Awspice Documentation

David Amrani Hernandez

Jan 21, 2020

Contents:

CHAPTER 1

Getting started

1.1 Requirements

Awspice is an abstraction layer of AWS, so it will be necessary to meet the following requirements:

We just need . . .	And it means. . .
AWS account	Have an Amazon Web Services account of any kind
IAM user	Enabled user with programmatic keys (access and secret key)
Permissions	Have permissions in the services and regions to use

1.2 Installation

```
pip install awspice
```

1.3 Configuration

The client is built and configured using `awspice.connect()`. This method indicates the type of authentication and region on which you are going to work. There are two ways to set your credentials *(Only one of the two can be used)*: * **Profile** *(Recommended)* – The access keys are stored in `~/.aws/credentials` file. (Read more) * **Access keys** – Typing the hard-coded access keys.

Parameter name	Default value	Description
region	eu-west-1	Region on which you are going to work.
profile	default	Name of the profile in ~/.aws/credentials file
access_key		User API access key
secret_key		User API secret key

```
import awspice

aws = awspice.connect() # Region: eu-west-1 | Profile: Default

aws = awspice.connect(region='us-west-2', profile='dev_profile')
aws = awspice.connect('us-west-2', access_key='AKIA**********', secret_key='/HR
↪$4**********')
```

1.4 Test it

To verify that the configuration has been correctly stored, you can run the following test. This test only checks that your user is registered and enabled on the AWS account set in the client's configuration.

```
import awspice

aws = awspice.connect(profile='<YOUR_PROFILE>')
aws.test()
```

1.5 Using boto3 client

If you want to use the native Boto3 client to perform some operation, you can also do it using the "client" attribute within each service. If you call the client through the class *ec2*, this will be the service on which the client will be configured. The region and authentication will be the same as the last call made.

```
import awspice

aws = awspice.connect(region='us-east-1', profile='sample')
aws.service.ec2.client.describe_instance_status(InstanceIds=['i-12345'])
```

CHAPTER 2

Services

2.1 Acm

awspice.services.acm.AcmService.list_certificates([...])	List all certificates
awspice.services.acm.AcmService.get_certificate_by(...)	Get certificate filtering by domain
awspice.services.acm.AcmService.get_certificate(arn)	Get certificate using CertificateArn (Ceritificate Identifier)

2.2 CostExplorer

awspice.services.ce.CostExplorerService.get_cost([...])	Get the cost of account or its elements.

2.3 Ec2

awspice.services.ec2.Ec2Service.set_tag(...)	Set tag for an instance
awspice.services.ec2.Ec2Service.get_amis([...])	Get all images
awspice.services.ec2.Ec2Service.get_ami_by(filters)	Get an ami for one or more regions that matches with filter
awspice.services.ec2.Ec2Service.get_amis_by(filters)	Get list of amis for one or more regions that matches with filter
awspice.services.ec2.Ec2Service.get_amis_by_distribution(distrib)	Get one or more Images filtering by distribution

Continued on next page

Table 3 – continued from previous page

awspice.services.ec2.Ec2Service.get_instances([...])	Get all instances for one or more regions.
awspice.services.ec2.Ec2Service.get_instance_by(filters)	Get an instance for one or more regions that matches with filter
awspice.services.ec2.Ec2Service.get_instances_by(filters)	Get an instance for one or more regions that matches with filter
awspice.services.ec2.Ec2Service.get_instances_status([...])	
awspice.services.ec2.Ec2Service.get_instance_status_by(filters)	
awspice.services.ec2.Ec2Service.get_instances_status_by(filters)	
awspice.services.ec2.Ec2Service.create_instances(...)	Create a new instance
awspice.services.ec2.Ec2Service.start_instances(...)	Stops an Amazon EC2 instance
awspice.services.ec2.Ec2Service.stop_instances(...)	Stops an Amazon EC2 instance
awspice.services.ec2.Ec2Service.get_volumes([...])	Get all volumes for one or more regions
awspice.services.ec2.Ec2Service.get_volume_by(filters)	Get a volume for one or more regions that matches with filters
awspice.services.ec2.Ec2Service.get_volumes_by(filters)	Get volumes for one or more regions that matches with filters
awspice.services.ec2.Ec2Service.get_snapshots()	Get all snapshots owned by self for the current region
awspice.services.ec2.Ec2Service.get_snapshot_by(filters)	Get a snapshot for a region tha matches with filters
awspice.services.ec2.Ec2Service.get_snapshots_by(filters)	Get all snapshots for the current region that matches with filters
awspice.services.ec2.Ec2Service.get_secgroups([...])	Get all security groups for the current region
awspice.services.ec2.Ec2Service.get_secgroup_by(filters)	Get security group for a region that matches with filters
awspice.services.ec2.Ec2Service.get_secgroups_by(filters)	Get all security groups for a region that matches with filters
awspice.services.ec2.Ec2Service.create_security_group(...)	Create a new Security Group
awspice.services.ec2.Ec2Service.delete_security_group(...)	Delete an existing Security Group
awspice.services.ec2.Ec2Service.get_addresses([...])	Get all IP Addresses for a region
awspice.services.ec2.Ec2Service.get_address_by(filters)	Get IP Addresses for a region that matches with filters
awspice.services.ec2.Ec2Service.get_vpcs([...])	Get all VPCs for a region
awspice.services.ec2.Ec2Service.get_default_vpc()	Get default Security Group

2.4 Elb

awspice.services.elb.ElbService.get_loadbalancers([...])	Get all Elastic Load Balancers for a region
awspice.services.elb.ElbService.get_loadbalancers_by(...)	Get loadbalancers which match with the filters
awspice.services.elb.ElbService.get_loadbalancer_by(...)	Get a load balancer for a region that matches with filter

2.5 Iam

awspice.services.iam.IamService.get_inactive_users()	Get users who have not logged in AWS since 1 year.
awspice.services.iam.IamService.get_users()	List all users for an AWS account
awspice.services.iam.IamService.get_access_keys(user)	
awspice.services.iam.IamService.get_access_key_last_used(...)	

2.6 Rds

awspice.services.rds.RdsService.get_database_by(filters)	
awspice.services.rds.RdsService.get_databases([...])	Get RDS instances in regions
awspice.services.rds.RdsService.get_snapshots([...])	Get RDS snapshots in regions

2.7 Route53

awspice.services.route53.Route53Service.list_hosted_zones()	List all hosted zones
awspice.services.route53.Route53Service.list_records(...)	List all records for a hosted zone
awspice.services.route53.Route53Service.list_records_by_domain(domain)	List all records of a hosted-zone domain

2.8 S3

awspice.services.s3.S3Service.upload_string_as_file(...)	Upload string as a file to S3 bucket

Continued on next page

Table 8 – continued from previous page

awspice.services.s3.S3Service.get_buckets()	Get all buckets in S3
awspice.services.s3.S3Service.get_bucket_acl(...)	
awspice.services.s3.S3Service.get_public_buckets()	Get all public buckets and its permissions
awspice.services.s3.S3Service.list_bucket_objects(bucket)	List objects stored in a bucket

CHAPTER 3

Modules

3.1 Finder

awspice.modules.finder.FinderModule.find_instance(filters)	Get an instance in different accounts and regions, using search filters.
awspice.modules.finder.FinderModule.find_instances([...])	Get instances in different accounts and regions, using search filters.
awspice.modules.finder.FinderModule.find_volume(filters)	Get a volume in different accounts and regions, using search filters.
awspice.modules.finder.FinderModule.find_volumes([...])	Get group of volumes in different accounts and regions, using search filters.
awspice.modules.finder.FinderModule.find_loadbalancer(filters)	Get a load balancer in different accounts and regions, using search filters.
awspice.modules.finder.FinderModule.find_loadbalancers([...])	Get load balancers in different accounts and regions, using search filters.
awspice.modules.finder.FinderModule.find_users([...])	Get IAM users in different accounts.
awspice.modules.finder.FinderModule.find_inactive_users([...])	Get inactive users in different accounts
awspice.modules.finder.FinderModule.find_buckets([...])	Search S3 buckets in different accounts.
awspice.modules.finder.FinderModule.find_rds_databases([...])	Get RDS databases in different accounts and regions.
awspice.modules.finder.FinderModule.find_rds_snapshots([...])	Get RDS snapshots in different accounts and regions.

