libcloudforensics

Google

Jul 06, 2020

GOOGLE CLOUD PLATFORM

CHAPTER

ONE

API DOCUMENTATION

You'll find links to the API documentation below.

1.1 GCP forensics package

1.1.1 Internal provider functions

GCP internal provider functions

libcloudforensics.providers.gcp.internal.build module

Google Cloud Build functionalities.

class `libcloudforensics.providers.gcp.internal.build.`**`GoogleCloudBuild`**(*project_id*)

Bases: `object`

Class to call Google Cloud Build APIs.

Dictionary objects content can be found in https://cloud.google.com/cloud-build/docs/api/reference/rest/v1/projects.builds

`gcb_api_client`

Client to interact with GCB APIs.

`BlockOperation`(*response*)

Block execution until API operation is finished.

Parameters **`response`** (`Dict`) – Google Cloud Build API response.

Returns

Holding the response of a get operation on an API object of type operations.

Return type Dict

Raises **`RuntimeError`** – If the Cloud Build failed or if getting the Cloud Build API operation object failed.

`CLOUD_BUILD_API_VERSION = 'v1'`

`CreateBuild`(*build_body*)

Create a cloud build.

Parameters **`build_body`** (`Dict`) – A dictionary that describes how to find the source code and how to build it.

Returns

Represents long-running operation that is the result of a network API call.

Return type Dict

GcbApi()

Get a Google Cloud Build service object.

Returns A Google Cloud Build service object.

Return type googleapiclient.discovery.Resource

libcloudforensics.providers.gcp.internal.common module

Common utilities.

libcloudforensics.providers.gcp.internal.common.**CreateService**(*service_name*, *api_version*)

Creates an GCP API service.

Parameters

- **service_name** (*str*) – Name of the GCP service to use.
- **api_version** (*str*) – Version of the GCP service API to use.

Returns API service resource.

Return type googleapiclient.discovery.Resource

Raises **RuntimeError** – If Application Default Credentials could not be obtained or if service build times out.

libcloudforensics.providers.gcp.internal.common.**ExecuteRequest**(*client*, *func*, *kwargs*, *throttle=False*)

Execute a request to the GCP API.

Parameters

- **client** (*googleapiclient.discovery.Resource*) – A GCP client object.
- **func** (*str*) – A GCP function to query from the client.
- **kwargs** (*Dict*) – A dictionary of parameters for the function func.
- **throttle** (*bool*) – A boolean indicating if requests should be throttled. This is necessary for some APIs (e.g. list logs) as there is an API rate limit. Default is False, i.e. requests are not throttled.

Returns A List of dictionaries (responses from the request).

Return type List[Dict]

Raises **RuntimeError** – If the request to the GCP API could not complete.

libcloudforensics.providers.gcp.internal.common.**FormatRFC3339**(*datetime_instance*)

Formats a datetime per RFC 3339.

Parameters **datetime_instance** (datetime) – The datetime group to be formatted.

Returns A string formatted as per RFC3339 (e.g 2018-05-11T12:34:56.992Z)

Return type str

libcloudforensics.providers.gcp.internal.common.**GenerateDiskName**(*snapshot*, *disk_name_prefix=None*)

Generate a new disk name for the disk to be created from the Snapshot.

The disk name must comply with the following RegEx:

- ^(?=.{1,63}$)[a-z]([-a-z0-9]*[a-z0-9])?$

i.e., it must be between 1 and 63 chars, the first character must be a lowercase letter, and all following characters must be a dash, lowercase letter, or digit, except the last character, which cannot be a dash.

Parameters

- **snapshot** (GoogleComputeSnapshot) – A disk's Snapshot.
- **disk_name_prefix** (*str*) – Optional. A prefix for the disk name.

Returns A name for the disk.

Return type str

Raises **ValueError** – If the disk name does not comply with the RegEx.

libcloudforensics.providers.gcp.internal.common.**GenerateUniqueInstanceName**(*prefix*, *truncate_at=None*)

Add a timestamp as a suffix to provided name and truncate at max limit.

Parameters

- **prefix** (*str*) – The name prefix to add the timestamp to and truncate.
- **truncate_at** (*int*) – Optional. The maximum length of the generated name. Default no limit.

Returns

The name after adding a timestamp. Ex: [prefix]-[TIMESTAMP('%Y%m%d%H%M%S')]

Return type str

class libcloudforensics.providers.gcp.internal.common.**GoogleCloudComputeClient**(*project_id=None*

Bases: object

Class representing Google Cloud Compute API client.

Request and response dictionary content is described here: https://cloud.google.com/compute/docs/reference/rest/v1

project_id

Project name.

Type str

BlockOperation(*response*, *zone=None*)

Block until API operation is finished.

Parameters

- **response** (*Dict*) – GCE API response.
- **zone** (*str*) – Optional. GCP zone to execute the operation in. None means GlobalZone.

Returns

Holding the response of a get operation on an API object of type zoneOperations or globalOperations.

Return type Dict

Raises **RuntimeError** – If API call failed.

COMPUTE_ENGINE_API_VERSION = 'v1'

GceApi()

Get a Google Compute Engine service object.

Returns

A Google Compute Engine service object.

Return type googleapiclient.discovery.Resource

libcloudforensics.providers.gcp.internal.compute module

Google Compute Engine functionalities.

class libcloudforensics.providers.gcp.internal.compute.**GoogleCloudCompute**(*project_id*, *default_zone=None*)

Bases: *libcloudforensics.providers.gcp.internal.common.GoogleCloudComputeClient*

Class representing all Google Cloud Compute objects in a project.

project_id

Project name.

default_zone

Default zone to create new resources in.

CreateDiskFromImage(*src_image*, *zone*, *name=None*)

Creates a GCE persistent disk from a GCE image.

Parameters

- **src_image** (GoogleComputeImage) – Source image for the disk.
- **zone** (*str*) – Zone to create the new disk in.
- **name** (*str*) – Optional. Name of the disk to create. Default is [src_image.name]-[TIMESTAMP('%Y%m%d%H%M%S')].

Returns A Google Compute Disk object.

Return type *GoogleComputeDisk*

Raises **ValueError** – If GCE disk name is invalid.

CreateDiskFromSnapshot(*snapshot*, *disk_name=None*, *disk_name_prefix=''*, *disk_type='pd-standard'*)

Create a new disk based on a Snapshot.

Parameters

- **snapshot** (GoogleComputeSnapshot) – Snapshot to use.
- **disk_name** (*str*) – Optional. String to use as new disk name.
- **disk_name_prefix** (*str*) – Optional. String to prefix the disk name with.

