 302
ERROR_DS_FILTER_UNKNOWN (*DirectoryStorageError* attribute), 262
ERROR_DS_FILTER_UNKNOWN (*Win32Error* attribute), 302
ERROR_DS_FILTER_USES_CONSTRUCTED_ATTRS (*DirectoryStorageError* attribute), 262
ERROR_DS_FILTER_USES_CONSTRUCTED_ATTRS (*Win32Error* attribute), 302
ERROR_DS_FOREST_VERSION_TOO_HIGH (*DirectoryStorageError* attribute), 262
ERROR_DS_FOREST_VERSION_TOO_HIGH (*Win32Error* attribute), 302
ERROR_DS_FOREST_VERSION_TOO_LOW (*DirectoryStorageError* attribute), 262
ERROR_DS_FOREST_VERSION_TOO_LOW (*Win32Error* attribute), 302
ERROR_DS_GC_NOT_AVAILABLE (*DirectoryStorageError* attribute), 262
ERROR_DS_GC_NOT_AVAILABLE (*Win32Error* attribute), 302
ERROR_DS_GC_REQUIRED (*DirectoryStorageError* attribute), 262
ERROR_DS_GC_REQUIRED (*Win32Error* attribute), 302
ERROR_DS_GCVERIFY_ERROR (*DirectoryStorageError* attribute), 262
ERROR_DS_GCVERIFY_ERROR (*Win32Error* attribute), 302
ERROR_DS_GENERIC_ERROR (*DirectoryStorageError* attribute), 262
ERROR_DS_GENERIC_ERROR (*Win32Error* attribute), 302
ERROR_DS_GLOBAL_CANT_HAVE_CROSSDOMAIN_MEMBER (*DirectoryStorageError* attribute), 262
ERROR_DS_GLOBAL_CANT_HAVE_CROSSDOMAIN_MEMBER (*Win32Error* attribute), 302
ERROR_DS_GLOBAL_CANT_HAVE_LOCAL_MEMBER (*DirectoryStorageError* attribute), 262
ERROR_DS_GLOBAL_CANT_HAVE_LOCAL_MEMBER (*Win32Error* attribute), 302
ERROR_DS_GLOBAL_CANT_HAVE_UNIVERSAL_MEMBER (*DirectoryStorageError* attribute), 262
ERROR_DS_GLOBAL_CANT_HAVE_UNIVERSAL_MEMBER (*Win32Error* attribute), 302
ERROR_DS_GOVERNSID_MISSING (*DirectoryStorageError* attribute), 262
ERROR_DS_GOVERNSID_MISSING (*Win32Error* attribute), 302
ERROR_DS_GROUP_CONVERSION_ERROR (*DirectoryStorageError* attribute), 262
ERROR_DS_GROUP_CONVERSION_ERROR (*Win32Error* attribute), 302

<i>ryStorageError attribute</i>), 262	<i>torageError attribute</i>), 262
ERROR_DS_GROUP_CONVERSION_ERROR (<i>Win32Error attribute</i>), 302	ERROR_DS_INIT_FAILURE_CONSOLE (<i>Win32Error attribute</i>), 302
ERROR_DS_HAVE_PRIMARY_MEMBERS (<i>DirectoryStorageError attribute</i>), 262	ERROR_DS_INSTALL_NO_SCH_VERSION_IN_INIFILE (<i>DirectoryStorageError attribute</i>), 262
ERROR_DS_HAVE_PRIMARY_MEMBERS (<i>Win32Error attribute</i>), 302	ERROR_DS_INSTALL_NO_SCH_VERSION_IN_INIFILE (<i>Win32Error attribute</i>), 303
ERROR_DS_HIERARCHY_TABLE_MALLOC_FAILED (<i>DirectoryStorageError attribute</i>), 262	ERROR_DS_INSTALL_NO_SRC_SCH_VERSION (<i>DirectoryStorageError attribute</i>), 262
ERROR_DS_HIERARCHY_TABLE_MALLOC_FAILED (<i>Win32Error attribute</i>), 302	ERROR_DS_INSTALL_NO_SRC_SCH_VERSION (<i>Win32Error attribute</i>), 303
ERROR_DS_HIERARCHY_TABLE_TOO_DEEP (<i>Win32Error attribute</i>), 302	ERROR_DS_INSTALL_SCHEMA_MISMATCH (<i>DirectoryStorageError attribute</i>), 262
ERROR_DS_ILLEGAL_BASE_SCHEMA_MOD (<i>DirectoryStorageError attribute</i>), 262	ERROR_DS_INSTALL_SCHEMA_MISMATCH (<i>Win32Error attribute</i>), 303
ERROR_DS_ILLEGAL_BASE_SCHEMA_MOD (<i>Win32Error attribute</i>), 302	ERROR_DS_INSUFF_ACCESS_RIGHTS (<i>DirectoryStorageError attribute</i>), 262
ERROR_DS_ILLEGAL_MOD_OPERATION (<i>DirectoryStorageError attribute</i>), 262	ERROR_DS_INSUFF_ACCESS_RIGHTS (<i>Win32Error attribute</i>), 303
ERROR_DS_ILLEGAL_MOD_OPERATION (<i>Win32Error attribute</i>), 302	ERROR_DS_INSUFFICIENT_ATTR_TO_CREATE_OBJECT (<i>DirectoryStorageError attribute</i>), 262
ERROR_DS_ILLEGAL_SUPERIOR (<i>DirectoryStorageError attribute</i>), 262	ERROR_DS_INSUFFICIENT_ATTR_TO_CREATE_OBJECT (<i>Win32Error attribute</i>), 303
ERROR_DS_ILLEGAL_SUPERIOR (<i>Win32Error attribute</i>), 302	ERROR_DS_INTERNAL_FAILURE (<i>DirectoryStorageError attribute</i>), 262
ERROR_DS_ILLEGAL_XDOM_MOVE_OPERATION (<i>DirectoryStorageError attribute</i>), 262	ERROR_DS_INTERNAL_FAILURE (<i>Win32Error attribute</i>), 303
ERROR_DS_ILLEGAL_XDOM_MOVE_OPERATION (<i>Win32Error attribute</i>), 302	ERROR_DS_INVALID_ATTRIBUTE_SYNTAX (<i>DirectoryStorageError attribute</i>), 262
ERROR_DS_INAPPROPRIATE_AUTH (<i>DirectoryStorageError attribute</i>), 262	ERROR_DS_INVALID_ATTRIBUTE_SYNTAX (<i>Win32Error attribute</i>), 303
ERROR_DS_INAPPROPRIATE_AUTH (<i>Win32Error attribute</i>), 302	ERROR_DS_INVALID_DMD (<i>DirectoryStorageError attribute</i>), 262
ERROR_DS_INAPPROPRIATE_MATCHING (<i>DirectoryStorageError attribute</i>), 262	ERROR_DS_INVALID_DMD (<i>Win32Error attribute</i>), 303
ERROR_DS_INAPPROPRIATE_MATCHING (<i>Win32Error attribute</i>), 302	ERROR_DS_INVALID_DN_SYNTAX (<i>DirectoryStorageError attribute</i>), 262
ERROR_DS_INCOMPATIBLE_CONTROLS_USED (<i>DirectoryStorageError attribute</i>), 262	ERROR_DS_INVALID_DN_SYNTAX (<i>Win32Error attribute</i>), 303
ERROR_DS_INCOMPATIBLE_CONTROLS_USED (<i>Win32Error attribute</i>), 302	ERROR_DS_INVALID_GROUP_TYPE (<i>DirectoryStorageError attribute</i>), 262
ERROR_DS_INCOMPATIBLE_VERSION (<i>DirectoryStorageError attribute</i>), 262	ERROR_DS_INVALID_GROUP_TYPE (<i>Win32Error attribute</i>), 303
ERROR_DS_INCOMPATIBLE_VERSION (<i>Win32Error attribute</i>), 302	ERROR_DS_INVALID_LDAP_DISPLAY_NAME (<i>DirectoryStorageError attribute</i>), 263
ERROR_DS_INCORRECT_ROLE_OWNER (<i>DirectoryStorageError attribute</i>), 262	ERROR_DS_INVALID_LDAP_DISPLAY_NAME (<i>Win32Error attribute</i>), 303
ERROR_DS_INCORRECT_ROLE_OWNER (<i>Win32Error attribute</i>), 302	ERROR_DS_INVALID_NAME_FOR_SPN (<i>DirectoryStorageError attribute</i>), 263
ERROR_DS_INIT_FAILURE (<i>DirectoryStorageError attribute</i>), 262	ERROR_DS_INVALID_NAME_FOR_SPN (<i>Win32Error attribute</i>), 303
ERROR_DS_INIT_FAILURE (<i>Win32Error attribute</i>), 302	ERROR_DS_INVALID_ROLE_OWNER (<i>DirectoryStorageError attribute</i>), 263
ERROR_DS_INIT_FAILURE_CONSOLE (<i>DirectoryStorageError attribute</i>), 262	ERROR_DS_INVALID_ROLE_OWNER (<i>Win32Error attribute</i>), 303

tribute), 303

ERROR_DS_INVALID_SCRIPT (*DirectoryStorageError attribute*), 263

ERROR_DS_INVALID_SCRIPT (*Win32Error attribute*), 303

ERROR_DS_INVALID_SEARCH_FLAG (*DirectoryStorageError attribute*), 263

ERROR_DS_INVALID_SEARCH_FLAG (*Win32Error attribute*), 303

ERROR_DS_INVALID_SEARCH_FLAG_SUBTREE (*Win32Error attribute*), 303

ERROR_DS_INVALID_SEARCH_FLAG_TUPLE (*Win32Error attribute*), 303

ERROR_DS_IS_LEAF (*DirectoryStorageError attribute*), 263

ERROR_DS_IS_LEAF (*Win32Error attribute*), 303

ERROR_DS_KEY_NOT_UNIQUE (*DirectoryStorageError attribute*), 263

ERROR_DS_KEY_NOT_UNIQUE (*Win32Error attribute*), 303

ERROR_DS_LDAP_SEND_QUEUE_FULL (*DirectoryStorageError attribute*), 263

ERROR_DS_LDAP_SEND_QUEUE_FULL (*Win32Error attribute*), 303

ERROR_DS_LINK_ID_NOT_AVAILABLE (*DirectoryStorageError attribute*), 263

ERROR_DS_LINK_ID_NOT_AVAILABLE (*Win32Error attribute*), 303

ERROR_DS_LOCAL_CANT_HAVE_CROSSDOMAIN_LOCAL_MEMBERSHIP (*DirectoryStorageError attribute*), 263

ERROR_DS_LOCAL_CANT_HAVE_CROSSDOMAIN_LOCAL_MEMBERSHIP (*Win32Error attribute*), 303

ERROR_DS_LOCAL_ERROR (*DirectoryStorageError attribute*), 263

ERROR_DS_LOCAL_ERROR (*Win32Error attribute*), 303

ERROR_DS_LOCAL_MEMBER_OF_LOCAL_ONLY (*DirectoryStorageError attribute*), 263

ERROR_DS_LOCAL_MEMBER_OF_LOCAL_ONLY (*Win32Error attribute*), 303

ERROR_DS_LOOP_DETECT (*DirectoryStorageError attribute*), 263

ERROR_DS_LOOP_DETECT (*Win32Error attribute*), 303

ERROR_DS_LOW_DSA_VERSION (*DirectoryStorageError attribute*), 263

ERROR_DS_LOW_DSA_VERSION (*Win32Error attribute*), 303

ERROR_DS_MACHINE_ACCOUNT_CREATED_PRENT4 (*DirectoryStorageError attribute*), 263

ERROR_DS_MACHINE_ACCOUNT_CREATED_PRENT4 (*Win32Error attribute*), 303

ERROR_DS_MACHINE_ACCOUNT_QUOTA_EXCEEDED (*DirectoryStorageError attribute*), 263

ERROR_DS_MACHINE_ACCOUNT_QUOTA_EXCEEDED (*Win32Error attribute*), 303

ERROR_DS_MASTERDSA_REQUIRED (*DirectoryStorageError attribute*), 263

ERROR_DS_MASTERDSA_REQUIRED (*Win32Error attribute*), 303

ERROR_DS_MAX_OBJ_SIZE_EXCEEDED (*DirectoryStorageError attribute*), 263

ERROR_DS_MAX_OBJ_SIZE_EXCEEDED (*Win32Error attribute*), 303

ERROR_DS_MEMBERSHIP_EVALUATED_LOCALLY (*DirectoryStorageError attribute*), 263

ERROR_DS_MEMBERSHIP_EVALUATED_LOCALLY (*Win32Error attribute*), 303

ERROR_DS_MISSING_EXPECTED_ATT (*DirectoryStorageError attribute*), 263

ERROR_DS_MISSING_EXPECTED_ATT (*Win32Error attribute*), 303

ERROR_DS_MISSING_FSMO_SETTINGS (*DirectoryStorageError attribute*), 263

ERROR_DS_MISSING_FSMO_SETTINGS (*Win32Error attribute*), 303

ERROR_DS_MISSING_INFRASTRUCTURE_CONTAINER (*DirectoryStorageError attribute*), 263

ERROR_DS_MISSING_INFRASTRUCTURE_CONTAINER (*Win32Error attribute*), 303

ERROR_DS_MISSING_REQUIRED_ATT (*DirectoryStorageError attribute*), 263

ERROR_DS_MISSING_REQUIRED_ATT (*Win32Error attribute*), 303

ERROR_DS_MISSING_SUPREF (*DirectoryStorageError attribute*), 263

ERROR_DS_MISSING_SUPREF (*Win32Error attribute*), 303

ERROR_DS_MODIFYDN_DISALLOWED_BY_FLAG (*DirectoryStorageError attribute*), 263

ERROR_DS_MODIFYDN_DISALLOWED_BY_FLAG (*Win32Error attribute*), 304

ERROR_DS_MODIFYDN_DISALLOWED_BY_INSTANCE_TYPE (*DirectoryStorageError attribute*), 263

ERROR_DS_MODIFYDN_DISALLOWED_BY_INSTANCE_TYPE (*Win32Error attribute*), 304

ERROR_DS_MODIFYDN_WRONG_GRANDPARENT (*DirectoryStorageError attribute*), 263

ERROR_DS_MODIFYDN_WRONG_GRANDPARENT (*Win32Error attribute*), 304

ERROR_DS_MUST_BE_RUN_ON_DST_DC (*DirectoryStorageError attribute*), 263

ERROR_DS_MUST_BE_RUN_ON_DST_DC (*Win32Error attribute*), 304

ERROR_DS_NAME_ERROR_DOMAIN_ONLY (*DirectoryStorageError attribute*), 263

ERROR_DS_NAME_ERROR_DOMAIN_ONLY (*Win32Error attribute*), 304

ERROR_DS_NAME_ERROR_NO_MAPPING (DirectoryStorageError attribute), 263	ERROR_DS_NAMING_MASTER_GC (Win32Error attribute), 304
ERROR_DS_NAME_ERROR_NO_MAPPING (Win32Error attribute), 304	ERROR_DS_NAMING_VIOLATION (DirectoryStorageError attribute), 264
ERROR_DS_NAME_ERROR_NO_SYNTACTICAL_MAPPING (DirectoryStorageError attribute), 263	ERROR_DS_NAMING_VIOLATION (Win32Error attribute), 304
ERROR_DS_NAME_ERROR_NO_SYNTACTICAL_MAPPING (Win32Error attribute), 304	ERROR_DS_NC_MUST_HAVE_NC_PARENT (DirectoryStorageError attribute), 264
ERROR_DS_NAME_ERROR_NOT_FOUND (DirectoryStorageError attribute), 263	ERROR_DS_NC_MUST_HAVE_NC_PARENT (Win32Error attribute), 304
ERROR_DS_NAME_ERROR_NOT_FOUND (Win32Error attribute), 304	ERROR_DS_NC_STILL_HAS_DSAS (DirectoryStorageError attribute), 264
ERROR_DS_NAME_ERROR_NOT_UNIQUE (DirectoryStorageError attribute), 263	ERROR_DS_NC_STILL_HAS_DSAS (Win32Error attribute), 304
ERROR_DS_NAME_ERROR_NOT_UNIQUE (Win32Error attribute), 304	ERROR_DS_NCNAME_MISSING_CR_REF (DirectoryStorageError attribute), 264
ERROR_DS_NAME_ERROR_RESOLVING (DirectoryStorageError attribute), 263	ERROR_DS_NCNAME_MISSING_CR_REF (Win32Error attribute), 304
ERROR_DS_NAME_ERROR_RESOLVING (Win32Error attribute), 304	ERROR_DS_NCNAME_MUST_BE_NC (DirectoryStorageError attribute), 264
ERROR_DS_NAME_ERROR_TRUST_REFERRAL (DirectoryStorageError attribute), 263	ERROR_DS_NCNAME_MUST_BE_NC (Win32Error attribute), 304
ERROR_DS_NAME_ERROR_TRUST_REFERRAL (Win32Error attribute), 304	ERROR_DS_NO_ATTRIBUTE_OR_VALUE (DirectoryStorageError attribute), 264
ERROR_DS_NAME_NOT_UNIQUE (DirectoryStorageError attribute), 263	ERROR_DS_NO_ATTRIBUTE_OR_VALUE (Win32Error attribute), 305
ERROR_DS_NAME_NOT_UNIQUE (Win32Error attribute), 304	ERROR_DS_NO_BEHAVIOR_VERSION_IN_MIXEDDOMAIN (DirectoryStorageError attribute), 264
ERROR_DS_NAME_REFERENCE_INVALID (DirectoryStorageError attribute), 264	ERROR_DS_NO_BEHAVIOR_VERSION_IN_MIXEDDOMAIN (Win32Error attribute), 305
ERROR_DS_NAME_REFERENCE_INVALID (Win32Error attribute), 304	ERROR_DS_NO_CHAINED_EVAL (DirectoryStorageError attribute), 264
ERROR_DS_NAME_TOO_LONG (DirectoryStorageError attribute), 264	ERROR_DS_NO_CHAINED_EVAL (Win32Error attribute), 305
ERROR_DS_NAME_TOO_LONG (Win32Error attribute), 304	ERROR_DS_NO_CHAINING (DirectoryStorageError attribute), 264
ERROR_DS_NAME_TOO_MANY_PARTS (DirectoryStorageError attribute), 264	ERROR_DS_NO_CHAINING (Win32Error attribute), 305
ERROR_DS_NAME_TOO_MANY_PARTS (Win32Error attribute), 304	ERROR_DS_NO_CHECKPOINT_WITH_PDC (DirectoryStorageError attribute), 264
ERROR_DS_NAME_TYPE_UNKNOWN (DirectoryStorageError attribute), 264	ERROR_DS_NO_CHECKPOINT_WITH_PDC (Win32Error attribute), 305
ERROR_DS_NAME_TYPE_UNKNOWN (Win32Error attribute), 304	ERROR_DS_NO_CROSSREF_FOR_NC (DirectoryStorageError attribute), 264
ERROR_DS_NAME_UNPARSEABLE (DirectoryStorageError attribute), 264	ERROR_DS_NO_CROSSREF_FOR_NC (Win32Error attribute), 305
ERROR_DS_NAME_UNPARSEABLE (Win32Error attribute), 304	ERROR_DS_NO_DELETED_NAME (DirectoryStorageError attribute), 264
ERROR_DS_NAME_VALUE_TOO_LONG (DirectoryStorageError attribute), 264	ERROR_DS_NO_DELETED_NAME (Win32Error attribute), 305
ERROR_DS_NAME_VALUE_TOO_LONG (Win32Error attribute), 304	ERROR_DS_NO_FPO_IN_UNIVERSAL_GROUPS (DirectoryStorageError attribute), 264
ERROR_DS_NAMING_MASTER_GC (DirectoryStorageError attribute), 264	ERROR_DS_NO_FPO_IN_UNIVERSAL_GROUPS (Win32Error attribute), 305

`ERROR_DS_NO_MORE_RIDS` (*DirectoryStorageError attribute*), 264

`ERROR_DS_NO_MORE_RIDS` (*Win32Error attribute*), 305

`ERROR_DS_NO_MSDS_INTID` (*DirectoryStorageError attribute*), 264

`ERROR_DS_NO_MSDS_INTID` (*Win32Error attribute*), 305

`ERROR_DS_NO_NEST_GLOBALGROUP_IN_MIXEDDOMAIN` (*DirectoryStorageError attribute*), 264

`ERROR_DS_NO_NEST_GLOBALGROUP_IN_MIXEDDOMAIN` (*Win32Error attribute*), 305

`ERROR_DS_NO_NEST_LOCALGROUP_IN_MIXEDDOMAIN` (*DirectoryStorageError attribute*), 265

`ERROR_DS_NO_NEST_LOCALGROUP_IN_MIXEDDOMAIN` (*Win32Error attribute*), 305

`ERROR_DS_NO_NTDSA_OBJECT` (*Win32Error attribute*), 305

`ERROR_DS_NO_OBJECT_MOVE_IN_SCHEMA_NC` (*DirectoryStorageError attribute*), 265

`ERROR_DS_NO_OBJECT_MOVE_IN_SCHEMA_NC` (*Win32Error attribute*), 305

`ERROR_DS_NO_PARENT_OBJECT` (*DirectoryStorageError attribute*), 265

`ERROR_DS_NO_PARENT_OBJECT` (*Win32Error attribute*), 305

`ERROR_DS_NO_PKT_PRIVACY_ON_CONNECTION` (*DirectoryStorageError attribute*), 265

`ERROR_DS_NO_PKT_PRIVACY_ON_CONNECTION` (*Win32Error attribute*), 305

`ERROR_DS_NO_RDN_DEFINED_IN_SCHEMA` (*DirectoryStorageError attribute*), 265

`ERROR_DS_NO_RDN_DEFINED_IN_SCHEMA` (*Win32Error attribute*), 305

`ERROR_DS_NO_REF_DOMAIN` (*DirectoryStorageError attribute*), 265

`ERROR_DS_NO_REF_DOMAIN` (*Win32Error attribute*), 305

`ERROR_DS_NO_REQUESTED_ATTRS_FOUND` (*DirectoryStorageError attribute*), 265

`ERROR_DS_NO_REQUESTED_ATTRS_FOUND` (*Win32Error attribute*), 305

`ERROR_DS_NO_RESULTS_RETURNED` (*DirectoryStorageError attribute*), 265

`ERROR_DS_NO_RESULTS_RETURNED` (*Win32Error attribute*), 305

`ERROR_DS_NO_RIDS_ALLOCATED` (*DirectoryStorageError attribute*), 265

`ERROR_DS_NO_RIDS_ALLOCATED` (*Win32Error attribute*), 305

`ERROR_DS_NO_SERVER_OBJECT` (*Win32Error attribute*), 305

`ERROR_DS_NO_SUCH_OBJECT` (*DirectoryStorageError attribute*), 265

`ERROR_DS_NO_SUCH_OBJECT` (*Win32Error attribute*), 305

`ERROR_DS_NO_TREE_DELETE_ABOVE_NC` (*DirectoryStorageError attribute*), 265

`ERROR_DS_NO_TREE_DELETE_ABOVE_NC` (*Win32Error attribute*), 305

`ERROR_DS_NON_ASQ_SEARCH` (*Win32Error attribute*), 304

`ERROR_DS_NON_BASE_SEARCH` (*DirectoryStorageError attribute*), 264

`ERROR_DS_NON_BASE_SEARCH` (*Win32Error attribute*), 304

`ERROR_DS_NONEXISTENT_MAY_HAVE` (*DirectoryStorageError attribute*), 264

`ERROR_DS_NONEXISTENT_MAY_HAVE` (*Win32Error attribute*), 304

`ERROR_DS_NONEXISTENT_MUST_HAVE` (*DirectoryStorageError attribute*), 264

`ERROR_DS_NONEXISTENT_MUST_HAVE` (*Win32Error attribute*), 304

`ERROR_DS_NONEXISTENT_POSS_SUP` (*DirectoryStorageError attribute*), 264

`ERROR_DS_NONEXISTENT_POSS_SUP` (*Win32Error attribute*), 304

`ERROR_DS_NONSAFE_SCHEMA_CHANGE` (*DirectoryStorageError attribute*), 264

`ERROR_DS_NONSAFE_SCHEMA_CHANGE` (*Win32Error attribute*), 304

`ERROR_DS_NOT_AN_OBJECT` (*DirectoryStorageError attribute*), 264

`ERROR_DS_NOT_AN_OBJECT` (*Win32Error attribute*), 304

`ERROR_DS_NOT_AUTHORTITIVE_FOR_DST_NC` (*DirectoryStorageError attribute*), 264

`ERROR_DS_NOT_AUTHORTITIVE_FOR_DST_NC` (*Win32Error attribute*), 304

`ERROR_DS_NOT_CLOSEST` (*DirectoryStorageError attribute*), 264

`ERROR_DS_NOT_CLOSEST` (*Win32Error attribute*), 304

`ERROR_DS_NOT_INSTALLED` (*DirectoryStorageError attribute*), 264

`ERROR_DS_NOT_INSTALLED` (*Win32Error attribute*), 304

`ERROR_DS_NOT_ON_BACKLINK` (*DirectoryStorageError attribute*), 264

`ERROR_DS_NOT_ON_BACKLINK` (*Win32Error attribute*), 304

`ERROR_DS_NOT_SUPPORTED` (*DirectoryStorageError attribute*), 264

`ERROR_DS_NOT_SUPPORTED` (*Win32Error attribute*), 305

`ERROR_DS_NOT_SUPPORTED_SORT_ORDER` (*DirectoryStorageError attribute*), 264

ERROR_DS_NOT_SUPPORTED_SORT_ORDER (Win32Error attribute), 305	ERROR_DS_OFFSET_RANGE_ERROR (DirectoryStorageError attribute), 265
ERROR_DS_NOTIFY_FILTER_TOO_COMPLEX (DirectoryStorageError attribute), 264	ERROR_DS_OFFSET_RANGE_ERROR (Win32Error attribute), 306
ERROR_DS_NOTIFY_FILTER_TOO_COMPLEX (Win32Error attribute), 304	ERROR_DS_OPERATIONS_ERROR (DirectoryStorageError attribute), 265
ERROR_DS_NTDSRIPT_PROCESS_ERROR (DirectoryStorageError attribute), 265	ERROR_DS_OPERATIONS_ERROR (Win32Error attribute), 306
ERROR_DS_NTDSRIPT_PROCESS_ERROR (Win32Error attribute), 305	ERROR_DS_OUT_OF_SCOPE (DirectoryStorageError attribute), 265
ERROR_DS_NTDSRIPT_SYNTAX_ERROR (DirectoryStorageError attribute), 265	ERROR_DS_OUT_OF_SCOPE (Win32Error attribute), 306
ERROR_DS_NTDSRIPT_SYNTAX_ERROR (Win32Error attribute), 305	ERROR_DS_OUT_OF_VERSION_STORE (DirectoryStorageError attribute), 265
ERROR_DS_OBJ_CLASS_NOT_DEFINED (DirectoryStorageError attribute), 265	ERROR_DS_OUT_OF_VERSION_STORE (Win32Error attribute), 306
ERROR_DS_OBJ_CLASS_NOT_DEFINED (Win32Error attribute), 305	ERROR_DS_PARAM_ERROR (DirectoryStorageError attribute), 265
ERROR_DS_OBJ_CLASS_NOT_SUBCLASS (DirectoryStorageError attribute), 265	ERROR_DS_PARAM_ERROR (Win32Error attribute), 306
ERROR_DS_OBJ_CLASS_NOT_SUBCLASS (Win32Error attribute), 305	ERROR_DS_PARENT_IS_AN_ALIAS (DirectoryStorageError attribute), 265
ERROR_DS_OBJ_CLASS_VIOLATION (DirectoryStorageError attribute), 265	ERROR_DS_PARENT_IS_AN_ALIAS (Win32Error attribute), 306
ERROR_DS_OBJ_CLASS_VIOLATION (Win32Error attribute), 305	ERROR_DS_PDC_OPERATION_IN_PROGRESS (DirectoryStorageError attribute), 265
ERROR_DS_OBJ_GUID_EXISTS (DirectoryStorageError attribute), 265	ERROR_DS_PDC_OPERATION_IN_PROGRESS (Win32Error attribute), 306
ERROR_DS_OBJ_GUID_EXISTS (Win32Error attribute), 305	ERROR_DS_POLICY_NOT_KNOWN (Win32Error attribute), 306
ERROR_DS_OBJ_NOT_FOUND (DirectoryStorageError attribute), 265	ERROR_DS_PROTOCOL_ERROR (DirectoryStorageError attribute), 265
ERROR_DS_OBJ_NOT_FOUND (Win32Error attribute), 305	ERROR_DS_PROTOCOL_ERROR (Win32Error attribute), 306
ERROR_DS_OBJ_STRING_NAME_EXISTS (DirectoryStorageError attribute), 265	ERROR_DS_RANGE_CONSTRAINT (DirectoryStorageError attribute), 265
ERROR_DS_OBJ_STRING_NAME_EXISTS (Win32Error attribute), 306	ERROR_DS_RANGE_CONSTRAINT (Win32Error attribute), 306
ERROR_DS_OBJ_TOO_LARGE (DirectoryStorageError attribute), 265	ERROR_DS_RDN_DOESNT_MATCH_SCHEMA (DirectoryStorageError attribute), 265
ERROR_DS_OBJ_TOO_LARGE (Win32Error attribute), 306	ERROR_DS_RDN_DOESNT_MATCH_SCHEMA (Win32Error attribute), 306
ERROR_DS_OBJECT_BEING_REMOVED (DirectoryStorageError attribute), 265	ERROR_DS_RECALCSHEMA_FAILED (DirectoryStorageError attribute), 265
ERROR_DS_OBJECT_BEING_REMOVED (Win32Error attribute), 305	ERROR_DS_RECALCSHEMA_FAILED (Win32Error attribute), 306
ERROR_DS_OBJECT_CLASS_REQUIRED (DirectoryStorageError attribute), 265	ERROR_DS_REFERRAL (DirectoryStorageError attribute), 265
ERROR_DS_OBJECT_CLASS_REQUIRED (Win32Error attribute), 305	ERROR_DS_REFERRAL (Win32Error attribute), 306
ERROR_DS_OBJECT_RESULTS_TOO_LARGE (DirectoryStorageError attribute), 265	ERROR_DS_REFERRAL_LIMIT_EXCEEDED (DirectoryStorageError attribute), 265
ERROR_DS_OBJECT_RESULTS_TOO_LARGE (Win32Error attribute), 305	ERROR_DS_REFERRAL_LIMIT_EXCEEDED (Win32Error attribute), 306
	ERROR_DS_REFUSING_FSMO_ROLES (DirectoryS-

