
cloudbridge Documentation

Release 1.0.2

GVL and Galaxy Projects

Dec 01, 2020

Contents

1	Usage example	3
2	Quick Reference	5
3	Installation	7
4	Documentation	9
4.1	Concepts and Organisation	9
4.2	Getting Started	10
4.3	Using CloudBridge	15
4.4	Contributor Guide	61
4.5	API reference	68
5	Page index	117
	Index	119

CloudBridge aims to provide a simple layer of abstraction over different cloud providers, reducing or eliminating the need to write conditional code for each cloud.

CHAPTER 1

Usage example

The simplest possible example for doing something useful with CloudBridge would look like the following.

```
from cloudbridge.factory import CloudProviderFactory, ProviderList

provider = CloudProviderFactory().create_provider(ProviderList.AWS, {})
print(provider.compute.instances.list())
```

In the example above, the `AWS_ACCESS_KEY` and `AWS_SECRET_KEY` environment variables must be set to your cloud credentials.

CHAPTER 2

Quick Reference

The following object graph shows how to access various provider services, and the resource that they return. Click on any object to drill down into its details.

CHAPTER 3

Installation

The latest release can always be installed from PyPI. For other installation options, see the [installation page](#):

```
pip install cloudbridge
```


4.1 Concepts and Organisation

4.1.1 Object types

Conceptually, CloudBridge consists of the following types of objects.

1. Providers - Represents a connection to a cloud provider, and is the gateway to using its services.
2. Services - Represents a service provided by a cloud provider, such as its compute service, storage service, networking service etc. Services may in turn be divided into smaller services. Smaller services tend to have uniform methods, such as create, find and list. For example, `InstanceService.list()`, `InstanceService.find()` etc. which can be used to access cloud resources. Larger services tend to provide organisational structure only. For example, the storage service provides access to the `VolumeService`, `SnapshotService` and `BucketService`.
3. Resources - resources are objects returned by a service, and represent a remote resource. For example, `InstanceService.list()` will return a list of Instance objects, which can be used to manipulate an instance. Similarly, `VolumeService.create()` will return a Volume object.

The actual source code structure of CloudBridge also mirrors this organisation.

4.1.2 Object identification and naming

In order to function uniformly across cloud providers, object identity and naming must be conceptually consistent. In CloudBridge, there are three main properties for identifying and naming an object.

1. Id - The *id* corresponds to a unique identifier that can be reliably used to reference a resource. All CloudBridge resources have an id. Most methods in CloudBridge services, such as *get*, use the *id* property to identify and retrieve objects.
2. Name - The *name* property is a more human-readable identifier for a particular resource, and is often useful to display to the end user instead of the *id*. While it is often unique, it is not guaranteed to be so, and therefore, the *id* property must always be used for uniquely identifying objects. All CloudBridge resources have a *name* property. The

name property is often assigned during resource creation, and is often derived from the *label* property by appending some unique characters to it. Once assigned however, it is unchangeable.

3. Label - Most resources also support a *label* property, which is a user changeable value that can be used to describe an object. When creating resources, cloudbridge often accepts a *label* property as a parameter. The *name* property is derived from the *label*, by appending some unique characters to it. However, there are some resources which do not support a *label* property, such as key pairs and buckets. In the latter case, the *name* can be specified during resource creation, but cannot be changed thereafter.

4.1.3 Detailed class relationships

The following diagram shows a typical provider object graph and the relationship between services.

Some services are nested. For example, to access the instance service, you can use *provider.compute.instances*. Similarly, to get a list of all instances, you can use the following code.

```
instances = provider.compute.instances.list()
print(instances[0].name)
```

4.2 Getting Started

This getting started guide will provide a quick tour of some CloudBridge features. For more details on individual features, see the [Using CloudBridge](#) section or the [API reference](#).

4.2.1 Installation

CloudBridge is available on PyPI so to install the latest available version, run:

```
pip install --upgrade cloudbridge
```

For common issues during setup, check the following section: *Common Setup Issues* <[topics/troubleshooting.html](#)>

4.2.2 Create a provider

To start, you will need to create a reference to a provider object. The provider object identifies the cloud you want to work with and supplies your credentials. Each provider instance is tied to a particular zone. Refer to the [Setup page](#) for more details information on provider configuration. Once a provider instance is created, the remainder of the code is the same for any provider.

AWS:

```
from cloudbridge.factory import CloudProviderFactory, ProviderList

config = {'aws_access_key': 'AKIAJW2XCYO4AF55XFEQ',
          'aws_secret_key': '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',
          'aws_zone_name': 'us-east-1a'}
provider = CloudProviderFactory().create_provider(ProviderList.AWS, config)
image_id = 'ami-0885b1f6bd170450c' # Ubuntu 20.04 (HVM)
```

OpenStack (with Keystone authentication v2):

```

from cloudbridge.factory import CloudProviderFactory, ProviderList

config = {'os_username': 'username',
          'os_password': 'password',
          'os_auth_url': 'authentication URL',
          'os_region_name': 'region name',
          'os_project_name': 'project name',
          'os_zone_name': 'zone_name'}
provider = CloudProviderFactory().create_provider(ProviderList.OPENSTACK,
                                                  config)
image_id = 'clf4b7bc-a563-4feb-b439-a2e071d861aa' # Ubuntu 14.04 @ NeCTAR

```

OpenStack (with Keystone authentication v3):

```

from cloudbridge.factory import CloudProviderFactory, ProviderList

config = {'os_username': 'username',
          'os_password': 'password',
          'os_auth_url': 'authentication URL',
          'os_project_name': 'project name',
          'os_project_domain_name': 'project domain name',
          'os_user_domain_name': 'domain name',
          'os_zone_name': 'zone_name'}
provider = CloudProviderFactory().create_provider(ProviderList.OPENSTACK,
                                                  config)
image_id = '46794408-6a80-44b1-bf5a-405127753f43' # Ubuntu 20.04@Jetstream

```

Azure:

```

from cloudbridge.factory import CloudProviderFactory, ProviderList

config = {'azure_subscription_id': 'REPLACE WITH ACTUAL VALUE',
          'azure_client_id': 'REPLACE WITH ACTUAL VALUE',
          'azure_secret': 'REPLACE WITH ACTUAL VALUE',
          'azure_tenant': 'REPLACE WITH ACTUAL VALUE',
          'azure_zone_name': 'zone_name'}
provider = CloudProviderFactory().create_provider(ProviderList.AZURE, config)
image_id = 'Canonical:UbuntuServer:16.04.0-LTS:latest' # Ubuntu 16.04

```

Google Compute Cloud:

```

from cloudbridge.factory import CloudProviderFactory, ProviderList

config = {'gcp_project_name': 'project name',
          'gcp_service_creds_file': 'service_file.json',
          'gcp_region_name': 'us-east1', # Use desired value
          'gcp_zone_name': 'us-east1-b'} # Use desired value
provider = CloudProviderFactory().create_provider(ProviderList.GCP, config)
image_id = 'https://www.googleapis.com/compute/v1/projects/ubuntu-os-cloud/global/
↳ images/ubuntu-1804-bionic-v20181222'

```

4.2.3 List some resources

Once you have a reference to a provider, explore the cloud platform:

```
provider.security.vm_firewalls.list()
provider.compute.vm_types.list()
provider.storage.snapshots.list()
provider.storage.buckets.list()
```

This will demonstrate the fact that the library was properly installed and your provider object is setup correctly. By itself, those commands are not very interesting so let's create a new instance we can ssh into using a key pair.

4.2.4 Create a key pair

We'll create a new key pair and save the private portion of the key to a file on disk as a read-only file.

```
import os
kp = provider.security.key_pairs.create('cb-keypair')
with open('cloudbridge_intro.pem', 'wb') as f:
    f.write(kp.material)
os.chmod('cloudbridge_intro.pem', 0o400)
```

4.2.5 Create a network

A cloudbridge instance should be launched into a private subnet. We'll create a private network and subnet, and make sure it has internet connectivity, by attaching an internet gateway to the subnet via a router.

```
net = provider.networking.networks.create(cidr_block='10.0.0.0/16',
                                          label='cb-network')
sn = net.subnets.create(
    cidr_block='10.0.0.0/28', label='cb-subnet')
router = provider.networking.routers.create(network=net, label='cb-router')
router.attach_subnet(sn)
gateway = net.gateways.get_or_create()
router.attach_gateway(gateway)
```

4.2.6 Create a VM firewall

Next, we need to create a VM firewall (also commonly known as a security group) and add a rule to allow ssh access. A VM firewall needs to be associated with a private network.

```
from cloudbridge.interfaces.resources import TrafficDirection
fw = provider.security.vm_firewalls.create(
    label='cb-firewall', description='A VM firewall used by
    CloudBridge', network=net)
fw.rules.create(TrafficDirection.INBOUND, 'tcp', 22, 22, '0.0.0.0/0')
```

4.2.7 Launch an instance

We can now launch an instance using the created key pair and security group. We will launch an instance type that has at least 2 CPUs and 4GB RAM. We will also add the network interface as a launch argument.


```

img = provider.compute.images.get(image_id)
vm_type = sorted([t for t in provider.compute.vm_types
                  if t.vcpus >= 2 and t.ram >= 4],
                 key=lambda x: x.vcpus*x.ram)[0]
inst = provider.compute.instances.create(
    image=img, vm_type=vm_type, label='cb-instance',
    subnet=sn, key_pair=kp, vm_firewalls=[fw])
# Wait until ready
inst.wait_till_ready() # This is a blocking call
# Show instance state
inst.state
# 'running'

```

Note: Note that we iterated through `provider.compute.vm_types` directly instead of calling `provider.compute.vm_types.list()`. This is because we need to iterate through all records in this case. The `list()` method may not always return all records, depending on the global limit for records, necessitating that additional records be paged in. See [Paging and iteration](#).

4.2.8 Assign a public IP address

To access the instance, let's assign a public IP address to the instance. For this step, we'll first need to allocate a floating IP address for our account and then associate it with the instance. Note that floating IPs are associated with an Internet Gateway so we allocate the IP under the gateway we dealt with earlier.

```

if not inst.public_ips:
    fip = gateway.floating_ips.create()
    inst.add_floating_ip(fip)
    inst.refresh()
inst.public_ips
# ['54.166.125.219']

```

From the command prompt, you can now ssh into the instance `ssh -i cloudbridge_intro.pem ubuntu@54.166.125.219`.

4.2.9 Get a resource

When a resource already exists, a reference to it can be retrieved using either its ID, name, or label. It is important to note that while IDs and names are unique, multiple resources of the same type could use the same label, thus the *find* method always returns a list, while the *get* method returns a single object. While the methods are similar across resources, they are explicitly listed in order to help map each resource with the service that handles it. Note that labeled resources allow to find by label, while unlabeled resources find by name or their special properties (eg: `public_ip` for floating IPs). For more detailed information on the types of resources and their provider mappings, see [Resource Types and Dashboard Mapping](#).

```

# Key Pair
kp = provider.security.key_pairs.get('keypair ID')
kp = provider.security.key_pairs.find(name='cb-keypair')[0]

# Floating IPs
fip = gateway.floating_ips.get('FloatingIP ID')
# Find using public IP address

```

(continues on next page)

(continued from previous page)

```

fip_list = gateway.floating_ips.find(public_ip='IP address')
# Find using name (the behavior of the `name` property can be
# cloud-dependent). More details can be found here <topics/resource_types_and_
↪mapping.html>`
fip_list = gateway.floating_ips.find(name='cb-fip')[0]

# Network
net = provider.networking.networks.get('network ID')
net_list = provider.networking.networks.find(label='my-network')
net = net_list[0]

# Subnet
sn = provider.networking.subnets.get('subnet ID')
# Unknown network
sn_list = provider.networking.subnets.find(label='cb-subnet')
# Known network
sn_list = provider.networking.subnets.find(network=net.id,
                                           label='cb-subnet')

sn = sn_list[0]

# Router
router = provider.networking.routers.get('router ID')
router_list = provider.networking.routers.find(label='cb-router')
router = router_list[0]

# Gateway
gateway = net.gateways.get_or_create()

# Firewall
fw = provider.security.vm_firewalls.get('firewall ID')
fw_list = provider.security.vm_firewalls.find(label='cb-firewall')
fw = fw_list[0]

# Instance
inst = provider.compute.instances.get('instance ID')
inst_list = provider.compute.instances.list(label='cb-instance')
inst = inst_list[0]

```

4.2.10 Cleanup

To wrap things up, let's clean up all the resources we have created

```

from cloudbridge.interfaces import InstanceState
inst.delete()
inst.wait_for([InstanceState.DELETED, InstanceState.UNKNOWN],
              terminal_states=[InstanceState.ERROR]) # Blocking call
fip.delete()
fw.delete()
kp.delete()
os.remove('cloudbridge_intro.pem')
router.detach_gateway(gateway)
router.detach_subnet(sn)
gateway.delete()
router.delete()
sn.delete()

```

(continues on next page)

(continued from previous page)

```
net.delete()
```

And that's it - a full circle in a few lines of code. You can now try the same with a different provider. All you will need to change is the cloud-specific data, namely the provider setup and the image ID.

4.3 Using CloudBridge

Introductions to all the key parts of CloudBridge you'll need to know:

4.3.1 Installation

Prerequisites: CloudBridge runs on Python 2.7 and higher. Python 3 is recommended.

We highly recommend installing CloudBridge in a [virtualenv](#). Creating a new virtualenv is simple:

```
pip install virtualenv
virtualenv .venv
source .venv/bin/activate
```

Latest stable release

The latest release of CloudBridge can be installed from PyPI:

```
pip install cloudbridge
```

Latest unreleased dev version

The development version of the library can be installed directly from the [GitHub repo](#):

```
$ pip install --upgrade git+https://github.com/CloudVE/cloudbridge.git
```

Developer installation

To install additional libraries required by CloudBridge contributors, such as [tox](#), clone the source code repository and run the following command from the repository root directory:

```
$ git clone https://github.com/CloudVE/cloudbridge.git
$ cd cloudbridge
$ pip install --upgrade --editable .[dev]
```

Checking installation

To check what version of the library you have installed, do the following:

```
import cloudbridge
cloudbridge.get_version()
```

4.3.2 Procuring access credentials

To initialize a connection to a cloud and get a provider object, you will need to provide the cloud's access credentials to CloudBridge. This page will walk you through the process of procuring credentials on different providers. For more information on using the obtained credentials with CloudBridge, see [Connection and Authentication Setup](#) page.

Microsoft Azure

The page linked below from the Microsoft documentation was used to create this section, and can be followed instead of this CloudBridge-specific documentation to procure Azure credentials <https://docs.microsoft.com/en-us/azure/azure-resource-manager/resource-group-create-service-principal-portal>.

In order to follow the instructions below, one needs a Microsoft Azure account and an active Subscription, as well as web portal access: <https://portal.azure.com/> The following section will walk you through the process of creating an application, which is required for API access, as well as help you locate all required Azure credentials for API access via CloudBridge.

Azure credentials require a *Subscription ID* which can be procured through *Subscriptions* in the Azure portal.