- **disk_type** (*str*) – Optional. URL of the disk type resource describing which disk type to use to create the disk. Default is pd-standard. Use pd-ssd to have a SSD disk. You can list all available disk types by running the following command: gcloud compute disk-types list

Returns Google Compute Disk.

Return type *GoogleComputeDisk*

Raises **RuntimeError** – If the disk exists already.

CreateImageFromDisk(*src_disk*, *name=None*)

Creates an image from a persistent disk.

Parameters

- **src_disk** (GoogleComputeDisk) – Source disk for the image.
- **name** (*str*) – Optional. Name of the image to create. Default is [src_disk.name]-[TIMESTAMP('%Y%m%d%H%M%S')].

Returns A Google Compute Image object.

Return type *GoogleComputeImage*

Raises **ValueError** – If the GCE Image name is invalid.

Disks(*refresh=True*)

Get all disks in the project.

Parameters **refresh** (*boolean*) – Optional. Returns refreshed result if True.

Returns

Dictionary mapping disk names (str) to their respective GoogleComputeDisk object.

Return type Dict[str, *GoogleComputeDisk*]

GetDisk(*disk_name*)

Get a GCP disk object.

Parameters **disk_name** (*str*) – Name of the disk.

Returns Disk object.

Return type *GoogleComputeDisk*

Raises **RuntimeError** – When the specified disk cannot be found in project.

GetInstance(*instance_name*)

Get instance from project.

Parameters **instance_name** (*str*) – The instance name.

Returns A Google Compute Instance object.

Return type *GoogleComputeInstance*

Raises **RuntimeError** – If instance does not exist.

GetOrCreateAnalysisVm(*vm_name*, *boot_disk_size*, *disk_type='pd-standard'*, *cpu_cores=4*, *image_project='ubuntu-os-cloud'*, *image_family='ubuntu-1804-lts'*, *packages=None*)

Get or create a new virtual machine for analysis purposes.

If none of the optional parameters are specified, then by default the analysis VM that will be created will run Ubuntu 18.04 LTS. A default set of forensic tools is also installed (a custom one may be provided using the 'packages' argument).

Parameters

- **vm_name** (*str*) – Name of the virtual machine.
- **boot_disk_size** (*int*) – The size of the analysis VM boot disk (in GB).
- **disk_type** (*str*) – Optional. URL of the disk type resource describing which disk type to use to create the disk. Default is pd-standard. Use pd-ssd to have a SSD disk.
- **cpu_cores** (*int*) – Optional. Number of CPU cores for the virtual machine.
- **image_project** (*str*) – Optional. Name of the project where the analysis VM image is hosted.
- **image_family** (*str*) – Optional. Name of the image to use to create the analysis VM.
- **packages** (*List[str]*) – Optional. List of packages to install in the VM.

Returns

A tuple with a virtual machine object and a boolean indicating if the virtual machine was created or not.

Return type Tuple(*GoogleComputeInstance*, bool)

Raises **RuntimeError** – If virtual machine cannot be created.

ImportImageFromStorage(*storage_image_path*, *image_name=None*, *bootable=False*, *os_name=None*, *guest_environment=True*)

Import GCE image from Cloud storage.

The import tool supports raw disk images and most virtual disk file formats, valid import formats are: [raw (dd), qcow2, qcow , vmdk, vdi, vhd, vhdx, qed, vpc].

Parameters

- **storage_image_path** (*str*) – Path to the source image in Cloud Storage.
- **image_name** (*str*) – Optional. Name of the imported image, default is "imported-image-" appended with a timestamp in "%Y%m%d%H%M%S" format.
- **bootable** (*bool*) – Optional. True if the imported image is bootable. Default is False. If True the os_name must be specified.
- **os_name** (*str*) – Optional. Name of the operating system on the bootable image. For supported versions please see: https://cloud.google.com/sdk/gcloud/reference/compute/images/import#–os # pylint: disable=line-too-long For known limitations please see: https://googlecloudplatform.github.io/compute-image-tools/image-import.html#compatibility-and-known-limitations # pylint: disable=line-too-long
- **guest_environment** (*bool*) – Optional. Install Google Guest Environment on a bootable image. Relevant only if image is bootable. Default True.

Returns A Google Compute Image object.

Return type *GoogleComputeImage*

Raises **ValueError** – If bootable is True and os_name not specified or if imported image name is invalid.

Instances(*refresh=True*)

Get all instances in the project.

Parameters **refresh** (*boolean*) – Optional. Returns refreshed result if True.

Returns

Dictionary mapping instance names (str) to their respective GoogleComputeInstance object.

Return type Dict[str, *GoogleComputeInstance*]

ListDiskByLabels(*labels_filter*, *filter_union=True*)

List Disks in a project with one/all of the provided labels.

This will call the _ListByLabel function on a disks() API object with the proper labels filter and return a Dict with name and metadata for each disk, e.g.:

{'disk-1': {'zone': 'us-central1-a', 'labels': {'id': '123'}}

Parameters

- **labels_filter** (*Dict[str, str]*) – A Dict of labels to find e.g. {'id': '123'}.
- **filter_union** (*bool*) – Optional. A Boolean; True to get the union of all filters, False to get the intersection.

Returns

Dictionary mapping disks to their respective GoogleComputeDisk object.

Return type Dict[str, *GoogleComputeDisk*]

ListDisks()

List disks in project.

Returns

Dictionary mapping disk names (str) to their respective GoogleComputeDisk object.

Return type Dict[str, *GoogleComputeDisk*]

ListInstanceByLabels(*labels_filter*, *filter_union=True*)

List VMs in a project with one/all of the provided labels.

This will call the _ListByLabel function on an instances() API object with the proper labels filter and return a Dict with name and metadata for each instance, e.g.:

{'instance-1': {'zone': 'us-central1-a', 'labels': {'id': '123'}}

Parameters

- **labels_filter** (*Dict[str, str]*) – A Dict of labels to find e.g. {'id': '123'}.
- **filter_union** (*bool*) – Optional. A Boolean; True to get the union of all filters, False to get the intersection.

Returns

Dictionary mapping instances to their respective GoogleComputeInstance object.

Return type Dict[str, *GoogleComputeInstance*]

ListInstances()

List instances in project.

Returns

Dictionary mapping instance names (str) to their respective GoogleComputeInstance object.

Return type Dict[str, *GoogleComputeInstance*]

class libcloudforensics.providers.gcp.internal.compute.**GoogleComputeDisk**(*project_id*, *zone*, *name*, *labels=None*)

Bases: *libcloudforensics.providers.gcp.internal.compute_base_resource.GoogleComputeBaseResource*

Class representing a Compute Engine disk.