storageError attribute), 266

ERROR_DS_REFUSING_FSMO_ROLES (*Win32Error attribute*), 306

ERROR_DS_REMOTE_CROSSREF_OP_FAILED (*DirectoryStorageError attribute*), 266

ERROR_DS_REMOTE_CROSSREF_OP_FAILED (*Win32Error attribute*), 306

ERROR_DS_REPL_LIFETIME_EXCEEDED (*DirectoryStorageError attribute*), 266

ERROR_DS_REPL_LIFETIME_EXCEEDED (*Win32Error attribute*), 306

ERROR_DS_REPLICA_SET_CHANGE_NOT_ALLOWED (*DirectoryStorageError attribute*), 266

ERROR_DS_REPLICA_SET_CHANGE_NOT_ALLOWED (*Win32Error attribute*), 306

ERROR_DS_REPLICATOR_ONLY (*DirectoryStorageError attribute*), 266

ERROR_DS_REPLICATOR_ONLY (*Win32Error attribute*), 306

ERROR_DS_RESERVED_LINK_ID (*DirectoryStorageError attribute*), 266

ERROR_DS_RESERVED_LINK_ID (*Win32Error attribute*), 306

ERROR_DS_RIDMGR_INIT_ERROR (*DirectoryStorageError attribute*), 266

ERROR_DS_RIDMGR_INIT_ERROR (*Win32Error attribute*), 306

ERROR_DS_ROLE_NOT_VERIFIED (*DirectoryStorageError attribute*), 266

ERROR_DS_ROLE_NOT_VERIFIED (*Win32Error attribute*), 306

ERROR_DS_ROOT_CANT_BE_SUBREF (*DirectoryStorageError attribute*), 266

ERROR_DS_ROOT_CANT_BE_SUBREF (*Win32Error attribute*), 306

ERROR_DS_ROOT_MUST_BE_NC (*DirectoryStorageError attribute*), 266

ERROR_DS_ROOT_MUST_BE_NC (*Win32Error attribute*), 306

ERROR_DS_ROOT_REQUIRES_CLASS_TOP (*DirectoryStorageError attribute*), 266

ERROR_DS_ROOT_REQUIRES_CLASS_TOP (*Win32Error attribute*), 306

ERROR_DS_SAM_INIT_FAILURE (*DirectoryStorageError attribute*), 266

ERROR_DS_SAM_INIT_FAILURE (*Win32Error attribute*), 306

ERROR_DS_SAM_INIT_FAILURE_CONSOLE (*DirectoryStorageError attribute*), 266

ERROR_DS_SAM_INIT_FAILURE_CONSOLE (*Win32Error attribute*), 306

ERROR_DS_SAM_NEED_BOOTKEY_FLOPPY (*DirectoryStorageError attribute*), 266

ERROR_DS_SAM_NEED_BOOTKEY_FLOPPY (*Win32Error attribute*), 306

ERROR_DS_SAM_NEED_BOOTKEY_PASSWORD (*DirectoryStorageError attribute*), 266

ERROR_DS_SAM_NEED_BOOTKEY_PASSWORD (*Win32Error attribute*), 306

ERROR_DS_SCHEMA_ALLOC_FAILED (*DirectoryStorageError attribute*), 266

ERROR_DS_SCHEMA_ALLOC_FAILED (*Win32Error attribute*), 306

ERROR_DS_SCHEMA_NOT_LOADED (*DirectoryStorageError attribute*), 266

ERROR_DS_SCHEMA_NOT_LOADED (*Win32Error attribute*), 306

ERROR_DS_SCHEMA_UPDATE_DISALLOWED (*DirectoryStorageError attribute*), 266

ERROR_DS_SCHEMA_UPDATE_DISALLOWED (*Win32Error attribute*), 306

ERROR_DS_SEC_DESC_INVALID (*DirectoryStorageError attribute*), 266

ERROR_DS_SEC_DESC_INVALID (*Win32Error attribute*), 307

ERROR_DS_SEC_DESC_TOO_SHORT (*DirectoryStorageError attribute*), 266

ERROR_DS_SEC_DESC_TOO_SHORT (*Win32Error attribute*), 307

ERROR_DS_SECURITY_CHECKING_ERROR (*DirectoryStorageError attribute*), 266

ERROR_DS_SECURITY_CHECKING_ERROR (*Win32Error attribute*), 306

ERROR_DS_SECURITY_ILLEGAL_MODIFY (*DirectoryStorageError attribute*), 266

ERROR_DS_SECURITY_ILLEGAL_MODIFY (*Win32Error attribute*), 306

ERROR_DS_SEMANTIC_ATT_TEST (*DirectoryStorageError attribute*), 266

ERROR_DS_SEMANTIC_ATT_TEST (*Win32Error attribute*), 307

ERROR_DS_SENSITIVE_GROUP_VIOLATION (*DirectoryStorageError attribute*), 266

ERROR_DS_SENSITIVE_GROUP_VIOLATION (*Win32Error attribute*), 307

ERROR_DS_SERVER_DOWN (*DirectoryStorageError attribute*), 266

ERROR_DS_SERVER_DOWN (*Win32Error attribute*), 307

ERROR_DS_SHUTTING_DOWN (*DirectoryStorageError attribute*), 266

ERROR_DS_SHUTTING_DOWN (*Win32Error attribute*), 307

ERROR_DS_SINGLE_USER_MODE_FAILED (*DirectoryStorageError attribute*), 266

ERROR_DS_SINGLE_USER_MODE_FAILED (*Win32Error attribute*), 307

ERROR_DS_SINGLE_VALUE_CONSTRAINT (*DirectoryStorageError attribute*), 266

ERROR_DS_SINGLE_VALUE_CONSTRAINT (*Win32Error attribute*), 307

- toryStorageError attribute*), 266
- ERROR_DS_SINGLE_VALUE_CONSTRAINT (*Win32Error attribute*), 307
- ERROR_DS_SIZELIMIT_EXCEEDED (*DirectoryStorageError attribute*), 266
- ERROR_DS_SIZELIMIT_EXCEEDED (*Win32Error attribute*), 307
- ERROR_DS_SORT_CONTROL_MISSING (*DirectoryStorageError attribute*), 266
- ERROR_DS_SORT_CONTROL_MISSING (*Win32Error attribute*), 307
- ERROR_DS_SOURCE_AUDITING_NOT_ENABLED (*DirectoryStorageError attribute*), 266
- ERROR_DS_SOURCE_AUDITING_NOT_ENABLED (*Win32Error attribute*), 307
- ERROR_DS_SOURCE_DOMAIN_IN_FOREST (*DirectoryStorageError attribute*), 266
- ERROR_DS_SOURCE_DOMAIN_IN_FOREST (*Win32Error attribute*), 307
- ERROR_DS_SRC_AND_DST_NC_IDENTICAL (*DirectoryStorageError attribute*), 266
- ERROR_DS_SRC_AND_DST_NC_IDENTICAL (*Win32Error attribute*), 307
- ERROR_DS_SRC_AND_DST_OBJECT_CLASS_MISMATCH (*DirectoryStorageError attribute*), 266
- ERROR_DS_SRC_AND_DST_OBJECT_CLASS_MISMATCH (*Win32Error attribute*), 307
- ERROR_DS_SRC_DC_MUST_BE_SP4_OR_GREATER (*DirectoryStorageError attribute*), 266
- ERROR_DS_SRC_DC_MUST_BE_SP4_OR_GREATER (*Win32Error attribute*), 307
- ERROR_DS_SRC_GUID_MISMATCH (*DirectoryStorageError attribute*), 266
- ERROR_DS_SRC_GUID_MISMATCH (*Win32Error attribute*), 307
- ERROR_DS_SRC_NAME_MISMATCH (*DirectoryStorageError attribute*), 267
- ERROR_DS_SRC_NAME_MISMATCH (*Win32Error attribute*), 307
- ERROR_DS_SRC_OBJ_NOT_GROUP_OR_USER (*DirectoryStorageError attribute*), 267
- ERROR_DS_SRC_OBJ_NOT_GROUP_OR_USER (*Win32Error attribute*), 307
- ERROR_DS_SRC_SID_EXISTS_IN_FOREST (*DirectoryStorageError attribute*), 267
- ERROR_DS_SRC_SID_EXISTS_IN_FOREST (*Win32Error attribute*), 307
- ERROR_DS_STRING_SD_CONVERSION_FAILED (*DirectoryStorageError attribute*), 267
- ERROR_DS_STRING_SD_CONVERSION_FAILED (*Win32Error attribute*), 307
- ERROR_DS_STRONG_AUTH_REQUIRED (*DirectoryStorageError attribute*), 267
- ERROR_DS_STRONG_AUTH_REQUIRED (*Win32Error attribute*), 307
- attribute*), 307
- ERROR_DS_SUB_CLS_TEST_FAIL (*DirectoryStorageError attribute*), 267
- ERROR_DS_SUB_CLS_TEST_FAIL (*Win32Error attribute*), 307
- ERROR_DS_SUBREF_MUST_HAVE_PARENT (*DirectoryStorageError attribute*), 267
- ERROR_DS_SUBREF_MUST_HAVE_PARENT (*Win32Error attribute*), 307
- ERROR_DS_SUBTREE_NOTIFY_NOT_NC_HEAD (*DirectoryStorageError attribute*), 267
- ERROR_DS_SUBTREE_NOTIFY_NOT_NC_HEAD (*Win32Error attribute*), 307
- ERROR_DS_SYNTAX_MISMATCH (*DirectoryStorageError attribute*), 267
- ERROR_DS_SYNTAX_MISMATCH (*Win32Error attribute*), 307
- ERROR_DS_THREAD_LIMIT_EXCEEDED (*DirectoryStorageError attribute*), 267
- ERROR_DS_THREAD_LIMIT_EXCEEDED (*Win32Error attribute*), 307
- ERROR_DS_TIMELIMIT_EXCEEDED (*DirectoryStorageError attribute*), 267
- ERROR_DS_TIMELIMIT_EXCEEDED (*Win32Error attribute*), 307
- ERROR_DS_TREE_DELETE_NOT_FINISHED (*DirectoryStorageError attribute*), 267
- ERROR_DS_TREE_DELETE_NOT_FINISHED (*Win32Error attribute*), 307
- ERROR_DS_UNABLE_TO_SURRENDER_ROLES (*DirectoryStorageError attribute*), 267
- ERROR_DS_UNABLE_TO_SURRENDER_ROLES (*Win32Error attribute*), 307
- ERROR_DS_UNAVAILABLE (*DirectoryStorageError attribute*), 267
- ERROR_DS_UNAVAILABLE (*Win32Error attribute*), 307
- ERROR_DS_UNAVAILABLE_CRIT_EXTENSION (*DirectoryStorageError attribute*), 267
- ERROR_DS_UNAVAILABLE_CRIT_EXTENSION (*Win32Error attribute*), 307
- ERROR_DS_UNICODEPWD_NOT_IN_QUOTES (*DirectoryStorageError attribute*), 267
- ERROR_DS_UNICODEPWD_NOT_IN_QUOTES (*Win32Error attribute*), 307
- ERROR_DS_UNIVERSAL_CANT_HAVE_LOCAL_MEMBER (*DirectoryStorageError attribute*), 267
- ERROR_DS_UNIVERSAL_CANT_HAVE_LOCAL_MEMBER (*Win32Error attribute*), 307
- ERROR_DS_UNKNOWN_ERROR (*DirectoryStorageError attribute*), 267
- ERROR_DS_UNKNOWN_ERROR (*Win32Error attribute*), 307
- ERROR_DS_UNKNOWN_OPERATION (*DirectoryStorageError attribute*), 267

ageError attribute), 267

ERROR_DS_UNKNOWN_OPERATION (*Win32Error attribute*), 307

ERROR_DS_UNWILLING_TO_PERFORM (*DirectoryStorageError attribute*), 267

ERROR_DS_UNWILLING_TO_PERFORM (*Win32Error attribute*), 307

ERROR_DS_USER_BUFFER_TOO_SMALL (*DirectoryStorageError attribute*), 267

ERROR_DS_USER_BUFFER_TOO_SMALL (*Win32Error attribute*), 308

ERROR_DS_VERSION_CHECK_FAILURE (*Win32Error attribute*), 308

ERROR_DS_WKO_CONTAINER_CANNOT_BE_SPECIAL (*DirectoryStorageError attribute*), 267

ERROR_DS_WKO_CONTAINER_CANNOT_BE_SPECIAL (*Win32Error attribute*), 308

ERROR_DS_WRONG_LINKED_ATT_SYNTAX (*DirectoryStorageError attribute*), 267

ERROR_DS_WRONG_LINKED_ATT_SYNTAX (*Win32Error attribute*), 308

ERROR_DS_WRONG_OM_OBJ_CLASS (*DirectoryStorageError attribute*), 267

ERROR_DS_WRONG_OM_OBJ_CLASS (*Win32Error attribute*), 308

ERROR_DUP_DOMAINNAME (*Win32Error attribute*), 308

ERROR_DUP_NAME (*Win32Error attribute*), 308

ERROR_DUPLICATE_SERVICE_NAME (*Win32Error attribute*), 308

ERROR_DUPLICATE_TAG (*Win32Error attribute*), 308

ERROR_DYNLINK_FROM_INVALID_RING (*Win32Error attribute*), 308

ERROR_EA_ACCESS_DENIED (*Win32Error attribute*), 308

ERROR_EA_FILE_CORRUPT (*Win32Error attribute*), 308

ERROR_EA_LIST_INCONSISTENT (*Win32Error attribute*), 308

ERROR_EA_TABLE_FULL (*Win32Error attribute*), 308

ERROR_EAS_DIDNT_FIT (*Win32Error attribute*), 308

ERROR_EAS_NOT_SUPPORTED (*Win32Error attribute*), 308

ERROR_EFS_ALG_BLOB_TOO_BIG (*Win32Error attribute*), 308

ERROR_EFS_DISABLED (*Win32Error attribute*), 308

ERROR_EFS_NOT_ALLOWED_IN_TRANSACTION (*Win32Error attribute*), 308

ERROR_EFS_SERVER_NOT_TRUSTED (*Win32Error attribute*), 308

ERROR_EFS_VERSION_NOT_SUPPORTED (*Win32Error attribute*), 308

ERROR_ELEVATION_REQUIRED (*Win32Error attribute*), 308

ERROR_EMPTY (*Win32Error attribute*), 308

ERROR_ENCRYPTION_FAILED (*Win32Error attribute*), 308

ERROR_END_OF_MEDIA (*Win32Error attribute*), 308

ERROR_ENLISTMENT_NOT_FOUND (*Win32Error attribute*), 308

ERROR_ENLISTMENT_NOT_SUPERIOR (*Win32Error attribute*), 308

ERROR_ENVVAR_NOT_FOUND (*Win32Error attribute*), 308

ERROR_EOM_OVERFLOW (*Win32Error attribute*), 308

ERROR_ERRORS_ENCOUNTERED (*Win32Error attribute*), 308

ERROR_EVALUATION_EXPIRATION (*Win32Error attribute*), 308

ERROR_EVENT_DONE (*Win32Error attribute*), 308

ERROR_EVENT_PENDING (*Win32Error attribute*), 308

ERROR_EVENTLOG_CANT_START (*Win32Error attribute*), 308

ERROR_EVENTLOG_FILE_CHANGED (*Win32Error attribute*), 308

ERROR_EVENTLOG_FILE_CORRUPT (*Win32Error attribute*), 308

ERROR_EXCEPTION_IN_RESOURCE_CALL (*Win32Error attribute*), 309

ERROR_EXCEPTION_IN_SERVICE (*Win32Error attribute*), 309

ERROR_EXCL_SEM_ALREADY_OWNED (*Win32Error attribute*), 309

ERROR_EXE_CANNOT_MODIFY_SIGNED_BINARY (*Win32Error attribute*), 309

ERROR_EXE_CANNOT_MODIFY_STRONG_SIGNED_BINARY (*Win32Error attribute*), 309

ERROR_EXE_MACHINE_TYPE_MISMATCH (*Win32Error attribute*), 309

ERROR_EXE_MARKED_INVALID (*Win32Error attribute*), 309

ERROR_EXTENDED_ERROR (*Win32Error attribute*), 309

ERROR_EXTRANEEOUS_INFORMATION (*Win32Error attribute*), 309

ERROR_FAIL_I24 (*Win32Error attribute*), 309

ERROR_FAIL_NOACTION_REBOOT (*Win32Error attribute*), 309

ERROR_FAIL_REBOOT_INITIATED (*Win32Error attribute*), 309

ERROR_FAIL_REBOOT_REQUIRED (*Win32Error attribute*), 309

ERROR_FAIL_RESTART (*Win32Error attribute*), 309

ERROR_FAIL_SHUTDOWN (*Win32Error attribute*), 309

ERROR_FAILED_DRIVER_ENTRY (*Win32Error attribute*), 309

ERROR_FAILED_SERVICE_CONTROLLER_CONNECT (*Win32Error attribute*), 309

ERROR_FATAL_APP_EXIT (*Win32Error attribute*), 309
 ERROR_FILE_CHECKED_OUT (*Win32Error attribute*), 309
 ERROR_FILE_CORRUPT (*Win32Error attribute*), 309
 ERROR_FILE_ENCRYPTED (*Win32Error attribute*), 309
 ERROR_FILE_EXISTS (*Win32Error attribute*), 309
 ERROR_FILE_IDENTITY_NOT_PERSISTENT (*Win32Error attribute*), 309
 ERROR_FILE_INVALID (*Win32Error attribute*), 309
 ERROR_FILE_NOT_ENCRYPTED (*Win32Error attribute*), 309
 ERROR_FILE_NOT_FOUND (*Win32Error attribute*), 309
 ERROR_FILE_OFFLINE (*Win32Error attribute*), 309
 ERROR_FILE_READ_ONLY (*Win32Error attribute*), 309
 ERROR_FILE_SYSTEM_LIMITATION (*Win32Error attribute*), 309
 ERROR_FILE_TOO_LARGE (*Win32Error attribute*), 309
 ERROR_FILEMARK_DETECTED (*Win32Error attribute*), 309
 ERROR_FILENAME_EXCED_RANGE (*Win32Error attribute*), 309
 ERROR_FIRMWARE_UPDATED (*Win32Error attribute*), 309
 ERROR_FLOAT_MULTIPLE_FAULTS (*Win32Error attribute*), 309
 ERROR_FLOAT_MULTIPLE_TRAPS (*Win32Error attribute*), 309
 ERROR_FLOATED_SECTION (*Win32Error attribute*), 309
 ERROR_FLOPPY_BAD_REGISTERS (*Win32Error attribute*), 310
 ERROR_FLOPPY_ID_MARK_NOT_FOUND (*Win32Error attribute*), 310
 ERROR_FLOPPY_UNKNOWN_ERROR (*Win32Error attribute*), 310
 ERROR_FLOPPY_VOLUME (*Win32Error attribute*), 310
 ERROR_FLOPPY_WRONG_CYLINDER (*Win32Error attribute*), 310
 ERROR_FORMS_AUTH_REQUIRED (*Win32Error attribute*), 310
 ERROR_FOUND_OUT_OF_SCOPE (*Win32Error attribute*), 310
 ERROR_FS_DRIVER_REQUIRED (*Win32Error attribute*), 310
 ERROR_FSFILTER_OP_COMPLETED_SUCCESSFULLY (*Win32Error attribute*), 310
 ERROR_FT_READ_RECOVERY_FROM_BACKUP (*Win32Error attribute*), 310
 ERROR_FT_WRITE_RECOVERY (*Win32Error attribute*), 310
 ERROR_FULL_BACKUP (*Win32Error attribute*), 310
 ERROR_FULLSCREEN_MODE (*Win32Error attribute*), 310
 ERROR_FUNCTION_FAILED (*Win32Error attribute*), 310
 ERROR_FUNCTION_NOT_CALLED (*Win32Error attribute*), 310
 ERROR_GEN_FAILURE (*Win32Error attribute*), 310
 ERROR_GENERIC_NOT_MAPPED (*Win32Error attribute*), 310
 ERROR_GLOBAL_ONLY_HOOK (*Win32Error attribute*), 310
 ERROR_GRACEFUL_DISCONNECT (*Win32Error attribute*), 310
 ERROR_GROUP_EXISTS (*Win32Error attribute*), 310
 ERROR_GROUP_NOT_AVAILABLE (*Win32Error attribute*), 310
 ERROR_GROUP_NOT_FOUND (*Win32Error attribute*), 310
 ERROR_GROUP_NOT_ONLINE (*Win32Error attribute*), 310
 ERROR_GUID_SUBSTITUTION_MADE (*Win32Error attribute*), 310
 ERROR_HANDLE_DISK_FULL (*Win32Error attribute*), 310
 ERROR_HANDLE_EOF (*Win32Error attribute*), 310
 ERROR_HANDLE_NO_LONGER_VALID (*Win32Error attribute*), 310
 ERROR_HANDLES_CLOSED (*Win32Error attribute*), 310
 ERROR_HIBERNATED (*Win32Error attribute*), 310
 ERROR_HIBERNATION_FAILURE (*Win32Error attribute*), 310
 ERROR_HOOK_NEEDS_HMOD (*Win32Error attribute*), 310
 ERROR_HOOK_NOT_INSTALLED (*Win32Error attribute*), 310
 ERROR_HOOK_TYPE_NOT_ALLOWED (*Win32Error attribute*), 310
 ERROR_HOST_NODE_NOT_AVAILABLE (*Win32Error attribute*), 310
 ERROR_HOST_NODE_NOT_GROUP_OWNER (*Win32Error attribute*), 310
 ERROR_HOST_NODE_NOT_RESOURCE_OWNER (*Win32Error attribute*), 310
 ERROR_HOST_UNREACHABLE (*Win32Error attribute*), 311
 ERROR_HOTKEY_ALREADY_REGISTERED (*Win32Error attribute*), 311
 ERROR_HOTKEY_NOT_REGISTERED (*Win32Error attribute*), 311
 ERROR_HWNDS_HAVE_DIFF_PARENT (*Win32Error attribute*), 311

`ERROR_ICM_NOT_ENABLED` (*Win32Error attribute*), 311

`ERROR_IEPORT_FULL` (*Win32Error attribute*), 311

`ERROR_ILL_FORMED_PASSWORD` (*Win32Error attribute*), 311

`ERROR_ILLEGAL_CHARACTER` (*Win32Error attribute*), 311

`ERROR_ILLEGAL_DLL_RELOCATION` (*Win32Error attribute*), 311

`ERROR_ILLEGAL_ELEMENT_ADDRESS` (*Win32Error attribute*), 311

`ERROR_ILLEGAL_FLOAT_CONTEXT` (*Win32Error attribute*), 311

`ERROR_IMAGE_MACHINE_TYPE_MISMATCH` (*Win32Error attribute*), 311

`ERROR_IMAGE_MACHINE_TYPE_MISMATCH_EXE` (*Win32Error attribute*), 311

`ERROR_IMAGE_NOT_AT_BASE` (*Win32Error attribute*), 311

`ERROR_IMPLICIT_TRANSACTION_NOT_SUPPORTED` (*Win32Error attribute*), 311

`ERROR_INC_BACKUP` (*Win32Error attribute*), 311

`ERROR_INCORRECT_ADDRESS` (*Win32Error attribute*), 311

`ERROR_INCORRECT_SIZE` (*Win32Error attribute*), 311

`ERROR_INDEX_ABSENT` (*Win32Error attribute*), 311

`ERROR_INDIGENOUS_TYPE` (*Win32Error attribute*), 311

`ERROR_INDOUBT_TRANSACTIONS_EXIST` (*Win32Error attribute*), 311

`ERROR_INFLOOP_IN_RELOC_CHAIN` (*Win32Error attribute*), 311

`ERROR_INSTALL_ALREADY_RUNNING` (*Win32Error attribute*), 311

`ERROR_INSTALL_FAILURE` (*Win32Error attribute*), 311

`ERROR_INSTALL_LANGUAGE_UNSUPPORTED` (*Win32Error attribute*), 311

`ERROR_INSTALL_LOG_FAILURE` (*Win32Error attribute*), 311

`ERROR_INSTALL_NOTUSED` (*Win32Error attribute*), 311

`ERROR_INSTALL_PACKAGE_INVALID` (*Win32Error attribute*), 311

`ERROR_INSTALL_PACKAGE_OPEN_FAILED` (*Win32Error attribute*), 311

`ERROR_INSTALL_PACKAGE_REJECTED` (*Win32Error attribute*), 311

`ERROR_INSTALL_PLATFORM_UNSUPPORTED` (*Win32Error attribute*), 311

`ERROR_INSTALL_REMOTE_DISALLOWED` (*Win32Error attribute*), 311

`ERROR_INSTALL_REMOTE_PROHIBITED` (*Win32Error attribute*), 311

`ERROR_INSTALL_SERVICE` (*Win32Error attribute*), 311

`ERROR_INSTALL_SERVICE_SAFEBOOT` (*Win32Error attribute*), 311

`ERROR_INSTALL_SOURCE_ABSENT` (*Win32Error attribute*), 311

`ERROR_INSTALL_SUSPEND` (*Win32Error attribute*), 312

`ERROR_INSTALL_TEMP_UNWRITABLE` (*Win32Error attribute*), 312

`ERROR_INSTALL_TRANSFORM_FAILURE` (*Win32Error attribute*), 312

`ERROR_INSTALL_TRANSFORM_REJECTED` (*Win32Error attribute*), 312

`ERROR_INSTALL_UI_FAILURE` (*Win32Error attribute*), 312

`ERROR_INSTALL_USEREXIT` (*Win32Error attribute*), 312

`ERROR_INSTRUCTION_MISALIGNMENT` (*Win32Error attribute*), 312

`ERROR_INSUFFICIENT_BUFFER` (*Win32Error attribute*), 312

`ERROR_INSUFFICIENT_LOGON_INFO` (*Win32Error attribute*), 312

`ERROR_INSUFFICIENT_POWER` (*Win32Error attribute*), 312

`ERROR_INSUFFICIENT_RESOURCE_FOR_SPECIFIED_SHARED_SECTION_SIZE` (*Win32Error attribute*), 312

`ERROR_INTERNAL_DB_CORRUPTION` (*Win32Error attribute*), 312

`ERROR_INTERNAL_DB_ERROR` (*Win32Error attribute*), 312

`ERROR_INTERNAL_ERROR` (*Win32Error attribute*), 312

`ERROR_INTERRUPT_STILL_CONNECTED` (*Win32Error attribute*), 312

`ERROR_INTERRUPT_VECTOR_ALREADY_CONNECTED` (*Win32Error attribute*), 312

`ERROR_INVALID_ACCEL_HANDLE` (*Win32Error attribute*), 312

`ERROR_INVALID_ACCESS` (*Win32Error attribute*), 312

`ERROR_INVALID_ACCOUNT_NAME` (*Win32Error attribute*), 312

`ERROR_INVALID_ACL` (*Win32Error attribute*), 312

`ERROR_INVALID_ADDRESS` (*Win32Error attribute*), 312

`ERROR_INVALID_AT_INTERRUPT_TIME` (*Win32Error attribute*), 312

`ERROR_INVALID_BLOCK` (*Win32Error attribute*), 312

`ERROR_INVALID_BLOCK_LENGTH` (*Win32Error attribute*), 312

`ERROR_INVALID_CATEGORY` (*Win32Error attribute*), 312

- 312
- ERROR_INVALID_CLEANER (*Win32Error attribute*), 312
- ERROR_INVALID_CLUSTER_IPV6_ADDRESS (*Win32Error attribute*), 312
- ERROR_INVALID_CMM (*Win32Error attribute*), 312
- ERROR_INVALID_COLORINDEX (*Win32Error attribute*), 312
- ERROR_INVALID_COLORSPACE (*Win32Error attribute*), 312
- ERROR_INVALID_COMBOBOX_MESSAGE (*Win32Error attribute*), 312
- ERROR_INVALID_COMMAND_LINE (*Win32Error attribute*), 312
- ERROR_INVALID_COMPUTERNAME (*Win32Error attribute*), 312
- ERROR_INVALID_CURSOR_HANDLE (*Win32Error attribute*), 312
- ERROR_INVALID_DATA (*Win32Error attribute*), 312
- ERROR_INVALID_DATATYPE (*Win32Error attribute*), 312
- ERROR_INVALID_DEVICE_OBJECT_PARAMETER (*Win32Error attribute*), 313
- ERROR_INVALID_DLL (*Win32Error attribute*), 313
- ERROR_INVALID_DOMAIN_ROLE (*Win32Error attribute*), 313
- ERROR_INVALID_DOMAIN_STATE (*Win32Error attribute*), 313
- ERROR_INVALID_DOMAINNAME (*Win32Error attribute*), 313
- ERROR_INVALID_DRIVE (*Win32Error attribute*), 313
- ERROR_INVALID_DRIVE_OBJECT (*Win32Error attribute*), 313
- ERROR_INVALID_DWP_HANDLE (*Win32Error attribute*), 313
- ERROR_INVALID_EA_HANDLE (*Win32Error attribute*), 313
- ERROR_INVALID_EA_NAME (*Win32Error attribute*), 313
- ERROR_INVALID_EDIT_HEIGHT (*Win32Error attribute*), 313
- ERROR_INVALID_ENVIRONMENT (*Win32Error attribute*), 313
- ERROR_INVALID_EVENT_COUNT (*Win32Error attribute*), 313
- ERROR_INVALID_EVENTNAME (*Win32Error attribute*), 313
- ERROR_INVALID_EXE_SIGNATURE (*Win32Error attribute*), 313
- ERROR_INVALID_FIELD (*Win32Error attribute*), 313
- ERROR_INVALID_FILTER_PROC (*Win32Error attribute*), 313
- ERROR_INVALID_FLAG_NUMBER (*Win32Error attribute*), 313
- ERROR_INVALID_FLAGS (*Win32Error attribute*), 313
- ERROR_INVALID_FORM_NAME (*Win32Error attribute*), 313
- ERROR_INVALID_FORM_SIZE (*Win32Error attribute*), 313
- ERROR_INVALID_FUNCTION (*Win32Error attribute*), 313
- ERROR_INVALID_GROUP_ATTRIBUTES (*Win32Error attribute*), 313
- ERROR_INVALID_GROUPNAME (*Win32Error attribute*), 313
- ERROR_INVALID_GW_COMMAND (*Win32Error attribute*), 313
- ERROR_INVALID_HANDLE (*Win32Error attribute*), 313
- ERROR_INVALID_HANDLE_STATE (*Win32Error attribute*), 313
- ERROR_INVALID_HOOK_FILTER (*Win32Error attribute*), 313
- ERROR_INVALID_HOOK_HANDLE (*Win32Error attribute*), 313
- ERROR_INVALID_HW_PROFILE (*Win32Error attribute*), 313
- ERROR_INVALID_ICON_HANDLE (*Win32Error attribute*), 313
- ERROR_INVALID_ID_AUTHORITY (*Win32Error attribute*), 313
- ERROR_INVALID_IMAGE_HASH (*Win32Error attribute*), 313
- ERROR_INVALID_INDEX (*Win32Error attribute*), 313
- ERROR_INVALID_KEYBOARD_HANDLE (*Win32Error attribute*), 313
- ERROR_INVALID_LB_MESSAGE (*Win32Error attribute*), 313
- ERROR_INVALID_LDT_DESCRIPTOR (*Win32Error attribute*), 314
- ERROR_INVALID_LDT_OFFSET (*Win32Error attribute*), 314
- ERROR_INVALID_LDT_SIZE (*Win32Error attribute*), 314
- ERROR_INVALID_LEVEL (*Win32Error attribute*), 314
- ERROR_INVALID_LIBRARY (*Win32Error attribute*), 314
- ERROR_INVALID_LIST_FORMAT (*Win32Error attribute*), 314
- ERROR_INVALID_LOGON_HOURS (*Win32Error attribute*), 314
- ERROR_INVALID_LOGON_TYPE (*Win32Error attribute*), 314
- ERROR_INVALID_MEDIA (*Win32Error attribute*), 314
- ERROR_INVALID_MEDIA_POOL (*Win32Error attribute*), 314
- ERROR_INVALID_MEMBER (*Win32Error attribute*), 314