3.2 Security

awspice.modules.security.SecurityModule.get_instance_portlisting(...)	List SecurityGroups and rules for an instance
awspice.modules.security.SecurityModule.get_region_portlisting(...)	List SecurityGroups and rules for all instances in region

3.3 Stats

awspice.modules.stats.StatsModule.get_stats([...])	Retrieve data about services in your AWS account like Volumes, Instances or Databases.
awspice.modules.stats.StatsModule.cost_saving([...])	List unused elements that carry expenses.

CHAPTER 4

FAQs & Troubleshooting

4.1 Frequently Asked Questions

4.1.1 Running Tests

At the moment this functionality is not available as they have not been mocked.

```
$ pip install -r requirements.txt
$ python -m unittest -v test
```

4.1.2 Generating Documentation

Sphinx is used for documentation. You can generate HTML locally with the following:

```
$ pip install -r requirements_dev.txt
$ cd docs
$ make html
```

4.2 Troubleshooting

4.2.1 TypeError: datetime is not JSON serializable

Sometimes Boto3 returns a non-serializable result to JSON and we get the following error when dumping that result: `TypeError: datetime.datetime (2015, 12, 3, 21, 20, 17, 326000, tzinfo = tzutc ()) is not JSON serializable`

You can solve it using this encoder in the following way:

```
import awspice

json.dumps(json, indent=4, cls=awspice.ClsEncoder)
```

CHAPTER 5

awspice

5.1 awspice package

5.1.1 Subpackages

awspice.modules package

Submodules

awspice.modules.finder module

class awspice.modules.finder.**FinderModule**(*aws*)
Bases: object

This class makes it easy to search for components in AWS.

aws
awspice client

find_instance(*filters*, *profiles=[]*, *regions=[]*)
Get an instance in different accounts and regions, using search filters.

find_instances(*filters=None*, *profiles=[]*, *regions=[]*)
Get instances in different accounts and regions, using search filters.

find_volume(*filters*, *profiles=[]*, *regions=[]*)
Get a volume in different accounts and regions, using search filters.

find_volumes(*filters=None*, *profiles=[]*, *regions=[]*)
Get group of volumes in different accounts and regions, using search filters.

find_loadbalancer(*filters*, *profiles=[]*, *regions=[]*)
Get a load balancer in different accounts and regions, using search filters.

find_loadbalancers (*filter_key=None*, *filter_value=None*, *profiles=[]*, *regions=[]*)
Get load balancers in different accounts and regions, using search filters.

find_users (*profiles=[]*)
Get IAM users in different accounts.

find_inactive_users (*profiles=[]*)
Get inactive users in different accounts

find_buckets (*profiles=[]*)
Search S3 buckets in different accounts.

find_rds_databases (*profiles=[]*, *regions=[]*)
Get RDS databases in different accounts and regions.

find_rds_snapshots (*profiles=[]*, *regions=[]*)
Get RDS snapshots in different accounts and regions.

__init__ (*aws*)
Initialize self. See help(type(self)) for accurate signature.

awspice.modules.security module

class awspice.modules.security.**SecurityModule**
Bases: object

This class facilitates methods for securing the AWS account

Methods are available to help improve AWS account security by detecting bad configurations.

classmethod get_instance_portlisting (*aws*, *instanceid*)
List SecurityGroups and rules for an instance

Parameters

- **aws** – AwsManager client
- **instanceid** – Id of instance to analyze

Returns Dictionary with instance and its SecurityGroups

classmethod get_region_portlisting (*aws*, *region*)
List SecurityGroups and rules for all instances in region

Parameters

- **aws** – AwsManager client
- **region** – Region to analyze

Returns Dictionary with regions, instances and its SecurityGroups

awspice.modules.stats module

class awspice.modules.stats.**StatsModule** (*aws*)
Bases: object

Class responsible for processing general data to the AWS account.

This class is dedicated to the global management of the AWS account in order to obtain statistics, costs or global information.

aws
awspice client

get_stats(*regions=[]*)
Retrieve data about services in your AWS account like Volumes, Instances or Databases.

Parameters

- **aws** – AwsManager client
- **region** – To retrieve data only of this region

Returns List of regions with its stats

cost_saving(*regions=[]*)
List unused elements that carry expenses.

Parameters **aws** – AwsManager client.

Returns Dict Region with a list of regions with its unused elements

__init__(*aws*)
Initialize self. See help(type(self)) for accurate signature.

Module contents

awspice.services package

Submodules

awspice.services.acm module

class awspice.services.acm.**AcmService**
Bases: *awspice.services.base.AwsBase*

Class belonging to the ACM certificate management service.

list_certificates(*regions=[]*)
List all certificates

Parameters **regions** (*lst*) – List of regions to list certificates

Returns List of certificates

get_certificate_by(*filter_key*, *filter_value*, *regions=[]*)
Get certificate filtering by domain

Parameters

- **filter_key** (*str*) – Name of the field to be searched. (Domain)
- **filter_value** (*str*) – Value for the previous field. (i.e.: google.es)
- **regions** (*lst*) – List of regions where the certificate can be.

Returns Certificate matched to the filter entered.

get_certificate(*arn*, *regions=[]*)
Get certificate using CertificateArn (Ceritificate Identifier)

Parameters

- **arn** (*str*) – ARN of the certificate

- **regions** (*lst*) – List of regions where the certificate can be.

Returns Certificate matched to the ARN entered.

__init__()
This constructor configures the corresponding service according to the class that calls it.

Every time the EC2Service Class is called (inherits from this class), this constructor will change the client's service to 'ec2'. And then, if ELBService service is called, this method is called again changing the service from 'ec2' to 'elb'.

Parameters **service** (*str*) – AWS service to uso

Returns None

awspice.services.base module

class awspice.services.base.**AwsBase**(*service*)
Bases: object

Base class from which all services inherit (ec2, s3, vpc . . .)

This class contains methods and properties that are common to all AWS services and should be accessible by all of them. This class is responsible for instantiating the client and processing information related to the accounts and regions.

client
Boto3 client

region
Current region used by the client

profile
Current profile used by the client

access_key
Current access key used by the client

secret_key
Current secret key used by the client

endpoints = None

region = None

profile = None

access_key = None

secret_key = None

pool = <awspice.helpers.ThreadPool object>

service_resources = ['ec2', 's3']

set_client(*service*)
Main method to set Boto3 client

Parameters

- **service** (*str*) – Service to use (i.e.: ec2, s3, vpc. . .)
- **region** (*str*) – Region name to use (i.e.: eu-central-1)
- **profile** (*str*) – Profile name set in ~/.aws/credentials file

- **access_key** (*str*) – API access key of your AWS account
- **secret_key** (*str*) – API secret key of your AWS account

Raises

- **ClientError** – Access keys are not valid or lack of permissions for a service/region
- **ProfileNotFound** – Profile name not found in credentials file

Returns None

classmethod set_auth_config(*region*, *profile=None*, *access_key=None*, *secret_key=None*)
Set properties like service, region or auth method to be used by boto3 client

Parameters

- **service** (*str*) – Service to use (i.e.: ec2, s3, vpc...)
- **region** (*str*) – Region name (i.e.: eu-central-1)
- **access_key** (*str*) – API Access key
- **secret_key** (*str*) – API Secret key
- **profile** (*str*) – Profile name set in ~/.aws/credentials file

classmethod get_client_vars()
Get information of the current client configuration Sometimes we need to store this variables, for example using threads, because AwsBase is constantly changing

Returns Array with current client configuration ({'region': 'eu-west-1', 'profile': 'default'})

Return type dict

classmethod inject_client_vars(*elements*, *client_conf=None*)
Insert in each item of a list, the region and the current credentials.

This function is called by all the methods of all the services that return a list of objects to identify in what region and account they have been found.

Parameters

- **elements** (*list*) – List of dictionaries
- **client_conf** (*dict*) – Array with the client configuration (see *get_client_vars*)

Returns list. Returns same list with the updated elements (region and authentication included)

region_in_regions(*region*, *regions*)
Check if region is in a complex list of regions

Parameters

- **region** (*str | lst*) – 'eu-west-1'}
- **regions** (*lst*) –

Examples

region_in_regions('eu-west-1', [{'RegionName': 'eu-west-1}])

Returns bool

classmethod validate_filters(*input_filters*, *accepted_filters*)
Transform filters into AWS filters format after validate them.