GetOperation()

Get API operation object for the disk.

Returns

An API operation object for a Google Compute Engine disk. https://cloud.google.com/compute/docs/reference/rest/v1/disks/get#response-body

Return type Dict

Snapshot(*snapshot_name=None*)

Create Snapshot of the disk.

The Snapshot name must comply with the following RegEx:

- ^(?=.{1,63}$)[a-z]([-a-z0-9]*[a-z0-9])?$

i.e., it must be between 1 and 63 chars, the first character must be a lowercase letter, and all following characters must be a dash, lowercase letter, or digit, except the last character, which cannot be a dash.

Parameters **snapshot_name** (*str*) – Optional. Name of the Snapshot.

Returns A Snapshot object.

Return type *GoogleComputeSnapshot*

Raises **ValueError** – If the name of the snapshot does not comply with the RegEx.

class libcloudforensics.providers.gcp.internal.compute.**GoogleComputeImage**(*project_id*, *zone*, *name*, *labels=None*)

Bases: *libcloudforensics.providers.gcp.internal.compute_base_resource.GoogleComputeBaseResource*

Class representing a Compute Engine Image.

Delete()

Delete Compute Disk Image from a project.

Return type None

ExportImage(*gcs_output_folder*, *output_name=None*)

Export compute image to Google Cloud storage.

Exported image is compressed and stored in .tar.gz format.

Parameters

- **gcs_output_folder** (*str*) – Folder path of the exported image.
- **output_name** (*str*) – Optional. Name of the output file. Name will be appended with .tar.gz. Default is [image_name].tar.gz.

Raises **RuntimeError** – If exported image name is invalid.

Return type None

GetOperation()

Get API operation object for the image.

Returns

Holding an API operation object for a Google Compute Engine Image. https://cloud.google.com/compute/docs/reference/rest/v1/images/get#response-body

Return type Dict

class libcloudforensics.providers.gcp.internal.compute.**GoogleComputeInstance**(*project_id*, *zone*, *name*, *labels=None*)

Bases: *libcloudforensics.providers.gcp.internal.compute_base_resource.GoogleComputeBaseResource*

Class representing a Google Compute Engine virtual machine.

AttachDisk(*disk*, *read_write=False*)

Attach a disk to the virtual machine.

Parameters

- **disk** (GoogleComputeDisk) – Disk to attach.
- **read_write** (*bool*) – Optional. Boolean indicating whether the disk should be attached in RW mode. Default is False (read-only).

Return type None

DetachDisk(*disk*)

Detach a disk from the virtual machine.

Parameters **disk** (GoogleComputeDisk) – Disk to detach.

Return type None

GetBootDisk()

Get the virtual machine boot disk.

Returns Disk object or None if no disk can be found.

Return type *GoogleComputeDisk*

GetDisk(*disk_name*)

Gets a disk attached to this virtual machine disk by name.

Parameters **disk_name** (*str*) – The name of the disk to get.

Returns Disk object.

Return type *GoogleComputeDisk*

Raises **RuntimeError** – If disk name is not found among those attached to the instance.

GetOperation()

Get API operation object for the virtual machine.

Returns

An API operation object for a Google Compute Engine virtual machine. https://cloud.google.com/compute/docs/reference/rest/v1/instances/get#response-body

Return type Dict

ListDisks()

List all disks for the virtual machine.

Returns

Dictionary mapping disk names to their respective GoogleComputeDisk object.

Return type Dict[str, *GoogleComputeDisk*]

Ssh()

Connect to the virtual machine over SSH.

Return type `None`

class libcloudforensics.providers.gcp.internal.compute.**GoogleComputeSnapshot**(*disk*, *name*)

Bases: *libcloudforensics.providers.gcp.internal.compute_base_resource.GoogleComputeBaseResource*

Class representing a Compute Engine Snapshot.

disk

Disk used for the Snapshot.

Type *GoogleComputeDisk*

Delete()

Delete a Snapshot.

Return type `None`

GetOperation()

Get API operation object for the Snapshot.

Returns

An API operation object for a Google Compute Engine Snapshot. https://cloud.google.com/compute/docs/reference/rest/v1/snapshots/get#response-body

Return type Dict

libcloudforensics.providers.gcp.internal.compute_base_resource module

Google Compute Engine resources.

class libcloudforensics.providers.gcp.internal.compute_base_resource.**GoogleComputeBaseReso**

Bases: *libcloudforensics.providers.gcp.internal.common.GoogleCloudComputeClient*

Base class representing a Computer Engine resource.

project_id

Google Cloud project ID.

Type str

zone

What zone the resource is in.

Type str

name
Name of the resource.

Type str

labels
Dictionary of labels for the resource, if existing.

Type Dict

AddLabels(*new_labels_dict*, *blocking_call=False*)
Add or update labels of a compute resource.

Parameters

- **new_labels_dict** (*Dict*) – A dictionary containing the labels to be added, ex:{"incident_id": "1234abcd"}.
- **blocking_call** (*bool*) – Optional. A boolean to decide whether the API call should be blocking or not, default is False.

Returns

The response of the API operation (a Dict if the call is successful).

Return type Optional[Any]

Raises **RuntimeError** – If the Compute resource Type is not one of instance, disk or snapshot.

FormOperation(*operation_name*)
Form an API operation object for the compute resource.

Example:[RESOURCE].FormOperation('setLabels')(**kwargs) [RESOURCE] can be type "instance", disk or "Snapshot".

Parameters **operation_name** (*str*) – The name of the API operation you need to perform.

Returns

An API operation object for the referenced compute resource.

Return type googleapiclient.discovery.Resource

Raises **RuntimeError** – If resource type is not defined as a type which extends the GoogleComputeBaseResource class.

FormatLogMessage(*message*)
Format log messages with project specific information.

Parameters **message** (*str*) – Message string to log.

Returns Formatted log message string.

Return type str

GetLabels()
Get all labels of a compute resource.

Returns A dictionary of all labels.

Return type Dict

GetOperation()
Abstract method to be implemented by child classes.

Raises **NotImplementedError** – If the child class doesn't implement GetOperation.

Return type `Dict[str, Any]`

GetResourceType()

Get the resource type from the resource key-value store.

Returns

Resource Type which is a string with one of the following values: compute#instance compute#disk compute#Snapshot

Return type str

GetSourceString()

API URL to the resource.

Returns The full API URL to the resource.

Return type str

GetValue(*key*)

Get specific value from the resource key value store.

Parameters **key** (*str*) – A key of type String to get key's corresponding value.

Returns Value of key or None if key is missing.