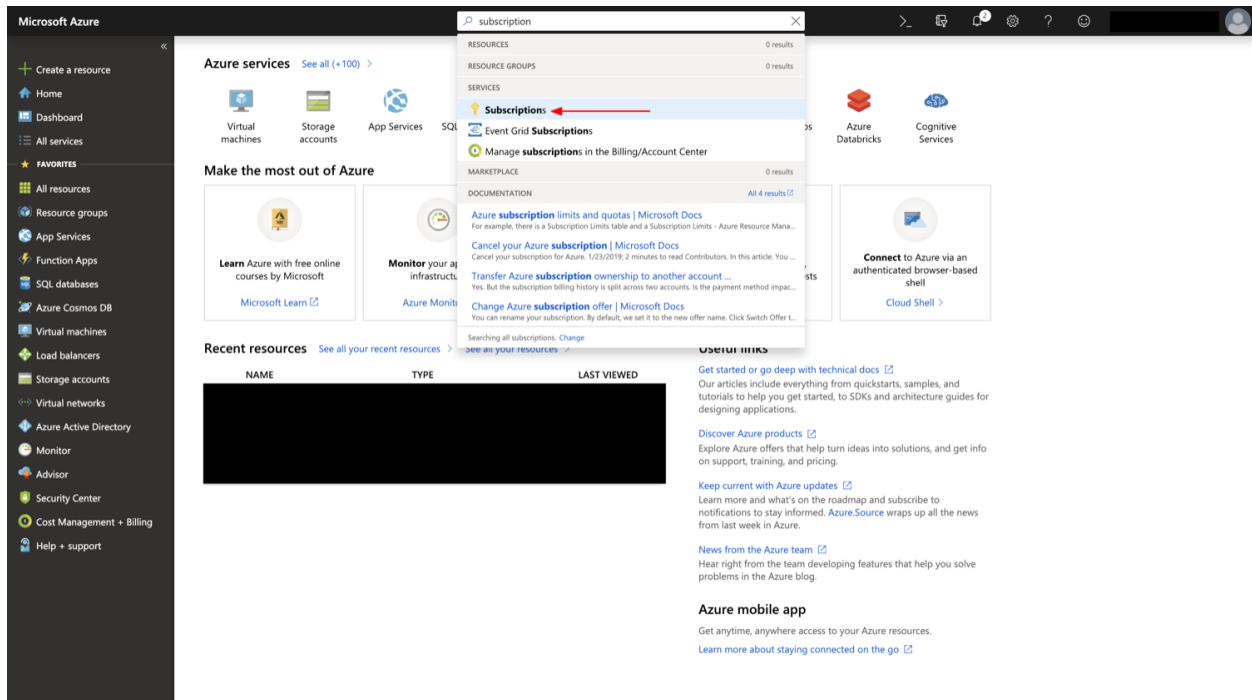


Fig. 1: Subscriptions can be accessed by searching, or by choosing *Subscriptions* in the *All Services* window

Another piece of credentials that already exists on any Azure account is a *Tenant ID* which will correspond to the *Active Directory ID*, which can be procured through the Azure Active Directory's *Properties*.

In order to access the API, an application needs to be registered and a key needs to be created. After creating an application through the *App Registrations* window under the *Active Directory*, the *Application ID* of the app will correspond to the *Client ID* in CloudBridge, and the generated value of its key, will correspond to the *Secret*.

Finally, in order to have appropriate permissions, you must assign an appropriate role to the newly created application. Permissions can be assigned at the level of the Subscription, or at the level of each Resource Group. *Contributor* access is recommended for general use in order to have sufficient permissions to create and manage all types of resources, but specific roles can also be assigned for more limited access.

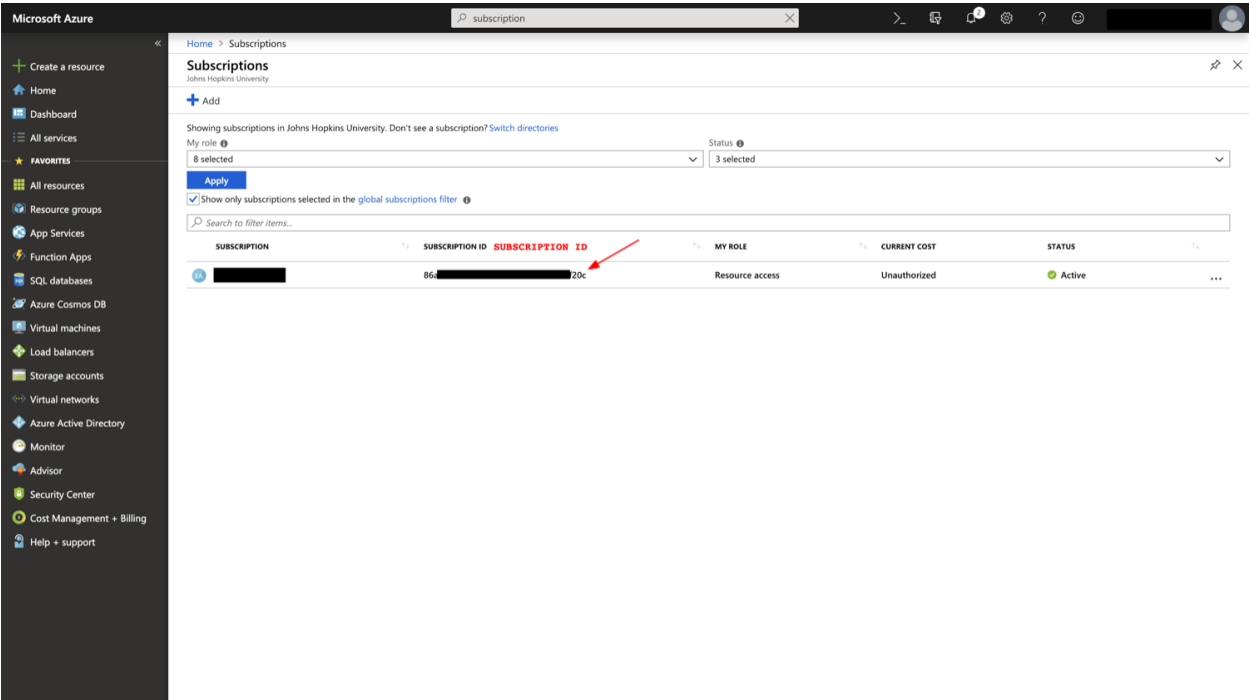


Fig. 2: The *Subscription ID* will be found in the list of subscriptions

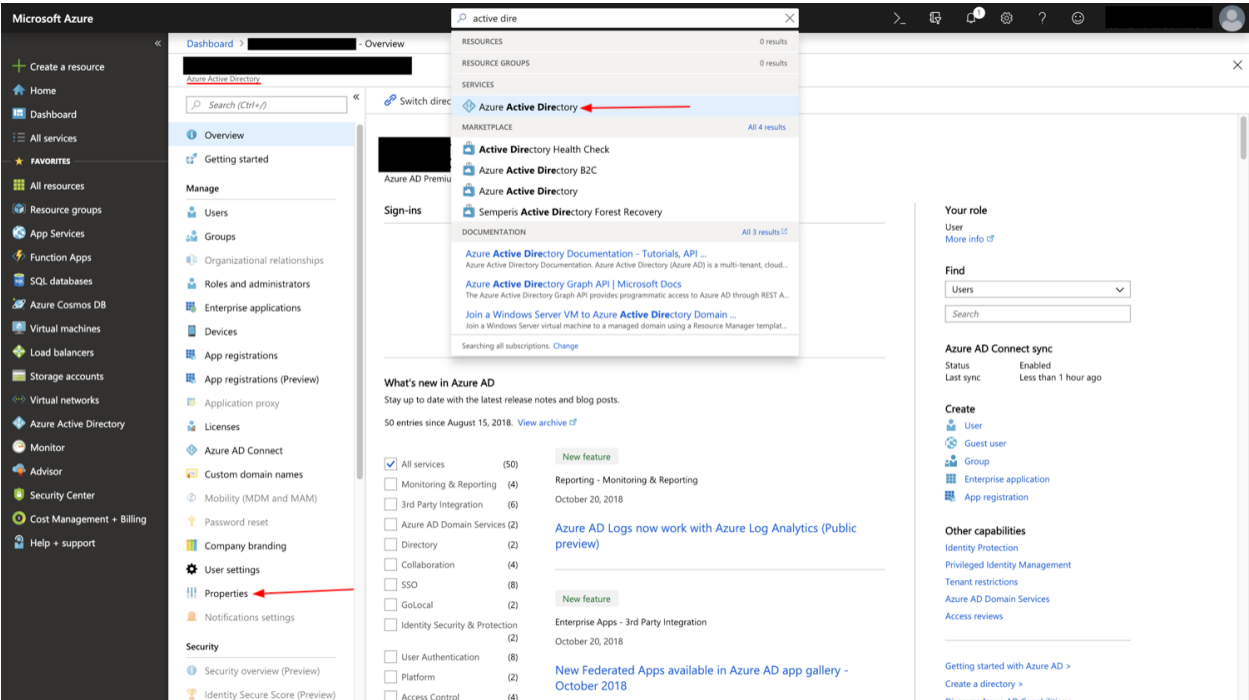


Fig. 3: The *Azure Active Directory* can be access by searching, or by choosing *Azure Active Directory* in the *All Services* window. *Azure Active Directory* is also a default favorite on the sidebar.

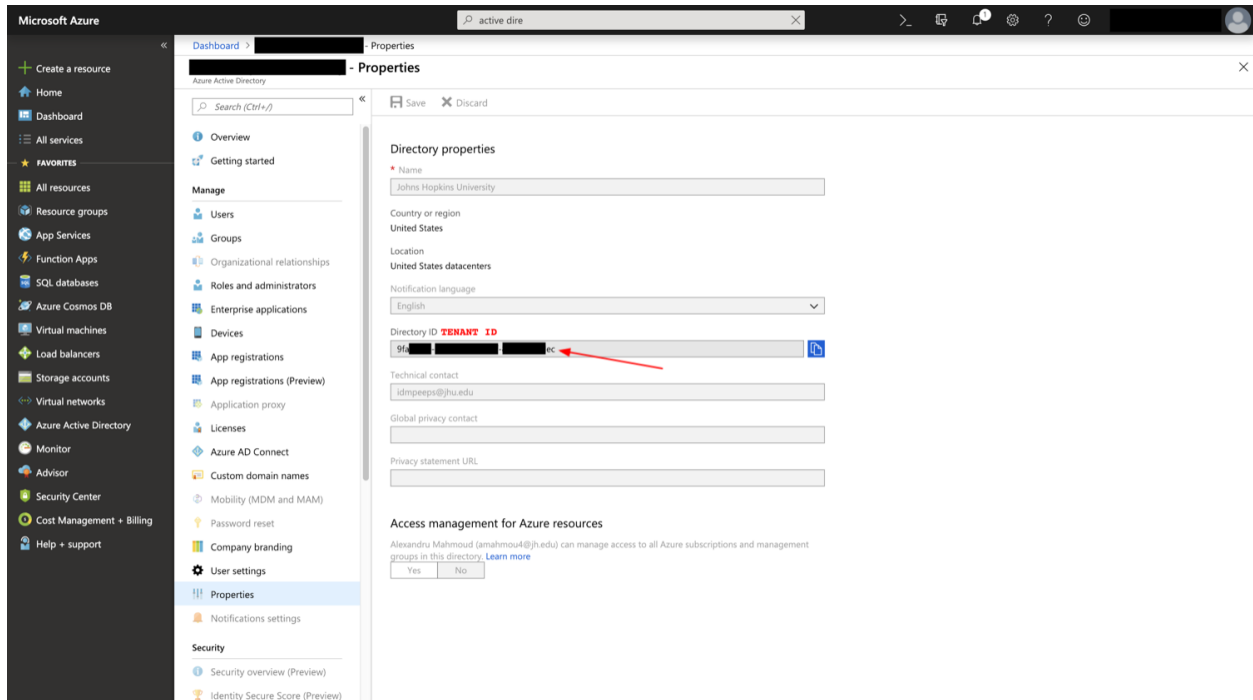


Fig. 4: The *Directory ID* will be found in the Directory's *Properties* section

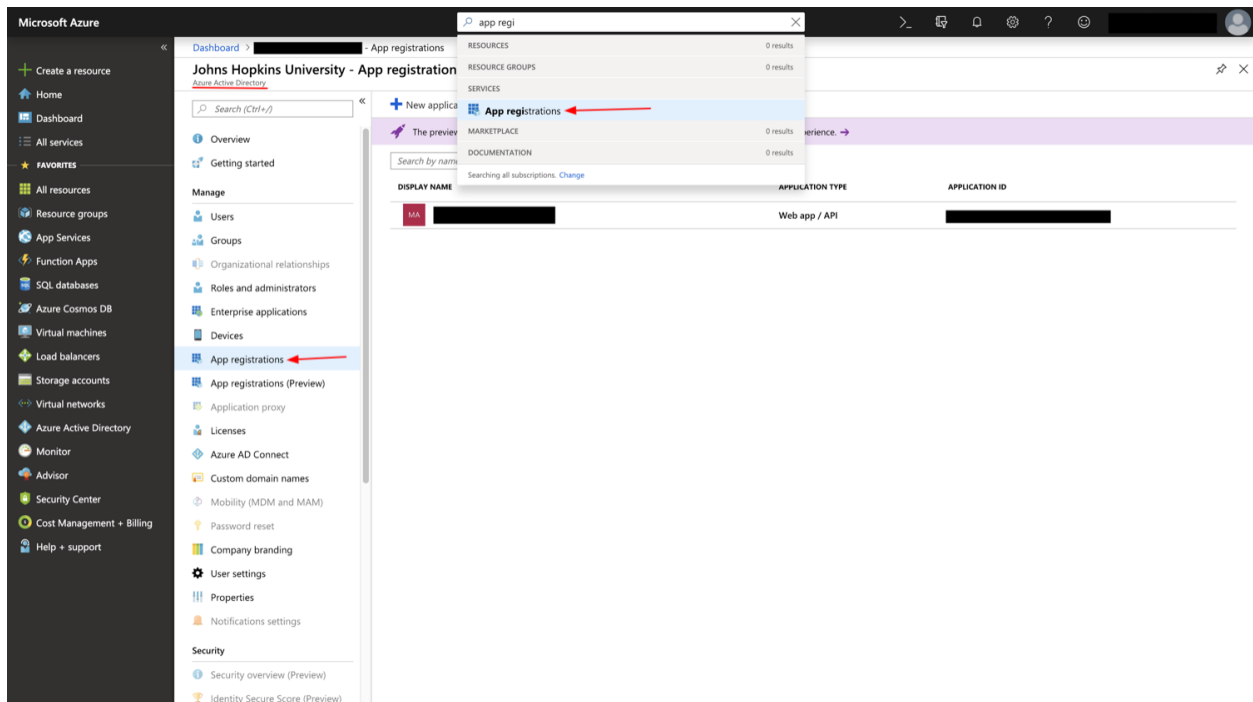


Fig. 5: *App Registrations* can be access by searching, or through choosing *App Registrations* under *Azure Active Directory*.