Parameters

- **input_filters** (*str*) – Items to validate
- **accepted_filters** (*list*) – Pre-validated list

Returns None

Raises **ValueError** – Filter is not in the accepted filter list

classmethod get_profiles()
Get a list of all available profiles in ~/.aws/credentials file

Returns list. List of strings with available profiles

change_profile(*profile*)
Change profile of the client

This method changes the account/profile used but keeps the same region and service

Parameters **profile** (*str*) – Name of the profile set in ~/.aws/credentials file

Examples

$ aws = awspice.connect() $ aws.service.ec2.change_profile('my_boring_company')

Returns None

parse_profiles(*profiles=[]*)
Validation method which get a profile or profile list and return the expected list of them

The purpose of this method is that a user can pass different types of data as a "profile" argument and obtain a valid output for any method that works with this type of data.

Parameters **profiles** (*list | str*) – String or list of string to parse

Examples

$ account_str = aws.service.ec2.parse_profiles('my_company') $ account_lst = aws.service.ec2.parse_profiles(['my_company']) $ accounts_lst = aws.service.ec2.parse_profiles(['my_company', 'other_company'])

Returns list. List of a strings with profile names

get_endpoints()
Get services and its regions and endpoints

Returns Dict with services (key) and its regions and Endpoints.

Return type dict

get_regions()
Get all available regions

Returns list. List of regions with 'Country' and 'RegionName'

change_region(*region*)
Change region of the client

This method changes the region used but keeps the same service and profile

Parameters **region** (*str*) – Region Name (ID) of AWS (i.e.: eu-central-1)

Examples

aws.service.ec2.change_region('eu-west-1')

Returns None

parse_regions (*regions=[]*, *default_all=False*)
Validation method which get a region or list of regions and return the expected list of them

The purpose of this method is that a user can pass different types of data as a "region" argument and obtain a valid output for any method that works with this type of data.

Parameters

- **regions** (*list | str*) – String or list of string to parse
- **default_all** (*bool*) – If the list of regions is empty and this argument is True, a list with all regions will be returned. This is useful when you do not know the data entry of type "region" and you want to search by default in all regions (if regions are empty means that the user does not know where an element is located).

Examples

AwsBase.region = aws.service.ec2.parse_regions([]) regions = aws.service.ec2.parse_regions('eu-west-1') regions = aws.service.ec2.parse_regions(['eu-west-1']) regions = aws.service.ec2.parse_regions(['eu-west-1', 'eu-west-2'])

Returns list. List of a strings with profile names

__init__ (*service*)
This constructor configures the corresponding service according to the class that calls it.

Every time the EC2Service Class is called (inherits from this class), this constructor will change the client's service to 'ec2'. And then, if ELBService service is called, this method is called again changing the service from 'ec2' to 'elb'.

Parameters **service** (*str*) – AWS service to uso

Returns None

awspice.services.ce module

class awspice.services.ce.**CostExplorerService**
Bases: *awspice.services.base.AwsBase*

Class belonging to the Cost Explorer service.

granularities = ['DAILY', 'MONTHLY']

filter_dimensions = ['AZ', 'INSTANCE_TYPE', 'LINKED_ACCOUNT', 'OPERATION', 'PURCHASE_T

group_dimensions = ['AZ', 'INSTANCE_TYPE', 'LEGAL_ENTITY_NAME', 'LINKED_ACCOUNT', 'OPE

get_cost (*from_date=None*, *to_date=None*, *interval='Monthly'*, *group_by=''*, *group_by_tag_value=''*, *filter_by={}*, *ec2_running_hours=False*)
Get the cost of account or its elements.

This method obtains costs of an account/s , one or several elements (substances, balancers, addresses) between two dates and granularized in days or months. If the date is not indicated, the cost of the last month will be returned.

Parameters

- **from_date** (*str*) – Date from which you want to obtain data. (Format: 2018-04-24)
- **to_date** (*str*) – Date until which you want to obtain data. (Format: 2018-04-24)
- **interval** (*str*) – Time interval to be analyzed. [MONTHLY | DAILY]
- **group_by** (*str*) – Group results by ['AZ', 'INSTANCE_TYPE', 'LEGAL_ENTITY_NAME', 'LINKED_ACCOUNT', 'OPERATION', 'PLATFORM', 'PURCHASE_TYPE', 'SERVICE', 'TAG', 'TENANCY', 'USAGE_TYPE']
- **group_by_tag_value** (*str*) – TAG key in case group_by set to 'TAG' (i.e. Name, Project or Environment)
- **filter_by** (*dict*) – Key of the filter and value. {'TAG_NAME': ['ec2-tagname', 'LINKED_ACCOUNT: ['1234']]}

Examples

get_cost(['machine-1', 'machine-2'], '2018-12-24', '2018-12-26', interval='daily') get_cost() # Get account cost

Returns List of days or months with the requested costs

Return type Costs (list)

__init__()

This constructor configures the corresponding service according to the class that calls it.

Every time the EC2Service Class is called (inherits from this class), this constructor will change the client's service to 'ec2'. And then, if ELBService service is called, this method is called again changing the service from 'ec2' to 'elb'.

Parameters **service** (*str*) – AWS service to uso

Returns None

awspice.services.ec2 module

class awspice.services.ec2.**Ec2Service**

Bases: *awspice.services.base.AwsBase*

Class belonging to the EC2 Computing service.

set_tag(*resource_id*, *tag_key*, *tag_value*, *regions=[]*)

Set tag for an instance

Parameters

- **elements_id** (*str*) – Id of resources to tag. (i.e: i-01234, vol-01234)
- **tag_key** (*str*) – Name of the element TAG (i.e: Name)
- **tag_value** (*str*) – Value of that Tag
- **regions** (*lst*) – Regions where to look for this element

Returns None

__init__()
This constructor configures the corresponding service according to the class that calls it.

Every time the EC2Service Class is called (inherits from this class), this constructor will change the client's service to 'ec2'. And then, if ELBService service is called, this method is called again changing the service from 'ec2' to 'elb'.

Parameters **service** (*str*) – AWS service to uso

Returns None

address_filters = {'domain': 'domain', 'instance': 'instance-id', 'privateip': 'pri

ami_distributions = {'amazon': 'amzn-ami-hvm-20*.*.*-x86_64-*', 'ubuntu': 'ubuntu/ima

ami_filters = {'architecture': 'architecture', 'id': 'image-id', 'name': 'name', 'o

create_instances(*name*, *key_name*, *allowed_range*, *ami=None*, *distribution=None*, *version=None*, *instance_type='t2.micro'*, *region=None*, *vpc=None*, *count=1*)
Create a new instance

Parameters

- **name** (*str*) – TagName of the instance
- **key_name** (*str*) – The name of the key pair (i.e: it_user)
- **allowed_range** (*str*) – Network range with access to instance (i.e: 10.0.0.0/32)
- **ami** (*str*) – Id of the ami (i.e: ami-12345)
- **instance_type** (*str*) – Type of hardware of the instance (i.e: t2.medium)
- **distribution** (*str*) – Instead of ami, select an OS: (i.e: ubuntu)
- **region** (*str*) – Name of the region where instance will be displayed
- **vpc** (*str*) – VPC identifier where the instance will be deployed.
- **count** (*int*) – Number of instances to launch

Returns List of launched instances

Return type Instances (lst)

create_security_group(*name*, *allowed_range*, *vpc_id=None*)
Create a new Security Group

Parameters

- **name** (*str*) – Name of the Security Group
- **allowed_range** (*str*) – Network range with permissions (i.e: 10.0.0.0/32)
- **vpc_id** (*str*) – Id of assigned VPC

Returns Identifier of the security group created.

Return type str

delete_security_group(*identifier*)
Delete an existing Security Group

Parameters **identifier** (*str*) – Id of the Security Group

Returns none

distrib_amis = {'redhat': 'ami-c86c3f23', 'ubuntu': 'ami-f90a4880', 'windows': 'ami

get_address_by (*filters*, *regions=[]*)
Get IP Addresses for a region that matches with filters

Parameters **regions** (*lst*) – Regions where to look for this element

Returns Dictionary with the address requested

Return type Address (dict)

get_addresses (*regions=[]*)
Get all IP Addresses for a region

Parameters **regions** (*lst*) – Regions where to look for this element

Returns List of dictionaries with the addresses requested

Return type Addresses (dict)

get_addresses_by (*filters*, *regions=[]*)
Get all IP Addresses for a region

Parameters **regions** (*lst*) – Regions where to look for this element

Returns List of dictionaries with the addresses requested

Return type Addresses (dict)

get_ami_by (*filters*, *regions=[]*)
Get an ami for one or more regions that matches with filter

Parameters

- **filter_key** (*str*) – Name of the filter
- **filter_value** (*str*) – Value of the filter
- **regions** (*lst*) – Regions where to look for this element

Returns Image requested

Return type Image (dict)

get_amis (*regions=[]*)
Get all images

Parameters **regions** (*lst*) – Regions where to look for this element

Returns List of all images

Return type Images (lst)

get_amis_by (*filters*, *regions=[]*, *return_first=False*)
Get list of amis for one or more regions that matches with filter

Parameters

- **filter_key** (*str*) – Name of the filter
- **filter_value** (*str*) – Value of the filter
- **regions** (*lst*) – Regions where to look for this element
- **return_first** (*bool*) – True if return first result

Returns List of requested images

Return type Images (lst)

get_amis_by_distribution(*distrib*, *version='*'*, *latest=False*, *regions=[]*)
Get one or more Images filtering by distribution

Parameters

- **distrib** (*str*) – Distribution of the image (i.e.: ubuntu)
- **version** (*str*) – Version of the system
- **latest** (*bool*) – True if only returns the newest item.
- **regions** (*lst*) – Regions where to look for this element

Returns List with the images requested.