Return type str

libcloudforensics.providers.gcp.internal.function module

Google Cloud Functions functionalities.

class libcloudforensics.providers.gcp.internal.function.**GoogleCloudFunction**(*project_id*)

Bases: `object`

Class to call Google Cloud Functions.

project_id

Google Cloud project ID.

gcf_api_client

Client to interact with GCF APIs.

CLOUD_FUNCTIONS_API_VERSION = 'v1'

ExecuteFunction(*function_name*, *region*, *args*)

Executes a Google Cloud Function.

Parameters

- **function_name** (*str*) – The name of the function to call.
- **region** (*str*) – Region to execute functions in.
- **args** (*Dict*) – Arguments to pass to the function. Dictionary content details can be found in https://cloud.google.com/functions/docs/reference/rest/v1/projects.locations.functions # pylint: disable=line-too-long

Returns Return value from function call.

Return type Dict[str, str]

Raises **RuntimeError** – When cloud function arguments cannot be serialized or when an HttpError is encountered.

GcfApi()
Get a Google Cloud Function service object.

Returns

A Google Cloud Function service object.

Return type googleapiclient.discovery.Resource

libcloudforensics.providers.gcp.internal.log module

Google Cloud Logging functionalities.

class libcloudforensics.providers.gcp.internal.log.**GoogleCloudLog**(*project_id*)
Bases: object

Class representing a Google Cloud Logs interface.

project_id
Google Cloud project ID.

gcl_api_client
Client to interact with GCP logging API.

Example use: # pylint: disable=line-too-long gcp = GoogleCloudLog(project_id='your_project_name') gcp.ListLogs() gcp.ExecuteQuery(filter='resource.type="gce_instance" labels."compute.googleapis.com/resource_name"="instance-1"') See https://cloud.google.com/logging/docs/view/advanced-queries for filter details.

ExecuteQuery(*qfilter*)
Query logs in GCP project.

Parameters **qfilter** (*str*) – The query filter to use.

Returns

Log entries returned by the query, e.g. [{'projectIds': [...], 'resourceNames': [...]}, {...}]

Return type List[Dict]

Raises **RuntimeError** – If API call failed.

GclApi()
Get a Google Compute Logging service object.

Returns

A Google Compute Logging service object.

Return type googleapiclient.discovery.Resource

LOGGING_API_VERSION = 'v2'

ListLogs()
List logs in project.

Returns The project logs available.

Return type List[str]

Raises **RuntimeError** – If API call failed.

libcloudforensics.providers.gcp.internal.monitoring module

Google Cloud Monitoring functionality.

class libcloudforensics.providers.gcp.internal.monitoring.**GoogleCloudMonitoring**(*project_id*)

Bases: object

Class to call Google Monitoring APIs.

https://cloud.google.com/monitoring/api/ref_v3/rest/v3/projects.timeSeries

project_id

Project name.

gcm_api_client

Client to interact with Monitoring APIs.

ActiveServices(*timeframe=30*)

List active services in the project (default: last 30 days).

Parameters **timeframe** (*int*) – Optional. The number (in days) for which to measure activity.

Returns Dictionary mapping service name to number of uses.

Return type Dict[str, int]

CLOUD_MONITORING_API_VERSION = 'v3'

GcmApi()

Get a Google Cloud Monitoring service object.

Returns

A Google Cloud Monitoring service object.

Return type googleapiclient.discovery.Resource

libcloudforensics.providers.gcp.internal.project module

Google Cloud Project resources and services.

class libcloudforensics.providers.gcp.internal.project.**GoogleCloudProject**(*project_id*, *default_zone=None*)

Bases: object

Class representing a Google Cloud Project.

project_id

Google Cloud project ID.

default_zone

Default zone to create new resources in.

Example use: gcp = GoogleCloudProject("your_project_name", "us-east") gcp.compute.ListInstances()

property build

Get a GoogleCloudBuild object for the project.

Returns Object that represents Google Cloud Build.

Return type *GoogleCloudBuild*

property compute
Get a GoogleCloudCompute object for the project.

Returns Object that represents Google Cloud Compute Engine.

Return type *GoogleCloudCompute*

property function
Get a GoogleCloudFunction object for the project.

Returns Object that represents Google Cloud Function.

Return type *GoogleCloudFunction*

property log
Get a GoogleCloudLog object for the project.

Returns Object that represents Google Cloud Logging.

Return type *GoogleCloudLog*

property monitoring
Get a GoogleCloudMonitoring object for the project.

Returns Object that represents Google Monitoring.

Return type *GoogleCloudMonitoring*

property storage
Get a GoogleCloudStorage object for the project.

Returns Object that represents Google Cloud Logging.

Return type *GoogleCloudLog*

libcloudforensics.providers.gcp.internal.storage module

Google Cloud Storage functionalities.

class libcloudforensics.providers.gcp.internal.storage.**GoogleCloudStorage**(*project_id=None*)
Bases: object

Class to call Google Cloud Storage APIs.

gcs_api_client
Client to interact with GCS APIs.

CLOUD_STORAGE_API_VERSION = 'v1'

GcsApi()
Get a Google Cloud Storage service object.

Returns A Google Cloud Storage service object.

Return type googleapiclient.discovery.Resource

GetObjectMetadata(*gcs_path*, *user_project=None*)
Get API operation object metadata for Google Cloud Storage object.

Parameters

- **gcs_path** (*str*) – File path to a resource in GCS. Ex: gs://bucket/folder/obj
- **user_project** (*str*) – The project ID to be billed for this request. Required for Requester Pays buckets.

Returns

An API operation object for a Google Cloud Storage object. https://cloud.google.com/storage/docs/json_api/v1/objects#resource

Return type Dict

`libcloudforensics.providers.gcp.internal.storage.`**`SplitGcsPath`**(*gcs_path*)

Split GCS path to bucket name and object URI.

Parameters **`gcs_path`** (*str*) – File path to a resource in GCS. Ex: gs://bucket/folder/obj

Returns Bucket name. Object URI.

Return type Tuple[str, str]

1.1.2 libcloudforensics.providers.gcp.forensics module

Forensics on GCP.

`libcloudforensics.providers.gcp.forensics.`**`CreateDiskCopy`**(*src_proj*, *dst_proj*, *instance_name*, *zone*, *disk_name=None*, *disk_type='pd-standard'*)

Creates a copy of a Google Compute Disk.