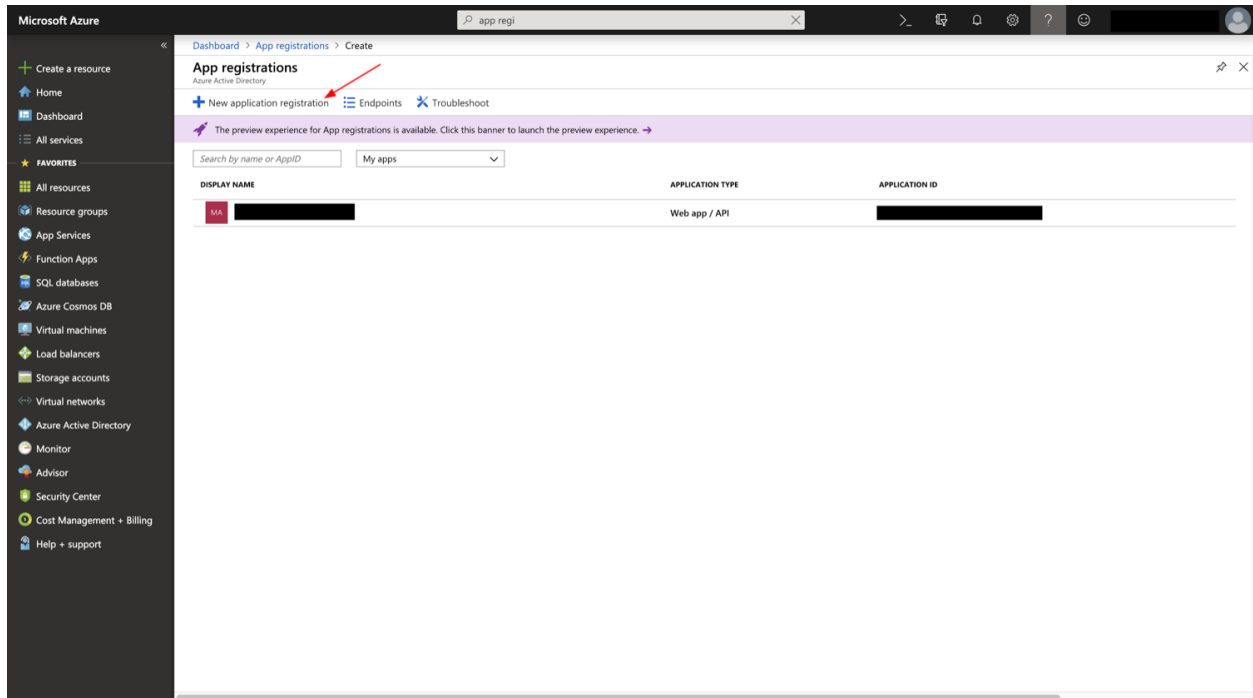


Fig. 6: The *New Application Registration* button will allow users to create a new application

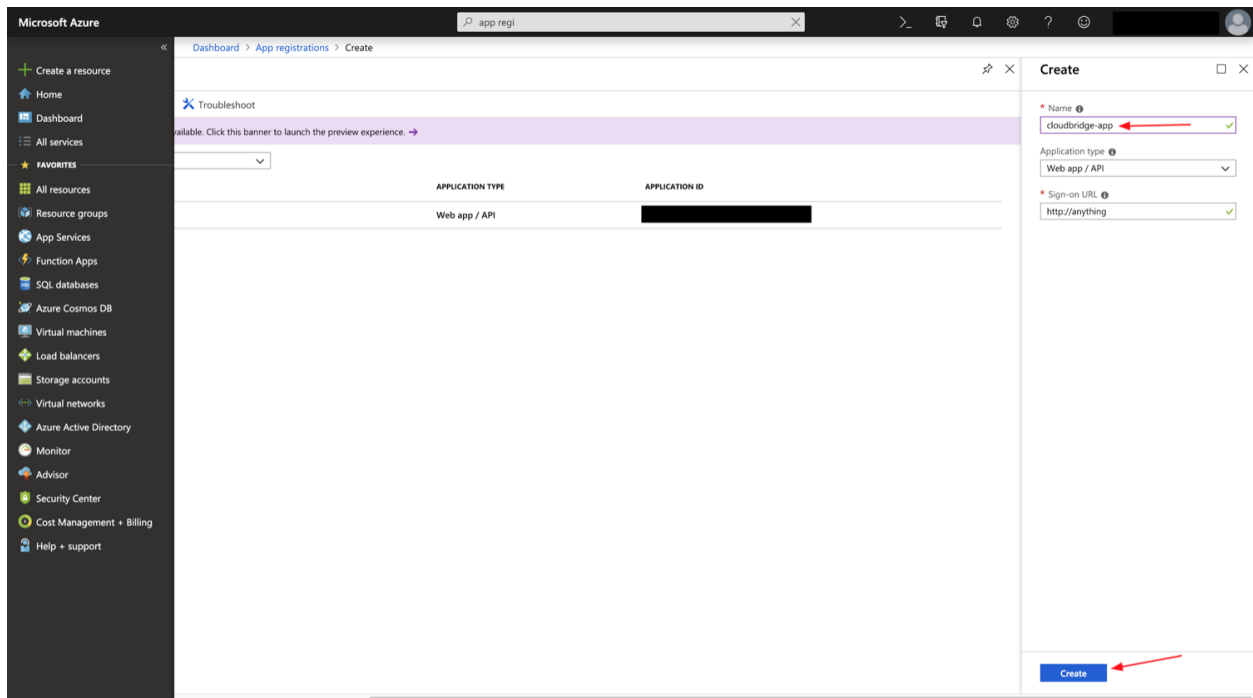


Fig. 7: The *Name* has to be unique within the subscription and will be used to identify the *Application* later on. The *Sign-on URL* can be any URL-looking string. It does not have to point towards anything.

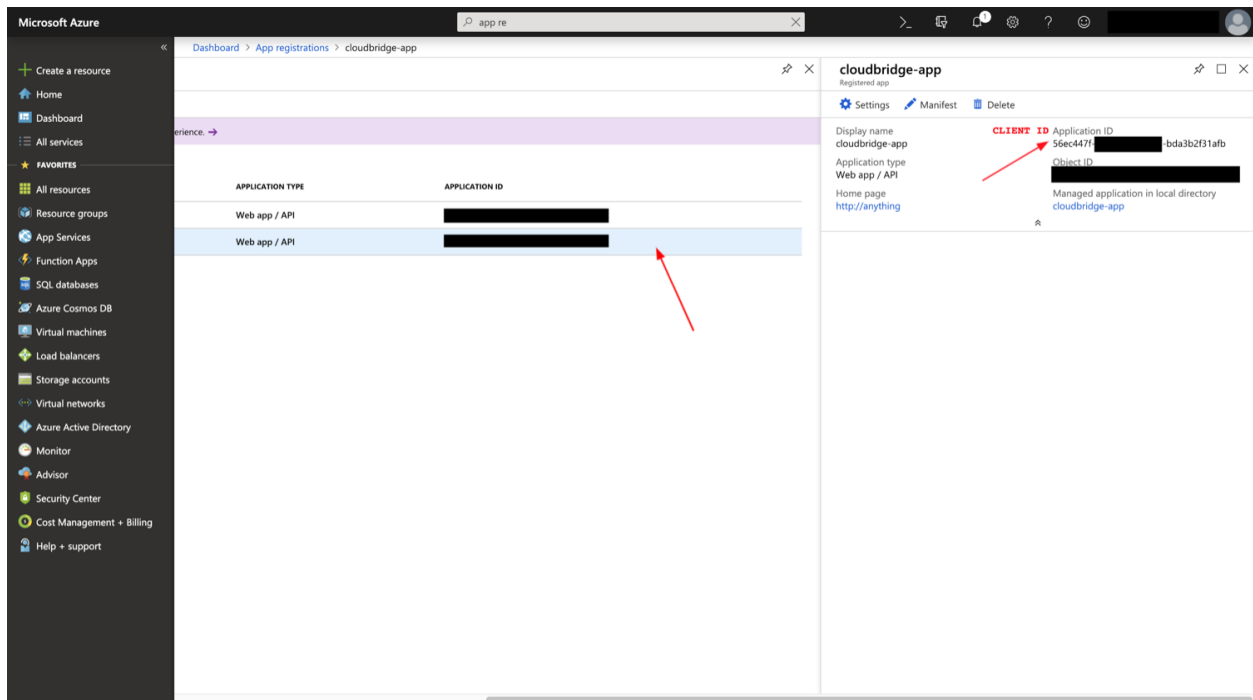


Fig. 8: After creating the application, one must select it, after which the *Application ID* will map to the *Client ID* in CloudBridge.

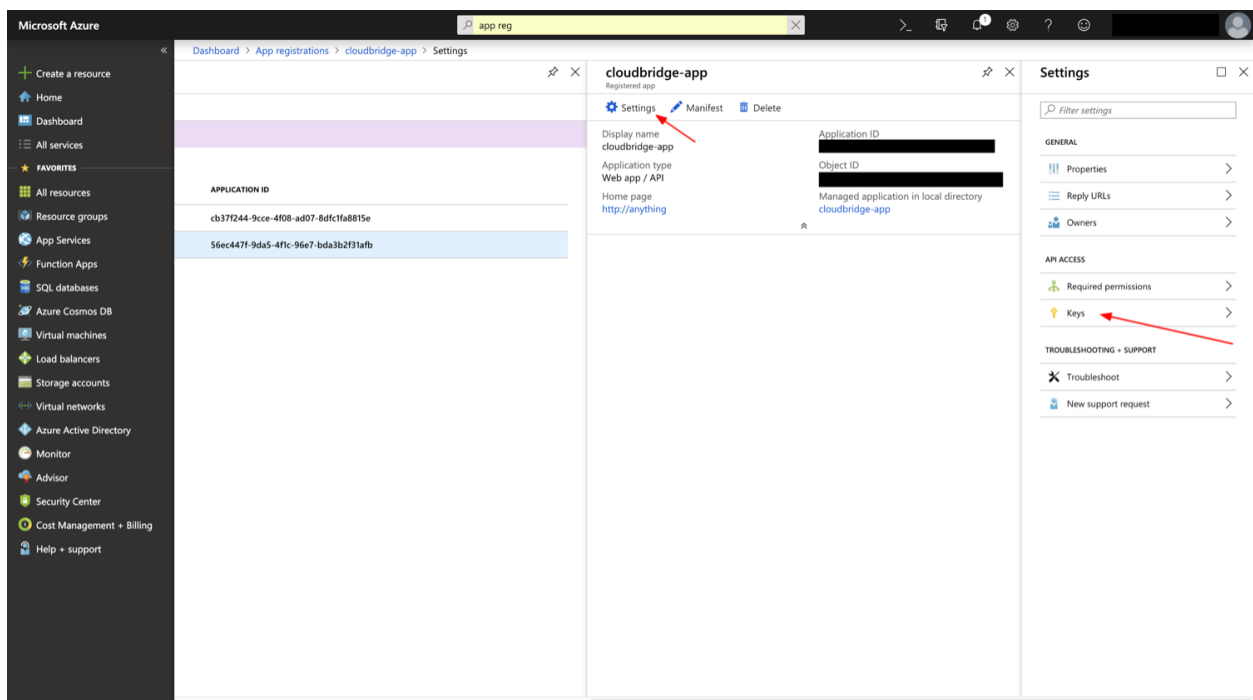


Fig. 9: In the application's *Settings* panel, under the *Keys* section, one will be able to create a new *Secret*.

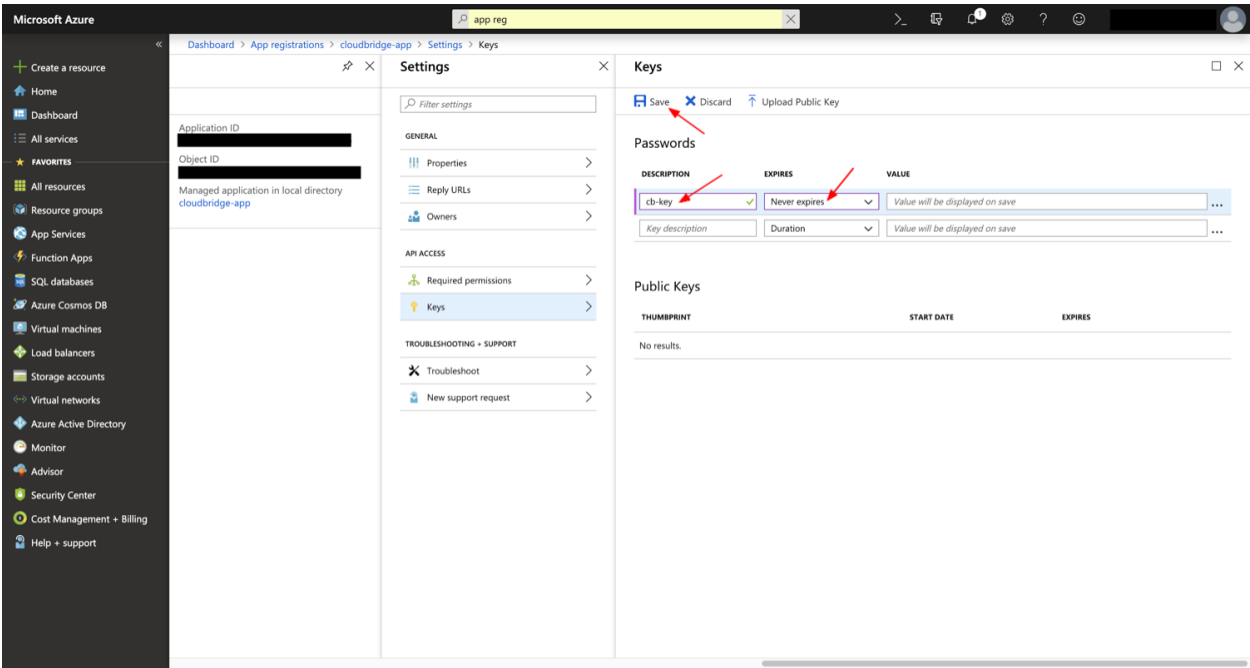


Fig. 10: Any name can be given to the key, and any expiration date, after which the *Save* button will generate the *Key* which will correspond to the *Secret* in CloudBridge.

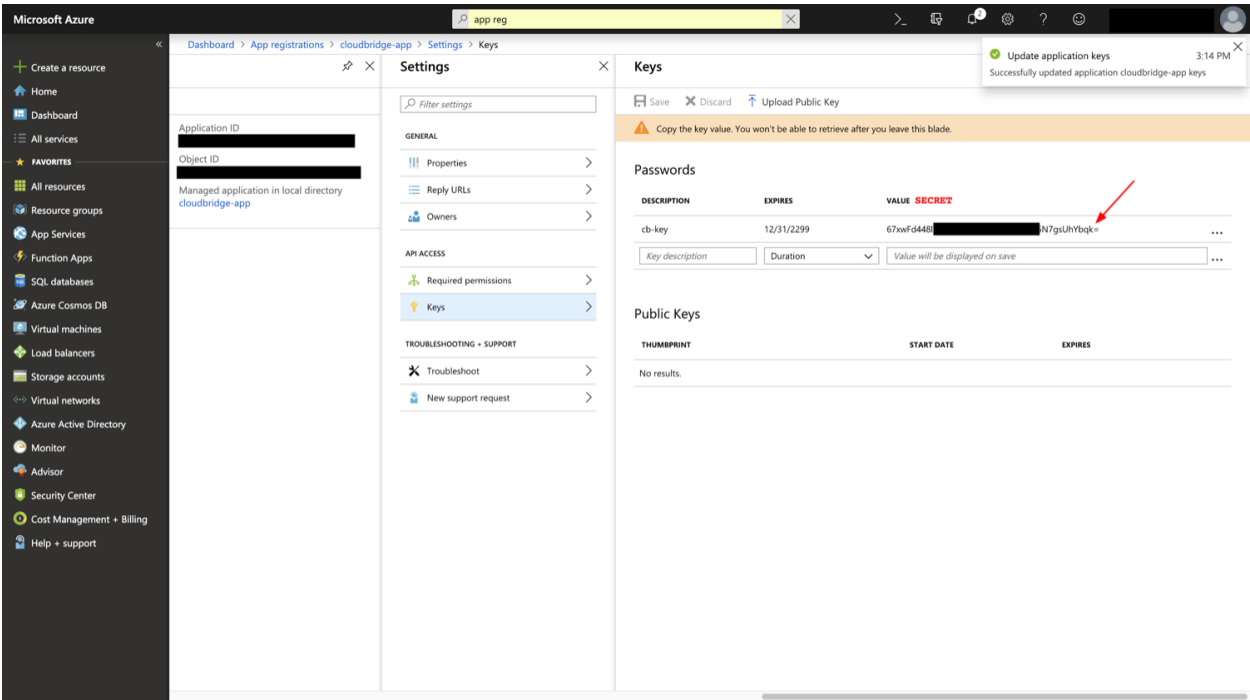


Fig. 11: The value of the key will correspond to the *Secret* in CloudBridge and needs to be saved at creation-time.