`ERROR_INVALID_MENU_HANDLE` (*Win32Error attribute*), 314

`ERROR_INVALID_MESSAGE` (*Win32Error attribute*), 314

`ERROR_INVALID_MESSAGEDEST` (*Win32Error attribute*), 314

`ERROR_INVALID_MESSAGE_NAME` (*Win32Error attribute*), 314

`ERROR_INVALID_MINALLOC_SIZE` (*Win32Error attribute*), 314

`ERROR_INVALID_MODULETYPE` (*Win32Error attribute*), 314

`ERROR_INVALID_MONITOR_HANDLE` (*Win32Error attribute*), 314

`ERROR_INVALID_MSGBOX_STYLE` (*Win32Error attribute*), 314

`ERROR_INVALID_NAME` (*Win32Error attribute*), 314

`ERROR_INVALID_NETNAME` (*Win32Error attribute*), 314

`ERROR_INVALID_OPERATION` (*Win32Error attribute*), 314

`ERROR_INVALID_OPERATION_ON_QUORUM` (*Win32Error attribute*), 314

`ERROR_INVALID_OPLOCK_PROTOCOL` (*Win32Error attribute*), 314

`ERROR_INVALID_ORDINAL` (*Win32Error attribute*), 314

`ERROR_INVALID_OWNER` (*Win32Error attribute*), 314

`ERROR_INVALID_PARAMETER` (*Win32Error attribute*), 314

`ERROR_INVALID_PASSWORD` (*Win32Error attribute*), 314

`ERROR_INVALID_PASSWORDNAME` (*Win32Error attribute*), 314

`ERROR_INVALID_PATCH_XML` (*Win32Error attribute*), 314

`ERROR_INVALID_PIXEL_FORMAT` (*Win32Error attribute*), 314

`ERROR_INVALID_PLUGPLAY_DEVICE_PATH` (*Win32Error attribute*), 314

`ERROR_INVALID_PORT_ATTRIBUTES` (*Win32Error attribute*), 314

`ERROR_INVALID_PRIMARY_GROUP` (*Win32Error attribute*), 314

`ERROR_INVALID_PRINT_MONITOR` (*Win32Error attribute*), 315

`ERROR_INVALID_PRINTER_COMMAND` (*Win32Error attribute*), 314

`ERROR_INVALID_PRINTER_NAME` (*Win32Error attribute*), 314

`ERROR_INVALID_PRINTER_STATE` (*Win32Error attribute*), 315

`ERROR_INVALID_PRIORITY` (*Win32Error attribute*), 315

`ERROR_INVALID_PROFILE` (*Win32Error attribute*), 315

`ERROR_INVALID_QUOTA_LOWER` (*Win32Error attribute*), 315

`ERROR_INVALID_REPARSE_DATA` (*Win32Error attribute*), 315

`ERROR_INVALID_SCROLLBAR_RANGE` (*Win32Error attribute*), 315

`ERROR_INVALID_SECURITY_DESCR` (*Win32Error attribute*), 315

`ERROR_INVALID_SEGDPL` (*Win32Error attribute*), 315

`ERROR_INVALID_SEGMENT_NUMBER` (*Win32Error attribute*), 315

`ERROR_INVALID_SEPARATOR_FILE` (*Win32Error attribute*), 315

`ERROR_INVALID_SERVER_STATE` (*Win32Error attribute*), 315

`ERROR_INVALID_SERVICE_ACCOUNT` (*Win32Error attribute*), 315

`ERROR_INVALID_SERVICE_CONTROL` (*Win32Error attribute*), 315

`ERROR_INVALID_SERVICE_LOCK` (*Win32Error attribute*), 315

`ERROR_INVALID_SERVICENAME` (*Win32Error attribute*), 315

`ERROR_INVALID_SHARENAME` (*Win32Error attribute*), 315

`ERROR_INVALID_SHOWWIN_COMMAND` (*Win32Error attribute*), 315

`ERROR_INVALID_SID` (*Win32Error attribute*), 315

`ERROR_INVALID_SIGNAL_NUMBER` (*Win32Error attribute*), 315

`ERROR_INVALID_SPI_VALUE` (*Win32Error attribute*), 315

`ERROR_INVALID_STACKSEG` (*Win32Error attribute*), 315

`ERROR_INVALID_STARTING_CODESEG` (*Win32Error attribute*), 315

`ERROR_INVALID_STATE` (*Win32Error attribute*), 315

`ERROR_INVALID_SUB_AUTHORITY` (*Win32Error attribute*), 315

`ERROR_INVALID_TABLE` (*Win32Error attribute*), 315

`ERROR_INVALID_TARGET_HANDLE` (*Win32Error attribute*), 315

`ERROR_INVALID_THREAD_ID` (*Win32Error attribute*), 315

`ERROR_INVALID_TIME` (*Win32Error attribute*), 315

`ERROR_INVALID_TRANSACTION` (*Win32Error attribute*), 315

`ERROR_INVALID_TRANSFORM` (*Win32Error attribute*), 315

`ERROR_INVALID_UNWIND_TARGET` (*Win32Error attribute*), 315

`ERROR_INVALID_USER_BUFFER` (*Win32Error attribute*), 315

`ERROR_INVALID_VARIANT` (*Win32Error attribute*), 315

`ERROR_INVALID_VERIFY_SWITCH` (*Win32Error attribute*), 315

`ERROR_INVALID_WINDOW_HANDLE` (*Win32Error attribute*), 315

`ERROR_INVALID_WINDOW_STYLE` (*Win32Error attribute*), 316

`ERROR_INVALID_WORKSTATION` (*Win32Error attribute*), 316

`ERROR_IO_DEVICE` (*Win32Error attribute*), 316

`ERROR_IO_INCOMPLETE` (*Win32Error attribute*), 316

`ERROR_IO_PENDING` (*Win32Error attribute*), 316

`ERROR_IO_PRIVILEGE_FAILED` (*Win32Error attribute*), 316

`ERROR_IO_REISSUE_AS_CACHED` (*Win32Error attribute*), 316

`ERROR_IOPL_NOT_ENABLED` (*Win32Error attribute*), 316

`ERROR_IP_ADDRESS_CONFLICT1` (*Win32Error attribute*), 316

`ERROR_IP_ADDRESS_CONFLICT2` (*Win32Error attribute*), 316

`ERROR_IRQ_BUSY` (*Win32Error attribute*), 316

`ERROR_IS_JOIN_PATH` (*Win32Error attribute*), 316

`ERROR_IS_JOIN_TARGET` (*Win32Error attribute*), 316

`ERROR_IS_JOINED` (*Win32Error attribute*), 316

`ERROR_IS_SUBST_PATH` (*Win32Error attribute*), 316

`ERROR_IS_SUBST_TARGET` (*Win32Error attribute*), 316

`ERROR_IS_SUBSTED` (*Win32Error attribute*), 316

`ERROR_ITERATED_DATA_EXCEEDS_64k` (*Win32Error attribute*), 316

`ERROR_JOIN_TO_JOIN` (*Win32Error attribute*), 316

`ERROR_JOIN_TO_SUBST` (*Win32Error attribute*), 316

`ERROR_JOURNAL_HOOK_SET` (*Win32Error attribute*), 316

`ERROR_KERNEL_APC` (*Win32Error attribute*), 316

`ERROR_KEY_DELETED` (*Win32Error attribute*), 316

`ERROR_KEY_HAS_CHILDREN` (*Win32Error attribute*), 316

`ERROR_KM_DRIVER_BLOCKED` (*Win32Error attribute*), 316

`ERROR_LABEL_TOO_LONG` (*Win32Error attribute*), 316

`ERROR_LAST_ADMIN` (*Win32Error attribute*), 316

`ERROR_LB_WITHOUT_TABSTOPS` (*Win32Error attribute*), 316

`ERROR_LIBRARY_FULL` (*Win32Error attribute*), 316

`ERROR_LIBRARY_OFFLINE` (*Win32Error attribute*), 316

`ERROR_LICENSE_QUOTA_EXCEEDED` (*Win32Error attribute*), 316

`ERROR_LISTBOX_ID_NOT_FOUND` (*Win32Error attribute*), 316

`ERROR_LM_CROSS_ENCRYPTION_REQUIRED` (*Win32Error attribute*), 316

`ERROR_LOCAL_USER_SESSION_KEY` (*Win32Error attribute*), 316

`ERROR_LOCK_FAILED` (*Win32Error attribute*), 316

`ERROR_LOCK_VIOLATION` (*Win32Error attribute*), 317

`ERROR_LOCKED` (*Win32Error attribute*), 316

`ERROR_LOG_APPENDED_FLUSH_FAILED` (*Win32Error attribute*), 317

`ERROR_LOG_ARCHIVE_IN_PROGRESS` (*Win32Error attribute*), 317

`ERROR_LOG_ARCHIVE_NOT_IN_PROGRESS` (*Win32Error attribute*), 317

`ERROR_LOG_BLOCK_INCOMPLETE` (*Win32Error attribute*), 317

`ERROR_LOG_BLOCK_INVALID` (*Win32Error attribute*), 317

`ERROR_LOG_BLOCK_VERSION` (*Win32Error attribute*), 317

`ERROR_LOG_BLOCKS_EXHAUSTED` (*Win32Error attribute*), 317

`ERROR_LOG_CANT_DELETE` (*Win32Error attribute*), 317

`ERROR_LOG_CLIENT_ALREADY_REGISTERED` (*Win32Error attribute*), 317

`ERROR_LOG_CLIENT_NOT_REGISTERED` (*Win32Error attribute*), 317

`ERROR_LOG_CONTAINER_LIMIT_EXCEEDED` (*Win32Error attribute*), 317

`ERROR_LOG_CONTAINER_OPEN_FAILED` (*Win32Error attribute*), 317

`ERROR_LOG_CONTAINER_READ_FAILED` (*Win32Error attribute*), 317

`ERROR_LOG_CONTAINER_STATE_INVALID` (*Win32Error attribute*), 317

`ERROR_LOG_CONTAINER_WRITE_FAILED` (*Win32Error attribute*), 317

`ERROR_LOG_CORRUPTION_DETECTED` (*Win32Error attribute*), 317

`ERROR_LOG_DEDICATED` (*Win32Error attribute*), 317

`ERROR_LOG_EPHEMERAL` (*Win32Error attribute*), 317

`ERROR_LOG_FILE_FULL` (*Win32Error attribute*), 317

`ERROR_LOG_FULL` (*Win32Error attribute*), 317

`ERROR_LOG_FULL_HANDLER_IN_PROGRESS` (*Win32Error attribute*), 317

`ERROR_LOG_GROWTH_FAILED` (*Win32Error attribute*), 317

`ERROR_LOG_HARD_ERROR` (*Win32Error attribute*), 317

`ERROR_LOG_INCONSISTENT_SECURITY` (*Win32Error attribute*), 317

`ERROR_LOG_INVALID_RANGE` (*Win32Error attribute*), 317

`ERROR_LOG_METADATA_CORRUPT` (*Win32Error attribute*), 317

`ERROR_LOG_METADATA_FLUSH_FAILED` (*Win32Error attribute*), 317

`ERROR_LOG_METADATA_INCONSISTENT` (*Win32Error attribute*), 318

`ERROR_LOG_METADATA_INVALID` (*Win32Error attribute*), 318

`ERROR_LOG_MULTIPLEXED` (*Win32Error attribute*), 318

`ERROR_LOG_NO_RESTART` (*Win32Error attribute*), 318

`ERROR_LOG_NOT_ENOUGH_CONTAINERS` (*Win32Error attribute*), 318

`ERROR_LOG_PINNED` (*Win32Error attribute*), 318

`ERROR_LOG_PINNED_ARCHIVE_TAIL` (*Win32Error attribute*), 318

`ERROR_LOG_PINNED_RESERVATION` (*Win32Error attribute*), 318

`ERROR_LOG_POLICY_ALREADY_INSTALLED` (*Win32Error attribute*), 318

`ERROR_LOG_POLICY_CONFLICT` (*Win32Error attribute*), 318

`ERROR_LOG_POLICY_INVALID` (*Win32Error attribute*), 318

`ERROR_LOG_POLICY_NOT_INSTALLED` (*Win32Error attribute*), 318

`ERROR_LOG_READ_CONTEXT_INVALID` (*Win32Error attribute*), 318

`ERROR_LOG_READ_MODE_INVALID` (*Win32Error attribute*), 318

`ERROR_LOG_RECORD_NONEXISTENT` (*Win32Error attribute*), 318

`ERROR_LOG_RECORDS_RESERVED_INVALID` (*Win32Error attribute*), 318

`ERROR_LOG_RESERVATION_INVALID` (*Win32Error attribute*), 318

`ERROR_LOG_RESIZE_INVALID_SIZE` (*Win32Error attribute*), 318

`ERROR_LOG_RESTART_INVALID` (*Win32Error attribute*), 318

`ERROR_LOG_SECTOR_INVALID` (*Win32Error attribute*), 318

`ERROR_LOG_SECTOR_PARITY_INVALID` (*Win32Error attribute*), 318

`ERROR_LOG_SECTOR_REMAPPED` (*Win32Error attribute*), 318

`ERROR_LOG_SPACE_RESERVED_INVALID` (*Win32Error attribute*), 318

`ERROR_LOG_START_OF_LOG` (*Win32Error attribute*), 318

`ERROR_LOG_STATE_INVALID` (*Win32Error attribute*), 318

`ERROR_LOG_TAIL_INVALID` (*Win32Error attribute*), 318

`ERROR_LOGIN_TIME_RESTRICTION` (*Win32Error attribute*), 317

`ERROR_LOGIN_WKSTA_RESTRICTION` (*Win32Error attribute*), 317

`ERROR_LOGON_FAILURE` (*Win32Error attribute*), 317

`ERROR_LOGON_NOT_GRANTED` (*Win32Error attribute*), 317

`ERROR_LOGON_SERVER_CONFLICT` (*Win32Error attribute*), 317

`ERROR_LOGON_SESSION_COLLISION` (*Win32Error attribute*), 317

`ERROR_LOGON_SESSION_EXISTS` (*Win32Error attribute*), 317

`ERROR_LOGON_TYPE_NOT_GRANTED` (*Win32Error attribute*), 317

`ERROR_LONGJUMP` (*Win32Error attribute*), 318

`ERROR_LOST_WRITEBEHIND_DATA` (*Win32Error attribute*), 318

`ERROR_LOST_WRITEBEHIND_DATA_LOCAL_DISK_ERROR` (*Win32Error attribute*), 318

`ERROR_LOST_WRITEBEHIND_DATA_NETWORK_DISCONNECTED` (*Win32Error attribute*), 318

`ERROR_LOST_WRITEBEHIND_DATA_NETWORK_SERVER_ERROR` (*Win32Error attribute*), 318

`ERROR_LUIDS_EXHAUSTED` (*Win32Error attribute*), 318

`ERROR_MAGAZINE_NOT_PRESENT` (*Win32Error attribute*), 318

`ERROR_MAPPED_ALIGNMENT` (*Win32Error attribute*), 318

`ERROR_MARSHALL_OVERFLOW` (*Win32Error attribute*), 318

`ERROR_MAX_SESSIONS_REACHED` (*Win32Error attribute*), 318

`ERROR_MAX_THRDS_REACHED` (*Win32Error attribute*), 319

`ERROR_MCA_EXCEPTION` (*Win32Error attribute*), 319

`ERROR_MCA_OCCURED` (*Win32Error attribute*), 319

`ERROR_MEDIA_CHANGED` (*Win32Error attribute*), 319

`ERROR_MEDIA_CHECK` (*Win32Error attribute*), 319

`ERROR_MEDIA_INCOMPATIBLE` (*Win32Error attribute*), 319

`ERROR_MEDIA_NOT_AVAILABLE` (*Win32Error attribute*), 319

`ERROR_MEDIA_OFFLINE` (*Win32Error attribute*), 319

`ERROR_MEDIA_UNAVAILABLE` (*Win32Error attribute*), 319

`ERROR_MEDIUM_NOT_ACCESSIBLE` (*Win32Error attribute*), 319

[ERROR_MEMBER_IN_ALIAS \(Win32Error attribute\), 319](#)
[ERROR_MEMBER_IN_GROUP \(Win32Error attribute\), 319](#)
[ERROR_MEMBER_NOT_IN_ALIAS \(Win32Error attribute\), 319](#)
[ERROR_MEMBER_NOT_IN_GROUP \(Win32Error attribute\), 319](#)
[ERROR_MEMBERS_PRIMARY_GROUP \(Win32Error attribute\), 319](#)
[ERROR_MEMORY_HARDWARE \(Win32Error attribute\), 319](#)
[ERROR_MENU_ITEM_NOT_FOUND \(Win32Error attribute\), 319](#)
[ERROR_MESSAGE_EXCEEDS_MAX_SIZE \(Win32Error attribute\), 319](#)
[ERROR_MESSAGE_SYNC_ONLY \(Win32Error attribute\), 319](#)
[ERROR_META_EXPANSION_TOO_LONG \(Win32Error attribute\), 319](#)
[ERROR_METAFILE_NOT_SUPPORTED \(Win32Error attribute\), 319](#)
[ERROR_MINIVERSION_INACCESSIBLE_FROM_SPECIFIED_TRANSACTION \(Win32Error attribute\), 319](#)
[ERROR_MISSING_SYSTEMFILE \(Win32Error attribute\), 319](#)
[ERROR_MOD_NOT_FOUND \(Win32Error attribute\), 319](#)
[ERROR_MORE_DATA \(Win32Error attribute\), 319](#)
[ERROR_MORE_WRITES \(Win32Error attribute\), 319](#)
[ERROR_MOUNT_POINT_NOT_RESOLVED \(Win32Error attribute\), 319](#)
[ERROR_MP_PROCESSOR_MISMATCH \(Win32Error attribute\), 319](#)
[ERROR_MR_MID_NOT_FOUND \(Win32Error attribute\), 319](#)
[ERROR_MULTIPLE_FAULT_VIOLATION \(Win32Error attribute\), 319](#)
[ERROR_MUTANT_LIMIT_EXCEEDED \(Win32Error attribute\), 319](#)
[ERROR_NEGATIVE_SEEK \(Win32Error attribute\), 319](#)
[ERROR_NESTING_NOT_ALLOWED \(Win32Error attribute\), 319](#)
[ERROR_NET_OPEN_FAILED \(Win32Error attribute\), 320](#)
[ERROR_NET_WRITE_FAULT \(Win32Error attribute\), 320](#)
[ERROR_NETLOGON_NOT_STARTED \(Win32Error attribute\), 319](#)
[ERROR_NETNAME_DELETED \(Win32Error attribute\), 319](#)
[ERROR_NETWORK_ACCESS_DENIED \(Win32Error attribute\), 319](#)
[ERROR_NETWORK_BUSY \(Win32Error attribute\), 320](#)
[ERROR_NETWORK_NOT_AVAILABLE \(Win32Error attribute\), 320](#)
[ERROR_NETWORK_UNREACHABLE \(Win32Error attribute\), 320](#)
[ERROR_NO_ASSOCIATION \(Win32Error attribute\), 321](#)
[ERROR_NO_BROWSER_SERVERS_FOUND \(Win32Error attribute\), 321](#)
[ERROR_NO_CALLBACK_ACTIVE \(Win32Error attribute\), 321](#)
[ERROR_NO_DATA \(Win32Error attribute\), 321](#)
[ERROR_NO_DATA_DETECTED \(Win32Error attribute\), 321](#)
[ERROR_NO_EFS \(Win32Error attribute\), 321](#)
[ERROR_NO_EVENT_PAIR \(Win32Error attribute\), 321](#)
[ERROR_NO_GUID_TRANSLATION \(Win32Error attribute\), 321](#)
[ERROR_NO_IMPERSONATION_TOKEN \(Win32Error attribute\), 321](#)
[ERROR_NO_INHERITANCE \(Win32Error attribute\), 321](#)
[ERROR_NO_LINK_TRACKING_IN_TRANSACTION \(Win32Error attribute\), 321](#)
[ERROR_NO_LOGON_SERVERS \(Win32Error attribute\), 321](#)
[ERROR_NO_MATCH \(Win32Error attribute\), 321](#)
[ERROR_NO_MEDIA_IN_DRIVE \(Win32Error attribute\), 321](#)
[ERROR_NO_MORE_DEVICES \(Win32Error attribute\), 321](#)
[ERROR_NO_MORE_FILES \(Win32Error attribute\), 321](#)
[ERROR_NO_MORE_ITEMS \(Win32Error attribute\), 321](#)
[ERROR_NO_MORE_MATCHES \(Win32Error attribute\), 321](#)
[ERROR_NO_MORE_SEARCH_HANDLES \(Win32Error attribute\), 321](#)
[ERROR_NO_MORE_USER_HANDLES \(Win32Error attribute\), 321](#)
[ERROR_NO_NET_OR_BAD_PATH \(Win32Error attribute\), 322](#)
[ERROR_NO_NETWORK \(Win32Error attribute\), 321](#)
[ERROR_NO_PAGEFILE \(Win32Error attribute\), 322](#)
[ERROR_NO_PROC_SLOTS \(Win32Error attribute\), 322](#)
[ERROR_NO_PROMOTION_ACTIVE \(DirectoryStorageError attribute\), 267](#)
[ERROR_NO_PROMOTION_ACTIVE \(Win32Error attribute\), 322](#)
[ERROR_NO_QUOTAS_FOR_ACCOUNT \(Win32Error attribute\), 322](#)
[ERROR_NO_RECOVERY_POLICY \(Win32Error attribute\), 322](#)
[ERROR_NO_RECOVERY_PROGRAM \(Win32Error attribute\), 322](#)
[ERROR_NO_SAVEPOINT_WITH_OPEN_FILES](#)

(*Win32Error attribute*), 322

ERROR_NO_SCROLLBARS (*Win32Error attribute*), 322

ERROR_NO_SECRETS (*Win32Error attribute*), 322

ERROR_NO_SECURITY_ON_OBJECT (*Win32Error attribute*), 322

ERROR_NO_SHUTDOWN_IN_PROGRESS (*Win32Error attribute*), 322

ERROR_NO_SIGNAL_SENT (*Win32Error attribute*), 322

ERROR_NO_SITE_SETTINGS_OBJECT (*Win32Error attribute*), 322

ERROR_NO_SITENAME (*Win32Error attribute*), 322

ERROR_NO_SPOOL_SPACE (*Win32Error attribute*), 322

ERROR_NO_SUCH_ALIAS (*Win32Error attribute*), 322

ERROR_NO_SUCH_DOMAIN (*Win32Error attribute*), 322

ERROR_NO_SUCH_GROUP (*Win32Error attribute*), 322

ERROR_NO_SUCH_LOGON_SESSION (*Win32Error attribute*), 322

ERROR_NO_SUCH_MEMBER (*Win32Error attribute*), 322

ERROR_NO_SUCH_PACKAGE (*Win32Error attribute*), 322

ERROR_NO_SUCH_PRIVILEGE (*Win32Error attribute*), 322

ERROR_NO_SUCH_SITE (*Win32Error attribute*), 322

ERROR_NO_SUCH_USER (*Win32Error attribute*), 322

ERROR_NO_SUPPORTING_DRIVES (*Win32Error attribute*), 322

ERROR_NO_SYSTEM_MENU (*Win32Error attribute*), 322

ERROR_NO_SYSTEM_RESOURCES (*Win32Error attribute*), 322

ERROR_NO_TOKEN (*Win32Error attribute*), 322

ERROR_NO_TRACKING_SERVICE (*Win32Error attribute*), 322

ERROR_NO_TRUST_LSA_SECRET (*Win32Error attribute*), 322

ERROR_NO_TRUST_SAM_ACCOUNT (*Win32Error attribute*), 322

ERROR_NO_TXF_METADATA (*Win32Error attribute*), 322

ERROR_NO_UNICODE_TRANSLATION (*Win32Error attribute*), 322

ERROR_NO_USER_KEYS (*Win32Error attribute*), 322

ERROR_NO_USER_SESSION_KEY (*Win32Error attribute*), 322

ERROR_NO_VOLUME_ID (*Win32Error attribute*), 323

ERROR_NO_VOLUME_LABEL (*Win32Error attribute*), 323

ERROR_NO_WILDCARD_CHARACTERS (*Win32Error attribute*), 323

ERROR_NO_WRITABLE_DC_FOUND (*Win32Error attribute*), 323

ERROR_NO_YIELD_PERFORMED (*Win32Error attribute*), 323

ERROR_NOACCESS (*Win32Error attribute*), 320

ERROR_NODE_CANNOT_BE_CLUSTERED (*Win32Error attribute*), 320

ERROR_NODE_CANT_HOST_RESOURCE (*Win32Error attribute*), 320

ERROR_NODE_NOT_AVAILABLE (*Win32Error attribute*), 320

ERROR_NOINTERFACE (*Win32Error attribute*), 320

ERROR_NOLOGON_INTERDOMAIN_TRUST_ACCOUNT (*Win32Error attribute*), 320

ERROR_NOLOGON_SERVER_TRUST_ACCOUNT (*Win32Error attribute*), 320

ERROR_NOLOGON_WORKSTATION_TRUST_ACCOUNT (*Win32Error attribute*), 320

ERROR_NON_MDICHILD_WINDOW (*Win32Error attribute*), 320

ERROR_NONE_MAPPED (*Win32Error attribute*), 320

ERROR_NONPAGED_SYSTEM_RESOURCES (*Win32Error attribute*), 320

ERROR_NOT_A_REPARSE_POINT (*Win32Error attribute*), 320

ERROR_NOT_ALL_ASSIGNED (*Win32Error attribute*), 320

ERROR_NOT_AUTHENTICATED (*Win32Error attribute*), 320

ERROR_NOT_CAPABLE (*Win32Error attribute*), 320

ERROR_NOT_CHILD_WINDOW (*Win32Error attribute*), 320

ERROR_NOT_CONNECTED (*Win32Error attribute*), 320

ERROR_NOT_CONTAINER (*Win32Error attribute*), 320

ERROR_NOT_DOS_DISK (*Win32Error attribute*), 320

ERROR_NOT_EMPTY (*Win32Error attribute*), 320

ERROR_NOT_ENOUGH_MEMORY (*Win32Error attribute*), 320

ERROR_NOT_ENOUGH_QUOTA (*Win32Error attribute*), 320

ERROR_NOT_ENOUGH_SERVER_MEMORY (*Win32Error attribute*), 320

ERROR_NOT_EXPORT_FORMAT (*Win32Error attribute*), 320

ERROR_NOT_FOUND (*Win32Error attribute*), 320

ERROR_NOT_JOINED (*Win32Error attribute*), 320

ERROR_NOT_LOCKED (*Win32Error attribute*), 320

ERROR_NOT_LOGGED_ON (*Win32Error attribute*), 320

ERROR_NOT_LOGON_PROCESS (*Win32Error attribute*), 321

ERROR_NOT_OWNER (*Win32Error attribute*), 321

ERROR_NOT_QUORUM_CAPABLE (*Win32Error attribute*), 321

ERROR_NOT_QUORUM_CLASS (*Win32Error attribute*), 321

[ERROR_NOT_READY \(Win32Error attribute\), 321](#)
[ERROR_NOT_REGISTRY_FILE \(Win32Error attribute\), 321](#)
[ERROR_NOT_SAFE_MODE_DRIVER \(Win32Error attribute\), 321](#)
[ERROR_NOT_SAFEBOOT_SERVICE \(Win32Error attribute\), 321](#)
[ERROR_NOT_SAME_DEVICE \(Win32Error attribute\), 321](#)
[ERROR_NOT_SNAPSHOT_VOLUME \(Win32Error attribute\), 321](#)
[ERROR_NOT_SUBSTED \(Win32Error attribute\), 321](#)
[ERROR_NOT_SUPPORTED \(Win32Error attribute\), 321](#)
[ERROR_NOT_SUPPORTED_ON_STANDARD_SERVER \(DirectoryStorageError attribute\), 267](#)
[ERROR_NOT_SUPPORTED_ON_STANDARD_SERVER \(Win32Error attribute\), 321](#)
[ERROR_NOT_TINY_STREAM \(Win32Error attribute\), 321](#)
[ERROR_NOHING_TO_TERMINATE \(Win32Error attribute\), 320](#)
[ERROR_NOTIFY_CLEANUP \(Win32Error attribute\), 320](#)
[ERROR_NOTIFY_ENUM_DIR \(Win32Error attribute\), 320](#)
[ERROR_NT_CROSS_ENCRYPTION_REQUIRED \(Win32Error attribute\), 323](#)
[ERROR_NTLM_BLOCKED \(Win32Error attribute\), 323](#)
[ERROR_NULL_LM_PASSWORD \(Win32Error attribute\), 323](#)
[ERROR_OBJECT_ALREADY_EXISTS \(Win32Error attribute\), 323](#)
[ERROR_OBJECT_IN_LIST \(Win32Error attribute\), 323](#)
[ERROR_OBJECT_NAME_EXISTS \(Win32Error attribute\), 323](#)
[ERROR_OBJECT_NO_LONGER_EXISTS \(Win32Error attribute\), 323](#)
[ERROR_OBJECT_NOT_FOUND \(Win32Error attribute\), 323](#)
[ERROR_OLD_WIN_VERSION \(Win32Error attribute\), 323](#)
[ERROR_OPEN_FAILED \(Win32Error attribute\), 323](#)
[ERROR_OPEN_FILES \(Win32Error attribute\), 323](#)
[ERROR_OPERATION_ABORTED \(Win32Error attribute\), 323](#)
[ERROR_OPERATION_NOT_SUPPORTED_IN_TRANSACTION \(Win32Error attribute\), 323](#)
[ERROR_OPLOCK_BREAK_IN_PROGRESS \(Win32Error attribute\), 323](#)
[ERROR_OPLOCK_NOT_GRANTED \(Win32Error attribute\), 323](#)
[ERROR_OUT_OF_PAPER \(Win32Error attribute\), 323](#)
[ERROR_OUT_OF_STRUCTURES \(Win32Error attribute\), 323](#)
[ERROR_OUTOFMEMORY \(Win32Error attribute\), 323](#)
[ERROR_PAGE_FAULT_COPY_ON_WRITE \(Win32Error attribute\), 323](#)
[ERROR_PAGE_FAULT_DEMAND_ZERO \(Win32Error attribute\), 323](#)
[ERROR_PAGE_FAULT_GUARD_PAGE \(Win32Error attribute\), 323](#)
[ERROR_PAGE_FAULT_PAGING_FILE \(Win32Error attribute\), 323](#)
[ERROR_PAGE_FAULT_TRANSITION \(Win32Error attribute\), 323](#)
[ERROR_PAGED_SYSTEM_RESOURCES \(Win32Error attribute\), 323](#)
[ERROR_PAGEFILE_CREATE_FAILED \(Win32Error attribute\), 323](#)
[ERROR_PAGEFILE_QUOTA \(Win32Error attribute\), 323](#)
[ERROR_PAGEFILE_QUOTA_EXCEEDED \(Win32Error attribute\), 323](#)
[ERROR_PARTIAL_COPY \(Win32Error attribute\), 323](#)
[ERROR_PARTITION_FAILURE \(Win32Error attribute\), 323](#)
[ERROR_PASSWORD_EXPIRED \(Win32Error attribute\), 323](#)
[ERROR_PASSWORD_MUST_CHANGE \(Win32Error attribute\), 323](#)
[ERROR_PASSWORD_RESTRICTION \(Win32Error attribute\), 324](#)
[ERROR_PATCH_MANAGED_ADVERTISED_PRODUCT \(Win32Error attribute\), 324](#)
[ERROR_PATCH_NO_SEQUENCE \(Win32Error attribute\), 324](#)
[ERROR_PATCH_PACKAGE_INVALID \(Win32Error attribute\), 324](#)
[ERROR_PATCH_PACKAGE_OPEN_FAILED \(Win32Error attribute\), 324](#)
[ERROR_PATCH_PACKAGE_REJECTED \(Win32Error attribute\), 324](#)
[ERROR_PATCH_PACKAGE_UNSUPPORTED \(Win32Error attribute\), 324](#)
[ERROR_PATCH_REMOVAL_DISALLOWED \(Win32Error attribute\), 324](#)
[ERROR_PATCH_REMOVAL_UNSUPPORTED \(Win32Error attribute\), 324](#)
[ERROR_PATCH_TARGET_NOT_FOUND \(Win32Error attribute\), 324](#)
[ERROR_PATH_BUSY \(Win32Error attribute\), 324](#)
[ERROR_PATH_NOT_FOUND \(Win32Error attribute\), 324](#)
[ERROR_PER_USER_TRUST_QUOTA_EXCEEDED \(Win32Error attribute\), 324](#)
[ERROR_PIPE_BUSY \(Win32Error attribute\), 324](#)
[ERROR_PIPE_CONNECTED \(Win32Error attribute\), 324](#)