Parameters

- **`src_proj`** (*str*) – Name of project that holds the disk to be copied.
- **`dst_proj`** (*str*) – Name of project to put the copied disk in.
- **`instance_name`** (*str*) – Instance using the disk to be copied.
- **`zone`** (*str*) – Zone where the new disk is to be created.
- **`disk_name`** (*str*) – Optional. Name of the disk to copy. If None, boot disk will be copied.
- **`disk_type`** (*str*) – Optional. URL of the disk type resource describing which disk type to use to create the disk. Default is pd-standard. Use pd-ssd to have a SSD disk.

Returns A Google Compute Disk object.

Return type *GoogleComputeDisk*

Raises **`RuntimeError`** – If there are errors copying the disk

`libcloudforensics.providers.gcp.forensics.`**`CreateDiskFromGCSImage`**(*project_id*, *storage_image_path*, *zone*, *name=None*)

Creates a GCE persistent disk from a image in GCS.

The method supports raw disk images and most virtual disk file formats. Valid import formats are: [raw (dd), qcow2, qcow , vmdk, vdi, vhd, vhdx, qed, vpc].

The created GCE disk might be larger than the original raw (dd) image stored in GCS to satisfy GCE capacity requirements: https://cloud.google.com/compute/docs/disks/#introduction However the bytes_count and the md5_hash values of the source image are returned with the newly created disk. The md5_hash can be used to verify the integrity of the created GCE disk, it must be compared with the hash of the created GCE disk from byte 0 to bytes_count. i.e: result['md5Hash'] = hash(created_gce_disk,

start_byte=0, end_byte=result['bytes_count'])

Parameters

- **project_id** (*str*) – Google Cloud Project ID.
- **storage_image_path** (*str*) – Path to the source image in GCS.
- **zone** (*str*) – Zone to create the new disk in.
- **name** (*str*) – Optional. Name of the disk to create. Default is imported-disk-[TIMESTAMP('%Y%m%d%H%M%S')].

Returns

A key value describing the imported GCE disk.

Ex: { 'project_id': 'fake-project', 'disk_name': 'fake-imported-disk', 'zone': 'fake-zone', 'bytes_count': '1234' # Content-Length of source image in bytes. 'md5Hash': 'Source Image MD5 hash string in hex'

}

Return type Dict

Raises **ValueError** – If the GCE disk name is invalid.

libcloudforensics.providers.gcp.forensics.**StartAnalysisVm**(*project*, *vm_name*, *zone*, *boot_disk_size*, *boot_disk_type*, *cpu_cores*, *attach_disks=None*, *image_project='ubuntu-os-cloud'*, *image_family='ubuntu-1804-lts'*)

Start a virtual machine for analysis purposes.

Parameters

- **project** (*str*) – Project id for virtual machine.
- **vm_name** (*str*) – The name of the virtual machine.
- **zone** (*str*) – Zone for the virtual machine.
- **boot_disk_size** (*int*) – The size of the analysis VM boot disk (in GB).
- **boot_disk_type** (*str*) – URL of the disk type resource describing which disk type to use to create the disk. Use pd-standard for a standard disk and pd-ssd for a SSD disk.
- **cpu_cores** (*int*) – The number of CPU cores to create the machine with.
- **attach_disks** (*List[str]*) – Optional. List of disk names to attach.
- **image_project** (*str*) – Optional. Name of the project where the analysis VM image is hosted.
- **image_family** (*str*) – Optional. Name of the image to use to create the analysis VM.

Returns

A tuple with a virtual machine object and a boolean indicating if the virtual machine was created or not.

Return type Tuple(*GoogleComputeInstance*, bool)

1.2 AWS forensics package

1.2.1 Internal provider functions

AWS internal provider functions

libcloudforensics.providers.aws.internal.account module

Library for incident response operations on AWS EC2.

Library to make forensic images of Amazon Elastic Block Store devices and create analysis virtual machine to be used in incident response.

class libcloudforensics.providers.aws.internal.account.**AWSAccount**(*default_availability_zone*, *aws_profile=None*)

Bases: object

Class representing an AWS account.

default_availability_zone

Default zone within the region to create new resources in.

Type str

aws_profile

The AWS profile defined in the AWS credentials file to use.

Type str

ClientApi(*service*, *region=None*)

Create an AWS client object.

Parameters

- **service** (*str*) – The AWS service to use.
- **region** (*str*) – Optional. The region in which to create new resources. If none provided, the default_region associated to the AWSAccount object will be used.

Returns An AWS EC2 client object.

Return type botocore.client.EC2

CreateKMSKey()

Create a KMS key.

Returns The KMS key ID for the key that was created.

Return type str

Raises **RuntimeError** – If the key could not be created.

CreateVolumeFromSnapshot(*snapshot*, *volume_name=None*, *volume_name_prefix=''*, *kms_key_id=None*, *tags=None*)

Create a new volume based on a snapshot.

Parameters

- **snapshot** (AWSSnapshot) – Snapshot to use.
- **volume_name** (*str*) – Optional. String to use as new volume name.
- **volume_name_prefix** (*str*) – Optional. String to prefix the volume name with.

- **kms_key_id** (*str*) – Optional. A KMS key id to encrypt the volume with.
- **tags** (*Dict[str, str]*) – Optional. A dictionary of tags to add to the volume, for example {'TicketID': 'xxx'}. An entry for the volume name is added by default.

Returns An AWS EBS Volume.

Return type *AWSVolume*

Raises

- **ValueError** – If the volume name does not comply with the RegEx.
- **RuntimeError** – If the volume could not be created.

DeleteKMSKey(*kms_key_id=None*)

Delete a KMS key.

Schedule the KMS key for deletion. By default, users have a 30 days window before the key gets deleted.

Parameters **kms_key_id** (*str*) – The ID of the KMS key to delete.

Raises **RuntimeError** – If the key could not be scheduled for deletion.

Return type None

GetAccountInformation(*info*)

Get information about the AWS account in use.

If the call succeeds, then the response from the STS API is expected to have the following entries:

- UserId
- Account
- Arn

See https://boto3.amazonaws.com/v1/documentation/api/1.9.42/reference/services/sts.html#STS.Client.get_caller_identity for more details. # pylint: disable=line-too-long

Parameters **info** (*str*) – The account information to retrieve. Must be one of [UserID, Account, Arn]

Returns The information requested.

Return type str

Raises **KeyError** – If the requested information doesn't exist.

GetInstanceById(*instance_id, region=None*)

Get an instance from an AWS account by its ID.

Parameters

- **instance_id** (*str*) – The instance id.
- **region** (*str*) – Optional. The region to look the instance in. If none provided, the default_region associated to the AWSAccount object will be used.

Returns An Amazon EC2 Instance object.

Return type *AWSInstance*

Raises **RuntimeError** – If instance does not exist.