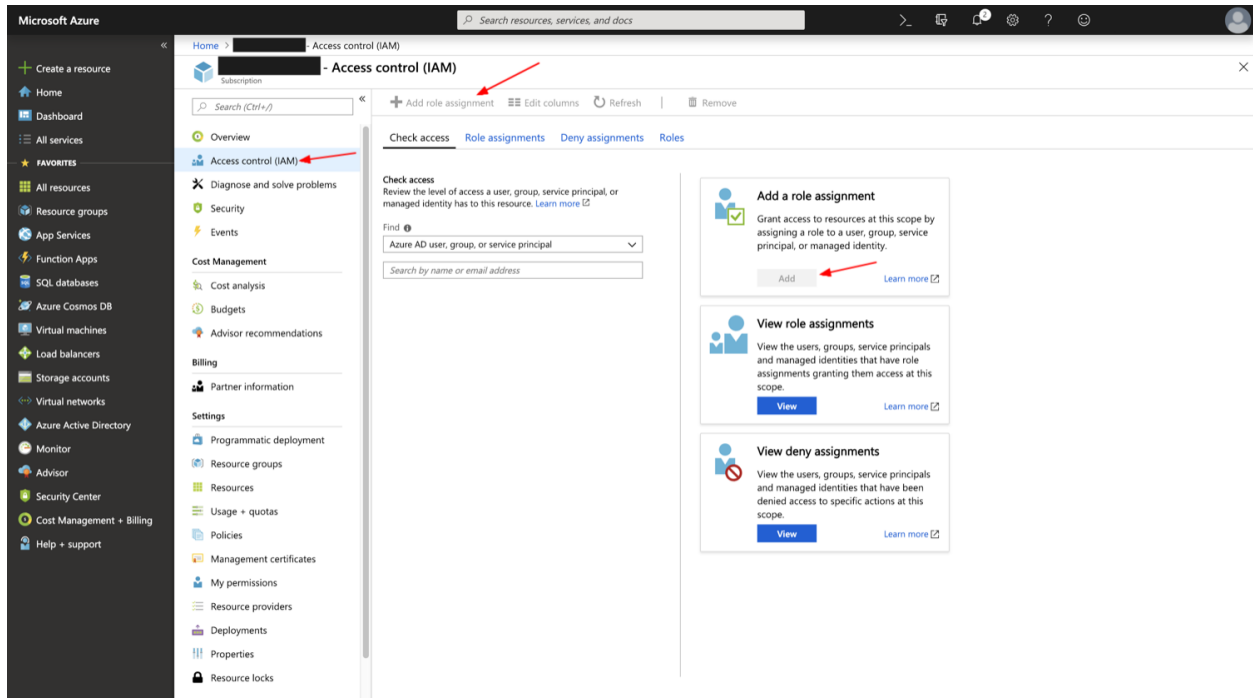


Fig. 12: Subscription-level access will allow the application to access resources from multiple resource groups.

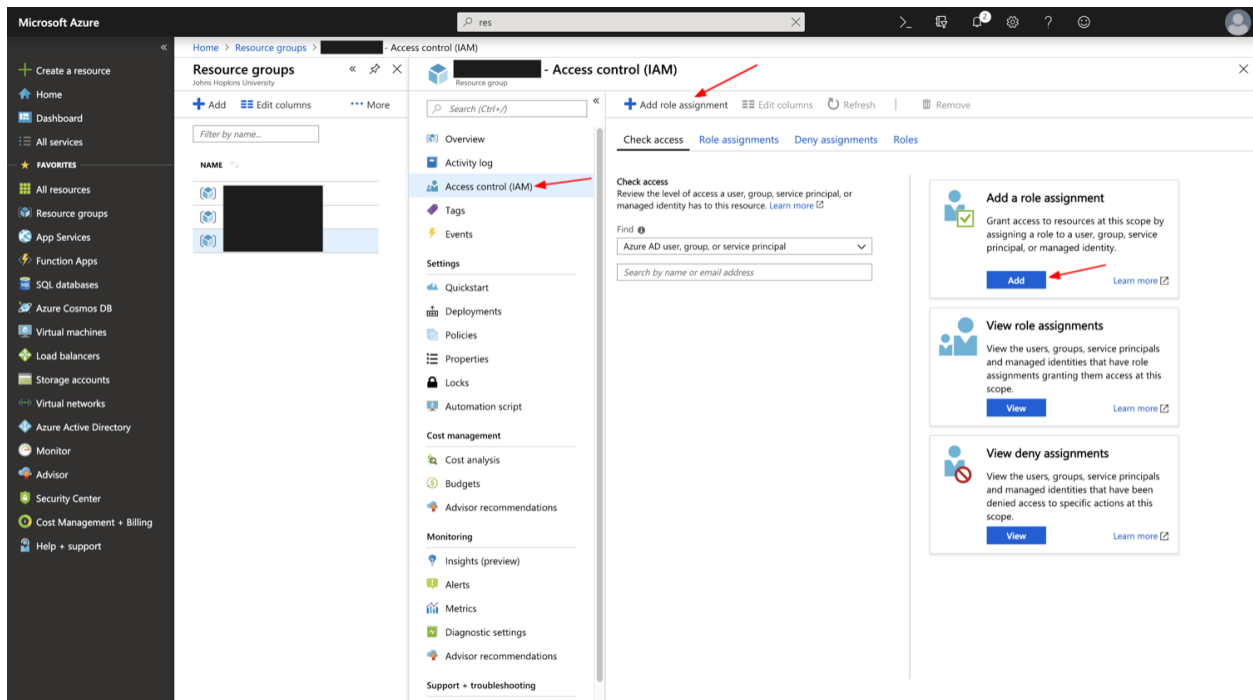


Fig. 13: When roles are set at the level of the Resource Group, one must specify this Resource Group as part of the credentials, as the application will not have enough permissions to create a Resource Group.

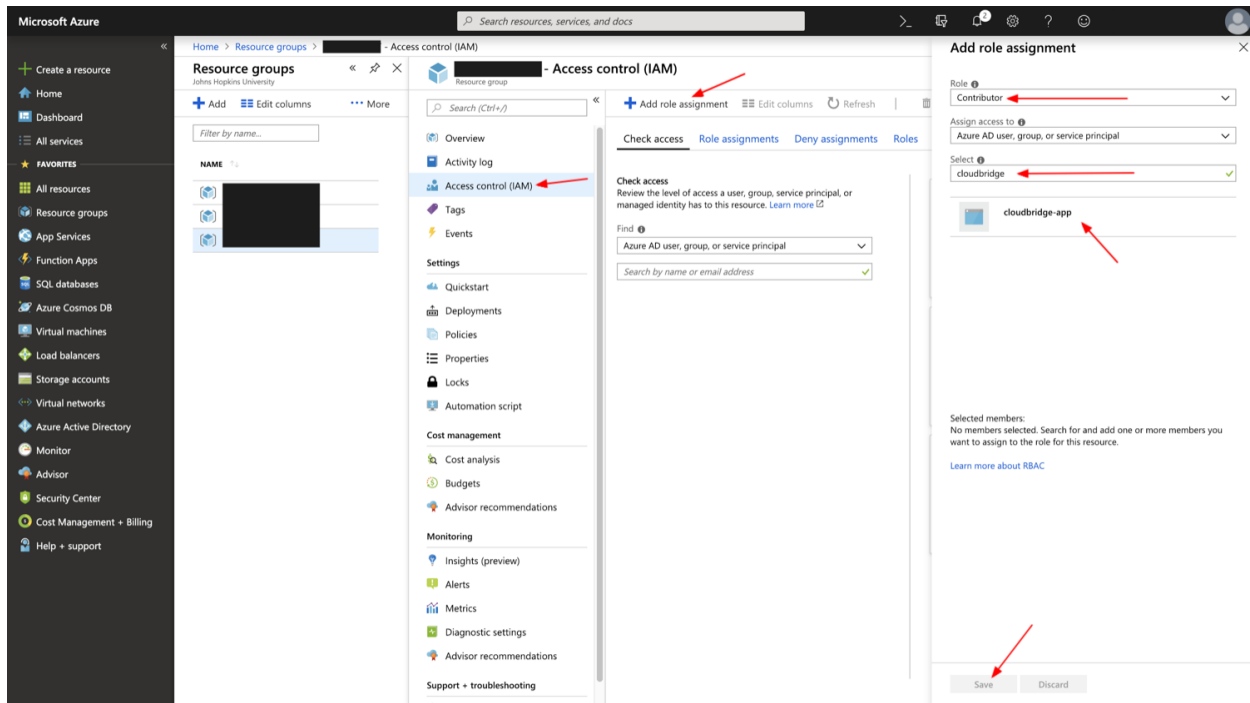


Fig. 14: Adding a role assignment to the application will give it appropriate permissions to manage resources.

Google Cloud Platform (GCP)

For Google Cloud Platform (GCP), create a service account followed by creating and downloading a key. Additional instructions are available at this link https://cloud.google.com/iam/docs/creating-managing-service-accounts#creating_a_service_account.

Start off by clicking on the *Create Service Account* button on the IAM & admin section of the Google Cloud Console: <https://console.cloud.google.com/iam-admin/serviceaccounts>.

Next, we provide a name for the service account and an informative description. Note that the supplied name is used to create an email address for the service account. Once created, this email cannot be changed.

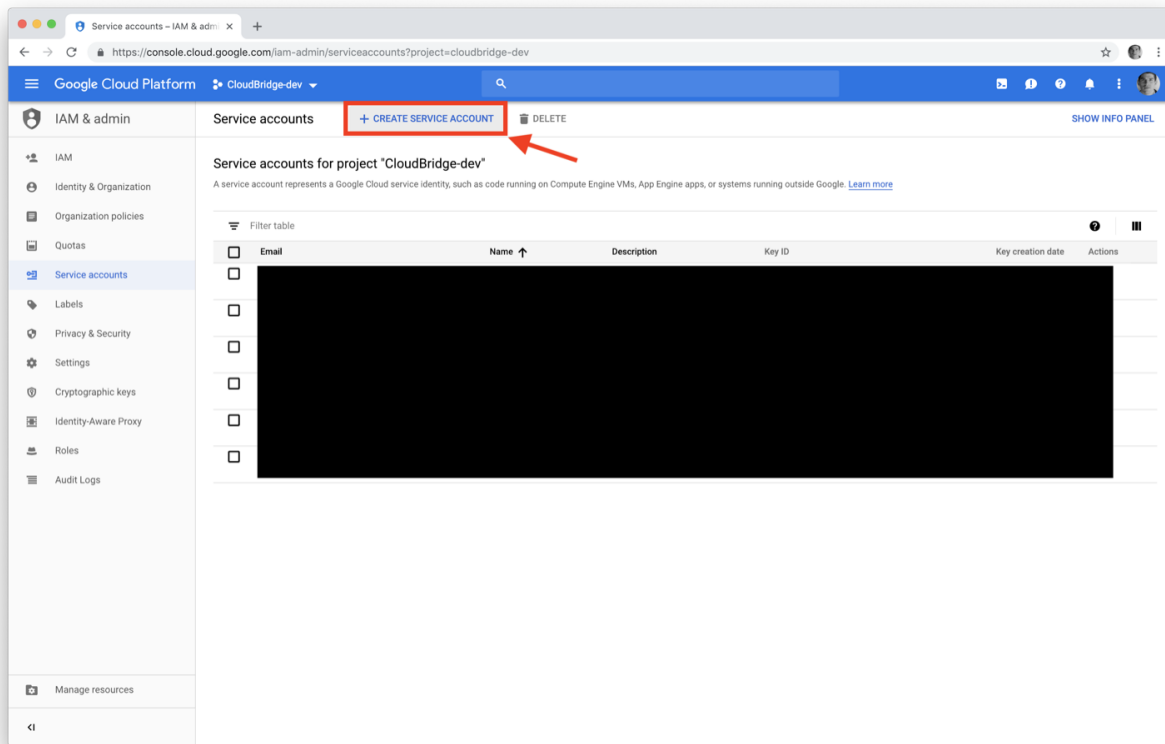
We then assign a role to the service account. Depending on what you will be doing with the service account, and CloudBridge, you can set granular access roles for the service account. The *Editor* role on the project is very broad and will allow you to exercise all of the CloudBridge capabilities.

After a service account has been created, we need to add a key to it.

Finally, choose the JSON format for the key when prompted. The file that is downloaded will be used with CloudBridge through the variables shown on the [Connection and Authentication Setup](#) page. Note that you may have multiple keys associated with the same service account.

The JSON credentials file will have a similar form to the example shown below, and can either be passed through an absolute path to the file, or through a variable containing the JSON dictionary itself.

```
{
  "type": "service_account",
  "project_id": "my-project",
  "private_key_id": "b12321312441245gerg245245g42c245g254t425",
  "private_key": "-----BEGIN PRIVATE KEY-----\nMIICWgIBAAKBgE1EJDPKM/2wck/\nCYZCS7F2cXoHXDBhXYtdeV+h70Nk+ABs6scAV\nnApYoobJAVpDeL+lutYAwbtscNz5K915DiNEkBf48LhfBWc5ea07OnC1OGC9\nkTHbM5iF5ak9N1LXT9wOTegKwtC9VjC\nng20QjFXAkLaDsFlAuICYaCBCXn1nUgNoYhaSEONwGnWIz376letXg\nmX+BALSPMFR/nhE6mbdmaL4OV1X8j8uf2VcrLfVFCZfhPu/TM5D6bVFYoQJBAJRHNKYU/csAB/\nNE/nzScJBv7PltOAoYpxbyFZblRwCv9mAn34382b0YBXbp3Giqvifs/\ntendlbBpAzzLm\n5gr8tZECQQCFZh4tNizeZZYUqkQxrxgqnnONey1hX7K+B1GyC6n2o26sE+I7cLi\nj\nn2kbnWoSFMAIdM2Hex\nWUSh2MRo8uB/74v123VD+VYAXTgtf3+JTzYbt1WK61TpuHQizEdRAkBjt8hL\nnBoNfJBUicXz0nuyzvyq10jREG+NjhRnAvFNbGS\nYbshW/jibYXwdj\nn-----END PRIVATE KEY-----",
}
```