Return type Image (lst)

get_default_vpc()
Get default Security Group

Returns Default security group resource

Return type SecurityGroup (dict)

get_instance_by(*filters*, *regions=[]*)
Get an instance for one or more regions that matches with filter

Parameters

- **filter_key** (*str*) – Name of the filter
- **filter_value** (*str*) – Value of the filter
- **regions** (*lst*) – Regions where to look for this element

Returns Dictionary with the instance requested

Return type Instance (dict)

get_instance_status_by(*filters*, *regions=[]*)

get_instances(*regions=[]*)
Get all instances for one or more regions.

Parameters **regions** (*lst*) – Regions where to look for this element

Returns List of dictionaries with the instances requested

Return type Instances (lst)

get_instances_by(*filters*, *regions=[]*, *return_first=False*)
Get an instance for one or more regions that matches with filter

Parameters

- **filter_key** (*str*) – Name of the filter
- **filter_value** (*str*) – Value of the filter
- **regions** (*lst*) – Regions where to look for this element
- **return_first** (*bool*) – Select to return the first match

Returns List of dictionaries with the instances requested

Return type Instances (lst)

get_instances_status(*regions=[]*)

get_instances_status_by(*filters*, *regions=[]*, *return_first=False*)

get_secgroup_by (*filters*, *regions=[]*)
Get security group for a region that matches with filters

Parameters

- **filter_key** (*str*) – Name of the filter
- **filter_value** (*str*) – Value of the filter

Returns Dictionaries with the security group requested

Return type SecurityGroup (dict)

get_secgroups (*regions=[]*)
Get all security groups for the current region

Returns List of dictionaries with the security groups requested

Return type SecurityGroups (lst)

get_secgroups_by (*filters*, *regions=[]*)
Get all security groups for a region that matches with filters

Parameters

- **filter_key** (*str*) – Name of the filter
- **filter_value** (*str*) – Value of the filter

Returns List of dictionaries with the security groups requested

Return type SecurityGroups (lst)

get_snapshot_by (*filters*)
Get a snapshot for a region tha matches with filters

Parameters

- **filter_key** (*str*) – Name of the filter
- **filter_value** (*str*) – Value of the filter

Returns Dictionary with the snapshot requested

Return type Snapshot (dict)

get_snapshots ()
Get all snapshots owned by self for the current region

Returns List of dictionaries with the snapshots requested

Return type Snapshots (lst)

get_snapshots_by (*filters*)
Get all snapshots for the current region that matches with filters

Parameters

- **filter_key** (*str*) – Name of the filter
- **filter_value** (*str*) – Value of the filter

Returns List of dictionaries with the snapshots requested

Return type Snapshots (lst)

get_volume_by (*filters*, *regions=[]*)
Get a volume for one or more regions that matches with filters

Parameters

- **filter_key** (*str*) – Name of the filter
- **filter_value** (*str*) – Value of the filter
- **regions** (*lst*) – Regions where to look for this element

Returns Dictionary with the volume requested

Return type Volume (dict)

get_volumes(*regions=[]*)
Get all volumes for one or more regions

Parameters **regions** (*lst*) – Regions where to look for this element

Returns List of dictionaries with the volumes requested

Return type Volumes (lst)

get_volumes_by(*filters*, *regions=[]*, *return_first=False*)
Get volumes for one or more regions that matches with filters

Parameters

- **filter_key** (*str*) – Name of the filter
- **filter_value** (*str*) – Value of the filter
- **regions** (*lst*) – Regions where to look for this element

Returns Dictionary with the volume requested

Return type Volume (dict)

get_vpcs(*regions=[]*)
Get all VPCs for a region

Returns List of dictionaries with the vpcs requested

Return type VPCs (lst)

instance_filters = {'dnsname': 'dns-name', 'id': 'instance-id', 'name': 'tag:Name',

instance_status_filters = {'event': 'event.code', 'instance-check': 'instance-status

secgroup_filters = {'description': 'description', 'fromport': 'ip-permission.from-po

snapshot_filters = {'id': 'snapshot-id', 'owner': 'owner-id', 'status': 'status', '

start_instances(*instance_ids*, *regions=[]*)
Stops an Amazon EC2 instance

Parameters **instance_ids** (*lst*) – List of identifiers of instances to be started.

Examples

$ aws.service.ec2.start_instances(instances=['i-001']) $ aws.service.ec2.start_instances(instances=['i-001', 'i-033'], regions=['eu-west-1', 'eu-central-1'])

Returns List of instances to be started, with their previous and current status.

Return type lst

stop_instances(*instance_ids*, *regions=[]*, *force=False*)
Stops an Amazon EC2 instance

Parameters **instance_ids** (*lst*) – List of identifiers of instances to be stopped.

Examples

$ aws.service.ec2.stop_instances(instances=['i-001']) $ aws.service.ec2.stop_instances(instances=['i-001', 'i-033'], regions=['eu-west-1', 'eu-central-1'])

Returns List of instances to be stopped, with their previous and current status.

Return type lst

volume_filters = {'autodelete': 'attachment.delete-on-termination', 'encrypted': 'en

awspice.services.elb module

class awspice.services.elb.**ElbService**

Bases: *awspice.services.base.AwsBase*

Class belonging to the Load Balancers service.

loadbalancer_filters = {'cname': '', 'domain': '', 'tagname': ''}

get_loadbalancers(*regions=[]*)

Get all Elastic Load Balancers for a region

Parameters **regions** (*list*) – Regions where to look for this element

Returns List of dictionaries with the load balancers requested

Return type LoadBalancers (list)

get_loadbalancers_by(*filter_key*, *filter_value*, *regions=[]*)

Get loadbalancers which match with the filters

Parameters

- **filter_key** (*str*) – [description]
- **filter_value** (*str*) – [description]
- **regions** (*list, optional*) – Defaults to []. List of regions to search in

Returns List of load balancers requested

Return type list

get_loadbalancer_by(*filter_key*, *filter_value*, *regions=[]*)

Get a load balancer for a region that matches with filter

Parameters

- **filter_key** (*str*) – Name of the filter
- **filter_value** (*str*) – Value of the filter
- **regions** (*list*) – Regions where to look for this element

Raises

- **dns.resolver.NXDOMAIN** – DNS Name not registered.
- **dns.resolver.NoAnswer** – DNS Name not found.

Returns Dictionary with the load balancer requested

Return type LoadBalancer (dict)

__init__()
This constructor configures the corresponding service according to the class that calls it.

Every time the EC2Service Class is called (inherits from this class), this constructor will change the client's service to 'ec2'. And then, if ELBService service is called, this method is called again changing the service from 'ec2' to 'elb'.

Parameters **service** (*str*) – AWS service to uso

Returns None

awspice.services.iam module

class awspice.services.iam.**IamService**
Bases: *awspice.services.base.AwsBase*

Class belonging to the IAM Identity & Access management service.

get_inactive_users()
Get users who have not logged in AWS since 1 year. This method returns users who haven't used their password and one of their keys in less than 9 months.

Returns List of inactive users

Return type list

get_users()
List all users for an AWS account

Returns List of all users

get_access_keys(*user*)

get_access_key_last_used(*accesskey*)

__init__()
This constructor configures the corresponding service according to the class that calls it.

Every time the EC2Service Class is called (inherits from this class), this constructor will change the client's service to 'ec2'. And then, if ELBService service is called, this method is called again changing the service from 'ec2' to 'elb'.