GetInstancesByName(*instance_name*, *region=None*)

Get all instances from an AWS account with matching name tag.

Parameters

- **instance_name** (*str*) – The instance name tag.
- **region** (*str*) – Optional. The region to look the instance in. If none provided, the default_region associated to the AWSAccount object will be used.

Returns

A list of EC2 Instance objects. If no instance with matching name tag is found, the method returns an empty list.

Return type List[*AWSInstance*]

GetInstancesByNameOrId(*instance_name=''*, *instance_id=''*, *region=None*)

Get instances from an AWS account by their name tag or an ID.

Exactly one of [instance_name, instance_id] must be specified. If looking up an instance by its ID, the method returns a list with exactly one element. If looking up instances by their name tag (which are not unique across instances), then the method will return a list of all instances with that name tag, or an empty list if no instances with matching name tag could be found.

Parameters

- **instance_name** (*str*) – Optional. The instance name tag of the instance to get.
- **instance_id** (*str*) – Optional. The instance id of the instance to get.
- **region** (*str*) – Optional. The region to look the instance in. If none provided, the default_region associated to the AWSAccount object will be used.

Returns A list of Amazon EC2 Instance objects.

Return type List[*AWSInstance*]

Raises **ValueError** – If both instance_name and instance_id are None or if both are set.

GetOrCreateAnalysisVm(*vm_name*, *boot_volume_size*, *ami*, *cpu_cores*, *packages=None*, *ssh_key_name=None*, *tags=None*)

Get or create a new virtual machine for analysis purposes.

Parameters

- **vm_name** (*str*) – The instance name tag of the virtual machine.
- **boot_volume_size** (*int*) – The size of the analysis VM boot volume (in GB).
- **ami** (*str*) – The Amazon Machine Image ID to use to create the VM.
- **cpu_cores** (*int*) – Number of CPU cores for the analysis VM.
- **packages** (*List[str]*) – Optional. List of packages to install in the VM.
- **ssh_key_name** (*str*) – Optional. A SSH key pair name linked to the AWS account to associate with the VM. If none provided, the VM can only be accessed through in-browser SSH from the AWS management console with the EC2 client connection package (ec2-instance-connect). Note that if this package fails to install on the target VM, then the VM will not be accessible. It is therefore recommended to fill in this parameter.
- **tags** (*Dict[str, str]*) – Optional. A dictionary of tags to add to the instance, for example {'TicketID': 'xxx'}. An entry for the instance name is added by default.

Returns

A tuple with an AWSInstance object and a boolean indicating if the virtual machine was created (True) or reused (False).

Return type Tuple[*AWSInstance*, bool]

Raises **RuntimeError** – If the virtual machine cannot be found or created.

GetVolumeById(*volume_id*, *region=None*)

Get a volume from an AWS account by its ID.

Parameters

- **volume_id** (*str*) – The volume id.
- **region** (*str*) – Optional. The region to look the volume in. If none provided, the default_region associated to the AWSAccount object will be used.

Returns An Amazon EC2 Volume object.

Return type *AWSVolume*

Raises **RuntimeError** – If volume does not exist.

GetVolumesByName(*volume_name*, *region=None*)

Get all volumes from an AWS account with matching name tag.

Parameters

- **volume_name** (*str*) – The volume name tag.
- **region** (*str*) – Optional. The region to look the volume in. If none provided, the default_region associated to the AWSAccount object will be used.

Returns

A list of EC2 Volume objects. If no volume with matching name tag is found, the method returns an empty list.

Return type List[*AWSVolume*]

GetVolumesByNameOrId(*volume_name=''*, *volume_id=''*, *region=None*)

Get a volume from an AWS account by its name tag or its ID.

Exactly one of [volume_name, volume_id] must be specified. If looking up a volume by its ID, the method returns a list with exactly one element. If looking up volumes by their name tag (which are not unique across volumes), then the method will return a list of all volumes with that name tag, or an empty list if no volumes with matching name tag could be found.

Parameters

- **volume_name** (*str*) – Optional. The volume name tag of the volume to get.
- **volume_id** (*str*) – Optional. The volume id of the volume to get.
- **region** (*str*) – Optional. The region to look the volume in. If none provided, the default_region associated to the AWSAccount object will be used.

Returns A list of Amazon EC2 Volume objects.

Return type List[*AWSVolume*]

Raises **ValueError** – If both volume_name and volume_id are None or if both are set.

ListImages(*qfilter=None*)

List AMI images.

Parameters

- **qfilter** (*List[Dict]*) – The filter expression.
- **https** (*See*) – //boto3.amazonaws.com/v1/documentation/api/latest/reference/services/ec2.html#EC2.Client.de
  # pylint: disable=line-too-long

Returns The list of images with their properties.

Return type List[Dict[str, Any]]

Raises **RuntimeError** – If the images could not be listed.

ListInstances (*region=None*, *filters=None*, *show_terminated=False*)
List instances of an AWS account.

Example usage:

ListInstances(region='us-east-1', filters=[{'Name':'instance-id', 'Values':['some-instance-id']}])

Parameters

- **region** (*str*) – Optional. The region from which to list instances. If none provided, the default_region associated to the AWSAccount object will be used.
- **filters** (*List[Dict]*) – Optional. Filters for the query. Filters are given as a list of dictionaries, e.g.: {'Name': 'someFilter', 'Values': ['value1', 'value2']}.
- **show_terminated** (*bool*) – Optional. Include terminated instances in the list.

Returns

Dictionary mapping instance IDs (str) to their respective AWSInstance object.

Return type Dict[str, AWInstance]

Raises **RuntimeError** – If instances can't be listed.

ListVolumes (*region=None*, *filters=None*)
List volumes of an AWS account.

Example usage: # List volumes attached to the instance 'some-instance-id' ListVolumes(filters=[

{'Name':'attachment.instance-id', 'Values':['some-instance-id']}])

Parameters

- **region** (*str*) – Optional. The region from which to list the volumes. If none provided, the default_region associated to the AWSAccount object will be used.
- **filters** (*List[Dict]*) – Optional. Filters for the query. Filters are given as a list of dictionaries, e.g.: {'Name': 'someFilter', 'Values': ['value1', 'value2']}.

Returns

Dictionary mapping volume IDs (str) to their respective AWSVolume object.

Return type Dict[str, *AWSVolume*]

Raises **RuntimeError** – If volumes can't be listed.

ResourceApi (*service*, *region=None*)
Create an AWS resource object.

Parameters

- **service** (*str*) – The AWS service to use.

- **region** (*str*) – Optional. The region in which to create new resources. If none provided, the default_region associated to the AWSAccount object will be used.