Google Cloud Platform CloudBridge-dev

IAM & admin

Service accounts

Create service account

1 Service account details — 2 Grant this service account access to project (optional)

Service account details

Service account name
cloudbridge-service-acct

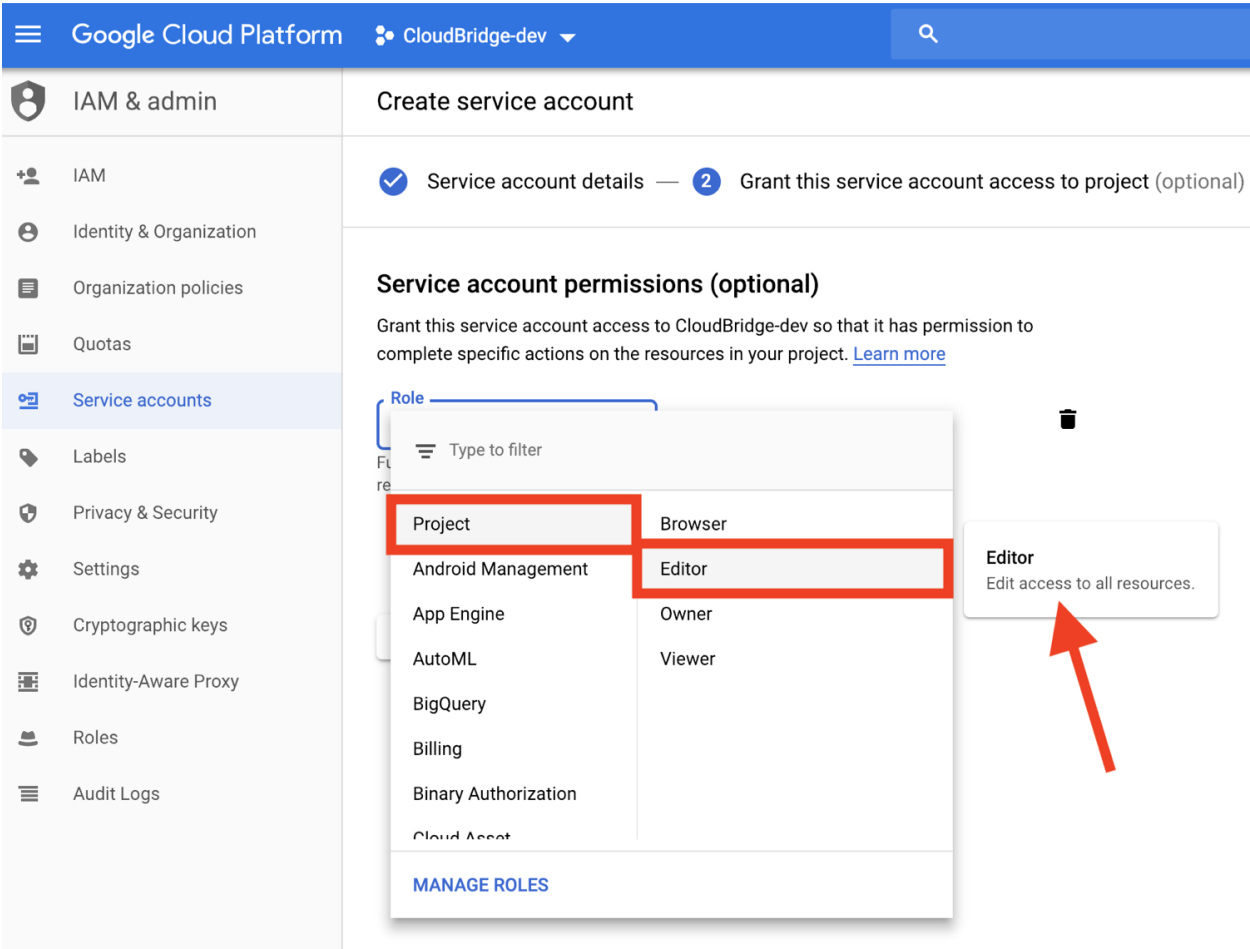
Display name for this service account

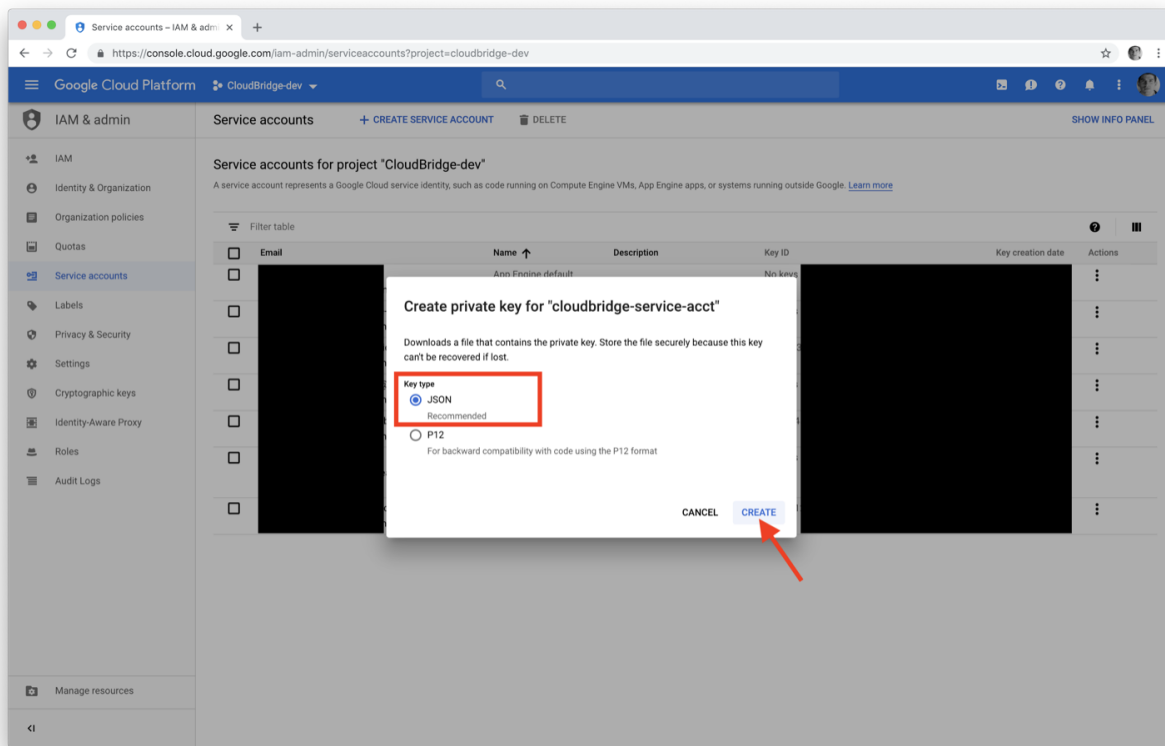
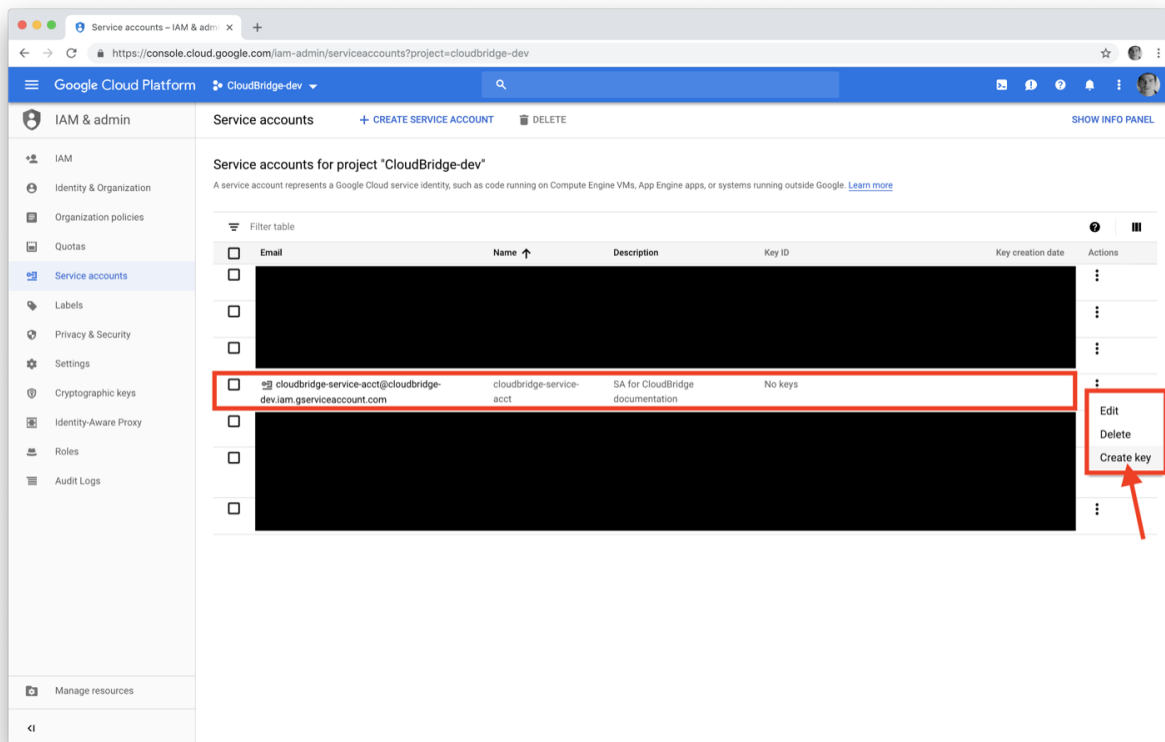
Service account ID
cloudbridge-service-acct @cloudbridge-dev.iam.gserviceaccount

Service account description
SA for CloudBridge documentation

Describe what this service account will do

CREATE CANCEL





(continued from previous page)

```

"client_email": "service-name@my-project.iam.gserviceaccount.com",
"client_id": "13451345134513451345",
"auth_uri": "https://accounts.google.com/o/oauth2/auth",
"token_uri": "https://oauth2.googleapis.com/token",
"auth_provider_x509_cert_url": "https://www.googleapis.com/oauth2/v1/certs",
"client_x509_cert_url": "https://www.googleapis.com/robot/v1/metadata/x509/service-
↪name%40my-project.iam.gserviceaccount.com"
}

```

To pass in the values as a properly escaped multi-line environment variable, use the following:

```
export GCP_SERVICE_CREDS_DICT=`cat ~/path/to/gcp-creds.json`
```

4.3.3 Setup

To initialize a connection to a cloud and get a provider object, you will need to provide the cloud's access credentials to CloudBridge. For more details on how to create and find these credentials, see the [Procuring Access Credentials](#) page. Note that you can selectively provide the credentials for any provider you want to use and do not have to provide credentials for all the providers. CloudBridge will consume the available credentials in one of following ways:

1. *Providing access credentials through a dictionary*
2. *Providing access credentials through environment variables*
3. *Providing access credentials in a CloudBridge config file*

Providing access credentials through a dictionary

You can initialize a simple config as follows. The key names are the same as the environment variables, in lower case. Note that the config dictionary will override environment values.

```

from cloudbridge.factory import CloudProviderFactory, ProviderList

## For AWS
config = {'aws_access_key' : '<your_access_key>',
          'aws_secret_key' : '<your_secret_key>'}
provider = CloudProviderFactory().create_provider(ProviderList.AWS, config)

## For Azure
config = {'azure_subscription_id': '<your_subscription_id>',
          'azure_client_id': '<your_client_id>',
          'azure_secret': '<your_secret>',
          'azure_tenant': '<your_tenant>',
          'azure_resource_group': '<your resource group>'}
provider = CloudProviderFactory().create_provider(ProviderList.AZURE, config)

## For GCP
config = {'gcp_service_creds_file': '<service_creds_file_name>.json'}
# Alternatively, we can supply a dictionary with the credentials values
# as the following:
gcp_creds = {
    "type": "service_account",

```

(continues on next page)

(continued from previous page)

```

    "project_id": "<project_name>",
    "private_key_id": "<private_key_id>",
    "private_key": "<private_key>",
    "client_email": "<client_email>",
    "client_id": "<client_id>",
    "auth_uri": "https://accounts.google.com/o/oauth2/auth",
    "token_uri": "https://oauth2.googleapis.com/token",
    "auth_provider_x509_cert_url": "https://www.googleapis.com/oauth2/v1/certs",
    "client_x509_cert_url": "https://www.googleapis.com/robot/v1/metadata/x509/
↪service-name%40my-project.iam.gserviceaccount.com"
}
config = {'gcp_service_creds_dict': gcp_creds}
# A third alternative is to use a GCP credentials object provided by the GCP python
# sdk. This is for advanced usage scenarios.
# e.g. credentials = AccessTokenCredentials(access_token, "MyAgent/1.0", None)
config = {'gcp_credentials_obj': credentials}
provider = CloudProviderFactory().create_provider(ProviderList.GCP, config)

## For OpenStack
config = {'os_username': '<your username>',
        'os_password': '<your password>',
        'os_auth_url': '<auth url>',
        'os_user_domain_name': '<user_domain_name>',
        'os_project_domain_id': '<project_domain_id>',
        'os_project_domain_name': '<project_domain_name>',
        'os_project_name': '<project_name>'}
provider = CloudProviderFactory().create_provider(ProviderList.OPENSTACK, config)

```

Some optional configuration values can only be provided through the config dictionary. These are listed below for each provider.

CloudBridge

Variable	Description
default_result_limit	Number of results that a <code>.list()</code> method should return. Default is 50.

AWS

Variable	Description
aws_region_name	Default region name. Default is <code>us-east-1</code> .
aws_zone_name	Default zone name. If not specified, defaults to first zone in default region. If specified, must match default region.
aws_session_token	Session key for your AWS account (if using temporary credentials).
ec2_endpoint_url	Endpoint to use. Default is <code>ec2.us-east-1.amazonaws.com</code> .
ec2_is_secure	True to use an SSL connection. Default is <code>True</code> .
ec2_validate_certs	Whether to use SSL certificate verification. Default is <code>False</code> .
s3_endpoint_url	Host connection endpoint. Default is <code>s3.amazonaws.com</code> .
s3_is_secure	True to use an SSL connection. Default is <code>True</code> .
s3_validate_certs	Whether to use SSL certificate verification. Default is <code>False</code> .

Azure

Variable	Description
<code>azure_access_token</code>	To sign requests to APIs protected by Azure.
<code>azure_public_key</code>	Storage table name where the key pairs are stored. Default is <code>cbcerts</code> .
<code>azure_region_name</code>	Default region to use for the current session. Default is <code>eastus</code> .
<code>aws_zone_name</code>	Default zone name. If not specified, defaults to first zone in default region. If specified, must match default region.
<code>azure_resource_group</code>	Azure resource group to use. Default is <code>cloudbridge</code> .
<code>azure_storage_account</code>	Azure storage account to use. Note that this value must be unique across Azure and all data in a given session is stored within the supplied storage account. Default <code>storacc</code> + first 6 chars of subscription id + first 6 chars of the supplied resource group.
<code>azure_vm_default_system_name</code>	System name for which supplied key pair will be placed.

GCP

Variable	Description
<code>gcp_region</code>	Default region to use for the current session. Default is <code>us-central1</code> .
<code>gcp_zone</code>	Default zone name. If not specified, defaults to first zone in default region. If specified, must match default region.
<code>gcp_vm_default_system_name</code>	System name for which supplied key pair will be placed.
<code>gcp_credentials_obj</code>	Provides obj to support advanced usage scenarios where an alternative authentication mechanism is required for GCP. This object replaces <code>GCP_SERVICE_CREDS_DICT</code> and is directly passed to the underlying python sdk's <code>build</code> method as <code>discovery.build('storage', 'v1', credentials=gcp_credentials_obj)</code> . You can pass in a manually constructed credentials object such as <code>creds = AccessTokenCredentials(access_token, "MyAgent/1.0", None)</code> . Refer to the GCP python sdk for available options.

OpenStack

Providing access credentials through environment variables

The following environment variables must be set, depending on the provider in use. For the meaning of the variables and default values, see the descriptions above.