Parameters **service** (*str*) – AWS service to uso

Returns None

awspice.services.rds module

class awspice.services.rds.**RdsService**
Bases: *awspice.services.base.AwsBase*

Class belonging to the Remote Database System service.

database_filters = {'cluster': 'db-cluster-id', 'id': 'db-instance-id'}

get_database_by(*filters*, *regions=[]*)

get_databases(*regions=[]*)
Get RDS instances in regions

Parameters **regions** (*list*) – Regions where you want to look for

Returns List of RDS dicts

Return type (list)

get_snapshots(*regions=[]*)
Get RDS snapshots in regions

Parameters **regions** (*list*) – Regions where you want to look for

Returns List of RDS dicts

Return type (list)

__init__()
This constructor configures the corresponding service according to the class that calls it.

Every time the EC2Service Class is called (inherits from this class), this constructor will change the client's service to 'ec2'. And then, if ELBService service is called, this method is called again changing the service from 'ec2' to 'elb'.

Parameters **service** (*str*) – AWS service to uso

Returns None

awspice.services.s3 module

class awspice.services.s3.**S3Service**
Bases: *awspice.services.base.AwsBase*

Class belonging to the S3 Storage service.

upload_string_as_file(*bucket_name*, *filepath*, *content*)
Upload string as a file to S3 bucket

Parameters

- **bucket_name** (*str*) – Name of the S3 bucket
- **filepath** (*str*) – File path which will be created. (i.e. 'folder1/folder2/filename.txt')
- **content** (*str*) – File content in string format.

Returns None

get_buckets()
Get all buckets in S3

Returns List of dictionaries with the buckets requested

Return type Buckets (list)

get_bucket_acl(*bucketname*)

get_public_buckets()
Get all public buckets and its permissions

This method returns all buckets in an AWS Account which have public permissions to read, write, read acl, write acl or even full control.

Returns List of dictionaries with the buckets requested

Return type Buckets-ACL (list)

list_bucket_objects(*bucket*)
List objects stored in a bucket

Parameters **bucket** (*str*) – Name of the bucket

Returns List of bucket objects

Return type list

__init__()
This constructor configures the corresponding service according to the class that calls it.

Every time the EC2Service Class is called (inherits from this class), this constructor will change the client's service to 'ec2'. And then, if ELBService service is called, this method is called again changing the service from 'ec2' to 'elb'.

Parameters **service** (*str*) – AWS service to uso

Returns None

Module contents

class awspice.services.**AwsBase**(*service*)
Bases: object

Base class from which all services inherit (ec2, s3, vpc . . .)

This class contains methods and properties that are common to all AWS services and should be accessible by all of them. This class is responsible for instantiating the client and processing information related to the accounts and regions.

client
Boto3 client

region
Current region used by the client

profile
Current profile used by the client

access_key
Current access key used by the client

secret_key
Current secret key used by the client

endpoints = None

region = None

profile = None

access_key = None

secret_key = None

pool = <awspice.helpers.ThreadPool object>

service_resources = ['ec2', 's3']

set_client(*service*)
Main method to set Boto3 client

Parameters

- **service** (*str*) – Service to use (i.e.: ec2, s3, vpc...)
- **region** (*str*) – Region name to use (i.e.: eu-central-1)
- **profile** (*str*) – Profile name set in ~/.aws/credentials file
- **access_key** (*str*) – API access key of your AWS account
- **secret_key** (*str*) – API secret key of your AWS account

Raises

- **ClientError** – Access keys are not valid or lack of permissions for a service/region
- **ProfileNotFound** – Profile name not found in credentials file

Returns None

classmethod set_auth_config(*region*, *profile=None*, *access_key=None*, *secret_key=None*)
Set properties like service, region or auth method to be used by boto3 client

Parameters

- **service** (*str*) – Service to use (i.e.: ec2, s3, vpc...)
- **region** (*str*) – Region name (i.e.: eu-central-1)
- **access_key** (*str*) – API Access key
- **secret_key** (*str*) – API Secret key
- **profile** (*str*) – Profile name set in ~/.aws/credentials file

classmethod get_client_vars()
Get information of the current client configuration Sometimes we need to store this variables, for example using threads, because AwsBase is constantly changing

Returns Array with current client configuration ({'region': 'eu-west-1', 'profile': 'default'})

Return type dict

classmethod inject_client_vars(*elements*, *client_conf=None*)
Insert in each item of a list, the region and the current credentials.

This function is called by all the methods of all the services that return a list of objects to identify in what region and account they have been found.

Parameters

- **elements** (*list*) – List of dictionaries
- **client_conf** (*dict*) – Array with the client configuration (see *get_client_vars*)

Returns list. Returns same list with the updated elements (region and authentication included)

region_in_regions(*region*, *regions*)
Check if region is in a complex list of regions

Parameters

- **region** (*str | lst*) – 'eu-west-1'}
- **regions** (*lst*) –

Examples

region_in_regions('eu-west-1', [{'RegionName': 'eu-west-1}])

Returns bool

classmethod validate_filters(*input_filters*, *accepted_filters*)
Transform filters into AWS filters format after validate them.

Parameters

- **input_filters** (*str*) – Items to validate
- **accepted_filters** (*list*) – Pre-validated list

Returns None

Raises **ValueError** – Filter is not in the accepted filter list

classmethod get_profiles()
Get a list of all available profiles in ~/.aws/credentials file

Returns list. List of strings with available profiles

change_profile(*profile*)
Change profile of the client

This method changes the account/profile used but keeps the same region and service

Parameters **profile** (*str*) – Name of the profile set in ~/.aws/credentials file

Examples

$ aws = awspice.connect() $ aws.service.ec2.change_profile('my_boring_company')

Returns None

parse_profiles(*profiles=[]*)
Validation method which get a profile or profile list and return the expected list of them

The purpose of this method is that a user can pass different types of data as a "profile" argument and obtain a valid output for any method that works with this type of data.

Parameters **profiles** (*list | str*) – String or list of string to parse

Examples

$ account_str = aws.service.ec2.parse_profiles('my_company') $ account_lst = aws.service.ec2.parse_profiles(['my_company']) $ accounts_lst = aws.service.ec2.parse_profiles(['my_company', 'other_company'])

Returns list. List of a strings with profile names

get_endpoints()
Get services and its regions and endpoints

Returns Dict with services (key) and its regions and Endpoints.

Return type dict

get_regions()
Get all available regions

Returns list. List of regions with 'Country' and 'RegionName'

change_region(*region*)
Change region of the client

This method changes the region used but keeps the same service and profile

Parameters **region** (*str*) – Region Name (ID) of AWS (i.e.: eu-central-1)

Examples

aws.service.ec2.change_region('eu-west-1')

Returns None

parse_regions(*regions=[]*, *default_all=False*)
Validation method which get a region or list of regions and return the expected list of them

The purpose of this method is that a user can pass different types of data as a "region" argument and obtain a valid output for any method that works with this type of data.

Parameters

- **regions** (*list | str*) – String or list of string to parse
- **default_all** (*bool*) – If the list of regions is empty and this argument is True, a list with all regions will be returned. This is useful when you do not know the data entry of type "region" and you want to search by default in all regions (if regions are empty means that the user does not know where an element is located).

Examples

AwsBase.region = aws.service.ec2.parse_regions([]) regions = aws.service.ec2.parse_regions('eu-west-1') regions = aws.service.ec2.parse_regions(['eu-west-1']) regions = aws.service.ec2.parse_regions(['eu-west-1', 'eu-west-2'])

Returns list. List of a strings with profile names

__init__(*service*)
This constructor configures the corresponding service according to the class that calls it.

Every time the EC2Service Class is called (inherits from this class), this constructor will change the client's service to 'ec2'. And then, if ELBService service is called, this method is called again changing the service from 'ec2' to 'elb'.

Parameters **service** (*str*) – AWS service to uso

Returns None

class awspice.services.**Ec2Service**
Bases: *awspice.services.base.AwsBase*

Class belonging to the EC2 Computing service.

set_tag(*resource_id*, *tag_key*, *tag_value*, *regions=[]*)
Set tag for an instance

Parameters

- **elements_id** (*str*) – Id of resources to tag. (i.e: i-01234, vol-01234)
- **tag_key** (*str*) – Name of the element TAG (i.e: Name)

- **tag_value** (*str*) – Value of that Tag
- **regions** (*lst*) – Regions where to look for this element

Returns None

__init__()
This constructor configures the corresponding service according to the class that calls it.

Every time the EC2Service Class is called (inherits from this class), this constructor will change the client's service to 'ec2'. And then, if ELBService service is called, this method is called again changing the service from 'ec2' to 'elb'.