324
ERROR_PIPE_LISTENING (*Win32Error attribute*), 324
ERROR_PIPE_LOCAL (*Win32Error attribute*), 324
ERROR_PIPE_NOT_CONNECTED (*Win32Error attribute*), 324
ERROR_PLUGPLAY_QUERY_VETOED (*Win32Error attribute*), 324
ERROR_PNP_BAD_MPS_TABLE (*Win32Error attribute*), 324
ERROR_PNP_INVALID_ID (*Win32Error attribute*), 324
ERROR_PNP_IRQ_TRANSLATION_FAILED (*Win32Error attribute*), 324
ERROR_PNP_REBOOT_REQUIRED (*Win32Error attribute*), 324
ERROR_PNP_RESTART_ENUMERATION (*Win32Error attribute*), 324
ERROR_PNP_TRANSLATION_FAILED (*Win32Error attribute*), 324
ERROR_POINT_NOT_FOUND (*Win32Error attribute*), 324
ERROR_POLICY_OBJECT_NOT_FOUND (*DirectoryStorageError attribute*), 267
ERROR_POLICY_OBJECT_NOT_FOUND (*Win32Error attribute*), 324
ERROR_POLICY_ONLY_IN_DS (*DirectoryStorageError attribute*), 267
ERROR_POLICY_ONLY_IN_DS (*Win32Error attribute*), 324
ERROR_POPUP_ALREADY_ACTIVE (*Win32Error attribute*), 324
ERROR_PORT_MESSAGE_TOO_LONG (*Win32Error attribute*), 324
ERROR_PORT_NOT_SET (*Win32Error attribute*), 324
ERROR_PORT_UNREACHABLE (*Win32Error attribute*), 324
ERROR_POSSIBLE_DEADLOCK (*Win32Error attribute*), 324
ERROR_PREDEFINED_HANDLE (*Win32Error attribute*), 324
ERROR_PRIMARY_TRANSPORT_CONNECT_FAILED (*Win32Error attribute*), 324
ERROR_PRINT_CANCELLED (*Win32Error attribute*), 325
ERROR_PRINT_JOB_RESTART_REQUIRED (*Win32Error attribute*), 325
ERROR_PRINT_MONITOR_ALREADY_INSTALLED (*Win32Error attribute*), 325
ERROR_PRINT_MONITOR_IN_USE (*Win32Error attribute*), 325
ERROR_PRINT_PROCESSOR_ALREADY_INSTALLED (*Win32Error attribute*), 325
ERROR_PRINTER_ALREADY_EXISTS (*Win32Error attribute*), 324
ERROR_PRINTER_DELETED (*Win32Error attribute*), 325
ERROR_PRINTER_DRIVER_ALREADY_INSTALLED (*Win32Error attribute*), 325
ERROR_PRINTER_DRIVER_BLOCKED (*Win32Error attribute*), 325
ERROR_PRINTER_DRIVER_DOWNLOAD_NEEDED (*Win32Error attribute*), 325
ERROR_PRINTER_DRIVER_IN_USE (*Win32Error attribute*), 325
ERROR_PRINTER_DRIVER_PACKAGE_IN_USE (*Win32Error attribute*), 325
ERROR_PRINTER_DRIVER_WARNED (*Win32Error attribute*), 325
ERROR_PRINTER_HAS_JOBS_QUEUED (*Win32Error attribute*), 325
ERROR_PRINTER_NOT_FOUND (*Win32Error attribute*), 325
ERROR_PRINTQ_FULL (*Win32Error attribute*), 325
ERROR_PRIVATE_DIALOG_INDEX (*Win32Error attribute*), 325
ERROR_PRIVILEGE_NOT_HELD (*Win32Error attribute*), 325
ERROR_PROC_NOT_FOUND (*Win32Error attribute*), 325
ERROR_PROCESS_ABORTED (*Win32Error attribute*), 325
ERROR_PROCESS_IN_JOB (*Win32Error attribute*), 325
ERROR_PROCESS_MODE_ALREADY_BACKGROUND (*Win32Error attribute*), 325
ERROR_PROCESS_MODE_NOT_BACKGROUND (*Win32Error attribute*), 325
ERROR_PROCESS_NOT_IN_JOB (*Win32Error attribute*), 325
ERROR_PRODUCT_UNINSTALLED (*Win32Error attribute*), 325
ERROR_PRODUCT_VERSION (*Win32Error attribute*), 325
ERROR_PROFILE_DOES_NOT_MATCH_DEVICE (*Win32Error attribute*), 325
ERROR_PROFILE_NOT_ASSOCIATED_WITH_DEVICE (*Win32Error attribute*), 325
ERROR_PROFILE_NOT_FOUND (*Win32Error attribute*), 325
ERROR_PROFILING_AT_LIMIT (*Win32Error attribute*), 325
ERROR_PROFILING_NOT_STARTED (*Win32Error attribute*), 325
ERROR_PROFILING_NOT_STOPPED (*Win32Error attribute*), 325
ERROR_PROMOTION_ACTIVE (*DirectoryStorageError attribute*), 267

ERROR_PROMOTION_ACTIVE (*Win32Error attribute*), 325
 ERROR_PROTOCOL_UNREACHABLE (*Win32Error attribute*), 325
 ERROR_PWD_HISTORY_CONFLICT (*Win32Error attribute*), 325
 ERROR_PWD_TOO_RECENT (*Win32Error attribute*), 325
 ERROR_PWD_TOO_SHORT (*Win32Error attribute*), 325
 ERROR_QUORUM_DISK_NOT_FOUND (*Win32Error attribute*), 326
 ERROR_QUORUM_NOT_ALLOWED_IN_THIS_GROUP (*Win32Error attribute*), 326
 ERROR_QUORUM_OWNER_ALIVE (*Win32Error attribute*), 326
 ERROR_QUORUM_RESOURCE (*Win32Error attribute*), 326
 ERROR_QUORUM_RESOURCE_ONLINE_FAILED (*Win32Error attribute*), 326
 ERROR_QUORUMLOG_OPEN_FAILED (*Win32Error attribute*), 326
 ERROR_QUOTA_LIST_INCONSISTENT (*Win32Error attribute*), 326
 ERROR_RANGE_LIST_CONFLICT (*Win32Error attribute*), 326
 ERROR_RANGE_NOT_FOUND (*Win32Error attribute*), 326
 ERROR_RDP_PROTOCOL_ERROR (*Win32Error attribute*), 326
 ERROR_READ_FAULT (*Win32Error attribute*), 326
 ERROR_REC_NON_EXISTENT (*Win32Error attribute*), 326
 ERROR_RECEIVE_EXPEDITED (*Win32Error attribute*), 326
 ERROR_RECEIVE_PARTIAL (*Win32Error attribute*), 326
 ERROR_RECEIVE_PARTIAL_EXPEDITED (*Win32Error attribute*), 326
 ERROR_RECOVERY_NOT_NEEDED (*Win32Error attribute*), 326
 ERROR_REDIR_PAUSED (*Win32Error attribute*), 326
 ERROR_REDIRECTOR_HAS_OPEN_HANDLES (*Win32Error attribute*), 326
 ERROR_REGISTRY_CORRUPT (*Win32Error attribute*), 326
 ERROR_REGISTRY_HIVE_RECOVERED (*Win32Error attribute*), 326
 ERROR_REGISTRY_IO_FAILED (*Win32Error attribute*), 326
 ERROR_REGISTRY_QUOTA_LIMIT (*Win32Error attribute*), 326
 ERROR_REGISTRY_RECOVERED (*Win32Error attribute*), 326
 ERROR_RELOC_CHAIN_XEEDS_SEGLIM (*Win32Error attribute*), 326
 ERROR_REM_NOT_LIST (*Win32Error attribute*), 326
 ERROR_REMOTE_FILE_VERSION_MISMATCH (*Win32Error attribute*), 326
 ERROR_REMOTE_PRINT_CONNECTIONS_BLOCKED (*Win32Error attribute*), 326
 ERROR_REMOTE_SESSION_LIMIT_EXCEEDED (*Win32Error attribute*), 326
 ERROR_REMOTE_STORAGE_MEDIA_ERROR (*Win32Error attribute*), 326
 ERROR_REMOTE_STORAGE_NOT_ACTIVE (*Win32Error attribute*), 326
 ERROR_REPARSE (*Win32Error attribute*), 326
 ERROR_REPARSE_ATTRIBUTE_CONFLICT (*Win32Error attribute*), 326
 ERROR_REPARSE_OBJECT (*Win32Error attribute*), 326
 ERROR_REPARSE_TAG_INVALID (*Win32Error attribute*), 326
 ERROR_REPARSE_TAG_MISMATCH (*Win32Error attribute*), 326
 ERROR_REPLY_MESSAGE_MISMATCH (*Win32Error attribute*), 326
 ERROR_REQ_NOT_ACCEP (*Win32Error attribute*), 327
 ERROR_REQUEST_ABORTED (*Win32Error attribute*), 327
 ERROR_REQUEST_OUT_OF_SEQUENCE (*Win32Error attribute*), 327
 ERROR_REQUEST_REFUSED (*Win32Error attribute*), 327
 ERROR_REQUIRES_INTERACTIVE_WINDOWSTATION (*Win32Error attribute*), 327
 ERROR_RESMON_CREATE_FAILED (*Win32Error attribute*), 327
 ERROR_RESMON_INVALID_STATE (*Win32Error attribute*), 327
 ERROR_RESMON_ONLINE_FAILED (*Win32Error attribute*), 327
 ERROR_RESOURCE_CALL_TIMED_OUT (*Win32Error attribute*), 327
 ERROR_RESOURCE_DATA_NOT_FOUND (*Win32Error attribute*), 327
 ERROR_RESOURCE_DISABLED (*Win32Error attribute*), 327
 ERROR_RESOURCE_FAILED (*Win32Error attribute*), 327
 ERROR_RESOURCE_LANG_NOT_FOUND (*Win32Error attribute*), 327
 ERROR_RESOURCE_NAME_NOT_FOUND (*Win32Error attribute*), 327
 ERROR_RESOURCE_NOT_AVAILABLE (*Win32Error attribute*), 327
 ERROR_RESOURCE_NOT_FOUND (*Win32Error attribute*), 327

`ERROR_RESOURCE_NOT_ONLINE` (*Win32Error attribute*), 327

`ERROR_RESOURCE_NOT_PRESENT` (*Win32Error attribute*), 327

`ERROR_RESOURCE_ONLINE` (*Win32Error attribute*), 327

`ERROR_RESOURCE_PROPERTIES_STORED` (*Win32Error attribute*), 327

`ERROR_RESOURCE_PROPERTY_UNCHANGEABLE` (*Win32Error attribute*), 327

`ERROR_RESOURCE_REQUIREMENTS_CHANGED` (*Win32Error attribute*), 327

`ERROR_RESOURCE_TYPE_NOT_FOUND` (*Win32Error attribute*), 327

`ERROR_RESOURCEMANAGER_NOT_FOUND` (*Win32Error attribute*), 327

`ERROR_RESOURCEMANAGER_READ_ONLY` (*Win32Error attribute*), 327

`ERROR_RESTART_APPLICATION` (*Win32Error attribute*), 327

`ERROR_RESUME_HIBERNATION` (*Win32Error attribute*), 327

`ERROR_RETRY` (*Win32Error attribute*), 327

`ERROR_REVISION_MISMATCH` (*Win32Error attribute*), 327

`ERROR_RING2_STACK_IN_USE` (*Win32Error attribute*), 327

`ERROR_RING2SEG_MUST_BE_MOVABLE` (*Win32Error attribute*), 327

`ERROR_RM_ALREADY_STARTED` (*Win32Error attribute*), 327

`ERROR_RM_DISCONNECTED` (*Win32Error attribute*), 327

`ERROR_RM_METADATA_CORRUPT` (*Win32Error attribute*), 327

`ERROR_RM_NOT_ACTIVE` (*Win32Error attribute*), 327

`ERROR_RMODE_APP` (*Win32Error attribute*), 327

`ERROR_ROLLBACK_TIMER_EXPIRED` (*Win32Error attribute*), 328

`ERROR_ROWSNOTRELEASED` (*Win32Error attribute*), 328

`ERROR_RPL_NOT_ALLOWED` (*Win32Error attribute*), 328

`ERROR_RXACT_COMMIT_FAILURE` (*Win32Error attribute*), 328

`ERROR_RXACT_COMMIT_NECESSARY` (*Win32Error attribute*), 328

`ERROR_RXACT_COMMITTED` (*Win32Error attribute*), 328

`ERROR_RXACT_INVALID_STATE` (*Win32Error attribute*), 328

`ERROR_RXACT_STATE_CREATED` (*Win32Error attribute*), 328

`ERROR_SAM_INIT_FAILURE` (*DirectoryStorageError attribute*), 267

`ERROR_SAM_INIT_FAILURE` (*Win32Error attribute*), 328

`ERROR_SAME_DRIVE` (*Win32Error attribute*), 328

`ERROR_SCOPE_NOT_FOUND` (*Win32Error attribute*), 328

`ERROR_SCREEN_ALREADY_LOCKED` (*Win32Error attribute*), 328

`ERROR_SECRET_TOO_LONG` (*Win32Error attribute*), 328

`ERROR_SECTOR_NOT_FOUND` (*Win32Error attribute*), 328

`ERROR_SEEK` (*Win32Error attribute*), 328

`ERROR_SEEK_ON_DEVICE` (*Win32Error attribute*), 328

`ERROR_SEGMENT_NOTIFICATION` (*Win32Error attribute*), 328

`ERROR_SEM_IS_SET` (*Win32Error attribute*), 328

`ERROR_SEM_NOT_FOUND` (*Win32Error attribute*), 328

`ERROR_SEM_OWNER_DIED` (*Win32Error attribute*), 328

`ERROR_SEM_TIMEOUT` (*Win32Error attribute*), 328

`ERROR_SEM_USER_LIMIT` (*Win32Error attribute*), 328

`ERROR_SERIAL_NO_DEVICE` (*Win32Error attribute*), 328

`ERROR_SERVER_DISABLED` (*Win32Error attribute*), 328

`ERROR_SERVER_HAS_OPEN_HANDLES` (*Win32Error attribute*), 328

`ERROR_SERVER_NOT_DISABLED` (*Win32Error attribute*), 328

`ERROR_SERVER_SID_MISMATCH` (*Win32Error attribute*), 328

`ERROR_SERVICE_ALREADY_RUNNING` (*Win32Error attribute*), 328

`ERROR_SERVICE_CANNOT_ACCEPT_CTRL` (*Win32Error attribute*), 328

`ERROR_SERVICE_DATABASE_LOCKED` (*Win32Error attribute*), 328

`ERROR_SERVICE_DEPENDENCY_DELETED` (*Win32Error attribute*), 328

`ERROR_SERVICE_DEPENDENCY_FAIL` (*Win32Error attribute*), 328

`ERROR_SERVICE_DISABLED` (*Win32Error attribute*), 328

`ERROR_SERVICE_DOES_NOT_EXIST` (*Win32Error attribute*), 328

`ERROR_SERVICE_EXISTS` (*Win32Error attribute*), 328

`ERROR_SERVICE_LOGON_FAILED` (*Win32Error attribute*), 328

`ERROR_SERVICE_MARKED_FOR_DELETE` (*Win32Error attribute*), 329

ERROR_SERVICE_NEVER_STARTED (*Win32Error attribute*), 329
 ERROR_SERVICE_NO_THREAD (*Win32Error attribute*), 329
 ERROR_SERVICE_NOT_ACTIVE (*Win32Error attribute*), 329
 ERROR_SERVICE_NOT_FOUND (*Win32Error attribute*), 329
 ERROR_SERVICE_NOT_IN_EXE (*Win32Error attribute*), 329
 ERROR_SERVICE_NOTIFICATION (*Win32Error attribute*), 329
 ERROR_SERVICE_REQUEST_TIMEOUT (*Win32Error attribute*), 329
 ERROR_SERVICE_SPECIFIC_ERROR (*Win32Error attribute*), 329
 ERROR_SERVICE_START_HANG (*Win32Error attribute*), 329
 ERROR_SESSION_CREDENTIAL_CONFLICT (*Win32Error attribute*), 329
 ERROR_SET_NOT_FOUND (*Win32Error attribute*), 329
 ERROR_SET_POWER_STATE_FAILED (*Win32Error attribute*), 329
 ERROR_SET_POWER_STATE_VETOED (*Win32Error attribute*), 329
 ERROR_SETCOUNT_ON_BAD_LB (*Win32Error attribute*), 329
 ERROR_SETMARK_DETECTED (*Win32Error attribute*), 329
 ERROR_SHARED_POLICY (*DirectoryStorageError attribute*), 267
 ERROR_SHARED_POLICY (*Win32Error attribute*), 329
 ERROR_SHARING_BUFFER_EXCEEDED (*Win32Error attribute*), 329
 ERROR_SHARING_PAUSED (*Win32Error attribute*), 329
 ERROR_SHARING_VIOLATION (*Win32Error attribute*), 329
 ERROR_SHUTDOWN_CLUSTER (*Win32Error attribute*), 329
 ERROR_SHUTDOWN_IN_PROGRESS (*Win32Error attribute*), 329
 ERROR_SIGNAL_PENDING (*Win32Error attribute*), 329
 ERROR_SIGNAL_REFUSED (*Win32Error attribute*), 329
 ERROR_SINGLE_INSTANCE_APP (*Win32Error attribute*), 329
 ERROR_SOME_NOT_MAPPED (*Win32Error attribute*), 329
 ERROR_SOURCE_ELEMENT_EMPTY (*Win32Error attribute*), 329
 ERROR_SPARSE_NOT_ALLOWED_IN_TRANSACTION (*Win32Error attribute*), 329
 ERROR_SPECIAL_ACCOUNT (*Win32Error attribute*), 329
 ERROR_SPECIAL_GROUP (*Win32Error attribute*), 329
 ERROR_SPECIAL_USER (*Win32Error attribute*), 329
 ERROR_SPL_NO_ADDJOB (*Win32Error attribute*), 329
 ERROR_SPL_NO_STARTDOC (*Win32Error attribute*), 329
 ERROR_SPOOL_FILE_NOT_FOUND (*Win32Error attribute*), 329
 ERROR_STACK_OVERFLOW (*Win32Error attribute*), 329
 ERROR_STACK_OVERFLOW_READ (*Win32Error attribute*), 329
 ERROR_STATIC_INIT (*Win32Error attribute*), 330
 ERROR_STOPPED_ON_SYMLINK (*Win32Error attribute*), 330
 ERROR_STREAM_MINIVERSION_NOT_FOUND (*Win32Error attribute*), 330
 ERROR_STREAM_MINIVERSION_NOT_VALID (*Win32Error attribute*), 330
 ERROR_SUBST_TO_JOIN (*Win32Error attribute*), 330
 ERROR_SUBST_TO_SUBST (*Win32Error attribute*), 330
 ERROR_SUCCESS (*Win32Error attribute*), 330
 ERROR_SUCCESS_REBOOT_INITIATED (*Win32Error attribute*), 330
 ERROR_SUCCESS_REBOOT_REQUIRED (*Win32Error attribute*), 330
 ERROR_SUCCESS_RESTART_REQUIRED (*Win32Error attribute*), 330
 ERROR_SWAPERROR (*Win32Error attribute*), 330
 ERROR_SYMLINK_CLASS_DISABLED (*Win32Error attribute*), 330
 ERROR_SYMLINK_NOT_SUPPORTED (*Win32Error attribute*), 330
 ERROR_SYNCHRONIZATION_REQUIRED (*Win32Error attribute*), 330
 ERROR_SYSTEM_HIVE_TOO_LARGE (*Win32Error attribute*), 330
 ERROR_SYSTEM_IMAGE_BAD_SIGNATURE (*Win32Error attribute*), 330
 ERROR_SYSTEM_POWERSTATE_COMPLEX_TRANSITION (*Win32Error attribute*), 330
 ERROR_SYSTEM_POWERSTATE_TRANSITION (*Win32Error attribute*), 330
 ERROR_SYSTEM_PROCESS_TERMINATED (*Win32Error attribute*), 330
 ERROR_SYSTEM_SHUTDOWN (*Win32Error attribute*), 330
 ERROR_SYSTEM_TRACE (*Win32Error attribute*), 330
 ERROR_TAG_NOT_FOUND (*Win32Error attribute*), 330
 ERROR_TAG_NOT_PRESENT (*Win32Error attribute*), 330
 ERROR_THREAD_1_INACTIVE (*Win32Error attribute*), 330