AWS

Variable	Required?
<code>AWS_ACCESS_KEY</code>	✓
<code>AWS_SECRET_KEY</code>	✓

Azure

Note that managing resources in Azure requires a Resource Group. If a Resource Group is not provided as part of the configuration, CloudBridge will attempt to create a Resource Group using the given credentials. This operation will

happen with the client initialization, and requires a “contributor” or “owner” role.

Similarly, a Storage Account is required when managing some resources, such as key pairs and buckets. If a Storage Account name is not provided as part of the configuration, CloudBridge will attempt to create the Storage Account when initializing the relevant services. This operation similarly requires a “contributor” or “owner” role.

For more information on roles, see <https://docs.microsoft.com/en-us/azure/role-based-access-control/overview>.

Variable	Required?
AZURE_CLIENT_ID	✓
AZURE_SECRET	✓
AZURE_SUBSCRIPTION_ID	✓
AZURE_TENANT	✓
AZURE_PUBLIC_KEY_STORAGE_TABLE_NAME	
AZURE_REGION_NAME	
AZURE_ZONE_NAME	
AZURE_RESOURCE_GROUP	
AZURE_STORAGE_ACCOUNT	
AZURE_VM_DEFAULT_USER_NAME	

GCP

Variable	Required?
GCP_SERVICE_CREDS_DICT or GCP_SERVICE_CREDS_FILE	✓
GCP_ZONE_NAME	
GCP_PROJECT_NAME	
GCP_REGION_NAME	

OpenStack

Variable	Required?
OS_AUTH_URL	✓
OS_USERNAME	✓
OS_PASSWORD	✓
OS_PROJECT_NAME	✓
OS_REGION_NAME	✓
OS_ZONE_NAME	✓
OS_COMPUTE_ZONE_NAME	
OS_NETWORKING_ZONE_NAME	
OS_SECURITY_ZONE_NAME	
OS_STORAGE_ZONE_NAME	
NOVA_SERVICE_NAME	
OS_AUTH_TOKEN	
OS_COMPUTE_API_VERSION	
OS_VOLUME_API_VERSION	
OS_STORAGE_URL	
OS_PROJECT_DOMAIN_ID	
OS_PROJECT_DOMAIN_NAME	
OS_USER_DOMAIN_NAME	

Once the environment variables are set, you can create a connection as follows, replacing `ProviderList.AWS` with the desired provider (AZURE, GCP, or OPENSTACK):

```
from cloudbridge.factory import CloudProviderFactory, ProviderList

provider = CloudProviderFactory().create_provider(ProviderList.AWS, {})
```

Providing access credentials in a CloudBridge config file

CloudBridge can also read credentials from a file on your local file system. The file should be placed in one of two locations: `/etc/cloudbridge.ini` or `~/.cloudbridge`. Each set of credentials should be delineated with the provider ID (e.g., `openstack`, `aws`, `azure`, `gcp`) with the necessary credentials being supplied in YAML format. Note that only one set of credentials per cloud provider type can be supplied (i.e., via this method, it is not possible to provide credentials for two different OpenStack clouds).

```
[aws]
aws_access_key: access key
aws_secret_key: secret key

[azure]
azure_subscription_id: subscription id
azure_tenant: tenant
azure_client_id: client id
azure_secret: secret
azure_resource_group: resource group

[gcp]
gcp_service_creds_file: absolute path to credentials file

[openstack]
os_username: username
os_password: password
os_auth_url: auth url
os_user_domain_name: user domain name
os_project_domain_id: project domain id
os_project_domain_name: project domain name
os_project_name: project name
```

Once the file is created, you can create a connection as follows, replacing `ProviderList.AWS` with the desired provider (AZURE, GCP, or OPENSTACK):

```
from cloudbridge.factory import CloudProviderFactory, ProviderList

provider = CloudProviderFactory().create_provider(ProviderList.AWS, {})
```

General configuration variables

In addition to the provider specific configuration variables above, there are some general configuration environment variables that apply to CloudBridge as a whole.

Variable	Description
CB_DEBUG	Setting CB_DEBUG=True will cause detailed debug output to be printed for each provider (including HTTP traces).
CB_TEST_PROVIDER	Set this value to a valid ProviderList value such as aws, to limit tests to that provider only.
CB_DEFAULT_SUBNET_NAME	Can be used for a subnet that will be considered the ‘default’ by the library. This default will be used only in cases there is no subnet marked as the default by the provider.
CB_DEFAULT_NETWORK_NAME	Can be used for a network that will be considered the ‘default’ by the library. This default will be used only in cases there is no network marked as the default by the provider.
CB_DEFAULT_IPV4_RANGE	Default IPv4 range when creating networks if one is not provided. This value is also used in tests.
CB_DEFAULT_SUBNET_IPV4_RANGE	Default IPv4 range used by CloudBridge if one is not specified by the user. Tests do not respect this variable.

4.3.4 Launching instances

Before being able to run below commands, you will need a `provider` object (see [this page](#)).

Common launch data

Before launching an instance, you need to decide what image to launch as well as what type of instance. We will create those objects here. The specified image ID is a base Ubuntu image on AWS so feel free to change it as desired. For instance type, we’re going to let CloudBridge figure out what’s the appropriate name on a given provider for an instance with at least 2 CPUs and 4 GB RAM.

```
img = provider.compute.images.get('ami-759bc50a') # Ubuntu 16.04 on AWS
vm_type = sorted([t for t in provider.compute.vm_types
                  if t.vcpus >= 2 and t.ram >= 4],
                 key=lambda x: x.vcpus*x.ram) [0]
```

In addition, CloudBridge instances must be launched into a private subnet. While it is possible to create complex network configurations as shown in the [Private networking](#) section, if you don’t particularly care in which subnet the instance is launched, CloudBridge provides a convenience function to quickly obtain a default subnet for use.

```
subnet = provider.networking.subnets.get_or_create_default()
```

When launching an instance, you can also specify several optional arguments such as the firewall (aka security group), a key pair, or instance user data. To allow you to connect to the launched instances, we will also supply those parameters (note that we’re making an assumption here these resources exist; if you don’t have those resources under your account, take a look at the [Getting Started](#) guide).

```
kp = provider.security.key_pairs.find(name='cloudbridge-intro') [0]
fw = provider.security.vm_firewalls.list() [0]
```

Launch an instance

Once we have all the desired pieces, we’ll use them to launch an instance. Note that the instance is launched in the provider’s default region and zone, and can be overridden by changing the provider config.

```
inst = provider.compute.instances.create(
    label='cloudbridge-vpc', image=img, vm_type=vm_type,
    subnet=subnet, key_pair=kp, vm_firewalls=[fw])
```

Private networking

Private networking gives you control over the networking setup for your instance(s) and is considered the preferred method for launching instances. To launch an instance with an explicit private network, you can create a custom network and make sure it has internet connectivity. You can then launch into that subnet.

```
net = self.provider.networking.networks.create(
    label='my-network', cidr_block='10.0.0.0/16')
sn = net.subnets.create(label='my-subnet', cidr_block='10.0.0.0/28')
# make sure subnet has internet access
router = self.provider.networking.routers.create(label='my-router', network=net)
router.attach_subnet(sn)
gateway = net.gateways.get_or_create()
router.attach_gateway(gateway)

inst = provider.compute.instances.create(
    label='cloudbridge-vpc', image=img, vm_type=vm_type,
    subnet=sn, key_pair=kp, vm_firewalls=[fw])
```

For more information on how to create and setup a private network, take a look at [Networking](#).

Block device mapping

Optionally, you may want to provide a block device mapping at launch, specifying volume or ephemeral storage mappings for the instance. While volumes can also be attached and mapped after instance boot using the volume service, specifying block device mappings at launch time is especially useful when it is necessary to resize the root volume.

The code below demonstrates how to resize the root volume. For more information, refer to [LaunchConfig](#).

```
lc = provider.compute.instances.create_launch_config()
lc.add_volume_device(source=img, size=11, is_root=True)
inst = provider.compute.instances.create(
    label='cloudbridge-bdm', image=img, vm_type=vm_type,
    launch_config=lc, key_pair=kp, vm_firewalls=[fw],
    subnet=subnet)
```

where `img` is the Image object to use for the root volume.

After launch

After an instance has launched, you can access its properties:

```
# Wait until ready
inst.wait_till_ready() # This is a blocking call
inst.state
# 'running'
```

Depending on the provider's networking setup, it may be necessary to explicitly assign a floating IP address to your instance. This can be done as follows:

```
# Create a new floating IP address
fip = provider.networking.floating_ips.create()
# Assign the desired IP to the instance
inst.add_floating_ip(fip)
inst.refresh()
inst.public_ips
# ['149.165.168.143']
```

4.3.5 Private networking

Private networking gives you control over the networking setup for your instance(s) and is considered the preferred method for launching instances. Also, providers these days are increasingly requiring use of private networks. All CloudBridge deployed VMs must be deployed into a particular subnet.

If you do not explicitly specify a private network to use when launching an instance, CloudBridge will attempt to use a default one. A ‘default’ network is one tagged as such by the native API. If such tag or functionality does not exist, CloudBridge will look for one with a predefined label (by default, called ‘cloudbridge-net’, which can be overridden with environment variable CB_DEFAULT_NETWORK_LABEL).

Once a VM is deployed, CloudBridge’s networking capabilities must address several common scenarios.

1. Allowing internet access from a launched VM

In the simplest scenario, a user may simply want to launch an instance and allow the instance to access the internet.

2. Allowing internet access to a launched VM

Alternatively, the user may want to allow the instance to be contactable from the internet. In a more complex scenario, a user may want to deploy VMs into several subnets, and deploy a gateway, jump host, or bastion host to access other VMs which are not directly connected to the internet. In the latter scenario, the gateway/jump host/bastion host will need to be contactable over the internet.

3. Secure access between subnets for n-tier applications

In this third scenario, a multi-tier app may be deployed into several subnets depending on their tier. For example, consider the following scenario:

- Tier 1/Subnet 1 - Web Server needs to be externally accessible over the internet. However, in this particular scenario, the web server itself does not need access to the internet.
- Tier 2/Subnet 2 - Application Server must only be able to communicate with the database server in Subnet 3, and receive communication from the Web Server in Subnet 1. However, we assume a special case here where the application server needs to access the internet.
- Tier 3/Subnet 3 - Database Server must only be able to receive incoming traffic from Tier 2, but must not be able to make outgoing traffic outside of its subnet.

At present, CloudBridge does not provide support for this scenario, primarily because OpenStack’s FwaaS (Firewall-as-a-Service) is not widely available.

1. Allowing internet access from a launched VM

Creating a private network is a simple, one-line command but appropriately connecting it so that it has uniform internet access across all providers is a multi-step process: (1) create a network; (2) create a subnet within this network; (3) create a router; (4) attach the router to the subnet; and (5) attach the router to the internet gateway.

When creating a network, we need to set an address pool. Any subsequent subnets you create must have a CIDR block that falls within the parent network's CIDR block. CloudBridge also defines a default IPv4 network range in `BaseNetwork.CB_DEFAULT_IPV4RANGE`. Below, we'll create a subnet starting from the beginning of the block and allow up to 16 IP addresses within a subnet (/28).

```
net = provider.networking.networks.create(
    label='my-network', cidr_block='10.0.0.0/16')
sn = net.subnets.create(label='my-subnet',
    cidr_block='10.0.0.0/28')
router = provider.networking.routers.create(label='my-router', network=net)
router.attach_subnet(sn)
gateway = net.gateways.get_or_create()
router.attach_gateway(gateway)
```

2. Allowing internet access to a launched VM

The additional step that's required here is to assign a floating IP to the VM:

```
net = provider.networking.networks.create(
    label='my-network', cidr_block='10.0.0.0/16')
sn = net.subnets.create(label='my-subnet', cidr_block='10.0.0.0/28')

vm = provider.compute.instances.create(label='my-inst', subnet=sn, ...)

router = provider.networking.routers.create(label='my-router', network=net)
router.attach_subnet(sn)
gateway = net.gateways.get_or_create()
router.attach_gateway(gateway)

fip = provider.networking.floating_ips.create()
vm.add_floating_ip(fip)
```

Retrieve an existing private network

If you already have existing networks, we can query for it:

```
provider.networking.networks.list() # Find a desired network ID
net = provider.networking.networks.get('desired network ID')
```

4.3.6 DNS Service

The DNS service provides a cloud-independent way to create and edit dns zones and records.

1. Creating a DNS zone

At the top-level, dns records are organized into zones. A zone is a portion of the dns namespace that's managed by a particular organization or group.

```
host_zone = provider.dns.host_zones.create("cloudve.org.", "admin@cloudve.org")
```

2. Create a DNS record

Once a zone is created, you can create records as required.

```
host_zone = provider.dns.host_zones.find(name="cloudve.org.")
# create an A record
rec1 = host_zone.records.create("mysubdomain.cloudve.org.", DnsRecordType.A, data='10.
↳1.1.1')
# create a wildcard record
rec2 = host_zone.records.create("*.cloudve.org.", DnsRecordType.A, data='10.1.1.2')
# create an MX record
MX_DATA = ['10 mx1.hello.com.', '20 mx2.hello.com.']
test_rec2 = host_zone.records.create("cloudve.org.", DnsRecordType.MX, data=MX_DATA,
↳ttl=300)
```

4.3.7 Object states and lifecycles

Overview

Some objects, namely, Instances, Volumes, Snapshots and Images, have a specific life-cycle and a set of states they can be in. For example, an Instance may be in state `RUNNING` and a volume maybe in state `AVAILABLE`. These provider specific states are mapped to a common vocabulary of states in CloudBridge.

In addition, it is common to want to wait for objects to reach a particular state. For example, wait for an instance to transition from `PENDING`->`RUNNING`. To facilitate this, all such objects in CloudBridge implement the `ObjectLifeCycleMixin`. Each object with a lifecycle has a state property, a refresh method and a `wait_for` method.

The state property will return the object's current state, and the `refresh()` method can be used to requery the server and update the object's state. The `wait_for` method can be used to wait for the object to transition to a desired state or set of states.

There is also a convenience method named `wait_till_ready()`, which will wait for the object to reach a ready-to-use state. A ready-to-use state would mean that the object has been successfully created and can be interacted with, and is not in an error state. Since this can be tedious to get right, the `wait_till_ready()` method encapsulates this behaviour. For example - in the case of an Instance, `wait_till_ready()` will internally call the `wait_for` method as follows:

```
self.wait_for(
    [InstanceState.RUNNING],
    terminal_states=[InstanceState.DELETED, InstanceState.ERROR],
    timeout=timeout,
    interval=interval)
```

This would cause the `wait_for` method to repeatedly call `refresh()` till the object's state reaches `RUNNING`. It will raise a `WaitStateException` if the timeout expires, or the object reaches a terminal state, such as `DELETED` or `ERROR`, in which case it is no longer reasonable to wait for the object to reach a running state.

Informational states and actionable states

As in the `wait_for` example above, some states are purely informational, and some states are actionable. Informational states are meant to provide information to the end-user, and should generally not be used by program logic to take actions. For example, it would be incorrect to write a `wait_for` function as follows:

```
wait_for([InstanceState.PENDING], ...)
```


This is because the PENDING state is fleeting and may occur too fast to be reliably detected by the client. In contrast, a state such as RUNNING is more stable and program logic can reasonably take actions based on such states. In the discussion that follows, we will specifically differentiate between informational and actionable states.

Instance states and lifecycle

The following states are defined for a CloudBridge Instance.