Parameters **service** (*str*) – AWS service to uso

Returns None

address_filters = {'domain': 'domain', 'instance': 'instance-id', 'privateip': 'pri

ami_distributions = {'amazon': 'amzn-ami-hvm-20*.*.*-x86_64-*', 'ubuntu': 'ubuntu/ima

ami_filters = {'architecture': 'architecture', 'id': 'image-id', 'name': 'name', 'o

create_instances(*name*, *key_name*, *allowed_range*, *ami=None*, *distribution=None*, *version=None*, *instance_type='t2.micro'*, *region=None*, *vpc=None*, *count=1*)
Create a new instance

Parameters

- **name** (*str*) – TagName of the instance
- **key_name** (*str*) – The name of the key pair (i.e: it_user)
- **allowed_range** (*str*) – Network range with access to instance (i.e: 10.0.0.0/32)
- **ami** (*str*) – Id of the ami (i.e: ami-12345)
- **instance_type** (*str*) – Type of hardware of the instance (i.e: t2.medium)
- **distribution** (*str*) – Instead of ami, select an OS: (i.e: ubuntu)
- **region** (*str*) – Name of the region where instance will be displayed
- **vpc** (*str*) – VPC identifier where the instance will be deployed.
- **count** (*int*) – Number of instances to launch

Returns List of launched instances

Return type Instances (lst)

create_security_group(*name*, *allowed_range*, *vpc_id=None*)
Create a new Security Group

Parameters

- **name** (*str*) – Name of the Security Group
- **allowed_range** (*str*) – Network range with permissions (i.e: 10.0.0.0/32)
- **vpc_id** (*str*) – Id of assigned VPC

Returns Identifier of the security group created.

Return type str

delete_security_group(*identifier*)
Delete an existing Security Group

Parameters **identifier** (*str*) – Id of the Security Group

Returns none

distrib_amis = {'redhat': 'ami-c86c3f23', 'ubuntu': 'ami-f90a4880', 'windows': 'ami-

get_address_by (*filters*, *regions=[]*)
Get IP Addresses for a region that matches with filters

Parameters **regions** (*lst*) – Regions where to look for this element

Returns Dictionary with the address requested

Return type Address (dict)

get_addresses (*regions=[]*)
Get all IP Addresses for a region

Parameters **regions** (*lst*) – Regions where to look for this element

Returns List of dictionaries with the addresses requested

Return type Addresses (dict)

get_addresses_by (*filters*, *regions=[]*)
Get all IP Addresses for a region

Parameters **regions** (*lst*) – Regions where to look for this element

Returns List of dictionaries with the addresses requested

Return type Addresses (dict)

get_ami_by (*filters*, *regions=[]*)
Get an ami for one or more regions that matches with filter

Parameters

- **filter_key** (*str*) – Name of the filter
- **filter_value** (*str*) – Value of the filter
- **regions** (*lst*) – Regions where to look for this element

Returns Image requested

Return type Image (dict)

get_amis (*regions=[]*)
Get all images

Parameters **regions** (*lst*) – Regions where to look for this element

Returns List of all images

Return type Images (lst)

get_amis_by (*filters*, *regions=[]*, *return_first=False*)
Get list of amis for one or more regions that matches with filter

Parameters

- **filter_key** (*str*) – Name of the filter
- **filter_value** (*str*) – Value of the filter
- **regions** (*lst*) – Regions where to look for this element
- **return_first** (*bool*) – True if return first result

Returns List of requested images

Return type Images (lst)

get_amis_by_distribution(*distrib*, *version='*'*, *latest=False*, *regions=[]*)
Get one or more Images filtering by distribution

Parameters

- **distrib** (*str*) – Distribution of the image (i.e.: ubuntu)
- **version** (*str*) – Version of the system
- **latest** (*bool*) – True if only returns the newest item.
- **regions** (*lst*) – Regions where to look for this element

Returns List with the images requested.

Return type Image (lst)

get_default_vpc()
Get default Security Group

Returns Default security group resource

Return type SecurityGroup (dict)

get_instance_by(*filters*, *regions=[]*)
Get an instance for one or more regions that matches with filter

Parameters

- **filter_key** (*str*) – Name of the filter
- **filter_value** (*str*) – Value of the filter
- **regions** (*lst*) – Regions where to look for this element

Returns Dictionary with the instance requested

Return type Instance (dict)

get_instance_status_by(*filters*, *regions=[]*)

get_instances(*regions=[]*)
Get all instances for one or more regions.

Parameters **regions** (*lst*) – Regions where to look for this element

Returns List of dictionaries with the instances requested

Return type Instances (lst)

get_instances_by(*filters*, *regions=[]*, *return_first=False*)
Get an instance for one or more regions that matches with filter

Parameters

- **filter_key** (*str*) – Name of the filter
- **filter_value** (*str*) – Value of the filter
- **regions** (*lst*) – Regions where to look for this element
- **return_first** (*bool*) – Select to return the first match

Returns List of dictionaries with the instances requested

Return type Instances (lst)

get_instances_status(*regions=[]*)

get_instances_status_by (*filters*, *regions=[]*, *return_first=False*)

get_secgroup_by (*filters*, *regions=[]*)
Get security group for a region that matches with filters

Parameters

- **filter_key** (*str*) – Name of the filter
- **filter_value** (*str*) – Value of the filter

Returns Dictionaries with the security group requested

Return type SecurityGroup (dict)

get_secgroups (*regions=[]*)
Get all security groups for the current region

Returns List of dictionaries with the security groups requested

Return type SecurityGroups (lst)

get_secgroups_by (*filters*, *regions=[]*)
Get all security groups for a region that matches with filters

Parameters

- **filter_key** (*str*) – Name of the filter
- **filter_value** (*str*) – Value of the filter

Returns List of dictionaries with the security groups requested

Return type SecurityGroups (lst)

get_snapshot_by (*filters*)
Get a snapshot for a region tha matches with filters

Parameters

- **filter_key** (*str*) – Name of the filter
- **filter_value** (*str*) – Value of the filter

Returns Dictionary with the snapshot requested

Return type Snapshot (dict)

get_snapshots ()
Get all snapshots owned by self for the current region

Returns List of dictionaries with the snapshots requested

Return type Snapshots (lst)

get_snapshots_by (*filters*)
Get all snapshots for the current region that matches with filters

Parameters

- **filter_key** (*str*) – Name of the filter
- **filter_value** (*str*) – Value of the filter

Returns List of dictionaries with the snapshots requested

Return type Snapshots (lst)

get_volume_by (*filters*, *regions=[]*)
Get a volume for one or more regions that matches with filters

Parameters

- **filter_key** (*str*) – Name of the filter
- **filter_value** (*str*) – Value of the filter
- **regions** (*lst*) – Regions where to look for this element

Returns Dictionary with the volume requested

Return type Volume (dict)

get_volumes (*regions=[]*)
Get all volumes for one or more regions

Parameters **regions** (*lst*) – Regions where to look for this element

Returns List of dictionaries with the volumes requested

Return type Volumes (lst)

get_volumes_by (*filters*, *regions=[]*, *return_first=False*)
Get volumes for one or more regions that matches with filters

Parameters

- **filter_key** (*str*) – Name of the filter
- **filter_value** (*str*) – Value of the filter
- **regions** (*lst*) – Regions where to look for this element

Returns Dictionary with the volume requested

Return type Volume (dict)

get_vpcs (*regions=[]*)
Get all VPCs for a region

Returns List of dictionaries with the vpcs requested

Return type VPCs (lst)

instance_filters = {'dnsname': 'dns-name', 'id': 'instance-id', 'name': 'tag:Name',

instance_status_filters = {'event': 'event.code', 'instance-check': 'instance-status

secgroup_filters = {'description': 'description', 'fromport': 'ip-permission.from-po

snapshot_filters = {'id': 'snapshot-id', 'owner': 'owner-id', 'status': 'status', '

start_instances (*instance_ids*, *regions=[]*)
Stops an Amazon EC2 instance

Parameters **instance_ids** (*lst*) – List of identifiers of instances to be started.

Examples

$ aws.service.ec2.start_instances(instances=['i-001']) $ aws.service.ec2.start_instances(instances=['i-001', 'i-033'], regions=['eu-west-1', 'eu-central-1'])

Returns List of instances to be started, with their previous and current status.