Returns An AWS EC2 resource object.

Return type boto3.resources.factory.ec2.ServiceResource

ShareKMSKeyWithAWSAccount (*kms_key_id*, *aws_account_id*)

Share a KMS key.

Parameters

- **kms_key_id** (*str*) – The KMS key ID of the key to share.
- **aws_account_id** (*str*) – The AWS Account ID to share the KMS key with.

Raises **RuntimeError** – If the key could not be shared.

Return type None

libcloudforensics.providers.aws.internal.common module

Common utilities.

libcloudforensics.providers.aws.internal.common.**CreateTags** (*resource*, *tags*)

Create AWS Tag Specifications.

Parameters

- **resource** (*str*) – The type of AWS resource.
- **tags** (*Dict[str, str]*) – A dictionary of tags to add to the resource.

Returns A dictionary for AWS Tag Specifications.

Return type Dict[str, Any]

libcloudforensics.providers.aws.internal.common.**ExecuteRequest** (*client*, *func*, *kwargs*)

Execute a request to the boto3 API.

Parameters

- **client** (*boto3.session.Session*) – A boto3 client object.
- **func** (*str*) – A boto3 function to query from the client.
- **kwargs** (*Dict*) – A dictionary of parameters for the function func. Expected keys are strings, values can be of multiple types. E.g.: {'InstanceIds': ['instance_id'], 'MaxResults': 12}.

Returns

A list of dictionaries (responses from the request), e.g. [{'Groups': [{...}], 'Instances': [{...}]}, {...}]

Return type List[Dict]

Raises **RuntimeError** – If the request to the boto3 API could not complete.

libcloudforensics.providers.aws.internal.common.**GetInstanceTypeByCPU** (*cpu_cores*)

Return the instance type for the requested number of CPU cores.

Parameters **cpu_cores** (*int*) – The number of requested cores.

Returns The type of instance that matches the number of cores.

Return type str

Raises **ValueError** – If the requested amount of cores is unavailable.

libcloudforensics.providers.aws.internal.ebs module

Disk functionality.

class libcloudforensics.providers.aws.internal.ebs.**AWSElasticBlockStore**(*aws_account*, *region*, *availability_zone*, *encrypted*, *name=None*)

Bases: object

Class representing an AWS EBS resource.

aws_account
The account for the resource.

Type *AWSAccount*

region
The region the EBS is in.

Type str

availability_zone
The zone within the region in which the EBS is.

Type str

encrypted
True if the EBS resource is encrypted, False otherwise.

Type bool

name
The name tag of the EBS resource, if existing.

Type str

class libcloudforensics.providers.aws.internal.ebs.**AWSSnapshot**(*snapshot_id*, *aws_account*, *region*, *availability_zone*, *volume*, *name=None*)

Bases: *libcloudforensics.providers.aws.internal.ebs.AWSElasticBlockStore*

Class representing an AWS EBS snapshot.

snapshot_id
The id of the snapshot.

Type str

aws_account
The account for the snapshot.

Type *AWSAccount*

region
The region the snapshot is in.

Type str

availability_zone
The zone within the region in which the snapshot is.

Type str

volume
The volume from which the snapshot was taken.

Type *AWSVolume*

name
The name tag of the snapshot, if existing.

Type str

Copy(*kms_key_id=None*, *delete=False*, *deletion_account=None*)
Copy a snapshot.

Parameters

- **kms_key_id** (*str*) – Optional. A KMS key id to encrypt the snapshot copy with. If set to None but the source snapshot is encrypted, then the copy will be encrypted too (with the key used by the source snapshot).
- **delete** (*bool*) – Optional. If set to True, the snapshot being copied will be deleted prior to returning the copy. Default is False.
- **deletion_account** (AWSAccount) – Optional. An AWSAccount object to use to delete the snapshot if 'delete' is set to True. Since accounts operate per region, this can be useful when copying snapshots across regions (which requires one AWSAccount object per region as per boto3.session.Session() requirements) and wanting to delete the source snapshot located in a different region than the copy being created.

Returns A copy of the snapshot.

Return type *AWSSnapshot*

Raises **RuntimeError** – If the snapshot could not be copied.

Delete()
Delete a snapshot.

Return type None

ShareWithAWSAccount(*aws_account_id*)
Share the snapshot with another AWS account ID.

Parameters **aws_account_id** (*str*) – The AWS Account ID to share the snapshot with.

Return type None

class libcloudforensics.providers.aws.internal.ebs.**AWSVolume**(*volume_id*, *aws_account*, *region*, *availability_zone*, *encrypted*, *name=None*, *device_name=None*)

Bases: *libcloudforensics.providers.aws.internal.ebs.AWSElasticBlockStore*

Class representing an AWS EBS volume.

volume_id
The id of the volume.

Type str

aws_account
The account for the volume.

Type *AWSAccount*

region
The region the volume is in.

Type str

availability_zone
The zone within the region in which the volume is.

Type str

encrypted
True if the volume is encrypted, False otherwise.

Type bool

name
The name tag of the volume, if existing.

Type str

device_name
The device name (e.g. /dev/spf) of the volume when it is attached to an instance, if applicable.

Type str

Delete()
Delete a volume.

Return type None

Snapshot(*tags=None*)
Create a snapshot of the volume.

Parameters **tags** (*Dict[str, str]*) – Optional. A dictionary of tags to add to the snapshot, for example {'Name': 'my-snapshot-name', 'TicketID': 'xxx'}.

Returns A snapshot object.

Return type *AWSSnapshot*

Raises

- **ValueError** – If the snapshot name does not comply with the RegEx.
- **RuntimeError** – If the snapshot could not be created.

libcloudforensics.providers.aws.internal.ec2 module

Instance functionality.

class libcloudforensics.providers.aws.internal.ec2.**AWSInstance**(*aws_account*, *instance_id*, *region*, *availability_zone*, *name=None*)

Bases: object

Class representing an AWS EC2 instance.

aws_account
The account for the instance.

Type *AWSAccount*

instance_id
The id of the instance.

Type str

region
The region the instance is in.

Type str

availability_zone
The zone within the region in which the instance is.

Type str

name
The name tag of the instance, if existing.

Type str

AttachVolume(*volume*, *device_name*)
Attach a volume to the AWS instance.

Parameters

- **volume** (AWSVolume) – The AWSVolume object to attach to the instance.
- **device_name** (*str*) – The device name for the volume (e.g. /dev/sdf).

Raises **RuntimeError** – If the volume could not be attached.

Return type None

GetBootVolume()
Get the instance's boot volume.

Returns Volume object if the volume is found.