State	Category	Description
UNKNOWN	actionable	Instance state is unknown. This means that the instance does not exist or CloudBridge does not know how to map this state.
PENDING	informational	Instance is pending
CONFIGURING	informational	Instance is being reconfigured in some way and may not be usable.
RUNNING	actionable	Instance is running.
REBOOTING	informational	Instance is rebooting.
DELETED	actionable	Instance is deleted. No further operations possible.
STOPPED	actionable	Instance is stopped. Instance can be resumed.
ERROR	actionable	Instance is in an error state. No further operations possible.

The lifecycle diagram is as follows:

Actionable states are shown in blue and informational states in cyan. Note that any state could potentially transition into an ERROR state. An Instance may initially be in an UNKNOWN state and transition into a PENDING state on launch. Similarly, it may transition into an UNKNOWN state after TERMINATION and the object no longer exists. More rarely, an instance may transition into an UNKNOWN state if CloudBridge does not know how to map the state reported by the cloud provider. Therefore, when writing a wait_for method, these potential transitions should be taken into account.

Volume states and lifecycle

The following states are defined for a CloudBridge Volume.

State	Cate- gory	Description
UN- KNOWN	action- able	Volume state is unknown. This means that the volume does not exist or CloudBridge does not know how to map this state.
CREAT- ING	informa- tional	Volume is pending
CONFIG- URING	informa- tional	Volume is being reconfigured in some way and may not be usable.
AVAIL- ABLE	action- able	Volume is unattached and available for use.
IN_USE	informa- tional	Volume is attached to an instance and in-use.
DELETED	action- able	Volume has been deleted. No further operations possible.
ERROR	action- able	Volume is in an error state. No further operations possible.

The lifecycle diagram is as follows:

Actionable states are shown in blue and informational states in cyan. Note that any state could potentially transition into an ERROR state. A Volume may initially be in an UNKNOWN state and transition into a CREATING state when created anew. Similarly, it may transition into an UNKNOWN state after DELETED and the object no longer exists. More rarely, a volume may transition into an UNKNOWN state if CloudBridge does not know how to map the state reported by the cloud provider. A Volume will typically transition through a CONFIGURING stage before going to an IN_USE stage and vice versa.

Snapshot states and lifecycle

The following states are defined for a CloudBridge Snapshot.

State	Cate- gory	Description
UN- KNOWN	action- able	Snapshot state is unknown. This means that the snapshot does not exist or CloudBridge does not know how to map this state.
PENDING	informa- tional	Snapshot is pending
CONFIG- URING	informa- tional	Snapshot is being reconfigured in some way and may not be usable.
AVAIL- ABLE	action- able	Snapshot is ready.
ERROR	action- able	Snapshot is in an error state. No further operations possible.

The lifecycle diagram is as follows:

Actionable states are shown in blue and informational states in cyan. Note that any state could potentially transition into an ERROR state. A Snapshot may initially be in an UNKNOWN state and transition into a PENDING state when created anew. Similarly, it may transition into an UNKNOWN state after deleted and the object no longer exists. More rarely, a snapshot may transition into an UNKNOWN state if CloudBridge does not know how to map the state reported by the cloud provider.

Image states and lifecycle

The following states are defined for a CloudBridge Image.

State	Category	Description
UNKNOWN	actionable	Image state is unknown. This means that the Image does not exist or CloudBridge does not know how to map this state.
PENDING	informational	Image is pending
AVAILABLE	actionable	Image is ready.
ERROR	actionable	Image is in an error state. No further operations possible.

The lifecycle diagram is as follows:

Actionable states are shown in blue and informational states in cyan. Note that any state could potentially transition into an ERROR state. An Image may initially be in an UNKNOWN state and transition into a PENDING state when created anew. Similarly, it may transition into an UNKNOWN state after deleted and the image no longer exists. More rarely, an Image may transition into an UNKNOWN state if CloudBridge does not know how to map the state reported by the cloud provider.

4.3.8 Paging and iteration

Overview

Most provider services have `list()` methods, and all list methods accept a `limit` parameter which specifies the maximum number of results to return. If a limit is not specified, CloudBridge will default to the global configuration variable `default_result_limit`, which can be modified through the provider config.

Since the returned result list may have more records available, CloudBridge will always return a `ResultList` object to assist with paging through additional results. A `ResultList` extends the standard `list` and the following example illustrates how to fetch additional records.

Example:

```
# get first page of results
rl = provider.compute.instances.list(limit=50)
for result in rl:
    print("Instance Data: {}".format(result))
if rl.supports_total:
    print("Total results: {}".format(rl.total_results))
else:
    print("Total records unknown,"
          "but has more data?: {}".format(rl.is_truncated))

# Page to next set of results
if (rl.is_truncated)
    rl = provider.compute.instances.list(limit=100,
                                         marker=rl.marker)
```

To ease development, CloudBridge also provides standard Python iterators that will page the results in for you automatically. Therefore, when you need to iterate through all available objects, the following shorthand is recommended:

Example:

```
# Iterate through all results
for instance in provider.compute.instances:
    print("Instance Data: {0}".format(instance))
```

4.3.9 Working with block storage

To add persistent storage to your cloud environments, you would use block storage devices, namely volumes and volume snapshots. A volume is attached to an instance and mounted as a file system for use by an application. A volume snapshot is a point-in-time snapshot of a volume that can be shared with other cloud users. Before a snapshot can be used, it is necessary to create a volume from it.

Volume storage

Operations, such as creating a new volume and listing the existing ones, are performed via the *VolumeService*. To start, let's create a 1GB volume.

```
vol = provider.storage.volumes.create('cloudbridge-vol', 1)
vol.wait_till_ready()
provider.storage.volumes.list()
```

Next, let's attach the volume to a running instance as device `/dev/sdh`:

```
vol.attach('i-dbf37022', '/dev/sdh') vol.refresh() vol.state # 'in-use'
```

Once attached, from within the instance, it is necessary to create a file system on the new volume and mount it.

Once you wish to detach a volume from an instance, it is necessary to unmount the file system from within the instance and detach it. The volume can then be attached to a different instance with all the data on it preserved.

```
vol.detach()
vol.refresh()
vol.state
# 'available'
```

Snapshot storage

A volume snapshot is created from an existing volume. Note that it may take a long time for a snapshot to become ready, particularly on AWS.

```
snap = vol.create_snapshot('cloudbridge-snap',
                           'A demo snapshot created via CloudBridge.')
snap.wait_till_ready()
snap.state
# 'available'
```

In order to make use of a snapshot, it is necessary to create a volume from it:

```
vol = provider.storage.volumes.create(
    'cloudbridge-snap-vol', 1, 'us-east-1e', snapshot=snap)
```

The newly created volume behaves just like any other volume and can be attached to an instance for use.

4.3.10 Working with object storage

Object storage provides a simple way to store and retrieve large amounts of unstructured data over HTTP. Object Storage is also referred to as Blob (Binary Large Object) Storage by Azure, and Simple Storage Service (S3) by Amazon.

Typically, you would store your objects within a Bucket, as it is known in AWS and GCP. A Bucket is also called a Container in OpenStack and Azure. In CloudBridge, we use the term Bucket.

Storing objects in a bucket

To store an object within a bucket, we need to first create a bucket or retrieve an existing bucket.

```
bucket = provider.storage.buckets.create('my-bucket')
bucket.objects.list()
```

Next, let's upload some data to this bucket. To efficiently upload a file, simple use the `upload_from_file` method.

```
obj = bucket.objects.create('my-data.txt')
obj.upload_from_file('/path/to/myfile.txt')
```

You can also use the `upload()` function to upload from an in memory stream. Note that, an object you create with `objects.create()` doesn't actually get persisted until you upload some content.

To locate and download this uploaded file again, you can do the following:

```
bucket = provider.storage.buckets.find(name='my-bucket')[0]
obj = bucket.objects.find(name='my-data.txt')[0]
print("Size: {0}, Modified: {1}".format(obj.size, obj.last_modified))
with open('/tmp/myfile.txt', 'wb') as f:
    obj.save_content(f)
```

Using tokens for authentication

Some providers may support using temporary credentials with a session token, in which case you will be able to access a particular bucket by using that session token.

```
provider = CloudProviderFactory().create_provider(
    ProviderList.AWS,
    {'aws_access_key': 'ACCESS_KEY',
     'aws_secret_key': 'SECRET_KEY',
     'aws_session_token': 'MY_SESSION_TOKEN'})
```

```
provider = CloudProviderFactory().create_provider(
    ProviderList.OPENSTACK,
    {'os_storage_url': 'SWIFT_STORAGE_URL',
     'os_auth_token': 'MY_SESSION_TOKEN'})
```

Once a provider is obtained, you can access the container as usual:

```
bucket = provider.storage.buckets.get(container)
obj = bucket.objects.create('my_object.txt')
obj.upload_from_file(source)
```

4.3.11 Resource Types and Dashboard Mapping

Cross-Platform Concepts

Given CloudBridge's goal to work uniformly across cloud providers, some compromises were necessary in order to bridge the many differences between providers' resources and features. Notably, in order to create a robust and conceptually consistent cross-cloud library, resources were separated into *labeled* and *unlabeled resources*, and were given three main properties: *ID*, *name*, and *label*.

The *ID* corresponds to a unique identifier that can be reliably used to reference a resource. Users can safely use an ID knowing that it will always point to the same resource. All resources have an *ID* property, thus making it the recommended property for reliably identifying a resource.

The *label* property, conversely, is a modifiable value that does not need to be unique. Unlike the *name* property, it is not used to identify a particular resource, but rather label a resource for easier distinction. Only labeled resources have the *label* property, and these resources require a *label* parameter be set at creation time.

The *name* property corresponds to an unchangeable and unique designation for a particular resource. This property is meant to be, in some ways, a more human-readable identifier. Thus, when no conceptually comparable property exists for a given resource in a particular provider, the *ID* is returned instead, as is the case for all OpenStack and some AWS resources. Given the discrepancy between providers, using the *name* property is not advisable for cross-cloud usage of the library. Labeled resources will use the label given at creation as a prefix to the set *name*, when this property is separable from the *ID* as is the case in Azure and some AWS resources. Finally, unlabeled resources will always support a *name*, and some unlabeled resources will require a *name* parameter at creation. Below is a list of all resources classified by whether they support a *label* property.

Labeled Resources	Unlabeled Resources
Instance	Key Pair
MachineImage	Bucket
Network	Bucket Object
Subnet	FloatingIP
Router	Internet Gateway
Volume	VMFirewall Rule
Snapshot	
VMFirewall	

Properties per Resource per Provider

For each provider, we documented the mapping of CloudBridge resources and properties to provider objects, as well as some useful dashboard navigation. These sections will thus present summary tables delineating the different types of CloudBridge resources, as well as present some design decisions made to preserve consistency across providers:

Detailed AWS Type and Resource Mappings

AWS Dashboard

AWS has a particular dashboard as resources are found within different services. The following table lists the dashboard location of each resource, and the below screenshot shows how the switch between the various services.

Instance	EC2
MachineImage (Private)	EC2
Volume	EC2
Snapshot	EC2
VMFirewall	EC2
FloatingIP	EC2
KeyPair	EC2
VMFirewallRule	EC2
Network	VPC
Subnet	VPC
Router	VPC
InternetGateway	VPC
Bucket	S2
BucketObject	S2

AWS - Labeled Resources

Labeled Resource	AWS Resource Type	CB ID	CB Name	CB Label
AWSInstance	Instance	Instance ID	Instance ID	tag:Name
AWSMachineImage	AMI	AMI ID	AMI Name	tag:Name
AWSNetwork	VPC	VPC ID	VPC ID	tag:Name
AWSSubnet	Subnet	Subnet ID	Subnet ID	tag:Name
AWSRouter	Route Table	Route Table ID	Route Table ID	tag:Name
AWSVolume	Volume	Volume ID	Volume ID	tag:Name
AWSSnapshot	Snapshot	Snapshot ID	Snapshot ID	tag:Name
AWSVMFirewall	Security Group	Group ID	Group Name	tag:Name

The resources listed above are labeled, they thus have both the *name* and *label* properties in CloudBridge. These resources require a mandatory *label* parameter at creation. For all labeled resources, the *label* property in AWS maps to the tag with *key:Name*. However, unlike in Azure where all resources have names, only some AWS resources have an unchangeable name by which to identify them. Thus, for most AWS resources, the *name* property maps to the ID, in order to preserve the concept of names being a unique identifier, even if they are not easily readable in this context. For resources that do support naming in AWS, the *name* will be generated from the *label* given at creation, consisting of up to 55 characters from the label, followed by a UUID. The label property can subsequently be changed, but the name property will be set at creation and remain unchanged. Finally, labeled resources support a *label* parameter for the *find* method in their corresponding services. The below screenshots will help map these properties to AWS objects in the web portal.

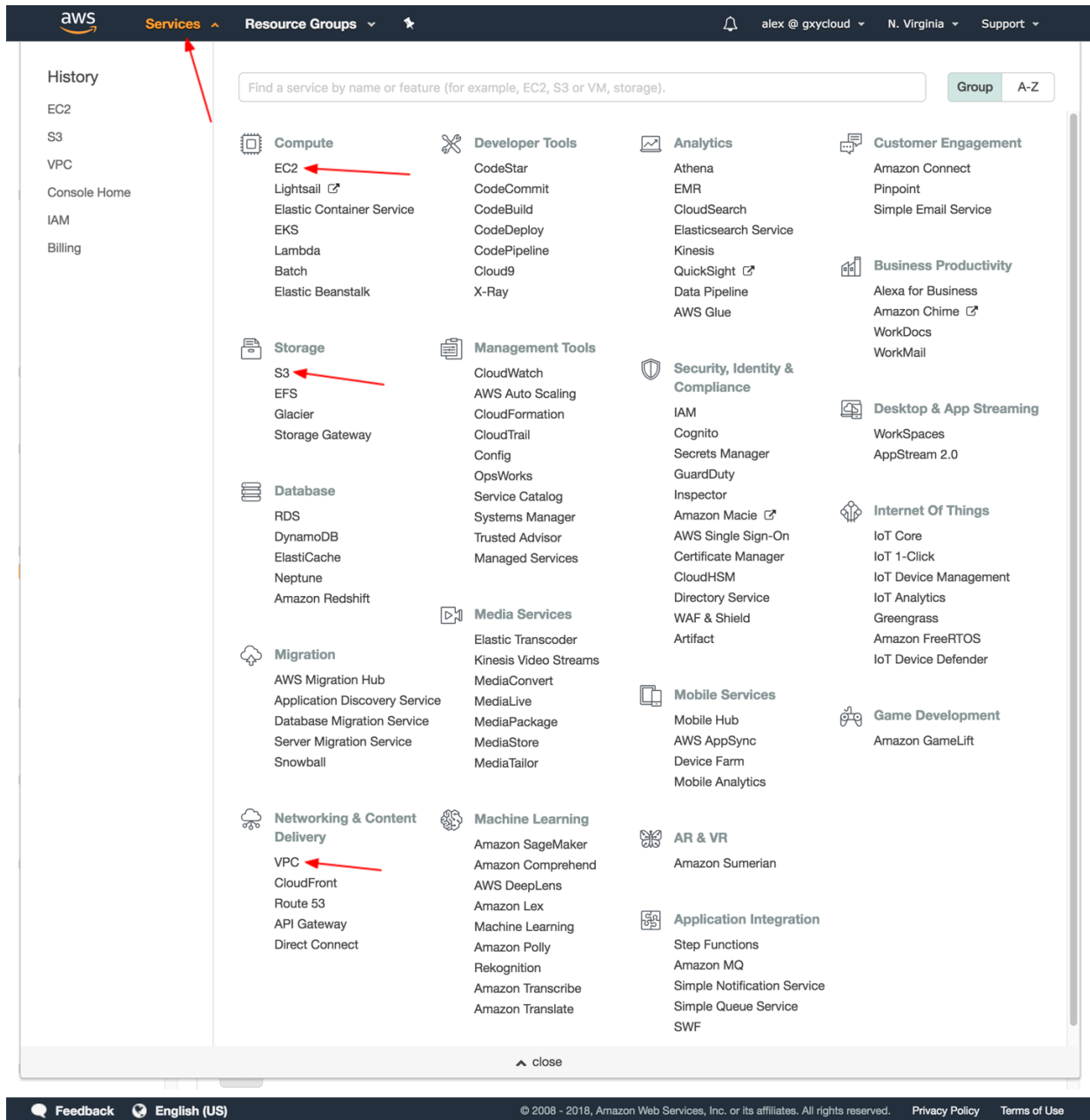


Fig. 15: Resources in AWS are separated into three dashboards depending on the type of service handling the resources