tribute), 330

ERROR_THREAD_MODE_ALREADY_BACKGROUND (*Win32Error attribute*), 330

ERROR_THREAD_MODE_NOT_BACKGROUND (*Win32Error attribute*), 330

ERROR_THREAD_NOT_IN_PROCESS (*Win32Error attribute*), 330

ERROR_THREAD_WAS_SUSPENDED (*Win32Error attribute*), 330

ERROR_TIMEOUT (*Win32Error attribute*), 330

ERROR_TIMER_NOT_CANCELED (*Win32Error attribute*), 330

ERROR_TIMER_RESOLUTION_NOT_SET (*Win32Error attribute*), 330

ERROR_TIMER_RESUME_IGNORED (*Win32Error attribute*), 330

ERROR_TLW_WITH_WSCHILD (*Win32Error attribute*), 330

ERROR_TM_IDENTITY_MISMATCH (*Win32Error attribute*), 330

ERROR_TM_INITIALIZATION_FAILED (*Win32Error attribute*), 330

ERROR_TM_VOLATILE (*Win32Error attribute*), 330

ERROR_TOKEN_ALREADY_IN_USE (*Win32Error attribute*), 331

ERROR_TOO_MANY_CMDS (*Win32Error attribute*), 331

ERROR_TOO_MANY_CONTEXT_IDS (*Win32Error attribute*), 331

ERROR_TOO_MANY_LINKS (*Win32Error attribute*), 331

ERROR_TOO_MANY_LUIDS_REQUESTED (*Win32Error attribute*), 331

ERROR_TOO_MANY_MODULES (*Win32Error attribute*), 331

ERROR_TOO_MANY_MUXWAITERS (*Win32Error attribute*), 331

ERROR_TOO_MANY_NAMES (*Win32Error attribute*), 331

ERROR_TOO_MANY_OPEN_FILES (*Win32Error attribute*), 331

ERROR_TOO_MANY_POSTS (*Win32Error attribute*), 331

ERROR_TOO_MANY_SECRETS (*Win32Error attribute*), 331

ERROR_TOO_MANY_SEM_REQUESTS (*Win32Error attribute*), 331

ERROR_TOO_MANY_SEMAPHORES (*Win32Error attribute*), 331

ERROR_TOO_MANY_SESS (*Win32Error attribute*), 331

ERROR_TOO_MANY_SIDS (*Win32Error attribute*), 331

ERROR_TOO_MANY_TCBS (*Win32Error attribute*), 331

ERROR_TOO_MANY_THREADS (*Win32Error attribute*), 331

ERROR_TRANSACTIONAL_MAPPING_UNSUPPORTED_REMOTE (*Win32Error attribute*), 331

ERROR_TRANSACTION_ALREADY_ABORTED (*Win32Error attribute*), 331

ERROR_TRANSACTION_ALREADY_COMMITTED (*Win32Error attribute*), 331

ERROR_TRANSACTION_FREEZE_IN_PROGRESS (*Win32Error attribute*), 331

ERROR_TRANSACTION_INTEGRITY_VIOLATED (*Win32Error attribute*), 331

ERROR_TRANSACTION_INVALID_MARSHALL_BUFFER (*Win32Error attribute*), 331

ERROR_TRANSACTION_NOT_ACTIVE (*Win32Error attribute*), 331

ERROR_TRANSACTION_NOT_FOUND (*Win32Error attribute*), 331

ERROR_TRANSACTION_NOT_JOINED (*Win32Error attribute*), 331

ERROR_TRANSACTION_NOT_REQUESTED (*Win32Error attribute*), 331

ERROR_TRANSACTION_NOT_ROOT (*Win32Error attribute*), 331

ERROR_TRANSACTION_OBJECT_EXPIRED (*Win32Error attribute*), 331

ERROR_TRANSACTION_PROPAGATION_FAILED (*Win32Error attribute*), 332

ERROR_TRANSACTION_RECORD_TOO_LONG (*Win32Error attribute*), 332

ERROR_TRANSACTION_REQUEST_NOT_VALID (*Win32Error attribute*), 332

ERROR_TRANSACTION_REQUIRED_PROMOTION (*Win32Error attribute*), 332

ERROR_TRANSACTION_RESPONSE_NOT_ENLISTED (*Win32Error attribute*), 332

ERROR_TRANSACTION_SCOPE_CALLBACKS_NOT_SET (*Win32Error attribute*), 332

ERROR_TRANSACTION_SUPERIOR_EXISTS (*Win32Error attribute*), 332

ERROR_TRANSACTIONAL_CONFLICT (*Win32Error attribute*), 331

ERROR_TRANSACTIONAL_OPEN_NOT_ALLOWED (*Win32Error attribute*), 331

ERROR_TRANSACTIONMANAGER_NOT_FOUND (*Win32Error attribute*), 331

ERROR_TRANSACTIONMANAGER_NOT_ONLINE (*Win32Error attribute*), 331

ERROR_TRANSACTIONMANAGER_RECOVERY_NAME_COLLISION (*Win32Error attribute*), 331

ERROR_TRANSACTIONS_NOT_FROZEN (*Win32Error attribute*), 331

ERROR_TRANSACTIONS_UNSUPPORTED_REMOTE (*Win32Error attribute*), 331

ERROR_TRANSFORM_NOT_SUPPORTED (*Win32Error attribute*), 332

ERROR_TRANSLATION_COMPLETE (*Win32Error attribute*), 331

tribute), 332

ERROR_TRANSPORT_FULL (*Win32Error attribute*), 332

ERROR_TRUST_FAILURE (*Win32Error attribute*), 332

ERROR_TRUSTED_DOMAIN_FAILURE (*Win32Error attribute*), 332

ERROR_TRUSTED_RELATIONSHIP_FAILURE (*Win32Error attribute*), 332

ERROR_TS_INCOMPATIBLE_SESSIONS (*Win32Error attribute*), 332

ERROR_TXF_ATTRIBUTE_CORRUPT (*Win32Error attribute*), 332

ERROR_TXF_DIR_NOT_EMPTY (*Win32Error attribute*), 332

ERROR_TXF_METADATA_ALREADY_PRESENT (*Win32Error attribute*), 332

ERROR_UNABLE_TO_CLEAN (*Win32Error attribute*), 332

ERROR_UNABLE_TO_EJECT_MOUNTED_MEDIA (*Win32Error attribute*), 332

ERROR_UNABLE_TO_INVENTORY_DRIVE (*Win32Error attribute*), 332

ERROR_UNABLE_TO_INVENTORY_SLOT (*Win32Error attribute*), 332

ERROR_UNABLE_TO_INVENTORY_TRANSPORT (*Win32Error attribute*), 332

ERROR_UNABLE_TO_LOAD_MEDIUM (*Win32Error attribute*), 332

ERROR_UNABLE_TO_LOCK_MEDIA (*Win32Error attribute*), 332

ERROR_UNABLE_TO_UNLOAD_MEDIA (*Win32Error attribute*), 332

ERROR_UNDEFINED_CHARACTER (*Win32Error attribute*), 332

ERROR_UNEXP_NET_ERR (*Win32Error attribute*), 332

ERROR_UNEXPECTED_MM_CREATE_ERR (*Win32Error attribute*), 332

ERROR_UNEXPECTED_MM_EXTEND_ERR (*Win32Error attribute*), 332

ERROR_UNEXPECTED_MM_MAP_ERROR (*Win32Error attribute*), 332

ERROR_UNEXPECTED_OMID (*Win32Error attribute*), 332

ERROR_UNHANDLED_EXCEPTION (*Win32Error attribute*), 332

ERROR_UNKNOWN_COMPONENT (*Win32Error attribute*), 332

ERROR_UNKNOWN_FEATURE (*Win32Error attribute*), 332

ERROR_UNKNOWN_PATCH (*Win32Error attribute*), 332

ERROR_UNKNOWN_PORT (*Win32Error attribute*), 332

ERROR_UNKNOWN_PRINT_MONITOR (*Win32Error attribute*), 333

ERROR_UNKNOWN_PRINTER_DRIVER (*Win32Error attribute*), 333

ERROR_UNKNOWN_PRINTPROCESSOR (*Win32Error attribute*), 333

ERROR_UNKNOWN_PRODUCT (*Win32Error attribute*), 333

ERROR_UNKNOWN_PROPERTY (*Win32Error attribute*), 333

ERROR_UNKNOWN_REVISION (*Win32Error attribute*), 333

ERROR_UNRECOGNIZED_MEDIA (*Win32Error attribute*), 333

ERROR_UNRECOGNIZED_VOLUME (*Win32Error attribute*), 333

ERROR_UNSUPPORTED_COMPRESSION (*Win32Error attribute*), 333

ERROR_UNSUPPORTED_TYPE (*Win32Error attribute*), 333

ERROR_UNWIND (*Win32Error attribute*), 333

ERROR_UNWIND_CONSOLIDATE (*Win32Error attribute*), 333

ERROR_USER_APC (*Win32Error attribute*), 333

ERROR_USER_DELETE_TRUST_QUOTA_EXCEEDED (*Win32Error attribute*), 333

ERROR_USER_EXISTS (*Win32Error attribute*), 333

ERROR_USER_MAPPED_FILE (*Win32Error attribute*), 333

ERROR_USER_PROFILE_LOAD (*Win32Error attribute*), 333

ERROR_VALIDATE_CONTINUE (*Win32Error attribute*), 333

ERROR_VC_DISCONNECTED (*Win32Error attribute*), 333

ERROR_VDM_HARD_ERROR (*Win32Error attribute*), 333

ERROR_VERIFIER_STOP (*Win32Error attribute*), 333

ERROR_VERSION_PARSE_ERROR (*Win32Error attribute*), 333

ERROR_VIRUS_DELETED (*Win32Error attribute*), 333

ERROR_VIRUS_INFECTED (*Win32Error attribute*), 333

ERROR_VOLSnap_HIBERNATE_READY (*Win32Error attribute*), 333

ERROR_VOLSnap_PREPARE_HIBERNATE (*Win32Error attribute*), 333

ERROR_VOLUME_CONTAINS_SYS_FILES (*Win32Error attribute*), 333

ERROR_VOLUME_DIRTY (*Win32Error attribute*), 333

ERROR_VOLUME_MOUNTED (*Win32Error attribute*), 333

ERROR_VOLUME_NOT_SIS_ENABLED (*Win32Error attribute*), 333

ERROR_VOLUME_NOT_SUPPORT_EFS (*Win32Error attribute*), 333

ERROR_WAIT_1 (*Win32Error attribute*), 333

`ERROR_WAIT_2` (*Win32Error attribute*), 333
`ERROR_WAIT_3` (*Win32Error attribute*), 333
`ERROR_WAIT_63` (*Win32Error attribute*), 333
`ERROR_WAIT_FOR_OPLOCK` (*Win32Error attribute*), 333
`ERROR_WAIT_NO_CHILDREN` (*Win32Error attribute*), 334
`ERROR_WAKE_SYSTEM` (*Win32Error attribute*), 334
`ERROR_WAKE_SYSTEM_DEBUGGER` (*Win32Error attribute*), 334
`ERROR_WAS_LOCKED` (*Win32Error attribute*), 334
`ERROR_WAS_UNLOCKED` (*Win32Error attribute*), 334
`ERROR_WINDOW_NOT_COMBOBOX` (*Win32Error attribute*), 334
`ERROR_WINDOW_NOT_DIALOG` (*Win32Error attribute*), 334
`ERROR_WINDOW_OF_OTHER_THREAD` (*Win32Error attribute*), 334
`ERROR_WINS_INTERNAL` (*Win32Error attribute*), 334
`ERROR_WMI_ALREADY_DISABLED` (*Win32Error attribute*), 334
`ERROR_WMI_ALREADY_ENABLED` (*Win32Error attribute*), 334
`ERROR_WMI_DP_FAILED` (*Win32Error attribute*), 334
`ERROR_WMI_DP_NOT_FOUND` (*Win32Error attribute*), 334
`ERROR_WMI_GUID_DISCONNECTED` (*Win32Error attribute*), 334
`ERROR_WMI_GUID_NOT_FOUND` (*Win32Error attribute*), 334
`ERROR_WMI_INSTANCE_NOT_FOUND` (*Win32Error attribute*), 334
`ERROR_WMI_INVALID_MOF` (*Win32Error attribute*), 334
`ERROR_WMI_INVALID_REGINFO` (*Win32Error attribute*), 334
`ERROR_WMI_ITEMID_NOT_FOUND` (*Win32Error attribute*), 334
`ERROR_WMI_READ_ONLY` (*Win32Error attribute*), 334
`ERROR_WMI_SERVER_UNAVAILABLE` (*Win32Error attribute*), 334
`ERROR_WMI_SET_FAILURE` (*Win32Error attribute*), 334
`ERROR_WMI_TRY_AGAIN` (*Win32Error attribute*), 334
`ERROR_WMI_UNRESOLVED_INSTANCE_REF` (*Win32Error attribute*), 334
`ERROR_WORKING_SET_QUOTA` (*Win32Error attribute*), 334
`ERROR_WOW_ASSERTION` (*Win32Error attribute*), 334
`ERROR_WRITE_FAULT` (*Win32Error attribute*), 334
`ERROR_WRITE_PROTECT` (*Win32Error attribute*), 334
`ERROR_WRONG_COMPARTMENT` (*Win32Error attribute*), 334
`ERROR_WRONG_DISK` (*Win32Error attribute*), 334
`ERROR_WRONG_EFS` (*Win32Error attribute*), 334
`ERROR_WRONG_PASSWORD` (*Win32Error attribute*), 334
`ERROR_WX86_ERROR` (*Win32Error attribute*), 334
`ERROR_WX86_WARNING` (*Win32Error attribute*), 334
`ERROR_XML_PARSE_ERROR` (*Win32Error attribute*), 334
`ERROR_XMLDSIG_ERROR` (*Win32Error attribute*), 334
`ErrorBaseClass` (*class in cbc_sdk.winerror*), 267
`ErrorMetaClass` (*class in cbc_sdk.winerror*), 268
`errors` (*Job attribute*), 152
`errors` (*WorkflowStatus attribute*), 130
`Event` (*class in cbc_sdk.endpoint_standard.base*), 78
`Event` (*class in cbc_sdk.platform.events*), 144
`event_count` (*Recommendation.RecommendationImpact attribute*), 79
`event_description` (*Observation attribute*), 155
`event_id` (*AuthEvent attribute*), 92
`event_id` (*Observation attribute*), 155
`event_network_inbound` (*Observation attribute*), 155
`event_network_local_ipv4` (*Observation attribute*), 155
`event_network_location` (*Observation attribute*), 155
`event_network_protocol` (*Observation attribute*), 155
`event_network_remote_ipv4` (*Observation attribute*), 155
`event_network_remote_port` (*Observation attribute*), 155
`event_type` (*Observation attribute*), 155
`EventFacet` (*class in cbc_sdk.platform.events*), 144
`EventFacet.Ranges` (*class in cbc_sdk.platform.events*), 145
`EventFacet.Terms` (*class in cbc_sdk.platform.events*), 145
`EventFacetQuery` (*class in cbc_sdk.platform.events*), 145
`EventQuery` (*class in cbc_sdk.platform.events*), 145
`events()` (*Process method*), 174
`exclude_appliance_uuid()` (*VCenterComputeResourceQuery method*), 207
`exclude_auto_scaling_group_name()` (*AWSComputeResourceQuery method*), 198
`exclude_availability_zone()` (*AWSComputeResourceQuery method*), 198
`exclude_cloud_provider_account_id()` (*AWSComputeResourceQuery method*), 198
`exclude_cloud_provider_resource_id()` (*AWSComputeResourceQuery method*), 198
`exclude_cloud_provider_tags()` (*AWSComputeResourceQuery method*), 199
`exclude_cluster_name()` (*VCenterComputeRe-*

- sourceQuery method*), 207
- `exclude_datacenter_name()` (*VCenterComputeResourceQuery method*), 207
- `exclude_device_guid()` (*VCenterComputeResourceQuery method*), 207
- `exclude_eligibility()` (*VCenterComputeResourceQuery method*), 207
- `exclude_eligibility_code()` (*VCenterComputeResourceQuery method*), 207
- `exclude_esx_host_name()` (*VCenterComputeResourceQuery method*), 207
- `exclude_esx_host_uuid()` (*VCenterComputeResourceQuery method*), 208
- `exclude_host_name()` (*VCenterComputeResourceQuery method*), 208
- `exclude_id()` (*AWSComputeResourceQuery method*), 199
- `exclude_installation_status()` (*AWSComputeResourceQuery method*), 199
- `exclude_installation_status()` (*VCenterComputeResourceQuery method*), 208
- `exclude_installation_type()` (*VCenterComputeResourceQuery method*), 208
- `exclude_ip_address()` (*VCenterComputeResourceQuery method*), 208
- `exclude_name()` (*AWSComputeResourceQuery method*), 199
- `exclude_name()` (*VCenterComputeResourceQuery method*), 208
- `exclude_os_architecture()` (*VCenterComputeResourceQuery method*), 208
- `exclude_os_description()` (*VCenterComputeResourceQuery method*), 208
- `exclude_os_type()` (*VCenterComputeResourceQuery method*), 209
- `exclude_platform()` (*AWSComputeResourceQuery method*), 199
- `exclude_platform_details()` (*AWSComputeResourceQuery method*), 199
- `exclude_region()` (*AWSComputeResourceQuery method*), 199
- `exclude_registration_id()` (*VCenterComputeResourceQuery method*), 209
- `exclude_subnet_id()` (*AWSComputeResourceQuery method*), 200
- `exclude_uuid()` (*VCenterComputeResourceQuery method*), 209
- `exclude_vcenter_host_url()` (*VCenterComputeResourceQuery method*), 209
- `exclude_vcenter_name()` (*VCenterComputeResourceQuery method*), 209
- `exclude_vcenter_uuid()` (*VCenterComputeResourceQuery method*), 209
- `exclude_virtual_private_cloud_id()` (*AWSComputeResourceQuery method*), 200
- `exclude_vmwaretools_version()` (*VCenterComputeResourceQuery method*), 209
- `execute_async()` (*AsyncQueryMixin method*), 215
- `expires` (*Grant attribute*), 150
- `expires()` (*SensorKitQuery method*), 196
- `export()` (*VulnerabilityAssetViewQuery method*), 188
- `export()` (*VulnerabilityQuery method*), 190
- `export_csv_as_file()` (*ResultQuery method*), 55
- `export_csv_as_lines()` (*ResultQuery method*), 55
- `export_csv_as_stream()` (*ResultQuery method*), 55
- `export_csv_as_string()` (*ResultQuery method*), 55
- `export_zipped_csv()` (*ResultQuery method*), 55
- ## F
- `facet()` (*BaseComputeResourceQuery method*), 204
- `facet_field()` (*FacetQuery method*), 50
- `FacetQuery` (class in *cbc_sdk.audit_remediation.base*), 50
- `FacetQuery` (class in *cbc_sdk.base*), 217
- `facets` (*AuthEventFacet.Ranges attribute*), 94
- `facets` (*AuthEventFacet.Terms attribute*), 94
- `facets` (*EnrichedEventFacet.Ranges attribute*), 76
- `facets` (*EnrichedEventFacet.Terms attribute*), 76
- `facets` (*EventFacet.Ranges attribute*), 145
- `facets` (*EventFacet.Terms attribute*), 145
- `facets` (*ObservationFacet.Ranges attribute*), 157
- `facets` (*ObservationFacet.Terms attribute*), 157
- `facets` (*ProcessFacet.Ranges attribute*), 177
- `facets` (*ProcessFacet.Terms attribute*), 177
- `facets()` (*BaseAlertSearchQuery method*), 119
- `facets()` (*DeviceSearchQuery method*), 140
- `facets()` (*Process method*), 174
- `facets()` (*USBDeviceQuery method*), 89
- `Facility` (class in *cbc_sdk.winerror*), 268
- `FACILITY_AAF` (*Facility attribute*), 268
- `FACILITY_ACS` (*Facility attribute*), 268
- `FACILITY_BACKGROUNDCOPY` (*Facility attribute*), 268
- `FACILITY_CERT` (*Facility attribute*), 268
- `FACILITY_CMI` (*Facility attribute*), 268
- `FACILITY_COMPLUS` (*Facility attribute*), 268
- `FACILITY_CONFIGURATION` (*Facility attribute*), 268
- `FACILITY_CONTROL` (*Facility attribute*), 268
- `FACILITY_DIRECTORYSERVICE` (*Facility attribute*), 268
- `FACILITY_DISPATCH` (*Facility attribute*), 268
- `FACILITY_DPLAY` (*Facility attribute*), 268
- `FACILITY_FVE` (*Facility attribute*), 268
- `FACILITY_FWP` (*Facility attribute*), 268
- `FACILITY_GRAPHICS` (*Facility attribute*), 268

- FACILITY_HTTP (*Facility attribute*), 268
- FACILITY_INTERNET (*Facility attribute*), 268
- FACILITY_ITF (*Facility attribute*), 268
- FACILITY_MEDIASERVER (*Facility attribute*), 268
- FACILITY_METADIRECTORY (*Facility attribute*), 268
- FACILITY_MSMQ (*Facility attribute*), 268
- FACILITY_NDIS (*Facility attribute*), 268
- FACILITY_NULL (*Facility attribute*), 268
- FACILITY_PLA (*Facility attribute*), 269
- FACILITY_RPC (*Facility attribute*), 269
- FACILITY_SCARD (*Facility attribute*), 269
- FACILITY_SECURITY (*Facility attribute*), 269
- FACILITY_SETUPAPI (*Facility attribute*), 269
- FACILITY_SHELL (*Facility attribute*), 269
- FACILITY_SSPI (*Facility attribute*), 269
- FACILITY_STATE_MANAGEMENT (*Facility attribute*), 269
- FACILITY_STORAGE (*Facility attribute*), 269
- FACILITY_SXS (*Facility attribute*), 269
- FACILITY_TPM_SERVICES (*Facility attribute*), 269
- FACILITY_TPM_SOFTWARE (*Facility attribute*), 269
- FACILITY_UMI (*Facility attribute*), 269
- FACILITY_URT (*Facility attribute*), 269
- FACILITY_USERMODE_COMMONLOG (*Facility attribute*), 269
- FACILITY_USERMODE_FILTER_MANAGER (*Facility attribute*), 269
- FACILITY_USERMODE_HYPERVISOR (*Facility attribute*), 269
- FACILITY_WIN32 (*Facility attribute*), 269
- FACILITY_WINDOWS (*Facility attribute*), 269
- FACILITY_WINDOWS_CE (*Facility attribute*), 269
- FACILITY_WINDOWS_DEFENDER (*Facility attribute*), 269
- FACILITY_WINDOWSUPDATE (*Facility attribute*), 269
- FACILITY_WINRM (*Facility attribute*), 269
- FAILED () (*in module cbc_sdk.winerror*), 268
- failed_ids (*WorkflowStatus attribute*), 130
- false_negatives (*NetworkThreatMetadata attribute*), 154
- false_positives (*NetworkThreatMetadata attribute*), 154
- Feed (*class in cbc_sdk.enterprise_edr.threat_intelligence*), 97
- feed (*Watchlist attribute*), 111
- Feed.FeedBuilder (*class in cbc_sdk.enterprise_edr.threat_intelligence*), 97
- FeedModel (*class in cbc_sdk.enterprise_edr.threat_intelligence*), 100
- FeedQuery (*class in cbc_sdk.enterprise_edr.threat_intelligence*), 100
- fetch_process_queries () (*CBCloudAPI method*), 254
- field (*DeviceFacet attribute*), 139
- field (*IOC_V2 attribute*), 103
- field (*ResultFacet attribute*), 54
- FieldDescriptor (*class in cbc_sdk.base*), 220
- fields (*AuthEventFacet.Ranges attribute*), 94
- fields (*AuthEventFacet.Terms attribute*), 94
- fields (*EnrichedEventFacet.Ranges attribute*), 76
- fields (*EnrichedEventFacet.Terms attribute*), 76
- fields (*EventFacet.Ranges attribute*), 145
- fields (*EventFacet.Terms attribute*), 145
- fields (*ObservationFacet.Ranges attribute*), 157
- fields (*ObservationFacet.Terms attribute*), 157
- fields (*ProcessFacet.Ranges attribute*), 177
- fields (*ProcessFacet.Terms attribute*), 177
- fields (*Result attribute*), 53
- fields_ (*Result attribute*), 53
- file_available (*Binary attribute*), 114
- file_description (*Binary attribute*), 114
- file_size (*Binary attribute*), 114
- file_version (*Binary attribute*), 114
- FileCredentialProvider (*class in cbc_sdk.credential_providers.file_credential_provider*), 70
- filemod_count (*AuthEvent attribute*), 92
- filename (*Recommendation.RecommendationNewRule attribute*), 80
- filename (*ReputationOverride attribute*), 180
- finished (*WorkflowStatus attribute*), 130
- first () (*IterableQueryMixin method*), 220
- first_event_time (*BaseAlert attribute*), 117
- first_name (*Device attribute*), 135
- first_name (*User attribute*), 184
- first_seen (*USBDevice attribute*), 84
- FNERR_BUFFERTOOSMALL (*CommDlgError attribute*), 256
- FNERR_FILENAMECODES (*CommDlgError attribute*), 256
- FNERR_INVALIDFILENAME (*CommDlgError attribute*), 256
- FNERR_SUBCLASSFAILURE (*CommDlgError attribute*), 256
- ForeignKeyFieldDescriptor (*class in cbc_sdk.base*), 220
- found (*Downloads attribute*), 114
- FRERR_BUFFERLENGTHZERO (*CommDlgError attribute*), 256
- FRERR_FINDREPLACECODES (*CommDlgError attribute*), 256
- from_type () (*cbc_sdk.workload.sensor_lifecycle.SensorKit class method*), 195

- FRS_ERR_AUTHENTICATION (Win32Error attribute), 335
- FRS_ERR_CHILD_TO_PARENT_COMM (Win32Error attribute), 335
- FRS_ERR_INSUFFICIENT_PRIV (Win32Error attribute), 335
- FRS_ERR_INTERNAL (Win32Error attribute), 335
- FRS_ERR_INTERNAL_API (Win32Error attribute), 335
- FRS_ERR_INVALID_API_SEQUENCE (Win32Error attribute), 335
- FRS_ERR_INVALID_SERVICE_PARAMETER (Win32Error attribute), 335
- FRS_ERR_PARENT_AUTHENTICATION (Win32Error attribute), 335
- FRS_ERR_PARENT_INSUFFICIENT_PRIV (Win32Error attribute), 335
- FRS_ERR_PARENT_TO_CHILD_COMM (Win32Error attribute), 335
- FRS_ERR_SERVICE_COMM (Win32Error attribute), 335
- FRS_ERR_STARTING_SERVICE (Win32Error attribute), 335
- FRS_ERR_STOPPING_SERVICE (Win32Error attribute), 335
- FRS_ERR_SYSVOL_DEMOTE (Win32Error attribute), 335
- FRS_ERR_SYSVOL_IS_BUSY (Win32Error attribute), 335
- FRS_ERR_SYSVOL_POPULATE (Win32Error attribute), 335
- FRS_ERR_SYSVOL_POPULATE_TIMEOUT (Win32Error attribute), 335
- FunctionalityDecommissioned, 236
- ## G
- get () (Connection method), 233
- get () (LiveResponseMemdump method), 249
- get () (NewBaseModel method), 222
- get_affected_assets () (Vulnerability method), 187
- get_assignment_mode () (CorePreventionRule-Config method), 170
- get_auditlogs () (CBCloudAPI method), 254
- get_auth_events_descriptions () (AuthEvent static method), 92
- get_cb_cloud_object () (in module *cbc_sdk.helpers*), 239
- get_config_template () (in module *cbc_sdk.workload.sensor_lifecycle.SensorKit* class method), 196
- get_credentials () (CredentialProvider method), 234
- get_credentials () (EnvironCredentialProvider method), 70
- get_credentials () (FileCredentialProvider method), 70
- get_credentials () (KeychainCredentialProvider method), 71
- get_credentials () (RegistryCredentialProvider method), 72
- get_default_provider () (DefaultProvider method), 70
- get_details () (AuthEvent method), 92
- get_details () (EnrichedEvent method), 75
- get_details () (Observation method), 155
- get_details () (Process method), 174
- get_endpoints () (USBDevice method), 84
- get_events () (CBAnalyticsAlert method), 123
- get_file () (CbLRSessionBase method), 243
- get_group_results () (ObservationQuery method), 158
- get_network_threat_metadata () (Observation method), 155
- get_notifications () (CBCloudAPI method), 254
- get_object () (BaseAPI method), 230
- get_object_by_name_or_id () (in module *cbc_sdk.helpers*), 239
- get_output_as_file () (Job method), 152
- get_output_as_lines () (Job method), 152
- get_output_as_stream () (Job method), 152
- get_output_as_string () (Job method), 152
- get_parameter () (PolicyRuleConfig method), 171
- get_permitted_role_urns () (in module *cbc_sdk.platform.grants.Grant* class method), 150
- get_policy_ruleconfig_parameter_schema () (CBCloudAPI method), 254
- get_process () (WatchlistAlert method), 129
- get_progress () (Job method), 152
- get_raw_data () (BaseAPI method), 230
- get_raw_file () (CbLRSessionBase method), 243
- get_registry_value () (CbLRSessionBase method), 243
- get_ruleconfig_parameter_schema () (Policy method), 165
- get_token () (Credentials method), 235
- get_token_type () (Credentials method), 235
- get_value () (Credentials method), 235
- get_vendors_and_products_seen () (in module *cbc_sdk.endpoint_standard.usb_device_control.USBDevice* class method), 84
- get_vulnerability_summary () (Device method), 135
- get_vulnerabilities () (Device method), 135
- GetFileJob (class in *cbc_sdk.live_response_api*), 248
- GetScode () (in module *cbc_sdk.winerror*), 269

`Grant` (class in `cbc_sdk.platform.grants`), 146

`grant()` (User method), 184

`Grant.GrantBuilder` (class in `cbc_sdk.platform.grants`), 146

`Grant.Profile` (class in `cbc_sdk.platform.grants`), 147

`Grant.ProfileBuilder` (class in `cbc_sdk.platform.grants`), 148

`GrantQuery` (class in `cbc_sdk.platform.grants`), 150

`group_details` (BaseAlert attribute), 117

`group_results()` (AuthEventQuery method), 95

H

`has_key()` (LRUCacheDict method), 214

`HRESULT_CODE()` (in module `cbc_sdk.winerror`), 269

`HRESULT_FACILITY()` (in module `cbc_sdk.winerror`), 269

`HRESULT_FROM_NT()` (in module `cbc_sdk.winerror`), 269

`HRESULT_FROM_WIN32()` (in module `cbc_sdk.winerror`), 269

`HRESULT_SEVERITY()` (in module `cbc_sdk.winerror`), 269

`http_request()` (Connection method), 233

I

`id` (BaseAlert attribute), 117

`id` (BaseAlert.Note attribute), 116

`id` (CorePreventionRuleConfig attribute), 170

`id` (Device attribute), 135

`id` (DeviceSummary attribute), 50

`id` (Feed attribute), 99

`id` (IOC_V2 attribute), 103

`id` (Job attribute), 152

`id` (Policy attribute), 165

`id` (PolicyRule attribute), 168

`id` (PolicyRuleConfig attribute), 171

`id` (Report attribute), 106

`id` (ReputationOverride attribute), 180

`id` (Result attribute), 53

`id` (Run attribute), 58

`id` (Template attribute), 64

`id` (USBDevice attribute), 85

`id` (USBDeviceApproval attribute), 87

`id` (USBDeviceBlock attribute), 88

`id` (Watchlist attribute), 111

`id` (WorkflowStatus attribute), 131

`id_` (WorkflowStatus attribute), 131

`ignore()` (IOC_V2 method), 103

`ignore()` (Report method), 106

`IGNORE_SYSTEM_PROXY` (CredentialValue attribute), 235

`ignored` (IOC_V2 attribute), 103

`ignored` (Report attribute), 106

`impact_` (Recommendation attribute), 81

`impact_score` (Recommendation.RecommendationImpact attribute), 79

`impacted_devices` (Recommendation.RecommendationImpact attribute), 79

`in_progress` (WorkflowStatus attribute), 131

`in_progress_count` (Run attribute), 58

`in_progress_count` (Template attribute), 64

`include_child_processes` (Recommendation.RecommendationNewRule attribute), 80

`include_child_processes` (ReputationOverride attribute), 180

`info_key` (Event attribute), 78

`ingress_time` (AuthEvent attribute), 93

`ingress_time` (Observation attribute), 156

`inherited_from` (CorePreventionRuleConfig attribute), 170

`inherited_from` (PolicyRuleConfig attribute), 171

`init_poolmanager()` (CBCSDKSessionAdapter method), 232

`INPLACE_E_FIRST` (RawErrorCode attribute), 277

`INPLACE_E_LAST` (RawErrorCode attribute), 277

`INPLACE_E_NOTOOLSPACE` (RawErrorCode attribute), 277

`INPLACE_E_NOTUNDOABLE` (RawErrorCode attribute), 277

`INPLACE_S_FIRST` (RawErrorCode attribute), 277

`INPLACE_S_LAST` (RawErrorCode attribute), 277

`install_sensor()` (AWSComputeResource method), 198

`install_sensor()` (BaseComputeResource method), 203

`install_sensor()` (VCenterComputeResource method), 206

`INTEGRATION` (CredentialValue attribute), 235

`interface_type` (USBDevice attribute), 85

`internal_name` (Binary attribute), 114

`InvalidHashError`, 237

`InvalidObjectError`, 237, 339

`IOC` (class in `cbc_sdk.enterprise_edr.threat_intelligence`), 101

`IOC_V2` (class in `cbc_sdk.enterprise_edr.threat_intelligence`), 101

`iocs` (Report attribute), 106

`iocs_` (Report attribute), 106

`iocs_v2` (Report attribute), 106

`ipv4` (IOC attribute), 101

`ipv6` (IOC attribute), 101

`ipv6_equality_format()` (`cbc_sdk.enterprise_edr.threat_intelligence.IOC_V2` class method), 103

`is_deleted` (PolicyRule attribute), 168

`is_dirty()` (MutableBaseModel method), 221

- is_system (*Policy attribute*), 165
- isoDateTimeFieldDescriptor (class in *cbc_sdk.base*), 220
- IterableQueryMixin (class in *cbc_sdk.base*), 220
- ## J
- Job (class in *cbc_sdk.platform.jobs*), 151
- job_id (*EnrichedEventFacet attribute*), 77
- job_id (*ProcessFacet attribute*), 177
- job_parameters (*Job attribute*), 152
- JobQuery (class in *cbc_sdk.platform.jobs*), 153
- jobrunner () (in module *cbc_sdk.live_response_api*), 250
- JobWorker (class in *cbc_sdk.live_response_api*), 248
- ## K
- KeychainCredentialProvider (class in *cbc_sdk.credential_providers.keychain_credential_provider*), 71
- kill_process () (*CbLRSessionBase method*), 244
- ## L
- lang_id (*Binary attribute*), 114
- last_contact_time (*Device attribute*), 135
- last_device_policy_changed_time (*Device attribute*), 135
- last_device_policy_requested_time (*Device attribute*), 135
- last_endpoint_id (*USBDevice attribute*), 85
- last_endpoint_name (*USBDevice attribute*), 85
- last_event_time (*BaseAlert attribute*), 117
- last_external_ip_address (*Device attribute*), 135
- last_internal_ip_address (*Device attribute*), 135
- last_location (*Device attribute*), 135
- last_name (*Device attribute*), 135
- last_name (*User attribute*), 184
- last_policy_id (*USBDevice attribute*), 85
- last_policy_updated_time (*Device attribute*), 135
- last_reported_time (*Device attribute*), 135
- last_reset_time (*Device attribute*), 135
- last_result_time (*Run attribute*), 58
- last_result_time (*Template attribute*), 64
- last_seen (*USBDevice attribute*), 85
- last_shutdown_time (*Device attribute*), 136
- last_update_time (*BaseAlert attribute*), 117
- last_update_time (*Job attribute*), 152
- last_update_time (*Workflow attribute*), 130
- last_update_timestamp (*Watchlist attribute*), 111
- latestRevision (*Policy attribute*), 165
- legacy (*Observation attribute*), 156
- legacy_alert_id (*BaseAlert attribute*), 117
- limit () (*FacetQuery method*), 218
- link (*IOC_V2 attribute*), 103
- link (*Report attribute*), 106
- linux_kernel_version (*Device attribute*), 136
- list_directory () (*CbLRSessionBase method*), 244
- list_processes () (*CbLRSessionBase method*), 244
- list_registry_keys_and_values () (*CbLRSessionBase method*), 245
- list_registry_values () (*CbLRSessionBase method*), 246
- live_response (*CBCloudAPI attribute*), 255
- LiveResponseError, 248
- LiveResponseJobScheduler (class in *cbc_sdk.live_response_api*), 248
- LiveResponseMemdump (class in *cbc_sdk.live_response_api*), 249
- LiveResponseSession (class in *cbc_sdk.live_response_api*), 249
- LiveResponseSessionManager (class in *cbc_sdk.live_response_api*), 249
- log (in module *cbc_sdk.base*), 228
- log (in module *cbc_sdk.endpoint_standard.base*), 78
- log (in module *cbc_sdk.endpoint_standard.recommendation*), 83
- log (in module *cbc_sdk.endpoint_standard.usb_device_control*), 90
- log (in module *cbc_sdk.enterprise_edr.threat_intelligence*), 112
- log (in module *cbc_sdk.platform.base*), 132
- log (in module *cbc_sdk.platform.grants*), 151
- log (in module *cbc_sdk.platform.users*), 186
- log (in module *cbc_sdk.platform.vulnerability_assessment*), 193
- log (in module *cbc_sdk.workload.vm_workloads_search*), 212
- login_id (*User attribute*), 184
- login_name (*User attribute*), 184
- login_user_name (*Device attribute*), 136
- lookup_error () (*cbc_sdk.winerror.ErrorBaseClass class method*), 267
- lr_session () (*Device method*), 136
- lru_cache_function () (in module *cbc_sdk.cache.lru*), 215
- LRUCacheFunction (class in *cbc_sdk.cache.lru*), 214
- LRUCacheDict (class in *cbc_sdk.cache.lru*), 213
- LRUCacheDict.EmptyCacheThread (class in *cbc_sdk.cache.lru*), 214
- ## M
- mac_address (*Device attribute*), 136
- managed_detection_response_permissions (*Policy attribute*), 165
- MARSHAL_E_FIRST (*RawErrorCode attribute*), 277