Return type *AWSVolume*

Raises **RuntimeError** – If no boot volume could be found.

GetVolume(*volume_id*)
Get a volume attached to the instance by ID.

Parameters **volume_id** (*str*) – The ID of the volume to get.

Returns The AWSVolume object.

Return type *AWSVolume*

Raises **RuntimeError** – If volume_id is not found amongst the volumes attached to the instance.

ListVolumes()

List all volumes for the instance.

Returns

Dictionary mapping volume IDs to their respective AWSVolume object.

Return type Dict[str, *AWSVolume*]

libcloudforensics.providers.aws.internal.log module

Log functionality.

class libcloudforensics.providers.aws.internal.log.**AWSCloudTrail**(*aws_account*)

Bases: object

Class representing an AWS CloudTrail service.

aws_account

The AWS account to use.

Type *AWSAccount*

LookupEvents(*qfilter=None*, *starttime=None*, *endtime=None*)

Lookup events in the CloudTrail logs of this account.

Example usage: # pylint: disable=line-too-long # qfilter = 'key,value' # starttime = datetime(2020,5,5,17,33,00) # LookupEvents(qfilter=qfilter, starttime=starttime) # Check documentation for qfilter details # https://boto3.amazonaws.com/v1/documentation/api/latest/reference/services/cloudtrail.html#CloudTrail.Client.lookup_events

Parameters

- **qfilter** (*string*) – Optional. Filter for the query including 1 key and value.
- **starttime** (*datetime*) – Optional. Start datetime to add to query filter.
- **endtime** (*datetime*) – Optional. End datetime to add to query filter.

Returns

A list of events. E.g. [{'EventId': 'id', ...}, {'EventId': ...}]

Return type List[Dict]

1.2.2 libcloudforensics.providers.aws.forensics module

Forensics on AWS.

libcloudforensics.providers.aws.forensics.**CreateVolumeCopy**(*zone*, *dst_zone=None*, *instance_id=None*, *volume_id=None*, *src_profile=None*, *dst_profile=None*, *tags=None*)

Create a copy of an AWS EBS Volume.

By default, the volume copy will be created in the same AWS account where the source volume sits. If you want the volume copy to be created in a different AWS account, you can specify one in the dst_profile parameter. The following example illustrates how you should configure your AWS credentials file for such a use case.

AWS credentials file [default] # default account to use with AWS aws_access_key_id=foo aws_secret_access_key=bar

[investigation] # source account for a particular volume to be copied from aws_access_key_id=foo1 aws_secret_access_key=bar1

[forensics] # destination account to create the volume copy in aws_access_key_id=foo2 aws_secret_access_key=bar2

Copies the boot volume from instance "instance_id" from the default AWS # account to the default AWS account. volume_copy = CreateDiskCopy(zone, instance_id='instance_id')

Copies the boot volume from instance "instance_id" from the default AWS # account to the 'forensics' AWS account. volume_copy = CreateDiskCopy(

zone, instance_id='instance_id', dst_profile='forensics')

Copies the boot volume from instance "instance_id" from the # 'investigation' AWS account to the 'forensics' AWS account. volume_copy = CreateDiskCopy(

zone, instance_id='instance_id', src_profile='investigation', dst_profile='forensics')

Parameters

- **zone** (*str*) – The AWS zone in which the volume is located, e.g. 'us-east-2b'.
- **dst_zone** (*str*) – Optional. The AWS zone in which to create the volume copy. By default, this is the same as 'zone'.
- **instance_id** (*str*) – Optional. Instance ID of the instance using the volume to be copied. If specified, the boot volume of the instance will be copied. If volume_id is also specified, then the volume pointed by that volume_id will be copied.
- **volume_id** (*str*) – Optional. ID of the volume to copy. If not set, then instance_id needs to be set and the boot volume will be copied.
- **src_profile** (*str*) – Optional. If the AWS account containing the volume that needs to be copied is different from the default account specified in the AWS credentials file then you can specify a different profile name here (see example above).
- **dst_profile** (*str*) – Optional. If the volume copy needs to be created in a different AWS account, you can specify a different profile name here (see example above).
- **tags** (*Dict[str, str]*) – Optional. A dictionary of tags to add to the volume copy, for example {'TicketID': 'xxx'}.

Returns An AWS EBS Volume object.

Return type *AWSVolume*

Raises

- **RuntimeError** – If there are errors copying the volume, or errors during KMS key creation/sharing if the target volume is encrypted.
- **ValueError** – If both instance_id and volume_id are missing.

`libcloudforensics.providers.aws.forensics.StartAnalysisVm`(*vm_name*, *default_availability_zone*, *boot_volume_size*, *ami=''*, *cpu_cores=4*, *attach_volumes=None*, *dst_profile=None*, *ssh_key_name=None*, *tags=None*)

Start a virtual machine for analysis purposes.

Look for an existing AWS instance with tag name vm_name. If found, this instance will be started and used as analysis VM. If not found, then a new vm with that name will be created, started and returned.

Parameters

- **vm_name** (*str*) – The name for the virtual machine.
- **default_availability_zone** (*str*) – Default zone within the region to create new resources in.
- **boot_volume_size** (*int*) – The size of the analysis VM boot volume (in GB).
- **ami** (*str*) – Optional. The Amazon Machine Image ID to use to create the VM. Default is a version of Ubuntu 18.04.
- **cpu_cores** (*int*) – Optional. The number of CPU cores to create the machine with. Default is 4.
- **attach_volumes** (*List[Tuple[str, str]]*) – Optional. List of tuples containing the volume IDs (str) to attach and their respective device name (str, e.g. /dev/sdf). Note that it is mandatory to provide a unique device name per volume to attach.
- **dst_profile** (*str*) – Optional. The AWS account in which to create the analysis VM. This is the profile name that is defined in your AWS credentials file.
- **ssh_key_name** (*str*) – Optional. A SSH key pair name linked to the AWS account to associate with the VM. If none provided, the VM can only be accessed through in-browser SSH from the AWS management console with the EC2 client connection package (ec2-instance-connect). Note that if this package fails to install on the target VM, then the VM will not be accessible. It is therefore recommended to fill in this parameter.
- **tags** (*Dict[str, str]*) – Optional. A dictionary of tags to add to the instance, for example {'TicketID': 'xxx'}. An entry for the instance name is added by default.

Returns

a tuple with a virtual machine object and a boolean indicating if the virtual machine was created or not.

Return type Tuple[*AWSInstance*, bool]

Raises **RuntimeError** – When multiple AMI images are returned.

PYTHON MODULE INDEX

l

INDEX

D

E

F

G

I

L

M

N

P

R

S

V

Z