Return type lst

stop_instances (*instance_ids*, *regions=[]*, *force=False*)
Stops an Amazon EC2 instance

Parameters **instance_ids** (*lst*) – List of identifiers of instances to be stopped.

Examples

$ aws.service.ec2.stop_instances(instances=['i-001']) $ aws.service.ec2.stop_instances(instances=['i-001', 'i-033'], regions=['eu-west-1', 'eu-central-1'])

Returns List of instances to be stopped, with their previous and current status.

Return type lst

volume_filters = {'autodelete': 'attachment.delete-on-termination', 'encrypted': 'en

class awspice.services.**ElbService**
Bases: *awspice.services.base.AwsBase*

Class belonging to the Load Balancers service.

loadbalancer_filters = {'cname': '', 'domain': '', 'tagname': ''}

get_loadbalancers (*regions=[]*)
Get all Elastic Load Balancers for a region

Parameters **regions** (*list*) – Regions where to look for this element

Returns List of dictionaries with the load balancers requested

Return type LoadBalancers (list)

get_loadbalancers_by (*filter_key*, *filter_value*, *regions=[]*)
Get loadbalancers which match with the filters

Parameters

- **filter_key** (*str*) – [description]
- **filter_value** (*str*) – [description]
- **regions** (*list, optional*) – Defaults to []. List of regions to search in

Returns List of load balancers requested

Return type list

get_loadbalancer_by (*filter_key*, *filter_value*, *regions=[]*)
Get a load balancer for a region that matches with filter

Parameters

- **filter_key** (*str*) – Name of the filter
- **filter_value** (*str*) – Value of the filter
- **regions** (*list*) – Regions where to look for this element

Raises

- **dns.resolver.NXDOMAIN** – DNS Name not registered.
- **dns.resolver.NoAnswer** – DNS Name not found.

Returns Dictionary with the load balancer requested

Return type LoadBalancer (dict)

__init__()
This constructor configures the corresponding service according to the class that calls it.

Every time the EC2Service Class is called (inherits from this class), this constructor will change the client's service to 'ec2'. And then, if ELBService service is called, this method is called again changing the service from 'ec2' to 'elb'.

Parameters **service** (*str*) – AWS service to uso

Returns None

class awspice.services.**IamService**
Bases: *awspice.services.base.AwsBase*

Class belonging to the IAM Identity & Access management service.

get_inactive_users()
Get users who have not logged in AWS since 1 year. This method returns users who haven't used their password and one of their keys in less than 9 months.

Returns List of inactive users

Return type list

get_users()
List all users for an AWS account

Returns List of all users

get_access_keys(*user*)

get_access_key_last_used(*accesskey*)

__init__()
This constructor configures the corresponding service according to the class that calls it.

Every time the EC2Service Class is called (inherits from this class), this constructor will change the client's service to 'ec2'. And then, if ELBService service is called, this method is called again changing the service from 'ec2' to 'elb'.

Parameters **service** (*str*) – AWS service to uso

Returns None

class awspice.services.**RdsService**
Bases: *awspice.services.base.AwsBase*

Class belonging to the Remote Database System service.

database_filters = {'cluster': 'db-cluster-id', 'id': 'db-instance-id'}

get_database_by(*filters*, *regions=[]*)

get_databases(*regions=[]*)
Get RDS instances in regions

Parameters **regions** (*list*) – Regions where you want to look for

Returns List of RDS dicts

Return type (list)

get_snapshots(*regions=[]*)
Get RDS snapshots in regions

Parameters **regions** (*list*) – Regions where you want to look for

Returns List of RDS dicts

Return type (list)

__init__()
This constructor configures the corresponding service according to the class that calls it.

Every time the EC2Service Class is called (inherits from this class), this constructor will change the client's service to 'ec2'. And then, if ELBService service is called, this method is called again changing the service from 'ec2' to 'elb'.

Parameters **service** (*str*) – AWS service to uso

Returns None

class awspice.services.**S3Service**
Bases: *awspice.services.base.AwsBase*

Class belonging to the S3 Storage service.

upload_string_as_file(*bucket_name*, *filepath*, *content*)
Upload string as a file to S3 bucket

Parameters

- **bucket_name** (*str*) – Name of the S3 bucket
- **filepath** (*str*) – File path which will be created. (i.e. 'folder1/folder2/filename.txt')
- **content** (*str*) – File content in string format.

Returns None

get_buckets()
Get all buckets in S3

Returns List of dictionaries with the buckets requested

Return type Buckets (list)

get_bucket_acl(*bucketname*)

get_public_buckets()
Get all public buckets and its permissions

This method returns all buckets in an AWS Account which have public permissions to read, write, read acl, write acl or even full control.

Returns List of dictionaries with the buckets requested

Return type Buckets-ACL (list)

list_bucket_objects(*bucket*)
List objects stored in a bucket

Parameters **bucket** (*str*) – Name of the bucket

Returns List of bucket objects

Return type list

__init__()
This constructor configures the corresponding service according to the class that calls it.

Every time the EC2Service Class is called (inherits from this class), this constructor will change the client's service to 'ec2'. And then, if ELBService service is called, this method is called again changing the service from 'ec2' to 'elb'.

Parameters **service** (*str*) – AWS service to uso

Returns None

class awspice.services.**AcmService**

Bases: *awspice.services.base.AwsBase*

Class belonging to the ACM certificate management service.

list_certificates(*regions=[]*)

List all certificates

Parameters **regions** (*lst*) – List of regions to list certificates

Returns List of certificates

get_certificate_by(*filter_key*, *filter_value*, *regions=[]*)

Get certificate filtering by domain

Parameters

- **filter_key** (*str*) – Name of the field to be searched. (Domain)
- **filter_value** (*str*) – Value for the previous field. (i.e.: google.es)
- **regions** (*lst*) – List of regions where the certificate can be.

Returns Certificate matched to the filter entered.

get_certificate(*arn*, *regions=[]*)

Get certificate using CertificateArn (Ceritificate Identifier)

Parameters

- **arn** (*str*) – ARN of the certificate
- **regions** (*lst*) – List of regions where the certificate can be.

Returns Certificate matched to the ARN entered.

__init__()

This constructor configures the corresponding service according to the class that calls it.

Every time the EC2Service Class is called (inherits from this class), this constructor will change the client's service to 'ec2'. And then, if ELBService service is called, this method is called again changing the service from 'ec2' to 'elb'.

Parameters **service** (*str*) – AWS service to uso

Returns None

class awspice.services.**CostExplorerService**

Bases: *awspice.services.base.AwsBase*

Class belonging to the Cost Explorer service.

granularities = ['DAILY', 'MONTHLY']

filter_dimensions = ['AZ', 'INSTANCE_TYPE', 'LINKED_ACCOUNT', 'OPERATION', 'PURCHASE_T

group_dimensions = ['AZ', 'INSTANCE_TYPE', 'LEGAL_ENTITY_NAME', 'LINKED_ACCOUNT', 'OPE

get_cost(*from_date=None*, *to_date=None*, *interval='Monthly'*, *group_by=''*, *group_by_tag_value=''*, *filter_by={}*, *ec2_running_hours=False*)

Get the cost of account or its elements.

This method obtains costs of an account/s , one or several elements (substances, balancers, addresses) between two dates and granularized in days or months. If the date is not indicated, the cost of the last month will be returned.

Parameters

- **from_date** (*str*) – Date from which you want to obtain data. (Format: 2018-04-24)
- **to_date** (*str*) – Date until which you want to obtain data. (Format: 2018-04-24)
- **interval** (*str*) – Time interval to be analyzed. [MONTHLY | DAILY]
- **group_by** (*str*) – Group results by ['AZ', 'INSTANCE_TYPE', 'LEGAL_ENTITY_NAME', 'LINKED_ACCOUNT', 'OPERATION', 'PLATFORM', 'PURCHASE_TYPE', 'SERVICE', 'TAG', 'TENANCY', 'USAGE_TYPE']
- **group_by_tag_value** (*str*) – TAG key in case group_by set to 'TAG' (i.e. Name, Project or Environment)
- **filter_by** (*dict*) – Key of the filter and value. {'TAG_NAME': ['ec2-tagname', 'LINKED_ACCOUNT: ['1234']]}

Examples

get_cost(['machine-1', 'machine-2'], '2018-12-24', '2018-12-26', interval='daily') get_cost() # Get account cost

Returns List of days or months with the requested costs

Return type Costs (list)

__init__()

This constructor configures the corresponding service according to the class that calls it.

Every time the EC2Service Class is called (inherits from this class), this constructor will change the client's service to 'ec2'. And then, if ELBService service is called, this method is called again changing the service from 'ec2' to 'elb'.