- MARSHAL_E_LAST (*RawErrorCode attribute*), 277
- MARSHAL_S_FIRST (*RawErrorCode attribute*), 277
- MARSHAL_S_LAST (*RawErrorCode attribute*), 277
- match_count (*Run attribute*), 58
- match_count (*Template attribute*), 64
- match_type (*IOC_V2 attribute*), 103
- matches_template() (*Grant.Profile method*), 148
- MAX_LIST_ITEM_RENDER (*NewBaseModel attribute*), 222
- MAX_RESULTS_LIMIT (in module *cbc_sdk.audit_remediation.base*), 52
- MAX_RETRY_COUNT (*CbLRSessionBase attribute*), 241
- MAX_VALUE_WIDTH (*NewBaseModel attribute*), 222
- md5 (*Binary attribute*), 114
- md5 (*IOC attribute*), 101
- MEM_E_INVALID_LINK (*RawErrorCode attribute*), 277
- MEM_E_INVALID_ROOT (*RawErrorCode attribute*), 277
- MEM_E_INVALID_SIZE (*RawErrorCode attribute*), 277
- memdump() (*CbLRSessionBase method*), 246
- message (*SensorKit attribute*), 196
- metrics (*DeviceSummary attribute*), 50
- metrics (*Result attribute*), 53
- metrics_ (*DeviceSummary attribute*), 50
- metrics_ (*Result attribute*), 53
- middle_name (*Device attribute*), 136
- MK_E_CANTOPENFILE (*RawErrorCode attribute*), 277
- MK_E_CONNECTMANUALLY (*RawErrorCode attribute*), 277
- MK_E_ENUMERATION_FAILED (*RawErrorCode attribute*), 277
- MK_E_EXCEEDEDDEADLINE (*RawErrorCode attribute*), 277
- MK_E_FIRST (*RawErrorCode attribute*), 277
- MK_E_INTERMEDIATEINTERFACENOTSUPPORTED (*RawErrorCode attribute*), 277
- MK_E_INVALIDEXTENSION (*RawErrorCode attribute*), 277
- MK_E_LAST (*RawErrorCode attribute*), 277
- MK_E_MUSTBOTHERUSER (*RawErrorCode attribute*), 277
- MK_E_NEEDGENERIC (*RawErrorCode attribute*), 277
- MK_E_NO_NORMALIZED (*RawErrorCode attribute*), 278
- MK_E_NOINVERSE (*RawErrorCode attribute*), 277
- MK_E_NOOBJECT (*RawErrorCode attribute*), 277
- MK_E_NOPREFIX (*RawErrorCode attribute*), 277
- MK_E_NOSTORAGE (*RawErrorCode attribute*), 278
- MK_E_NOTBINDABLE (*RawErrorCode attribute*), 278
- MK_E_NOTBOUND (*RawErrorCode attribute*), 278
- MK_E_SYNTAX (*RawErrorCode attribute*), 278
- MK_E_UNAVAILABLE (*RawErrorCode attribute*), 278
- MK_S_FIRST (*RawErrorCode attribute*), 278
- MK_S_LAST (*RawErrorCode attribute*), 278
- model_base_directory (*CbMetaModel attribute*), 216
- model_classes (*CbMetaModel attribute*), 216
- ModelNotFound, 237
- modload_count (*AuthEvent attribute*), 93
- monitored_assets (*Vulnerability.OrgSummary attribute*), 187
- MoreThanOneResultError, 237, 339
- MutableBaseModel (class in *cbc_sdk.base*), 221
- ## N
- name (*CorePreventionRuleConfig attribute*), 170
- name (*Device attribute*), 136
- name (*Feed attribute*), 99
- name (*Policy attribute*), 165
- name (*PolicyRuleConfig attribute*), 171
- name (*Run attribute*), 58
- name (*Template attribute*), 64
- name (*Watchlist attribute*), 111
- name() (*RunQuery method*), 61
- netconn_count (*AuthEvent attribute*), 93
- NetworkThreatMetadata (class in *cbc_sdk.platform.network_threat_metadata*), 153
- new_object() (*cbc_sdk.base.NewBaseModel class method*), 222
- new_rule_ (*Recommendation attribute*), 81
- NewBaseModel (class in *cbc_sdk.base*), 221
- newer_run_create_time (*Differential attribute*), 67
- newer_run_id (*Differential attribute*), 67
- newer_run_id() (*DifferentialQuery method*), 68
- newer_run_not_responded_devices (*Differential attribute*), 67
- no_match_count (*Run attribute*), 59
- no_match_count (*Template attribute*), 64
- NonQueryableModel, 237
- normalize_org() (in module *cbc_sdk.platform.grants*), 151
- normalize_profile_list() (in module *cbc_sdk.platform.users*), 186
- not_() (*QueryBuilder method*), 225
- not_() (*QueryBuilderSupportMixin method*), 226
- not_started_count (*Run attribute*), 59
- not_started_count (*Template attribute*), 64
- not_supported_count (*Run attribute*), 59
- not_supported_count (*Template attribute*), 64
- note (*BaseAlert.Note attribute*), 116
- notes (*USBDeviceApproval attribute*), 87
- notes_() (*BaseAlert method*), 117
- notes_present (*BaseAlert attribute*), 117

notification_listener() (CBCloudAPI method), 255

notify_on_finish (Run attribute), 59

notify_on_finish (Template attribute), 65

notify_on_finish() (RunQuery method), 61

nsx_available (Device attribute), 136

nsx_remediation() (Device method), 136

NSXJobError, 237

NSXRemediationJob (class in cbc_sdk.workload.nsx_remediation), 194

NTE_BAD_ALGID (RawErrorCode attribute), 278

NTE_BAD_DATA (RawErrorCode attribute), 278

NTE_BAD_FLAGS (RawErrorCode attribute), 278

NTE_BAD_HASH (RawErrorCode attribute), 278

NTE_BAD_HASH_STATE (RawErrorCode attribute), 278

NTE_BAD_KEY (RawErrorCode attribute), 278

NTE_BAD_KEY_STATE (RawErrorCode attribute), 278

NTE_BAD_KEYSET (RawErrorCode attribute), 278

NTE_BAD_KEYSET_PARAM (RawErrorCode attribute), 278

NTE_BAD_LEN (RawErrorCode attribute), 278

NTE_BAD_PROV_TYPE (RawErrorCode attribute), 278

NTE_BAD_PROVIDER (RawErrorCode attribute), 278

NTE_BAD_PUBLIC_KEY (RawErrorCode attribute), 278

NTE_BAD_SIGNATURE (RawErrorCode attribute), 278

NTE_BAD_TYPE (RawErrorCode attribute), 278

NTE_BAD_UID (RawErrorCode attribute), 278

NTE_BAD_VER (RawErrorCode attribute), 278

NTE_DOUBLE_ENCRYPT (RawErrorCode attribute), 278

NTE_EXISTS (RawErrorCode attribute), 278

NTE_FAIL (RawErrorCode attribute), 278

NTE_KEYSET_ENTRY_BAD (RawErrorCode attribute), 278

NTE_KEYSET_NOT_DEF (RawErrorCode attribute), 278

NTE_NO_KEY (RawErrorCode attribute), 278

NTE_NO_MEMORY (RawErrorCode attribute), 278

NTE_NOT_FOUND (RawErrorCode attribute), 278

NTE_OP_OK (RawErrorCode attribute), 278

NTE_PERM (RawErrorCode attribute), 278

NTE_PROV_DLL_NOT_FOUND (RawErrorCode attribute), 279

NTE_PROV_TYPE_ENTRY_BAD (RawErrorCode attribute), 279

NTE_PROV_TYPE_NO_MATCH (RawErrorCode attribute), 279

NTE_PROV_TYPE_NOT_DEF (RawErrorCode attribute), 279

NTE_PROVIDER_DLL_FAIL (RawErrorCode attribute), 278

NTE_SIGNATURE_FILE_BAD (RawErrorCode attribute), 279

NTE_SYS_ERR (RawErrorCode attribute), 279

num_found (AuthEventFacet attribute), 94

num_found (EnrichedEventFacet attribute), 77

num_found (ObservationFacet attribute), 157

num_found (ProcessFacet attribute), 177

num_hits (WorkflowStatus attribute), 131

num_success (WorkflowStatus attribute), 131

O

object_rule_configs (Policy attribute), 165

object_rule_configs_list (Policy attribute), 166

object_rules (Policy attribute), 166

ObjectFieldDescriptor (class in cbc_sdk.base), 222

ObjectNotFoundError, 238, 339

Observation (class in cbc_sdk.platform.observations), 154

observation_description (Observation attribute), 156

observation_id (Observation attribute), 156

observation_type (Observation attribute), 156

ObservationFacet (class in cbc_sdk.platform.observations), 156

ObservationFacet.Ranges (class in cbc_sdk.platform.observations), 157

ObservationFacet.Terms (class in cbc_sdk.platform.observations), 157

ObservationGroup (class in cbc_sdk.platform.observations), 157

ObservationQuery (class in cbc_sdk.platform.observations), 158

older_run_create_time (Differential attribute), 67

older_run_id (Differential attribute), 67

older_run_id() (DifferentialQuery method), 68

older_run_not_responded_devices (Differential attribute), 67

OLE_E_ADVDF (RawErrorCode attribute), 279

OLE_E_ADVISENOTSUPPORTED (RawErrorCode attribute), 279

OLE_E_BLANK (RawErrorCode attribute), 279

OLE_E_CANT_BINDTOSOURCE (RawErrorCode attribute), 279

OLE_E_CANT_GETMONIKER (RawErrorCode attribute), 279

OLE_E_CANTCONVERT (RawErrorCode attribute), 279

OLE_E_CLASSDIFF (RawErrorCode attribute), 279

OLE_E_ENUM_NOMORE (RawErrorCode attribute), 279

OLE_E_FIRST (RawErrorCode attribute), 279

OLE_E_INVALIDHWND (RawErrorCode attribute), 279

OLE_E_INVALIDRECT (RawErrorCode attribute), 279

- OLE_E_LAST (*RawErrorCode* attribute), 279
 - OLE_E_NOCACHE (*RawErrorCode* attribute), 279
 - OLE_E_NOCONNECTION (*RawErrorCode* attribute), 279
 - OLE_E_NOSTORAGE (*RawErrorCode* attribute), 279
 - OLE_E_NOT_INPLACEACTIVE (*RawErrorCode* attribute), 279
 - OLE_E_NOTRUNNING (*RawErrorCode* attribute), 279
 - OLE_E_OLEVERB (*RawErrorCode* attribute), 279
 - OLE_E_PROMPTSAVECANCELLED (*RawErrorCode* attribute), 279
 - OLE_E_STATIC (*RawErrorCode* attribute), 279
 - OLE_E_WRONGCOMPOBJ (*RawErrorCode* attribute), 279
 - OLE_S_FIRST (*RawErrorCode* attribute), 279
 - OLE_S_LAST (*RawErrorCode* attribute), 279
 - OLEOBJ_E_FIRST (*RawErrorCode* attribute), 279
 - OLEOBJ_E_INVALIDVERB (*RawErrorCode* attribute), 279
 - OLEOBJ_E_LAST (*RawErrorCode* attribute), 279
 - OLEOBJ_E_NOVERBS (*RawErrorCode* attribute), 279
 - OLEOBJ_S_FIRST (*RawErrorCode* attribute), 279
 - OLEOBJ_S_LAST (*RawErrorCode* attribute), 279
 - one() (*IterableQueryMixin* method), 220
 - OpenKey() (in *module cbc_sdk.credential_providers.registry_credential_provider*), 71
 - operation (*PolicyRule* attribute), 169
 - operation (*Recommendation.RecommendationNewRule* attribute), 80
 - OperationCancelled, 238
 - or_() (*AuthEventQuery* method), 96
 - or_() (*EnrichedEventQuery* method), 77
 - or_() (*ObservationQuery* method), 158
 - or_() (*QueryBuilder* method), 226
 - or_() (*QueryBuilderSupportMixin* method), 227
 - OR_INVALID_OID (*Win32Error* attribute), 335
 - OR_INVALID_OXID (*Win32Error* attribute), 335
 - OR_INVALID_SET (*Win32Error* attribute), 335
 - org_admin_version (*User* attribute), 184
 - org_adoption (*Recommendation.RecommendationImpact* attribute), 79
 - org_groups (*Grant.Profile* attribute), 148
 - org_id (*AuthEvent* attribute), 93
 - org_id (*Observation* attribute), 156
 - org_id (*User* attribute), 184
 - org_key (*BaseAlert* attribute), 117
 - ORG_KEY (*CredentialValue* attribute), 235
 - org_key (*Job* attribute), 152
 - org_key (*Policy* attribute), 166
 - org_key (*Run* attribute), 59
 - org_key (*Template* attribute), 65
 - org_key (*USBDevice* attribute), 85
 - org_key (*User* attribute), 184
 - org_ref (*Grant* attribute), 150
 - org_urn (*CBCloudAPI* attribute), 255
 - org_urn (*User* attribute), 185
 - organization_id (*Device* attribute), 136
 - organization_name (*Device* attribute), 136
 - orgs (*Grant.Profile* attribute), 148
 - original_document (*NewBaseModel* attribute), 222
 - original_filename (*Binary* attribute), 114
 - os (*Device* attribute), 136
 - os_info (*Vulnerability* attribute), 187
 - os_product_id (*Vulnerability* attribute), 187
 - os_type (*Binary* attribute), 114
 - os_version (*Device* attribute), 136
 - override_list (*Recommendation.RecommendationNewRule* attribute), 80
 - override_list (*ReputationOverride* attribute), 180
 - override_type (*Recommendation.RecommendationNewRule* attribute), 80
 - override_type (*ReputationOverride* attribute), 180
 - owner (*Feed* attribute), 99
 - owner_id (*Job* attribute), 152
- ## P
- PaginatedQuery (class in *cbc_sdk.base*), 223
 - parameters (*CorePreventionRuleConfig* attribute), 170
 - parameters (*PolicyRuleConfig* attribute), 171
 - parent_guid (*AuthEvent* attribute), 93
 - parent_guid (*Observation* attribute), 156
 - parent_pid (*AuthEvent* attribute), 93
 - parent_pid (*Observation* attribute), 156
 - parents (*Process* attribute), 175
 - passive_mode (*Device* attribute), 136
 - path (*Recommendation.RecommendationNewRule* attribute), 80
 - path (*ReputationOverride* attribute), 180
 - PDERR_CREATEICFAILURE (*CommDlgError* attribute), 256
 - PDERR_DEFAULTDIFFERENT (*CommDlgError* attribute), 256
 - PDERR_DNDMMISMATCH (*CommDlgError* attribute), 256
 - PDERR_GETDEVMODEFAIL (*CommDlgError* attribute), 257
 - PDERR_INITFAILURE (*CommDlgError* attribute), 257
 - PDERR_LOADDRVFAILURE (*CommDlgError* attribute), 257
 - PDERR_NODEFAULTPRN (*CommDlgError* attribute), 257
 - PDERR_NODEVICES (*CommDlgError* attribute), 257

PDERR_PARSEFAILURE (*CommDlgError attribute*), 257
 PDERR_PRINTERCODES (*CommDlgError attribute*), 257
 PDERR_PRINTERNOTFOUND (*CommDlgError attribute*), 257
 PDERR_RETDEFFAILED (*CommDlgError attribute*), 257
 PDERR_SETUPFAILURE (*CommDlgError attribute*), 257
 perform_action() (*Vulnerability method*), 187
 PERSIST_E_NOTSELSIZING (*RawErrorCode attribute*), 279
 PERSIST_E_SIZEDEFINITE (*RawErrorCode attribute*), 280
 PERSIST_E_SIZEINDEFINITE (*RawErrorCode attribute*), 280
 phone (*User attribute*), 185
 PlatformModel (*class in cbc_sdk.platform.base*), 131
 Policy (*class in cbc_sdk.platform.policies*), 159
 policy (*Policy attribute*), 166
 Policy.PolicyBuilder (*class in cbc_sdk.platform.policies*), 160
 policy_id (*BaseAlert attribute*), 117
 policy_id (*Device attribute*), 136
 policy_id (*USBDeviceBlock attribute*), 88
 policy_id() (*RunQuery method*), 61
 policy_name (*BaseAlert attribute*), 117
 policy_name (*Device attribute*), 136
 policy_override (*Device attribute*), 136
 PolicyQuery (*class in cbc_sdk.platform.policies*), 167
 PolicyRule (*class in cbc_sdk.platform.policies*), 168
 PolicyRuleConfig (*class in cbc_sdk.platform.policy_ruleconfigs*), 170
 poll_status() (*in module cbc_sdk.live_response_api*), 250
 position (*Policy attribute*), 166
 post() (*Connection method*), 233
 post_multipart() (*BaseAPI method*), 230
 post_object() (*BaseAPI method*), 230
 primary_key (*AuthEvent attribute*), 93
 primary_key (*AuthEventFacet attribute*), 94
 primary_key (*BaseAlert attribute*), 117
 primary_key (*BaseAlert.Note attribute*), 116
 primary_key (*BaseComputeResource attribute*), 203
 primary_key (*Binary attribute*), 114
 primary_key (*Binary.Summary attribute*), 113
 primary_key (*Device attribute*), 136
 primary_key (*DeviceFacet attribute*), 139
 primary_key (*DeviceSummary attribute*), 50
 primary_key (*Downloads.FoundItem attribute*), 114
 primary_key (*EnrichedEvent attribute*), 76
 primary_key (*EnrichedEventFacet attribute*), 77
 primary_key (*Event attribute*), 78, 144
 primary_key (*EventFacet attribute*), 145
 primary_key (*Feed attribute*), 99
 primary_key (*Grant attribute*), 150
 primary_key (*Grant.Profile attribute*), 148
 primary_key (*IOC_V2 attribute*), 103
 primary_key (*Job attribute*), 152
 primary_key (*NetworkThreatMetadata attribute*), 154
 primary_key (*NewBaseModel attribute*), 222
 primary_key (*Observation attribute*), 156
 primary_key (*ObservationFacet attribute*), 157
 primary_key (*Policy attribute*), 166
 primary_key (*PolicyRule attribute*), 169
 primary_key (*PolicyRuleConfig attribute*), 171
 primary_key (*Process attribute*), 175
 primary_key (*Process.Summary attribute*), 173
 primary_key (*Process.Tree attribute*), 173
 primary_key (*ProcessFacet attribute*), 177
 primary_key (*Recommendation attribute*), 81
 primary_key (*Report attribute*), 106
 primary_key (*ReportSeverity attribute*), 108
 primary_key (*ReputationOverride attribute*), 180
 primary_key (*Result attribute*), 53
 primary_key (*Result.Device attribute*), 52
 primary_key (*ResultFacet attribute*), 54
 primary_key (*Run attribute*), 59
 primary_key (*Template attribute*), 65
 primary_key (*USBDevice attribute*), 85
 primary_key (*USBDeviceApproval attribute*), 87
 primary_key (*USBDeviceBlock attribute*), 88
 primary_key (*User attribute*), 185
 primary_key (*Vulnerability attribute*), 188
 primary_key (*WorkflowStatus attribute*), 131
 principal (*Grant attribute*), 150
 principal_name (*Grant attribute*), 150
 priority_level (*Policy attribute*), 166
 priorityLevel (*Policy attribute*), 166
 private_build (*Binary attribute*), 114
 Process (*class in cbc_sdk.platform.processes*), 172
 Process.Summary (*class in cbc_sdk.platform.processes*), 172
 Process.Tree (*class in cbc_sdk.platform.processes*), 173
 process_guid (*AuthEvent attribute*), 93
 process_guid (*Observation attribute*), 156
 process_hash (*AuthEvent attribute*), 93
 process_hash (*Observation attribute*), 156
 process_limits() (*CBCloudAPI method*), 255
 process_md5 (*Process attribute*), 175
 process_name (*AuthEvent attribute*), 93
 process_name (*Observation attribute*), 156
 process_pid (*AuthEvent attribute*), 93
 process_pid (*Observation attribute*), 156
 process_pids (*Process attribute*), 175
 process_sha256 (*EnrichedEvent attribute*), 76

process_sha256 (*Process attribute*), 175
 process_username (*AuthEvent attribute*), 93
 process_username (*Observation attribute*), 156
 ProcessFacet (*class in cbc_sdk.platform.processes*), 176
 ProcessFacet.Ranges (*class in cbc_sdk.platform.processes*), 177
 ProcessFacet.Terms (*class in cbc_sdk.platform.processes*), 177
 product_description (*Binary attribute*), 114
 product_id (*USBDevice attribute*), 85
 product_id (*USBDeviceApproval attribute*), 87
 product_info (*Vulnerability attribute*), 188
 product_name (*Binary attribute*), 114
 product_name (*USBDevice attribute*), 85
 product_name (*USBDeviceApproval attribute*), 87
 product_version (*Binary attribute*), 114
 profile_uuid (*Grant.Profile attribute*), 148
 profiles (*Grant attribute*), 150
 profiles_ (*Grant attribute*), 150
 progress (*Job attribute*), 153
 provider_url (*Feed attribute*), 99
 PROXY (*CredentialValue attribute*), 235
 put () (*Connection method*), 233
 put_file () (*CbLRSessionBase method*), 246
 put_object () (*BaseAPI method*), 231

Q

quarantine () (*Device method*), 136
 quarantine () (*DeviceSearchQuery method*), 140
 quarantined (*Device attribute*), 137
 Query (*class in cbc_sdk.base*), 223
 query (*IOC attribute*), 101
 query_device_summaries () (*Result method*), 53
 query_device_summaries () (*Run method*), 59
 query_device_summary_facets () (*Result method*), 53
 query_devices () (*DeviceFacet.DeviceFacetValue method*), 138
 query_facets () (*Run method*), 59
 query_result_facets () (*Result method*), 53
 query_results () (*Run method*), 59
 query_runs () (*Template method*), 65
 QueryBuilder (*class in cbc_sdk.base*), 225
 QueryBuilderSupportMixin (*class in cbc_sdk.base*), 226
 QuerySyntaxError, 238
 QueryValueEx () (*in module cbc_sdk.credential_providers.registry_credential_provider*), 71
 queued (*WorkflowStatus attribute*), 131

R

raise_unless_json () (*BaseAPI method*), 231

ranges (*AuthEventFacet attribute*), 94
 ranges (*EnrichedEventFacet attribute*), 77
 ranges (*ObservationFacet attribute*), 157
 ranges (*ProcessFacet attribute*), 177
 ranges_ (*AuthEventFacet attribute*), 94
 ranges_ (*EnrichedEventFacet attribute*), 77
 ranges_ (*EventFacet attribute*), 145
 ranges_ (*ObservationFacet attribute*), 157
 ranges_ (*ProcessFacet attribute*), 177
 RawErrorCode (*class in cbc_sdk.winerror*), 269
 read_iocs () (*in module cbc_sdk.helpers*), 240
 Recommendation (*class in cbc_sdk.endpoint_standard.recommendation*), 78
 Recommendation.RecommendationApplication (*class in cbc_sdk.endpoint_standard.recommendation*), 79
 Recommendation.RecommendationImpact (*class in cbc_sdk.endpoint_standard.recommendation*), 79
 Recommendation.RecommendationNewRule (*class in cbc_sdk.endpoint_standard.recommendation*), 79
 Recommendation.RecommendationWorkflow (*class in cbc_sdk.endpoint_standard.recommendation*), 80
 RecommendationQuery (*class in cbc_sdk.endpoint_standard.recommendation*), 82
 recommended_query_id (*Run attribute*), 59
 recommended_query_id (*Template attribute*), 65
 ref_id (*Recommendation attribute*), 81
 ref_id (*Recommendation.RecommendationWorkflow attribute*), 81
 refresh () (*MutableBaseModel method*), 221
 refresh () (*NewBaseModel method*), 222
 refresh () (*UnrefreshableModel method*), 228
 REGDB_E_CLASSNOTREG (*RawErrorCode attribute*), 280
 REGDB_E_FIRST (*RawErrorCode attribute*), 280
 REGDB_E_IIDNOTREG (*RawErrorCode attribute*), 280
 REGDB_E_INVALIDVALUE (*RawErrorCode attribute*), 280
 REGDB_E_KEYMISSING (*RawErrorCode attribute*), 280
 REGDB_E_LAST (*RawErrorCode attribute*), 280
 REGDB_E_READREGDB (*RawErrorCode attribute*), 280
 REGDB_E_WRITEREGDB (*RawErrorCode attribute*), 280
 REGDB_S_FIRST (*RawErrorCode attribute*), 280
 REGDB_S_LAST (*RawErrorCode attribute*), 280
 registered_time (*Device attribute*), 137
 RegistryCredentialProvider (*class in cbc_sdk.credential_providers.registry_credential_provider*), 71

- 71
- regmod_count (*AuthEvent attribute*), 93
- reject() (*Recommendation method*), 81
- remediation (*Workflow attribute*), 130
- remove_iocs() (*Report method*), 106
- remove_iocs_by_id() (*Report method*), 107
- replace_reports() (*Feed method*), 99
- replace_reports_rawdata() (*Feed method*), 99
- replace_rule() (*Policy method*), 166
- replace_rule_config() (*Policy method*), 166
- Report (*class in cbc_sdk.enterprise_edr.threat_intelligence*), 103
- Report.ReportBuilder (*class in cbc_sdk.enterprise_edr.threat_intelligence*), 104
- report_id (*ReportSeverity attribute*), 108
- report_ids (*Watchlist attribute*), 111
- ReportQuery (*class in cbc_sdk.enterprise_edr.threat_intelligence*), 107
- reports (*Feed attribute*), 99
- reports (*Watchlist attribute*), 111
- ReportSeverity (*class in cbc_sdk.enterprise_edr.threat_intelligence*), 108
- reputation_override() (*Recommendation method*), 82
- ReputationOverride (*class in cbc_sdk.platform.reputation*), 178
- ReputationOverrideQuery (*class in cbc_sdk.platform.reputation*), 180
- request_session() (*CbLRManagerBase method*), 240
- required (*PolicyRule attribute*), 169
- requires_boolean_value() (*CredentialValue method*), 235
- reset() (*MutableBaseModel method*), 221
- reset() (*Recommendation method*), 82
- reset_google_authenticator_registration() (*User method*), 185
- Result (*class in cbc_sdk.audit_remediation.base*), 52
- Result.Device (*class in cbc_sdk.audit_remediation.base*), 52
- Result.Fields (*class in cbc_sdk.audit_remediation.base*), 52
- Result.Metrics (*class in cbc_sdk.audit_remediation.base*), 53
- result_url (*AuthEventFacet attribute*), 94
- result_url (*EnrichedEventFacet attribute*), 77
- result_url (*ObservationFacet attribute*), 157
- result_url (*Process.Summary attribute*), 173
- result_url (*Process.Tree attribute*), 174
- result_url (*ProcessFacet attribute*), 177
- ResultFacet (*class in cbc_sdk.audit_remediation.base*), 54
- ResultFacet.Values (*class in cbc_sdk.audit_remediation.base*), 54
- ResultFromCode() (*in module cbc_sdk.winerror*), 286
- ResultQuery (*class in cbc_sdk.audit_remediation.base*), 54
- results (*FacetQuery attribute*), 219
- results (*FeedQuery attribute*), 100
- results (*ReportQuery attribute*), 108
- results (*SimpleQuery attribute*), 227
- results (*SummaryQuery attribute*), 177
- results (*WatchlistQuery attribute*), 112
- role (*User attribute*), 185
- roles (*Grant attribute*), 150
- roles (*Grant.Profile attribute*), 148
- RPC_E_ACCESS_DENIED (*RawErrorCode attribute*), 280
- RPC_E_ATTEMPTED_MULTITHREAD (*RawErrorCode attribute*), 280
- RPC_E_CALL_CANCELED (*RawErrorCode attribute*), 280
- RPC_E_CALL_COMPLETE (*RawErrorCode attribute*), 280
- RPC_E_CALL_REJECTED (*RawErrorCode attribute*), 280
- RPC_E_CANTCALLOUT_AGAIN (*RawErrorCode attribute*), 280
- RPC_E_CANTCALLOUT_INASYNCCALL (*RawErrorCode attribute*), 280
- RPC_E_CANTCALLOUT_INEXTERNALCALL (*RawErrorCode attribute*), 280
- RPC_E_CANTCALLOUT_ININPUTSYNCCALL (*RawErrorCode attribute*), 280
- RPC_E_CANTPOST_INSEND_CALL (*RawErrorCode attribute*), 280
- RPC_E_CANTTRANSMIT_CALL (*RawErrorCode attribute*), 280
- RPC_E_CHANGED_MODE (*RawErrorCode attribute*), 280
- RPC_E_CLIENT_CANTMARSHAL_DATA (*RawErrorCode attribute*), 280
- RPC_E_CLIENT_CANTUNMARSHAL_DATA (*RawErrorCode attribute*), 280
- RPC_E_CLIENT_DIED (*RawErrorCode attribute*), 280
- RPC_E_CONNECTION_TERMINATED (*RawErrorCode attribute*), 280
- RPC_E_DISCONNECTED (*RawErrorCode attribute*), 280
- RPC_E_FAULT (*RawErrorCode attribute*), 280
- RPC_E_INVALID_CALLDATA (*RawErrorCode attribute*), 280
- RPC_E_INVALID_DATA (*RawErrorCode attribute*), 280

`RPC_E_INVALID_DATAPACKET` (*RawErrorCode attribute*), 280

`RPC_E_INVALID_EXTENSION` (*RawErrorCode attribute*), 280

`RPC_E_INVALID_HEADER` (*RawErrorCode attribute*), 280

`RPC_E_INVALID_IPID` (*RawErrorCode attribute*), 281

`RPC_E_INVALID_OBJECT` (*RawErrorCode attribute*), 281

`RPC_E_INVALID_OBJREF` (*RawErrorCode attribute*), 281

`RPC_E_INVALID_PARAMETER` (*RawErrorCode attribute*), 281

`RPC_E_INVALIDMETHOD` (*RawErrorCode attribute*), 280

`RPC_E_NO_CONTEXT` (*RawErrorCode attribute*), 281

`RPC_E_NO_GOOD_SECURITY_PACKAGES` (*RawErrorCode attribute*), 281

`RPC_E_NO_SYNC` (*RawErrorCode attribute*), 281

`RPC_E_NOT_REGISTERED` (*RawErrorCode attribute*), 281

`RPC_E_OUT_OF_RESOURCES` (*RawErrorCode attribute*), 281

`RPC_E_REMOTE_DISABLED` (*RawErrorCode attribute*), 281

`RPC_E_RETRY` (*RawErrorCode attribute*), 281

`RPC_E_SERVER_CANTMARSHAL_DATA` (*RawErrorCode attribute*), 281

`RPC_E_SERVER_CANTUNMARSHAL_DATA` (*RawErrorCode attribute*), 281

`RPC_E_SERVER_DIED` (*RawErrorCode attribute*), 281

`RPC_E_SERVER_DIED_DNE` (*RawErrorCode attribute*), 281

`RPC_E_SERVERCALL_REJECTED` (*RawErrorCode attribute*), 281

`RPC_E_SERVERCALL_RETRYLATER` (*RawErrorCode attribute*), 281

`RPC_E_SERVERFAULT` (*RawErrorCode attribute*), 281

`RPC_E_SYS_CALL_FAILED` (*RawErrorCode attribute*), 281

`RPC_E_THREAD_NOT_INIT` (*RawErrorCode attribute*), 281

`RPC_E_TIMEOUT` (*RawErrorCode attribute*), 281

`RPC_E_TOO_LATE` (*RawErrorCode attribute*), 281

`RPC_E_UNEXPECTED` (*RawErrorCode attribute*), 281

`RPC_E_UNSECURE_CALL` (*RawErrorCode attribute*), 281

`RPC_E_VERSION_MISMATCH` (*RawErrorCode attribute*), 281

`RPC_E_WRONG_THREAD` (*RawErrorCode attribute*), 281

`RPC_S_ADDRESS_ERROR` (*Win32Error attribute*), 335

`RPC_S_ALREADY_LISTENING` (*Win32Error attribute*), 335

`RPC_S_ALREADY_REGISTERED` (*Win32Error attribute*), 335

`RPC_S_BINDING_HAS_NO_AUTH` (*Win32Error attribute*), 335

`RPC_S_BINDING_INCOMPLETE` (*Win32Error attribute*), 335

`RPC_S_CALL_CANCELLED` (*Win32Error attribute*), 335

`RPC_S_CALL_FAILED` (*Win32Error attribute*), 335

`RPC_S_CALL_FAILED_DNE` (*Win32Error attribute*), 335

`RPC_S_CALL_IN_PROGRESS` (*Win32Error attribute*), 335

`RPC_S_CALLPENDING` (*RawErrorCode attribute*), 281

`RPC_S_CANNOT_SUPPORT` (*Win32Error attribute*), 335

`RPC_S_CANT_CREATE_ENDPOINT` (*Win32Error attribute*), 335

`RPC_S_COMM_FAILURE` (*Win32Error attribute*), 335

`RPC_S_DUPLICATE_ENDPOINT` (*Win32Error attribute*), 335

`RPC_S_ENTRY_ALREADY_EXISTS` (*Win32Error attribute*), 335

`RPC_S_ENTRY_NOT_FOUND` (*Win32Error attribute*), 335

`RPC_S_ENTRY_TYPE_MISMATCH` (*Win32Error attribute*), 335

`RPC_S_FP_DIV_ZERO` (*Win32Error attribute*), 336

`RPC_S_FP_OVERFLOW` (*Win32Error attribute*), 336

`RPC_S_FP_UNDERFLOW` (*Win32Error attribute*), 336

`RPC_S_GROUP_MEMBER_NOT_FOUND` (*Win32Error attribute*), 336

`RPC_S_GRP_ELT_NOT_ADDED` (*Win32Error attribute*), 336

`RPC_S_GRP_ELT_NOT_REMOVED` (*Win32Error attribute*), 336

`RPC_S_INCOMPLETE_NAME` (*Win32Error attribute*), 336

`RPC_S_INTERFACE_NOT_EXPORTED` (*Win32Error attribute*), 336

`RPC_S_INTERFACE_NOT_FOUND` (*Win32Error attribute*), 336

`RPC_S_INTERNAL_ERROR` (*Win32Error attribute*), 336

`RPC_S_INVALID_ASYNC_CALL` (*Win32Error attribute*), 336

`RPC_S_INVALID_ASYNC_HANDLE` (*Win32Error attribute*), 336

`RPC_S_INVALID_AUTH_IDENTITY` (*Win32Error attribute*), 336

`RPC_S_INVALID_BINDING` (*Win32Error attribute*), 336

`RPC_S_INVALID_BOUND` (*Win32Error attribute*), 336

[RPC_S_INVALID_ENDPOINT_FORMAT \(Win32Error attribute\), 336](#)
[RPC_S_INVALID_NAF_ID \(Win32Error attribute\), 336](#)
[RPC_S_INVALID_NAME_SYNTAX \(Win32Error attribute\), 336](#)
[RPC_S_INVALID_NET_ADDR \(Win32Error attribute\), 336](#)
[RPC_S_INVALID_NETWORK_OPTIONS \(Win32Error attribute\), 336](#)
[RPC_S_INVALID_OBJECT \(Win32Error attribute\), 336](#)
[RPC_S_INVALID_RPC_PROTSEQ \(Win32Error attribute\), 336](#)
[RPC_S_INVALID_STRING_BINDING \(Win32Error attribute\), 336](#)
[RPC_S_INVALID_STRING_UUID \(Win32Error attribute\), 336](#)
[RPC_S_INVALID_TAG \(Win32Error attribute\), 336](#)
[RPC_S_INVALID_TIMEOUT \(Win32Error attribute\), 336](#)
[RPC_S_INVALID_VERS_OPTION \(Win32Error attribute\), 336](#)
[RPC_S_MAX_CALLS_TOO_SMALL \(Win32Error attribute\), 336](#)
[RPC_S_NAME_SERVICE_UNAVAILABLE \(Win32Error attribute\), 336](#)
[RPC_S_NO_BINDINGS \(Win32Error attribute\), 336](#)
[RPC_S_NO_CALL_ACTIVE \(Win32Error attribute\), 337](#)
[RPC_S_NO_CONTEXT_AVAILABLE \(Win32Error attribute\), 337](#)
[RPC_S_NO_ENDPOINT_FOUND \(Win32Error attribute\), 337](#)
[RPC_S_NO_ENTRY_NAME \(Win32Error attribute\), 337](#)
[RPC_S_NO_INTERFACES \(Win32Error attribute\), 337](#)
[RPC_S_NO_MORE_BINDINGS \(Win32Error attribute\), 337](#)
[RPC_S_NO_MORE_MEMBERS \(Win32Error attribute\), 337](#)
[RPC_S_NO_PRINC_NAME \(Win32Error attribute\), 337](#)
[RPC_S_NO_PROTSEQS \(Win32Error attribute\), 337](#)
[RPC_S_NO_PROTSEQS_REGISTERED \(Win32Error attribute\), 337](#)
[RPC_S_NOT_ALL_OBJS_EXPORTED \(Win32Error attribute\), 336](#)
[RPC_S_NOT_ALL_OBJS_UNEXPORTED \(Win32Error attribute\), 336](#)
[RPC_S_NOT_CANCELLED \(Win32Error attribute\), 336](#)
[RPC_S_NOT_LISTENING \(Win32Error attribute\), 336](#)
[RPC_S_NOT_RPC_ERROR \(Win32Error attribute\), 336](#)
[RPC_S_NOTHING_TO_EXPORT \(Win32Error attribute\), 336](#)
[RPC_S_OBJECT_NOT_FOUND \(Win32Error attribute\), 337](#)
[RPC_S_OUT_OF_RESOURCES \(Win32Error attribute\), 337](#)
[RPC_S_PRF_ELT_NOT_ADDED \(Win32Error attribute\), 337](#)
[RPC_S_PRF_ELT_NOT_REMOVED \(Win32Error attribute\), 337](#)
[RPC_S_PROCNUM_OUT_OF_RANGE \(Win32Error attribute\), 337](#)
[RPC_S_PROFILE_NOT_ADDED \(Win32Error attribute\), 337](#)
[RPC_S_PROTOCOL_ERROR \(Win32Error attribute\), 337](#)
[RPC_S_PROTSEQ_NOT_FOUND \(Win32Error attribute\), 337](#)
[RPC_S_PROTSEQ_NOT_SUPPORTED \(Win32Error attribute\), 337](#)
[RPC_S_PROXY_ACCESS_DENIED \(Win32Error attribute\), 337](#)
[RPC_S_SEC_PKG_ERROR \(Win32Error attribute\), 337](#)
[RPC_S_SEND_INCOMPLETE \(Win32Error attribute\), 337](#)
[RPC_S_SERVER_TOO_BUSY \(Win32Error attribute\), 337](#)
[RPC_S_SERVER_UNAVAILABLE \(Win32Error attribute\), 337](#)
[RPC_S_STRING_TOO_LONG \(Win32Error attribute\), 337](#)
[RPC_S_TYPE_ALREADY_REGISTERED \(Win32Error attribute\), 337](#)
[RPC_S_UNKNOWN_AUTHN_LEVEL \(Win32Error attribute\), 337](#)
[RPC_S_UNKNOWN_AUTHN_SERVICE \(Win32Error attribute\), 337](#)
[RPC_S_UNKNOWN_AUTHN_TYPE \(Win32Error attribute\), 337](#)
[RPC_S_UNKNOWN_AUTHZ_SERVICE \(Win32Error attribute\), 337](#)
[RPC_S_UNKNOWN_IF \(Win32Error attribute\), 337](#)
[RPC_S_UNKNOWN_MGR_TYPE \(Win32Error attribute\), 337](#)
[RPC_S_UNSUPPORTED_AUTHN_LEVEL \(Win32Error attribute\), 337](#)
[RPC_S_UNSUPPORTED_NAME_SYNTAX \(Win32Error attribute\), 337](#)
[RPC_S_UNSUPPORTED_TRANS_SYN \(Win32Error attribute\), 337](#)
[RPC_S_UNSUPPORTED_TYPE \(Win32Error attribute\), 337](#)
[RPC_S_UUID_LOCAL_ONLY \(Win32Error attribute\), 338](#)
[RPC_S_UUID_NO_ADDRESS \(Win32Error attribute\), 338](#)
[RPC_S_WAITONTIMER \(RawErrorCode attribute\), 281](#)

RPC_S_WRONG_KIND_OF_BINDING (*Win32Error attribute*), 338
 RPC_S_ZERO_DIVIDE (*Win32Error attribute*), 338
 RPC_X_BAD_STUB_DATA (*Win32Error attribute*), 338
 RPC_X_BYTE_COUNT_TOO_SMALL (*Win32Error attribute*), 338
 RPC_X_ENUM_VALUE_OUT_OF_RANGE (*Win32Error attribute*), 338
 RPC_X_INVALID_ES_ACTION (*Win32Error attribute*), 338
 RPC_X_INVALID_PIPE_OBJECT (*Win32Error attribute*), 338
 RPC_X_NO_MORE_ENTRIES (*Win32Error attribute*), 338
 RPC_X_NULL_REF_POINTER (*Win32Error attribute*), 338
 RPC_X_PIPE_CLOSED (*Win32Error attribute*), 338
 RPC_X_PIPE_DISCIPLINE_ERROR (*Win32Error attribute*), 338
 RPC_X_PIPE_EMPTY (*Win32Error attribute*), 338
 RPC_X_SS_CANNOT_GET_CALL_HANDLE (*Win32Error attribute*), 338
 RPC_X_SS_CHAR_TRANS_OPEN_FAIL (*Win32Error attribute*), 338
 RPC_X_SS_CHAR_TRANS_SHORT_FILE (*Win32Error attribute*), 338
 RPC_X_SS_CONTEXT_DAMAGED (*Win32Error attribute*), 338
 RPC_X_SS_HANDLES_MISMATCH (*Win32Error attribute*), 338
 RPC_X_SS_IN_NULL_CONTEXT (*Win32Error attribute*), 338
 RPC_X_WRONG_ES_VERSION (*Win32Error attribute*), 338
 RPC_X_WRONG_PIPE_ORDER (*Win32Error attribute*), 338
 RPC_X_WRONG_PIPE_VERSION (*Win32Error attribute*), 338
 RPC_X_WRONG_STUB_VERSION (*Win32Error attribute*), 338
 rule_id (*Observation attribute*), 156
 rules (*Policy attribute*), 166
 Run (*class in cbc_sdk.audit_remediation.base*), 57
 run () (*GetFileJob method*), 248
 run () (*JobWorker method*), 248
 run () (*LiveResponseJobScheduler method*), 249
 run () (*LRUCacheDict.EmptyCacheThread method*), 214
 run_id () (*FacetQuery method*), 51
 run_id () (*ResultQuery method*), 55
 run_job () (*JobWorker method*), 248
 RunHistory (*class in cbc_sdk.audit_remediation.base*), 60
 RunHistoryQuery (*class in cbc_sdk.audit_remediation.base*), 60
 RUNNING_STATUSES (*NSXRemediationJob attribute*), 194
 RunQuery (*class in cbc_sdk.audit_remediation.base*), 60

S

save () (*Feed method*), 99
 save () (*MutableBaseModel method*), 221
 save () (*Watchlist method*), 111
 save_watchlist () (*Report method*), 107
 scan_last_action_time (*Device attribute*), 137
 scan_last_complete_time (*Device attribute*), 137
 scan_status (*Device attribute*), 137
 schedule (*Run attribute*), 59
 schedule (*Template attribute*), 65
 schedule () (*RunQuery method*), 61
 SCHEMA_IOC2V2 (*FeedModel attribute*), 100
 SCHEMA_REPORT (*FeedModel attribute*), 100
 SCORE_CODE () (*in module cbc_sdk.winerror*), 286
 SCORE_FACILITY () (*in module cbc_sdk.winerror*), 286
 SCORE_SEVERITY () (*in module cbc_sdk.winerror*), 286
 scriptload_count (*AuthEvent attribute*), 93
 search_suggestions () (*AuthEvent static method*), 93
 search_suggestions () (*BaseAlert static method*), 117
 search_suggestions () (*Observation static method*), 156
 select () (*BaseAPI method*), 231
 select_class_instance () (*in module cbc_sdk.connection*), 234
 sensor_config_url (*SensorKit attribute*), 196
 sensor_out_of_date (*Device attribute*), 137
 sensor_settings (*Policy attribute*), 166
 sensor_states (*Device attribute*), 137
 sensor_type (*SensorKit attribute*), 196
 sensor_url (*SensorKit attribute*), 196
 sensor_version (*Device attribute*), 137
 SensorKit (*class in cbc_sdk.workload.sensor_lifecycle*), 195
 SensorKitQuery (*class in cbc_sdk.workload.sensor_lifecycle*), 196
 serial_number (*USBDevice attribute*), 85
 serial_number (*USBDeviceApproval attribute*), 87
 ServerError, 238, 339
 session_status () (*LiveResponseSessionManager method*), 250
 set_ad_group_ids () (*DeviceSearchQuery method*), 141
 set_alert_ids () (*BaseAlertSearchQuery method*), 119

`set_alertable()` (*Feed.FeedBuilder method*), 98
`set_alerts_enabled()` (*Watchlist.WatchlistBuilder method*), 109
`set_appliance_uuid()` (*VCenterComputeResourceQuery method*), 209
`set_assignment_mode()` (*CorePreventionRuleConfig method*), 170
`set_auth_method()` (*User.UserBuilder method*), 182
`set_auto_delete_bad_hash_delay()` (*Policy.PolicyBuilder method*), 162
`set_auto_deregister_interval()` (*Policy.PolicyBuilder method*), 162
`set_auto_scaling_group_name()` (*AWSComputeResourceQuery method*), 200
`set_auto_scaling_group_name()` (*DeviceSearchQuery method*), 141
`set_availability_zone()` (*AWSComputeResourceQuery method*), 200
`set_avira_protection_cloud()` (*Policy.PolicyBuilder method*), 162
`set_blocked_threat_categories()` (*CBAAnalyticsAlertSearchQuery method*), 124
`set_categories()` (*BaseAlertSearchQuery method*), 119
`set_category()` (*Feed.FeedBuilder method*), 98
`set_cloud_provider_account_id()` (*AWSComputeResourceQuery method*), 200
`set_cloud_provider_account_id()` (*DeviceSearchQuery method*), 141
`set_cloud_provider_resource_id()` (*AWSComputeResourceQuery method*), 200
`set_cloud_provider_tags()` (*AWSComputeResourceQuery method*), 200
`set_cluster_name()` (*VCenterComputeResourceQuery method*), 210
`set_cluster_names()` (*ContainerRuntimeAlertSearchQuery method*), 125
`set_conditions()` (*Grant.ProfileBuilder method*), 148
`set_create_time()` (*BaseAlertSearchQuery method*), 119
`set_datacenter_name()` (*VCenterComputeResourceQuery method*), 210
`set_deployment_type()` (*DeviceSearchQuery method*), 141
`set_deployment_type()` (*VulnerabilityQuery method*), 190
`set_description()` (*Policy.PolicyBuilder method*), 162
`set_description()` (*Report.ReportBuilder method*), 104
`set_description()` (*Watchlist.WatchlistBuilder method*), 109
`set_device_guid()` (*VCenterComputeResourceQuery method*), 210
`set_device_ids()` (*BaseAlertSearchQuery method*), 119
`set_device_ids()` (*DeviceSearchQuery method*), 141
`set_device_ids()` (*DifferentialQuery method*), 69
`set_device_ids()` (*FacetQuery method*), 51
`set_device_ids()` (*ResultQuery method*), 56
`set_device_ids()` (*USBDeviceApprovalQuery method*), 87
`set_device_locations()` (*CBAAnalyticsAlertSearchQuery method*), 124
`set_device_names()` (*BaseAlertSearchQuery method*), 119
`set_device_names()` (*FacetQuery method*), 51
`set_device_names()` (*ResultQuery method*), 56
`set_device_os()` (*BaseAlertSearchQuery method*), 120
`set_device_os()` (*FacetQuery method*), 51
`set_device_os()` (*ResultQuery method*), 56
`set_device_os_versions()` (*BaseAlertSearchQuery method*), 120
`set_device_type()` (*VulnerabilityQuery method*), 190
`set_device_username()` (*BaseAlertSearchQuery method*), 120
`set_disabled()` (*Grant.Profile method*), 148
`set_disabled()` (*Grant.ProfileBuilder method*), 149
`set_egress_group_ids()` (*ContainerRuntimeAlertSearchQuery method*), 125
`set_egress_group_names()` (*ContainerRuntimeAlertSearchQuery method*), 126
`set_eligibility()` (*VCenterComputeResourceQuery method*), 210
`set_eligibility_code()` (*VCenterComputeResourceQuery method*), 210
`set_email()` (*User.UserBuilder method*), 182
`set_endpoint_names()` (*USBDeviceQuery method*), 89
`set_esx_host_name()` (*VCenterComputeResourceQuery method*), 210
`set_esx_host_uuid()` (*VCenterComputeResourceQuery method*), 210
`set_exclude_sensor_versions()` (*DeviceSearchQuery method*), 141
`set_expiration()` (*Grant.Profile method*), 148
`set_expiration()` (*Grant.ProfileBuilder method*), 149
`set_external_device_friendly_names()` (*DeviceControlAlertSearchQuery method*), 128
`set_external_device_ids()` (*DeviceControlAlertSearchQuery method*), 128
`set_fields()` (*Query method*), 224

`set_first_name()` (*User.UserBuilder* method), 182
`set_group_results()` (*BaseAlertSearchQuery* method), 120
`set_hashes()` (*RecommendationQuery* method), 82
`set_highest_risk_score()` (*VulnerabilityQuery* method), 190
`set_host_name()` (*VCenterComputeResourceQuery* method), 211
`set_id()` (*AWSComputeResourceQuery* method), 201
`set_installation_status()` (*AWSComputeResourceQuery* method), 201
`set_installation_status()` (*VCenterComputeResourceQuery* method), 211
`set_installation_type()` (*VCenterComputeResourceQuery* method), 211
`set_ip_address()` (*VCenterComputeResourceQuery* method), 211
`set_ip_reputations()` (*ContainerRuntimeAlertSearchQuery* method), 126
`set_kill_chain_statuses()` (*CBAnalyticsAlertSearchQuery* method), 124
`set_last_contact_time()` (*DeviceSearchQuery* method), 141
`set_last_name()` (*User.UserBuilder* method), 182
`set_last_sync_ts()` (*VulnerabilityQuery* method), 191
`set_legacy_alert_ids()` (*BaseAlertSearchQuery* method), 120
`set_link()` (*Report.ReportBuilder* method), 105
`set_managed_detection_response_permissions()` (*Policy.PolicyBuilder* method), 162
`set_max_rows()` (*DeviceSearchQuery* method), 142
`set_max_rows()` (*USBDeviceQuery* method), 89
`set_minimum_severity()` (*BaseAlertSearchQuery* method), 120
`set_name()` (*AWSComputeResourceQuery* method), 201
`set_name()` (*Feed.FeedBuilder* method), 98
`set_name()` (*Policy.PolicyBuilder* method), 162
`set_name()` (*VCenterComputeResourceQuery* method), 211
`set_name()` (*VulnerabilityQuery* method), 191
`set_name()` (*Watchlist.WatchlistBuilder* method), 109
`set_namespaces()` (*ContainerRuntimeAlertSearchQuery* method), 126
`set_not_blocked_threat_categories()` (*CBAnalyticsAlertSearchQuery* method), 124
`set_on_access_scan()` (*Policy.PolicyBuilder* method), 162
`set_on_demand_scan()` (*Policy.PolicyBuilder* method), 162
`set_on_demand_scan_schedule()` (*Policy.PolicyBuilder* method), 163
`set_org()` (*Grant.GrantBuilder* method), 147
`set_orgs()` (*Grant.ProfileBuilder* method), 149
`set_os()` (*DeviceSearchQuery* method), 142
`set_os_arch()` (*VulnerabilityQuery* method), 191
`set_os_architecture()` (*VCenterComputeResourceQuery* method), 211
`set_os_description()` (*VCenterComputeResourceQuery* method), 211
`set_os_name()` (*VulnerabilityQuery* method), 191
`set_os_product_id()` (*AffectedAssetQuery* method), 186
`set_os_type()` (*VCenterComputeResourceQuery* method), 211
`set_os_type()` (*VulnerabilityQuery* method), 191
`set_os_version()` (*VulnerabilityQuery* method), 192
`set_override_list()` (*ReputationOverrideQuery* method), 180
`set_override_type()` (*ReputationOverrideQuery* method), 181
`set_parameter()` (*PolicyRuleConfig* method), 171
`set_phone()` (*User.UserBuilder* method), 183
`set_platform()` (*AWSComputeResourceQuery* method), 201
`set_platform_details()` (*AWSComputeResourceQuery* method), 201
`set_policy_applied()` (*CBAnalyticsAlertSearchQuery* method), 124
`set_policy_ids()` (*BaseAlertSearchQuery* method), 120
`set_policy_ids()` (*DeviceSearchQuery* method), 142
`set_policy_ids()` (*FacetQuery* method), 51
`set_policy_ids()` (*ResultQuery* method), 56
`set_policy_names()` (*BaseAlertSearchQuery* method), 120
`set_policy_names()` (*FacetQuery* method), 52
`set_policy_names()` (*ResultQuery* method), 56
`set_policy_types()` (*RecommendationQuery* method), 82
`set_ports()` (*ContainerRuntimeAlertSearchQuery* method), 126
`set_principal_name()` (*Grant.GrantBuilder* method), 147
`set_priority()` (*Policy.PolicyBuilder* method), 163
`set_process_names()` (*BaseAlertSearchQuery* method), 121
`set_process_sha256()` (*BaseAlertSearchQuery* method), 121
`set_product_ids()` (*DeviceControlAlertSearchQuery* method), 128
`set_product_names()` (*DeviceControlAlertSearchQuery* method), 128
`set_product_names()` (*USBDeviceApprovalQuery* method), 87

`set_product_names()` (*USBDeviceQuery method*), 89
`set_profile_expiration()` (*User method*), 185
`set_protocols()` (*ContainerRuntimeAlertSearch-Query method*), 126
`set_provider_url()` (*Feed.FeedBuilder method*), 98
`set_reason_code()` (*CBAnalyticsAlertSearchQuery method*), 124
`set_region()` (*AWSComputeResourceQuery method*), 201
`set_registration_id()` (*VCenterComputeResourceQuery method*), 212
`set_registry_value()` (*CbLRSessionBase method*), 246
`set_remote_domains()` (*ContainerRuntimeAlertSearchQuery method*), 126
`set_remote_ips()` (*ContainerRuntimeAlertSearch-Query method*), 126
`set_replica_ids()` (*ContainerRuntimeAlertSearchQuery method*), 127
`set_reputations()` (*BaseAlertSearchQuery method*), 121
`set_role()` (*User.UserBuilder method*), 183
`set_roles()` (*Grant.GrantBuilder method*), 147
`set_roles()` (*Grant.ProfileBuilder method*), 149
`set_rows()` (*AsyncProcessQuery method*), 171
`set_rows()` (*AuthEventQuery method*), 96
`set_rows()` (*EnrichedEventQuery method*), 77
`set_rows()` (*FacetQuery method*), 219
`set_rows()` (*ObservationQuery method*), 158
`set_rows()` (*Query method*), 224
`set_rule_ids()` (*ContainerRuntimeAlertSearch-Query method*), 127
`set_rule_names()` (*ContainerRuntimeAlertSearch-Query method*), 127
`set_run_states()` (*CBAnalyticsAlertSearchQuery method*), 124
`set_sensor_actions()` (*CBAnalyticsAlertSearch-Query method*), 125
`set_serial_numbers()` (*DeviceControlAlertSearchQuery method*), 128
`set_serial_numbers()` (*USBDeviceQuery method*), 90
`set_severity()` (*Report.ReportBuilder method*), 105
`set_severity()` (*VulnerabilityOrgSummaryQuery method*), 188
`set_severity()` (*VulnerabilityQuery method*), 192
`set_signature_update()` (*Policy.PolicyBuilder method*), 163
`set_signature_update_schedule()` (*Policy.PolicyBuilder method*), 163
`set_source_label()` (*Feed.FeedBuilder method*), 98
`set_start()` (*Query method*), 224
`set_status()` (*DeviceSearchQuery method*), 142
`set_statuses()` (*FacetQuery method*), 52
`set_statuses()` (*RecommendationQuery method*), 83
`set_statuses()` (*ResultQuery method*), 56
`set_statuses()` (*USBDeviceQuery method*), 90
`set_subnet_id()` (*AWSComputeResourceQuery method*), 201
`set_summary()` (*Feed.FeedBuilder method*), 98
`set_sync_status()` (*VulnerabilityQuery method*), 192
`set_sync_type()` (*VulnerabilityQuery method*), 192
`set_system()` (*PolicyQuery method*), 167
`set_tags()` (*BaseAlertSearchQuery method*), 121
`set_tags_enabled()` (*Watchlist.WatchlistBuilder method*), 110
`set_target_priorities()` (*BaseAlertSearch-Query method*), 121
`set_target_priorities()` (*DeviceSearchQuery method*), 142
`set_template_ids()` (*RunHistoryQuery method*), 60
`set_threat_cause_vectors()` (*CBAnalyticAlertSearchQuery method*), 125
`set_threat_ids()` (*BaseAlertSearchQuery method*), 121
`set_time_range()` (*BaseAlertSearchQuery method*), 121
`set_time_range()` (*FacetQuery method*), 219
`set_time_range()` (*Query method*), 224
`set_time_range()` (*SummaryQuery method*), 178
`set_timestamp()` (*Report.ReportBuilder method*), 105
`set_title()` (*Report.ReportBuilder method*), 105
`set_types()` (*BaseAlertSearchQuery method*), 122
`set_update_servers_offsite()` (*Policy.PolicyBuilder method*), 163
`set_update_servers_onsite()` (*Policy.PolicyBuilder method*), 163
`set_update_servers_override()` (*Policy.PolicyBuilder method*), 164
`set_uuid()` (*VCenterComputeResourceQuery method*), 212
`set_vcenter()` (*VulnerabilityOrgSummaryQuery method*), 189
`set_vcenter()` (*VulnerabilityQuery method*), 192
`set_vcenter_host_url()` (*VCenterComputeResourceQuery method*), 212
`set_vcenter_name()` (*VCenterComputeResource-Query method*), 212
`set_vcenter_uuid()` (*VCenterComputeResource-Query method*), 212