Parameters **service** (*str*) – AWS service to uso

Returns None

class awspice.services.**Route53Service**

Bases: *awspice.services.base.AwsBase*

Class belonging to the Route 53 DNS Service

get_domains()

Get hosted zones and its records

Returns List of Hosted Zones with Records

Return type (lst)

list_hosted_zones()

List all hosted zones

Returns List of hosted zones

list_records(*hosted_zone_id*)

List all records for a hosted zone

Parameters **zone** (*hosted*) – The ID of the hosted zone that contains the resource record sets that you want to list

Returns List of DNS records

list_records_by_domain(*domain*)
List all records of a hosted-zone domain

Parameters **domain** (*str*) – The DOMAIN name of the hosted zone that contains the resource record sets that you want to list

Returns List of DNS records

__init__()
This constructor configures the corresponding service according to the class that calls it.

Every time the EC2Service Class is called (inherits from this class), this constructor will change the client's service to 'ec2'. And then, if ELBService service is called, this method is called again changing the service from 'ec2' to 'elb'.

Parameters **service** (*str*) – AWS service to uso

Returns None

5.1.2 Submodules

5.1.3 awspice.helpers module

class awspice.helpers.**Worker**(*tasks*)
Bases: threading.Thread

Thread executing tasks from a given tasks queue http://code.activestate.com/recipes/577187-python-thread-pool/

__init__(*tasks*)
This constructor should always be called with keyword arguments. Arguments are:

group should be None; reserved for future extension when a ThreadGroup class is implemented.

target is the callable object to be invoked by the run() method. Defaults to None, meaning nothing is called.

name is the thread name. By default, a unique name is constructed of the form "Thread-N" where N is a small decimal number.

args is the argument tuple for the target invocation. Defaults to ().

kwargs is a dictionary of keyword arguments for the target invocation. Defaults to {}.

If a subclass overrides the constructor, it must make sure to invoke the base class constructor (Thread.__init__()) before doing anything else to the thread.

run()
Method representing the thread's activity.

You may override this method in a subclass. The standard run() method invokes the callable object passed to the object's constructor as the target argument, if any, with sequential and keyword arguments taken from the args and kwargs arguments, respectively.

class awspice.helpers.**ThreadPool**(*num_threads*)
Bases: object

Pool of threads consuming tasks from a queue http://code.activestate.com/recipes/577187-python-thread-pool/

__init__(*num_threads*)
Initialize self. See help(type(self)) for accurate signature.

add_task(*func*, **args*, ***kargs*)
Add a task to the queue

wait_completion()
Wait for completion of all the tasks in the queue

class awspice.helpers.**ClsEncoder**(**, skipkeys=False, ensure_ascii=True, check_circular=True, allow_nan=True, sort_keys=False, indent=None, separators=None, default=None*)
Bases: json.encoder.JSONEncoder

JSON encoder extension.

Sometimes Boto3 returns a non-serializable result to JSON and we get the following error when dumping that result: *TypeError: datetime.datetime (2015, 12, 3, 21, 20, 17, 326000, tzinfo = tzutc ()) is not JSON serializable*
Solve it using this class encoder in `cls` argument

Examples

json.dumps(results, indent=4, cls=awspice.ClsEncoder)

default(*obj*)
Implement this method in a subclass such that it returns a serializable object for `o`, or calls the base implementation (to raise a `TypeError`).

For example, to support arbitrary iterators, you could implement default like this:

```
def default(self, o):
    try:
        iterable = iter(o)
    except TypeError:
        pass
    else:
        return list(iterable)
    # Let the base class default method raise the TypeError
    return JSONEncoder.default(self, o)
```

awspice.helpers.**ip_in_aws**(*ip*)
Check if an IP address is from AWS

Parameters **ip** – Address to check

Returns bool

awspice.helpers.**extract_region_from_ip**(*ip*)
Get the region where a IP is located and if it's on AWS

Parameters **ip** (*str*) – IP address

Returns It's in AWS / Region where the IP is located

Return type (bool, str)

awspice.helpers.**dnsinfo_from_ip**(*ip*)
Returns the DNS name of an IP address

Parameters **ip** – Address of the element.

Examples

dns = get_dnsname_from_ip('8.8.8.8')

Returns {'region': 'eu-west-1', 'service': 'ec2'}

Return type dict

5.1.4 awspice.manager module

class awspice.manager.**AwsManager**(*region='eu-west-1'*, *profile=None*, *access_key=None*, *secret_key=None*)

Bases: object

Main class that provides access to services (ec2, s3, vpc ...) and modules (finder, stats ..)

This master class provides access to individual services through the "services" property, and also to other complex modules such as "finder", "stats" and "security".

aws
Object of type #ServiceManager that provides access to the other services.

service

finder

security

stats

test()
Method to verify that the loaded configuration is correct and access with the AWS API is correct

Returns boolean. True if the test was successful, false if it failed.

__init__(*region='eu-west-1'*, *profile=None*, *access_key=None*, *secret_key=None*)
Initialization and configuration of the client

Parameters

- **region** (*str*) – Region in which to make queries and operations.
- **profile** (*str*) – Name of the AWS profile set in ~/.aws/credentials file
- **access_key** (*str*) – API access key of your AWS account
- **secret_key** (*str*) – API secret key of your AWS account

Returns None

5.1.5 awspice.servicemanager module

class awspice.servicemanager.**ServiceManager**(*region*, *profile=None*, *access_key=None*, *secret_key=None*)

Bases: object

Parent class that provides access to services.

For each service (ec2, s3, vpc ...) you are given access through a property of this class. This property will return an instance of the corresponding class, for example Ec2Service or VpcService. Each class of service (Ec2Service, S3Service ...) inherits from the AwsBase class.

ec2

elb

acm

iam

rds

s3

ce

route53

classmethod get_auth_config()
Get the configuration of the client currently configured

This method allows us to work with multiple accounts and different authentication methods (keys and profiles) without getting lost.

Returns

A dictionary with the type of authentication used and the associated value. The secret_key is not returned for security reasons.

{'Authorization': {'Type': 'Profile', 'Value': 'MyBoringCompany'}}

__init__(*region*, *profile=None*, *access_key=None*, *secret_key=None*)
Constructor of the parent class of the services.

With this method you can modify the configuration of the awspice client. It allows us to change the profile, the region or the access codes.

Parameters

- **region** (*str*) – Region in which to make queries and operations.
- **profile** (*str*) – Name of the AWS profile set in ~/.aws/credentials file
- **access_key** (*str*) – API access key of your AWS account
- **secret_key** (*str*) – API secret key of your AWS account

5.1.6 Module contents

CHAPTER 6

What is Awspice?

Is a wrapper tool of Boto3 library to list inventory and manage your AWS infrastructure The objective of the wrapper is to abstract the use of AWS, being able to dig through all the data of our account, and for example you will be able of:

- Run a ssh-command for all instances in all regions
- List all instances with exposed critical ports like 22 or 3389
- Get info about all certificates of your account/s
- Obtain all the infrastructure after a domain associated with a balancer

CHAPTER 7

Installation

```
pip install awspice
```

CHAPTER 8

Configuration

The client is built and configured using `awspice.connect()`. This method indicates the type of authentication and region on which you are going to work.

```
import awspice

aws = awspice.connect() # Region: eu-west-1 | Profile: Default

aws = awspice.connect(region='us-west-2', profile='dev_profile')
aws = awspice.connect('us-west-2', access_key='AKIA**********', secret_key='/HR
↪$4**********')
```

CHAPTER 9

Test it

To verify that the configuration has been correctly stored, you can run the following test. This test only checks that your user is registered and enabled on the AWS account set in the client's configuration.

```
import awspice

aws = awspice.connect(profile='<YOUR_PROFILE>')
aws.test()
```

CHAPTER 10

Using boto3 client

If you want to use the native Boto3 client to perform some operation, you can also do it using the "client" attribute within each service. If you call the client through the class *ec2*, this will be the service on which the client will be configured. The region and authentication will be the same as the last call made.

```
import awspice

aws = awspice.connect(region='us-east-1', profile='sample')
aws.service.ec2.client.describe_instance_status(InstanceIds=['i-12345'])
```

10.1 Indices and tables

- genindex
- modindex

10.2 Contact me

Author: David Amrani Hernández
Github: @davidmoremad

Python Module Index

a

Index

Symbols

A

C

D

E

F

G

I

L

P

R

S

T

U

V

W