`set_vendor_ids()` (*DeviceControlAlertSearchQuery method*), 128

`set_vendor_names()` (*DeviceControlAlertSearchQuery method*), 129

`set_vendor_names()` (*USBDeviceApprovalQuery method*), 87

`set_vendor_names()` (*USBDeviceQuery method*), 90

`set_virtual_private_cloud_id()` (*AWSComputeResourceQuery method*), 201

`set_virtual_private_cloud_id()` (*DeviceSearchQuery method*), 143

`set_visibility()` (*Report.ReportBuilder method*), 105

`set_visibility()` (*VulnerabilityOrgSummaryQuery method*), 189

`set_visibility()` (*VulnerabilityQuery method*), 192

`set_vm_id()` (*VulnerabilityQuery method*), 193

`set_vmwaretools_version()` (*VCenterComputeResourceQuery method*), 212

`set_vuln_count()` (*VulnerabilityQuery method*), 193

`set_watchlist_ids()` (*WatchlistAlertSearchQuery method*), 129

`set_watchlist_names()` (*WatchlistAlertSearchQuery method*), 129

`set_workflows()` (*BaseAlertSearchQuery method*), 122

`set_workload_ids()` (*ContainerRuntimeAlertSearchQuery method*), 127

`set_workload_kinds()` (*ContainerRuntimeAlertSearchQuery method*), 127

`set_workload_names()` (*ContainerRuntimeAlertSearchQuery method*), 127

`severity` (*BaseAlert attribute*), 117

`severity` (*Report attribute*), 107

`severity` (*ReportSeverity attribute*), 108

`severity_levels()` (*Vulnerability.OrgSummary method*), 187

`severity_summary` (*Vulnerability.OrgSummary attribute*), 187

`sha256` (*Binary attribute*), 114

`sha256_hash` (*Recommendation.RecommendationNewRule attribute*), 80

`sha256_hash` (*ReputationOverride attribute*), 180

`SHOW_ATTR` (*Process.Summary attribute*), 173

`SHOW_ATTR` (*Process.Tree attribute*), 173

`siblings` (*Process attribute*), 175

`signed_by` (*Recommendation.RecommendationNewRule attribute*), 80

`signed_by` (*ReputationOverride attribute*), 180

`SimpleQuery` (*class in cbc_sdk.base*), 227

`size()` (*LRUCacheDict method*), 214

`sort()` (*SimpleQuery method*), 227

`sort_by()` (*BaseAlertSearchQuery method*), 122

`sort_by()` (*BaseComputeResourceQuery method*), 204

`sort_by()` (*DeviceSearchQuery method*), 143

`sort_by()` (*Query method*), 225

`sort_by()` (*RecommendationQuery method*), 83

`sort_by()` (*ReputationOverrideQuery method*), 181

`sort_by()` (*ResultQuery method*), 57

`sort_by()` (*RunHistoryQuery method*), 60

`sort_by()` (*TemplateHistoryQuery method*), 66

`sort_by()` (*USBDeviceQuery method*), 90

`sort_by()` (*VulnerabilityQuery method*), 193

`source_label` (*Feed attribute*), 99

`SPAPI_E_BAD_INTERFACE_INSTALLSECT` (*RawErrorCode attribute*), 281

`SPAPI_E_BAD_SECTION_NAME_LINE` (*RawErrorCode attribute*), 281

`SPAPI_E_BAD_SERVICE_INSTALLSECT` (*RawErrorCode attribute*), 281

`SPAPI_E_CANT_LOAD_CLASS_ICON` (*RawErrorCode attribute*), 281

`SPAPI_E_CLASS_MISMATCH` (*RawErrorCode attribute*), 281

`SPAPI_E_DEVICE_INTERFACE_ACTIVE` (*RawErrorCode attribute*), 281

`SPAPI_E_DEVICE_INTERFACE_REMOVED` (*RawErrorCode attribute*), 281

`SPAPI_E_DEVINFO_DATA_LOCKED` (*RawErrorCode attribute*), 281

`SPAPI_E_DEVINFO_LIST_LOCKED` (*RawErrorCode attribute*), 282

`SPAPI_E_DEVINFO_NOT_REGISTERED` (*RawErrorCode attribute*), 282

`SPAPI_E_DEVINST_ALREADY_EXISTS` (*RawErrorCode attribute*), 282

`SPAPI_E_DI_BAD_PATH` (*RawErrorCode attribute*), 282

`SPAPI_E_DI_DO_DEFAULT` (*RawErrorCode attribute*), 282

`SPAPI_E_DI_DONT_INSTALL` (*RawErrorCode attribute*), 282

`SPAPI_E_DI_NOFILECOPY` (*RawErrorCode attribute*), 282

`SPAPI_E_DI_POSTPROCESSING_REQUIRED` (*RawErrorCode attribute*), 282

`SPAPI_E_DUPLICATE_FOUND` (*RawErrorCode attribute*), 282

`SPAPI_E_ERROR_NOT_INSTALLED` (*RawErrorCode attribute*), 282

`SPAPI_E_EXPECTED_SECTION_NAME` (*RawErrorCode attribute*), 282

- SPAPI_E_FILEQUEUE_LOCKED (*RawErrorCode attribute*), 282
- SPAPI_E_GENERAL_SYNTAX (*RawErrorCode attribute*), 282
- SPAPI_E_INVALID_CLASS (*RawErrorCode attribute*), 282
- SPAPI_E_INVALID_CLASS_INSTALLER (*RawErrorCode attribute*), 282
- SPAPI_E_INVALID_COINSTALLER (*RawErrorCode attribute*), 282
- SPAPI_E_INVALID_DEVINST_NAME (*RawErrorCode attribute*), 282
- SPAPI_E_INVALID_FILTER_DRIVER (*RawErrorCode attribute*), 282
- SPAPI_E_INVALID_HWPROFILE (*RawErrorCode attribute*), 282
- SPAPI_E_INVALID_INF_LOGCONFIG (*RawErrorCode attribute*), 282
- SPAPI_E_INVALID_MACHINENAME (*RawErrorCode attribute*), 282
- SPAPI_E_INVALID_PROPPAGE_PROVIDER (*RawError-Code attribute*), 282
- SPAPI_E_INVALID_REFERENCE_STRING (*RawError-Code attribute*), 282
- SPAPI_E_INVALID_REG_PROPERTY (*RawError-Code attribute*), 282
- SPAPI_E_KEY_DOES_NOT_EXIST (*RawErrorCode attribute*), 282
- SPAPI_E_LINE_NOT_FOUND (*RawErrorCode attribute*), 282
- SPAPI_E_MACHINE_UNAVAILABLE (*RawErrorCode attribute*), 282
- SPAPI_E_NO_ASSOCIATED_CLASS (*RawErrorCode attribute*), 282
- SPAPI_E_NO_ASSOCIATED_SERVICE (*RawError-Code attribute*), 282
- SPAPI_E_NO_CLASS_DRIVER_LIST (*RawError-Code attribute*), 282
- SPAPI_E_NO_CLASSINSTALL_PARAMS (*RawError-Code attribute*), 282
- SPAPI_E_NO_COMPAT_DRIVERS (*RawErrorCode attribute*), 282
- SPAPI_E_NO_CONFIGMGR_SERVICES (*RawError-Code attribute*), 282
- SPAPI_E_NO_DEFAULT_DEVICE_INTERFACE (*RawErrorCode attribute*), 282
- SPAPI_E_NO_DEVICE_ICON (*RawErrorCode attribute*), 282
- SPAPI_E_NO_DEVICE_SELECTED (*RawErrorCode attribute*), 282
- SPAPI_E_NO_DRIVER_SELECTED (*RawErrorCode attribute*), 283
- SPAPI_E_NO_INF (*RawErrorCode attribute*), 283
- SPAPI_E_NO_SUCH_DEVICE_INTERFACE (*RawError-Code attribute*), 283
- SPAPI_E_NO_SUCH_DEVINST (*RawErrorCode attribute*), 283
- SPAPI_E_NO_SUCH_INTERFACE_CLASS (*RawError-Code attribute*), 283
- SPAPI_E_REMOTE_COMM_FAILURE (*RawErrorCode attribute*), 283
- SPAPI_E_SECTION_NAME_TOO_LONG (*RawError-Code attribute*), 283
- SPAPI_E_SECTION_NOT_FOUND (*RawErrorCode attribute*), 283
- SPAPI_E_WRONG_INF_STYLE (*RawErrorCode attribute*), 283
- special_build (*Binary attribute*), 114
- sql (*Run attribute*), 59
- sql (*Template attribute*), 65
- SSL_CERT_FILE (*CredentialValue attribute*), 235
- SSL_FORCE_TLS_1_2 (*CredentialValue attribute*), 235
- SSL_VERIFY (*CredentialValue attribute*), 235
- SSL_VERIFY_HOSTNAME (*CredentialValue attribute*), 235
- start_memdump() (*CbLRSessionBase method*), 247
- start_request() (*cbc_sdk.workload.nsx_remediation.NSXRemediation class method*), 194
- state (*Workflow attribute*), 130
- status (*Device attribute*), 137
- status (*DeviceSummary attribute*), 50
- status (*Job attribute*), 153
- status (*NSXRemediationJob attribute*), 195
- status (*Recommendation attribute*), 82
- status (*Recommendation.RecommendationWorkflow attribute*), 81
- status (*Result attribute*), 54
- status (*Run attribute*), 59
- status (*Template attribute*), 65
- status (*USBDevice attribute*), 85
- status (*WorkflowStatus attribute*), 131
- status_update_time (*Run attribute*), 59
- status_update_time (*Template attribute*), 65
- STG_E_ABNORMALAPIEXIT (*RawErrorCode attribute*), 283
- STG_E_ACCESSDENIED (*RawErrorCode attribute*), 283
- STG_E_BADBASEADDRESS (*RawErrorCode attribute*), 283
- STG_E_CANTSAVE (*RawErrorCode attribute*), 283
- STG_E_DISKISWRITEPROTECTED (*RawErrorCode attribute*), 283
- STG_E_DOCFILECORRUPT (*RawErrorCode attribute*), 283
- STG_E_EXTANTMARSHALLINGS (*RawErrorCode attribute*), 283

STG_E_FILEALREADYEXISTS (*RawErrorCode attribute*), 283

STG_E_FILENOTFOUND (*RawErrorCode attribute*), 283

STG_E_INCOMPLETE (*RawErrorCode attribute*), 283

STG_E_INSUFFICIENTMEMORY (*RawErrorCode attribute*), 283

STG_E_INUSE (*RawErrorCode attribute*), 283

STG_E_INVALIDFLAG (*RawErrorCode attribute*), 283

STG_E_INVALIDFUNCTION (*RawErrorCode attribute*), 283

STG_E_INVALIDHANDLE (*RawErrorCode attribute*), 283

STG_E_INVALIDHEADER (*RawErrorCode attribute*), 283

STG_E_INVALIDNAME (*RawErrorCode attribute*), 283

STG_E_INVALIDPARAMETER (*RawErrorCode attribute*), 283

STG_E_INVALIDPOINTER (*RawErrorCode attribute*), 283

STG_E_LOCKVIOLATION (*RawErrorCode attribute*), 283

STG_E_MEDIUMFULL (*RawErrorCode attribute*), 283

STG_E_NOMOREFILES (*RawErrorCode attribute*), 283

STG_E_NOTCURRENT (*RawErrorCode attribute*), 283

STG_E_NOTFILEBASEDSTORAGE (*RawErrorCode attribute*), 283

STG_E_OLDDLL (*RawErrorCode attribute*), 283

STG_E_OLDFORMAT (*RawErrorCode attribute*), 283

STG_E_PATHNOTFOUND (*RawErrorCode attribute*), 283

STG_E_PROPSETMISMATCHED (*RawErrorCode attribute*), 284

STG_E_READFAULT (*RawErrorCode attribute*), 284

STG_E_REVERTED (*RawErrorCode attribute*), 284

STG_E_SEEKERROR (*RawErrorCode attribute*), 284

STG_E_SHAREREQUIRED (*RawErrorCode attribute*), 284

STG_E_SHAREVIOLATION (*RawErrorCode attribute*), 284

STG_E_TERMINATED (*RawErrorCode attribute*), 284

STG_E_TOOMANYOPENFILES (*RawErrorCode attribute*), 284

STG_E_UNIMPLEMENTEDFUNCTION (*RawErrorCode attribute*), 284

STG_E_UNKNOWN (*RawErrorCode attribute*), 284

STG_E_WRITEFAULT (*RawErrorCode attribute*), 284

STG_S_BLOCK (*RawErrorCode attribute*), 284

STG_S_CANNOTCONSOLIDATE (*RawErrorCode attribute*), 284

STG_S_CONSOLIDATIONFAILED (*RawErrorCode attribute*), 284

STG_S_CONVERTED (*RawErrorCode attribute*), 284

STG_S_MONITORING (*RawErrorCode attribute*), 284

STG_S_MULTIPLEOPENS (*RawErrorCode attribute*), 284

STG_S_RETRYNOW (*RawErrorCode attribute*), 284

stop () (*Run method*), 59

stop () (*Template method*), 65

stop_keepalive_thread () (*CbLRManagerBase method*), 241

submit () (*DifferentialQuery method*), 69

submit () (*RunQuery method*), 62

submit () (*VulnerabilityOrgSummaryQuery method*), 189

submit_job () (*CbLRManagerBase method*), 241

submit_job () (*LiveResponseJobScheduler method*), 249

submit_job () (*LiveResponseSessionManager method*), 250

submit_url (*AuthEventFacet attribute*), 94

submit_url (*EnrichedEventFacet attribute*), 77

submit_url (*ObservationFacet attribute*), 157

submit_url (*ProcessFacet attribute*), 177

SUCCEEDED () (*in module cbc_sdk.winerror*), 286

success_count (*Run attribute*), 59

success_count (*Template attribute*), 65

summarize () (*AWSComputeResourceQuery method*), 202

summary (*Binary attribute*), 114

summary (*Feed attribute*), 99

summary (*Process attribute*), 175

summary_format (*Process.Summary attribute*), 173

summary_format (*Process.Tree attribute*), 174

SummaryQuery (*class in cbc_sdk.platform.processes*), 177

swagger_meta_file (*Vulnerability.AssetView attribute*), 187

SwaggerLoader (*class in cbc_sdk.base*), 228

systemPolicy (*Policy attribute*), 166

T

tags (*BaseAlert attribute*), 117

tags (*Report attribute*), 107

tags_enabled (*Watchlist attribute*), 111

target_priority_type (*Device attribute*), 137

target_value (*BaseAlert attribute*), 117

Template (*class in cbc_sdk.audit_remediation.base*), 63

template_id (*Run attribute*), 59

template_id (*Template attribute*), 65

TemplateHistory (*class in cbc_sdk.audit_remediation.base*), 65

TemplateHistoryQuery (*class in cbc_sdk.audit_remediation.base*), 65

terms (*AuthEventFacet attribute*), 95

terms (*EnrichedEventFacet attribute*), 77

terms (*ObservationFacet attribute*), 157

- `terms` (*ProcessFacet attribute*), 177
- `terms_` (*AuthEventFacet attribute*), 95
- `terms_` (*EnrichedEventFacet attribute*), 77
- `terms_` (*EventFacet attribute*), 145
- `terms_` (*ObservationFacet attribute*), 157
- `terms_` (*ProcessFacet attribute*), 177
- `threat_id` (*BaseAlert attribute*), 118
- `threat_public_comment` (*NetworkThreatMetadata attribute*), 154
- `time_received` (*DeviceSummary attribute*), 50
- `time_received` (*Result attribute*), 54
- `timeout()` (*AsyncProcessQuery method*), 172
- `timeout()` (*AuthEventQuery method*), 96
- `timeout()` (*EnrichedEventQuery method*), 77
- `timeout()` (*FacetQuery method*), 219
- `timeout()` (*ObservationQuery method*), 159
- `timeout()` (*SummaryQuery method*), 178
- `timeout_time` (*Run attribute*), 60
- `timeout_time` (*Template attribute*), 65
- `TimeoutError`, 238, 340
- `timestamp` (*Report attribute*), 107
- `title` (*Report attribute*), 107
- `to_dict()` (*Credentials method*), 235
- `TOKEN` (*CredentialValue attribute*), 235
- `total_results` (*DeviceSummary attribute*), 50
- `total_results` (*Run attribute*), 60
- `total_results` (*Template attribute*), 65
- `touch()` (*MutableBaseModel method*), 221
- `trademark` (*Binary attribute*), 114
- `tree` (*Process attribute*), 175
- `TRUST_E_ACTION_UNKNOWN` (*RawErrorCode attribute*), 284
- `TRUST_E_BAD_DIGEST` (*RawErrorCode attribute*), 284
- `TRUST_E_BASIC_CONSTRAINTS` (*RawErrorCode attribute*), 284
- `TRUST_E_CERT_SIGNATURE` (*RawErrorCode attribute*), 284
- `TRUST_E_COUNTER_SIGNER` (*RawErrorCode attribute*), 284
- `TRUST_E_FAIL` (*RawErrorCode attribute*), 284
- `TRUST_E_FINANCIAL_CRITERIA` (*RawErrorCode attribute*), 284
- `TRUST_E_NO_SIGNER_CERT` (*RawErrorCode attribute*), 284
- `TRUST_E_NOSIGNATURE` (*RawErrorCode attribute*), 284
- `TRUST_E_PROVIDER_UNKNOWN` (*RawErrorCode attribute*), 284
- `TRUST_E_SUBJECT_FORM_UNKNOWN` (*RawErrorCode attribute*), 284
- `TRUST_E_SUBJECT_NOT_TRUSTED` (*RawErrorCode attribute*), 284
- `TRUST_E_SYSTEM_ERROR` (*RawErrorCode attribute*), 284
- `TRUST_E_TIME_STAMP` (*RawErrorCode attribute*), 284
- `try_json()` (*in module cbc_sdk.connection*), 234
- `type` (*BaseAlert attribute*), 118
- `type` (*Job attribute*), 153
- `type` (*Recommendation.RecommendationApplication attribute*), 79
- `TYPE_E_AMBIGUOUSNAME` (*RawErrorCode attribute*), 284
- `TYPE_E_BADMODULEKIND` (*RawErrorCode attribute*), 284
- `TYPE_E_BUFFER_TOO_SMALL` (*RawErrorCode attribute*), 284
- `TYPE_E_CANTCREATETMPFILE` (*RawErrorCode attribute*), 284
- `TYPE_E_CANTLOADLIBRARY` (*RawErrorCode attribute*), 285
- `TYPE_E_CIRCULARTYPE` (*RawErrorCode attribute*), 285
- `TYPE_E_DLLFUNCTIONNOTFOUND` (*RawErrorCode attribute*), 285
- `TYPE_E_DUPLICATEID` (*RawErrorCode attribute*), 285
- `TYPE_E_ELEMENTNOTFOUND` (*RawErrorCode attribute*), 285
- `TYPE_E_FIELDNOTFOUND` (*RawErrorCode attribute*), 285
- `TYPE_E_INCONSISTENTPROPFUNCS` (*RawErrorCode attribute*), 285
- `TYPE_E_INVALIDID` (*RawErrorCode attribute*), 285
- `TYPE_E_INVALIDSTATE` (*RawErrorCode attribute*), 285
- `TYPE_E_INVDATAREAD` (*RawErrorCode attribute*), 285
- `TYPE_E_IOERROR` (*RawErrorCode attribute*), 285
- `TYPE_E_LIBNOTREGISTERED` (*RawErrorCode attribute*), 285
- `TYPE_E_NAMECONFLICT` (*RawErrorCode attribute*), 285
- `TYPE_E_OUTOFBOUNDS` (*RawErrorCode attribute*), 285
- `TYPE_E_QUALIFIEDNAMEDISALLOWED` (*RawErrorCode attribute*), 285
- `TYPE_E_REGISTRYACCESS` (*RawErrorCode attribute*), 285
- `TYPE_E_SIZE_TOO_BIG` (*RawErrorCode attribute*), 285
- `TYPE_E_TYPEMISMATCH` (*RawErrorCode attribute*), 285
- `TYPE_E_UNDEFINEDTYPE` (*RawErrorCode attribute*), 285
- `TYPE_E_UNKNOWNLCID` (*RawErrorCode attribute*), 285

TYPE_E_UNSUPFORMAT (*RawErrorCode* attribute), 285

TYPE_E_WRONGTYPEKIND (*RawErrorCode* attribute), 285

U

UnauthorizedError, 239

unignore() (*IOC_V2* method), 103

unignore() (*Report* method), 107

uninstall_code (*Device* attribute), 137

uninstall_sensor() (*Device* method), 137

uninstall_sensor() (*DeviceSearchQuery* method), 143

UnrefreshableModel (class in *cbc_sdk.base*), 228

update() (*BaseAlert* method), 118

update() (*BaseAlertSearchQuery* method), 122

update() (*Feed* method), 99

update() (*Report* method), 107

update() (*Watchlist* method), 111

update_criteria() (*CriteriaBuilderSupportMixin* method), 216

update_policy() (*Device* method), 137

update_policy() (*DeviceSearchQuery* method), 143

update_sensor_version() (*Device* method), 137

update_sensor_version() (*DeviceSearchQuery* method), 143

update_threat() (*BaseAlert* method), 118

update_time (*Grant* attribute), 150

update_time (*Recommendation* attribute), 82

update_time (*Recommendation.RecommendationImpact* attribute), 79

update_time (*Recommendation.RecommendationWorkflow* attribute), 81

updated_at (*USBDevice* attribute), 85

updated_at (*USBDeviceApproval* attribute), 87

updated_at (*USBDeviceBlock* attribute), 88

updated_by (*Grant* attribute), 150

updated_by (*USBDeviceApproval* attribute), 87

url (*BaseAPI* attribute), 231

URL (*CredentialValue* attribute), 235

urlobject (*BaseAlert* attribute), 118

urlobject (*BaseAlert.Note* attribute), 116

urlobject (*BaseComputeResource* attribute), 203

urlobject (*CBAanalyticsAlert* attribute), 123

urlobject (*ContainerRuntimeAlert* attribute), 125

urlobject (*Device* attribute), 137

urlobject (*DeviceControlAlert* attribute), 128

urlobject (*DeviceFacet* attribute), 139

urlobject (*DeviceSummary* attribute), 50

urlobject (*DeviceSummaryFacet* attribute), 50

urlobject (*Differential* attribute), 67

urlobject (*Downloads* attribute), 115

urlobject (*Event* attribute), 78, 144

urlobject (*EventFacet* attribute), 145

urlobject (*Feed* attribute), 100

urlobject (*Grant* attribute), 150

urlobject (*Grant.Profile* attribute), 148

urlobject (*Job* attribute), 153

urlobject (*NetworkThreatMetadata* attribute), 154

urlobject (*Policy* attribute), 166

urlobject (*PolicyRuleConfig* attribute), 171

urlobject (*Process* attribute), 176

urlobject (*Process.Summary* attribute), 173

urlobject (*Process.Tree* attribute), 174

urlobject (*Recommendation* attribute), 82

urlobject (*Report* attribute), 107

urlobject (*ReputationOverride* attribute), 180

urlobject (*Result* attribute), 54

urlobject (*ResultFacet* attribute), 54

urlobject (*Run* attribute), 60

urlobject (*Template* attribute), 65

urlobject (*USBDevice* attribute), 85

urlobject (*USBDeviceApproval* attribute), 87

urlobject (*USBDeviceBlock* attribute), 88

urlobject (*User* attribute), 185

urlobject (*Vulnerability* attribute), 188

urlobject (*Vulnerability.AssetView* attribute), 187

urlobject (*Vulnerability.OrgSummary* attribute), 187

urlobject (*Watchlist* attribute), 112

urlobject (*WatchlistAlert* attribute), 129

urlobject_history (*RunHistory* attribute), 60

urlobject_history (*TemplateHistory* attribute), 65

urlobject_single (*BaseAlert* attribute), 118

urlobject_single (*BaseAlert.Note* attribute), 116

urlobject_single (*BaseComputeResource* attribute), 203

urlobject_single (*Binary* attribute), 114

urlobject_single (*Binary.Summary* attribute), 113

urlobject_single (*Device* attribute), 137

urlobject_single (*Feed* attribute), 100

urlobject_single (*Grant* attribute), 150

urlobject_single (*Grant.Profile* attribute), 148

urlobject_single (*Job* attribute), 153

urlobject_single (*Policy* attribute), 166

urlobject_single (*Recommendation* attribute), 82

urlobject_single (*Report* attribute), 107

urlobject_single (*ReputationOverride* attribute), 180

urlobject_single (*Run* attribute), 60

urlobject_single (*Template* attribute), 65

urlobject_single (*USBDevice* attribute), 85

urlobject_single (*USBDeviceApproval* attribute), 87

urlobject_single (*USBDeviceBlock* attribute), 88

urlobject_single (*User* attribute), 185

urlobject_single (*Watchlist* attribute), 112

urlobject_single (*WorkflowStatus* attribute), 131

- urn (*User attribute*), 185
- USBDevice (class in *VALID_FACET_FIELDS (BaseAlertSearchQuery attribute)*, 118
cbc_sdk.endpoint_standard.usb_device_control), 83
- USBDeviceApproval (class in *VALID_FACET_FIELDS (DeviceSearchQuery attribute)*, 139
cbc_sdk.endpoint_standard.usb_device_control), 85
- USBDeviceApprovalQuery (class in *VALID_FACET_FIELDS (USBDeviceQuery attribute)*, 89
cbc_sdk.endpoint_standard.usb_device_control), 87
- USBDeviceBlock (class in *VALID_GROUP_FIELDS (AuthEventQuery attribute)*, 95
cbc_sdk.endpoint_standard.usb_device_control), 88
- USBDeviceBlockQuery (class in *VALID_GROUP_FIELDS (ObservationQuery attribute)*, 158
cbc_sdk.endpoint_standard.usb_device_control), 88
- USBDeviceQuery (class in *VALID_INSTALLATION_STATUS (AWSComputeResourceQuery attribute)*, 198
cbc_sdk.endpoint_standard.usb_device_control), 89
- User (class in *cbc_sdk.platform.users*), 181
- User.UserBuilder (class in *VALID_INSTALLATION_STATUS (VCenterComputeResourceQuery attribute)*, 207
cbc_sdk.platform.users), 182
- user_ids() (*UserQuery method*), 185
- UserQuery (class in *cbc_sdk.platform.users*), 185
- ## V
- VALID_ACTIONS (*PolicyRule attribute*), 168
- VALID_ALERT_TYPES (*BaseAlertSearchQuery attribute*), 118
- VALID_APP_KEYS (*PolicyRule attribute*), 168
- VALID_APP_TYPES (*PolicyRule attribute*), 168
- VALID_ARCHITECTURES (*SensorKit attribute*), 195
- VALID_CATEGORIES (*BaseAlertSearchQuery attribute*), 118
- VALID_DAYS (*Policy attribute*), 164
- VALID_DEPLOYMENT_TYPE (*BaseComputeResourceQuery attribute*), 203
- VALID_DEPLOYMENT_TYPES (*DeviceSearchQuery attribute*), 139
- VALID_DEVICE_TYPE (*VulnerabilityQuery attribute*), 189
- VALID_DEVICE_TYPES (*SensorKit attribute*), 195
- VALID_DIRECTIONS (*BaseComputeResourceQuery attribute*), 203
- VALID_DIRECTIONS (*DeviceSearchQuery attribute*), 139
- VALID_DIRECTIONS (*ReputationOverrideQuery attribute*), 180
- VALID_DIRECTIONS (*VulnerabilityQuery attribute*), 189
- VALID_DOWNLOAD_FORMATS (*BaseComputeResourceQuery attribute*), 203
- VALID_ELIGIBILITY (*VCenterComputeResourceQuery attribute*), 207
- VALID_FACET_FIELDS (*BaseAlertSearchQuery attribute*), 118
- VALID_FACET_FIELDS (*DeviceSearchQuery attribute*), 139
- VALID_FACET_FIELDS (*USBDeviceQuery attribute*), 89
- VALID_GROUP_FIELDS (*AuthEventQuery attribute*), 95
- VALID_GROUP_FIELDS (*ObservationQuery attribute*), 158
- VALID_INSTALLATION_STATUS (*AWSComputeResourceQuery attribute*), 198
- VALID_INSTALLATION_STATUS (*VCenterComputeResourceQuery attribute*), 207
- VALID_KILL_CHAIN_STATUSES (*CBAnalyticsAlertSearchQuery attribute*), 123
- VALID_LOCATIONS (*CBAnalyticsAlertSearchQuery attribute*), 123
- VALID_OPERATIONS (*PolicyRule attribute*), 168
- VALID_OS (*DeviceSearchQuery attribute*), 139
- VALID_OS_ARCHITECTURE (*VCenterComputeResourceQuery attribute*), 207
- VALID_OS_TYPE (*VCenterComputeResourceQuery attribute*), 207
- VALID_OS_TYPE (*VulnerabilityQuery attribute*), 189
- VALID_POLICY_APPLIED (*CBAnalyticsAlertSearchQuery attribute*), 123
- VALID_POLICY_TYPES (*RecommendationQuery attribute*), 82
- VALID_PRIORITIES (*DeviceSearchQuery attribute*), 139
- VALID_PRIORITIES (*Policy attribute*), 164
- VALID_REPUTATIONS (*BaseAlertSearchQuery attribute*), 118
- VALID_REPUTATIONS (*PolicyRule attribute*), 168
- valid_rule_configs() (*Policy method*), 166
- VALID_RUN_STATES (*CBAnalyticsAlertSearchQuery attribute*), 123
- VALID_SCAN_MODES (*Policy attribute*), 164
- VALID_SCAN_OPTIONS (*Policy attribute*), 164
- VALID_SENSOR_ACTIONS (*CBAnalyticsAlertSearchQuery attribute*), 123
- VALID_SENSOR_SETTINGS (*Policy attribute*), 164
- VALID_SEVERITY (*VulnerabilityOrgSummaryQuery attribute*), 188
- VALID_SEVERITY (*VulnerabilityQuery attribute*), 189
- VALID_STATUSES (*DeviceSearchQuery attribute*), 139
- VALID_STATUSES (*RecommendationQuery attribute*), 82
- VALID_STATUSES (*USBDeviceQuery attribute*), 89
- VALID_SYNC_STATUS (*VulnerabilityQuery attribute*), 189
- VALID_SYNC_TYPE (*VulnerabilityQuery attribute*), 189

- VALID_TAGS (*NSXRemediationJob* attribute), 194
- VALID_THREAT_CATEGORIES (*CBAnalyticsAlertSearchQuery* attribute), 123
- VALID_THREAT_CAUSE_VECTORS (*CBAnalyticsAlertSearchQuery* attribute), 123
- VALID_TYPES (*SensorKit* attribute), 195
- VALID_VISIBILITY (*VulnerabilityOrgSummaryQuery* attribute), 188
- VALID_VISIBILITY (*VulnerabilityQuery* attribute), 189
- VALID_WORKFLOW_VALS (*BaseAlertSearchQuery* attribute), 118
- validate() (*Feed* method), 100
- validate() (*IOC* method), 101
- validate() (*IOC_V2* method), 103
- validate() (*MutableBaseModel* method), 221
- validate() (*PolicyRule* method), 169
- validate() (*PolicyRuleConfig* method), 171
- validate() (*Report* method), 107
- validate() (*Watchlist* method), 112
- validate_process_query() (*CBCloudAPI* method), 255
- validation_url (*AuthEvent* attribute), 93
- validation_url (*Event* attribute), 144
- validation_url (*Observation* attribute), 156
- validation_url (*Process* attribute), 176
- value (*Recommendation.RecommendationApplication* attribute), 79
- values (*ComputeResourceFacet* attribute), 205
- values (*DeviceFacet* attribute), 139
- values (*IOC_V2* attribute), 103
- values (*ResultFacet* attribute), 54
- values_ (*DeviceFacet* attribute), 139
- values_ (*ResultFacet* attribute), 54
- VCenterComputeResource (class in *cbc_sdk.workload.vm_workloads_search*), 205
- VCenterComputeResourceQuery (class in *cbc_sdk.workload.vm_workloads_search*), 206
- vdi_base_device (*Device* attribute), 138
- vendor_id (*USBDevice* attribute), 85
- vendor_id (*USBDeviceApproval* attribute), 87
- vendor_name (*USBDevice* attribute), 85
- vendor_name (*USBDeviceApproval* attribute), 87
- version (*Policy* attribute), 167
- VIEW_E_DRAW (*RawErrorCode* attribute), 285
- VIEW_E_FIRST (*RawErrorCode* attribute), 285
- VIEW_E_LAST (*RawErrorCode* attribute), 285
- VIEW_S_FIRST (*RawErrorCode* attribute), 285
- VIEW_S_LAST (*RawErrorCode* attribute), 285
- virtual_machine (*Device* attribute), 138
- virtualization_provider (*Device* attribute), 138
- visibility (*Report* attribute), 107
- vuln_info (*Vulnerability* attribute), 188
- Vulnerability (class in *cbc_sdk.platform.vulnerability_assessment*), 186
- Vulnerability.AssetView (class in *cbc_sdk.platform.vulnerability_assessment*), 187
- Vulnerability.OrgSummary (class in *cbc_sdk.platform.vulnerability_assessment*), 187
- vulnerability_refresh() (*Device* method), 138
- VulnerabilityAssetViewQuery (class in *cbc_sdk.platform.vulnerability_assessment*), 188
- VulnerabilityOrgSummaryQuery (class in *cbc_sdk.platform.vulnerability_assessment*), 188
- VulnerabilityQuery (class in *cbc_sdk.platform.vulnerability_assessment*), 189
- ## W
- wait() (*LiveResponseMemdump* method), 249
- WAIT_TIMEOUT (*Win32Error* attribute), 338
- walk() (*CbLRSessionBase* method), 247
- Watchlist (class in *cbc_sdk.enterprise_edr.threat_intelligence*), 108
- Watchlist.WatchlistBuilder (class in *cbc_sdk.enterprise_edr.threat_intelligence*), 109
- WatchlistAlert (class in *cbc_sdk.platform.alerts*), 129
- WatchlistAlertSearchQuery (class in *cbc_sdk.platform.alerts*), 129
- WatchlistQuery (class in *cbc_sdk.enterprise_edr.threat_intelligence*), 112
- where() (*FeedQuery* method), 101
- where() (*QueryBuilder* method), 226
- where() (*QueryBuilderSupportMixin* method), 227
- where() (*ReportQuery* method), 108
- where() (*RunQuery* method), 63
- where() (*SimpleQuery* method), 228
- win16_E_ABORT (*RawErrorCode* attribute), 285
- win16_E_ACCESSDENIED (*RawErrorCode* attribute), 285
- win16_E_FAIL (*RawErrorCode* attribute), 285
- win16_E_HANDLE (*RawErrorCode* attribute), 285
- win16_E_INVALIDARG (*RawErrorCode* attribute), 285
- win16_E_NOINTERFACE (*RawErrorCode* attribute), 285
- win16_E_NOTIMPL (*RawErrorCode* attribute), 285

`win16_E_OUTOFMEMORY` (*RawErrorCode attribute*),
285

`win16_E_POINTER` (*RawErrorCode attribute*), 285

`Win32Error` (*class in cbc_sdk.winerror*), 286

`windows_event_id` (*AuthEvent attribute*), 94

`windows_platform` (*Device attribute*), 138

`WorkerStatus` (*class in cbc_sdk.live_response_api*),
250

`workflow` (*BaseAlert attribute*), 118

`Workflow` (*class in cbc_sdk.platform.alerts*), 129

`workflow` (*WorkflowStatus attribute*), 131

`workflow_` (*BaseAlert attribute*), 118

`workflow_` (*Recommendation attribute*), 82

`workflow_` (*WorkflowStatus attribute*), 131

`WorkflowStatus` (*class in cbc_sdk.platform.alerts*),
130

`WorkItem` (*class in cbc_sdk.live_response_api*), 250

Y

`yaml_constructors` (*SwaggerLoader attribute*), 228