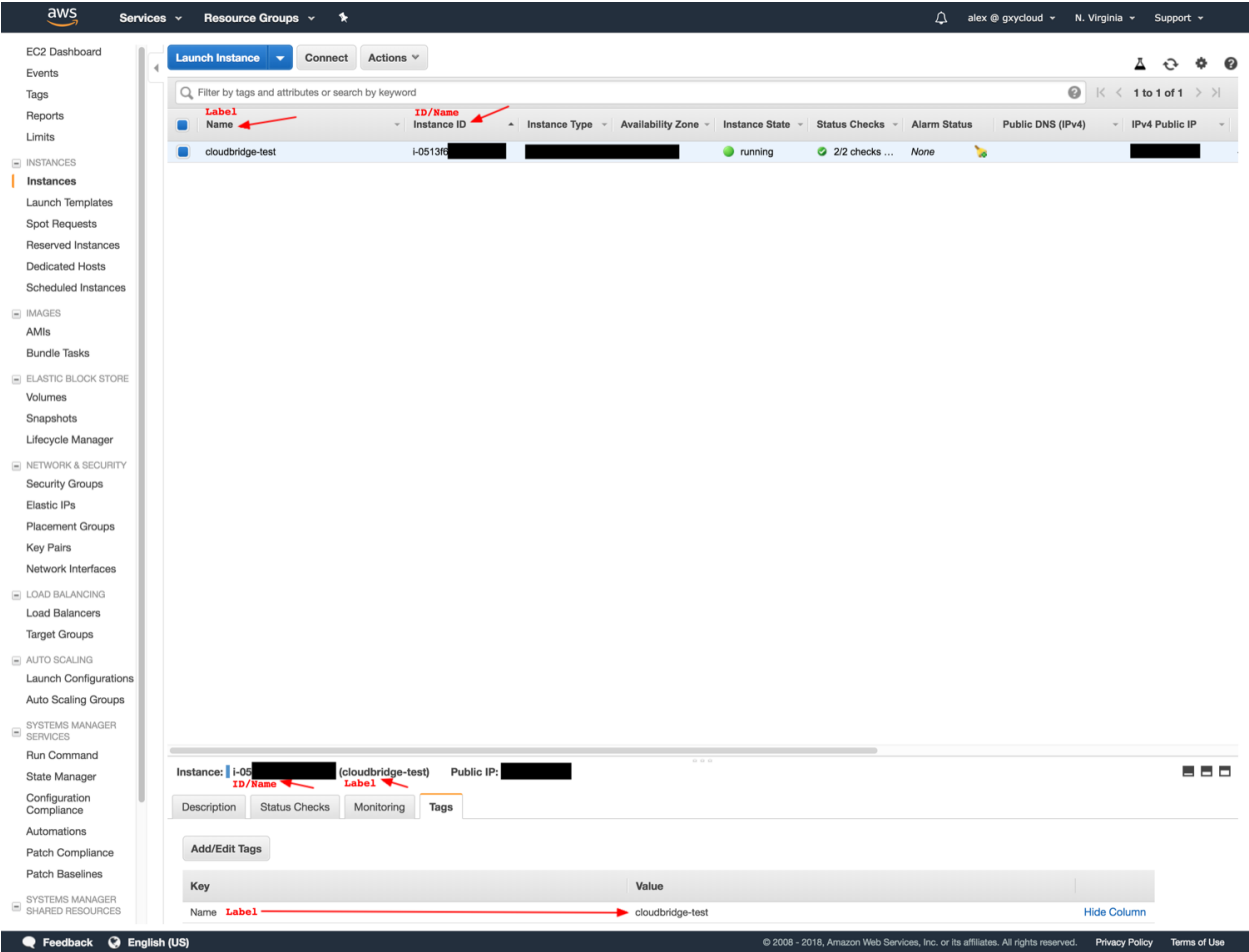


Fig. 16: The CloudBridge *name* and *ID* properties map to the unchangeable resource ID in AWS when the resource does not allow for an unchangeable name. The *label* property maps to the tag with key 'Name' for all resources in AWS. By default, this label will appear in the first column.

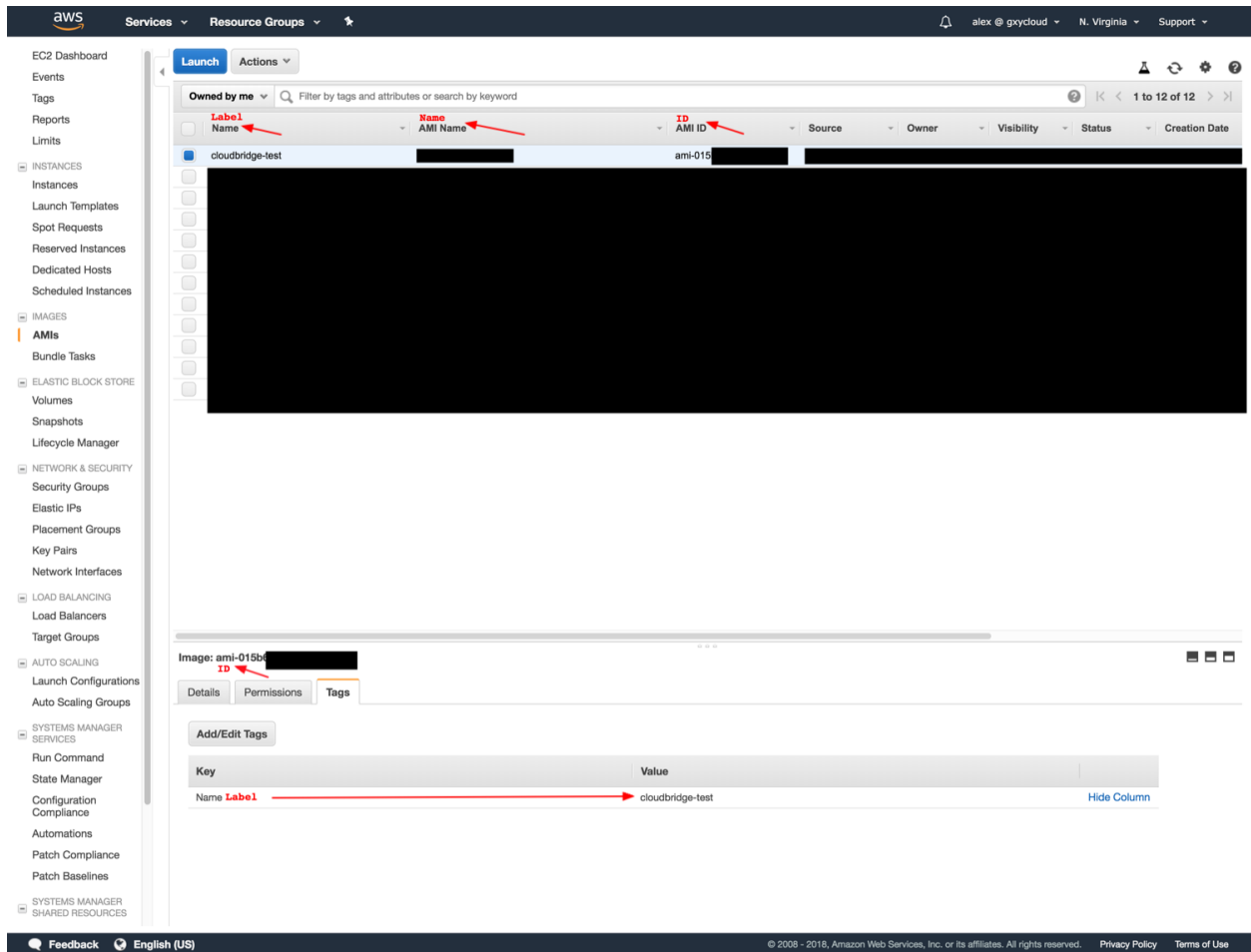


Fig. 17: When an AWS resource allows for an unchangeable name, the CloudBridge *ID* property maps to the Resource ID, while the *Name* property maps to the Resource Name. The *label* property maps to the tag with key ‘Name’ for all resources in AWS. By default, this label will appear in the first column.

AWS - Unlabeled Resources

Unlabeled Resource	Re- source	AWS Resource Type	CB ID	CB Name	CB Label
AWSKeyPair		Key Pair	Name	Name	•
AWSBucket		Bucket	Name	Name	•
AWSBucketObject		Bucket Object	Key	Key	•

The resources listed above are unlabeled. They thus only have the *name* property in CloudBridge. These resources require a mandatory *name* parameter at creation, which will directly map to the unchangeable *name* property. Additionally, for these resources, the *ID* property also maps to the *name* in AWS, as these resources don't have an *ID* in the traditional sense and can be located by name. Finally, unlabeled resources support a *name* parameter for the *find* method in their corresponding services.

AWS - Special Unlabeled Resources

Unlabeled Resource	Re- source	AWS Resource Type	CB ID	CB Name	CB Label
AWSFloatingIP		Elastic IP	ID	[public_ip]	•
AWSInternetGateway		Internet Gateway	ID	tag:Name	•
AWSVMFirewallRule		Network Security Rules	ID	Generated: [direction]- [protocol]- [from_port]- [to_port]-[cidr]- [fw_id]	•

While these resources are similarly unlabeled, they do not follow the same general rules as the ones listed above. Firstly, they differ by the fact that they take neither a *name* nor a *label* parameter at creation. Moreover, each of them has other special properties.

The FloatingIP resource has a traditional resource ID, but instead of a traditional name, its *name* property maps to its Public IP. Moreover, the corresponding *find* method for Floating IPs can thus help find a resource by *Public IP Address*.

In terms of the gateway, given that gateways are not their own objects in other providers, we do not treat them like labeled resources in AWS although they could support labels. Thus, the internet gateway create method does not take a name parameter, and the *name* property is set automatically to a default value. Note that since this value is stored in the tag with key Name, the AWS dashboard does allow for its modification, although that is not encouraged as the default name is expected for the *get_or_create* method.

Finally, Firewall Rules in AWS differ from traditional unlabeled resources by the fact that they do not take a *name* parameter at creation, and the *name* property is automatically generated from the rule's properties, as shown above. These rules can be found within each Firewall (i.e. Security Group) in the AWS EC2 portal, and will not have any name in the AWS dashboard

aws Services Resource Groups alex @ gxycld Global Support

Welcome to Amazon S3. Create new buckets or select an existing bucket to view and configure properties. Documentation

Amazon S3 Discover the new console Quick tips

Search for buckets

+ Create bucket Delete bucket Empty bucket

20 Buckets 9 Public 1 Regions

ID/Name Bucket name	Access	Region	Date created
	Not public *	US East (N. Virginia)	
	Not public *	US East (N. Virginia)	
	Not public *	US East (N. Virginia)	
	Public	US East (N. Virginia)	
	Not public *	US East (N. Virginia)	
	Not public *	US East (N. Virginia)	
	Public	US East (N. Virginia)	
	Public	US East (N. Virginia)	
	Not public *	US East (N. Virginia)	
	Public	US East (N. Virginia)	
	Public	US East (N. Virginia)	
	Not public *	US East (N. Virginia)	
	Not public *	US East (N. Virginia)	
	Not public *	US East (N. Virginia)	
	Public	US East (N. Virginia)	
	Not public *	US East (N. Virginia)	
	Public	US East (N. Virginia)	
	Public	US East (N. Virginia)	
	Public	US East (N. Virginia)	

Fig. 18: Buckets can be found in the Amazon S3 portal. BucketObjects are contained within each Bucket.

Detailed Azure Type and Resource Mappings

Azure - Labeled Resources

Labeled CloudBridge Resource	Azure Resource Type	CB ID	CB Name	CB Label
AzureInstance	Virtual Machine	ID	Name	tag:Label
AzureMachineImage (Private) AzureMachineImage (Marketplace Image)	Image Virtual-MachineImage	ID ID	Name Name URN	tag:Label tag:Label URN
AzureNetwork	Virtual Network	ID	Name	tag:Label
AzureSubnet	Subnet	ID	Network-Name/SubnetName	Net-network:tag:SubnetLabel_SubnetName
AzureRouter	Route Table	ID	Name	tag:Label
AzureVolume	Disk	ID	Name	tag:Label
AzureSnapshot	Snapshot	ID	Name	tag:Label
AzureVMFirewall	Network security group	ID	Name	tag:Label

The resources listed above are labeled, they thus have both the *name* and *label* properties in CloudBridge. These resources require a mandatory *label* parameter at creation. The *label* will then be used to create the *name*, which will consist of up to 55 characters from the label, followed by a UUID. The label property can subsequently be changed, but the name property will remain unchanged, as it is part of the ID. Finally, labeled resources support a *label* parameter for the *find* method in their corresponding services. The below screenshots will help map these properties to Azure objects in the web portal. Additionally, although Azure Security Groups are not associated with a specific network, such an association is done in CloudBridge, due to its necessity in AWS. As such, the VMFirewall creation method requires a *network* parameter and the association is accomplished in OpenStack through a tag with the key *network_id*.

Two labeled resources are exceptions to the general trends presented above, namely public images (i.e. Azure Marketplace Images) and subnets.

These public images can be found in the Azure Marketplace, and cannot be found on a user's dashboard. A Marketplace Image can be passed either by URN, or by public ID, and does not need to be linked to a user. While all Marketplace images will not be listed by the *find* or *list* methods at the moment, a pre-set list of popular images is built into CloudBridge for that purpose. However, one can choose to list all Marketplace Images using the *list_marketplace_images* function in the azure client. Specifically, this can be done as follows:

```
# List all images
# Note that in September 2018, around 10 minutes of wall time were required
# to fetch the entire list
provider.azure_client.list_marketplace_images()
# List all images published by Canonical
provider.azure_client.list_marketplace_images(publisher='Canonical')
# List all Ubuntu images
provider.azure_client.list_marketplace_images(publisher='Canonical',
                                              offer='UbuntuServer')
# List all Ubuntu 16.04 images
provider.azure_client.list_marketplace_images(publisher='Canonical',
                                              offer='UbuntuServer',
                                              sku='16.04.0-LTS')
# The ID of the listed object can then be used to retrieve an instance
img = provider.compute.images.get
    ('/Subscriptions/{subscriptionID}/Providers/Microsoft.Compute/\
Locations/{regionName}/Publishers/Canonical/ArtifactTypes/VMImage\
/Offer/UbuntuServer/Skus/16.04.0-LTS/Versions/16.04.201808140')
```

(continues on next page)

Microsoft Azure

Home > All resources

All resources

+ Add Edit columns Refresh Assign tags Delete

Subscriptions: AnswerALS Azure Sponsorship – Don't see a subscription? [Open Directory](#) + [Subscription settings](#)

Filter by name... All resource groups All types All locations All tags No grouping

1 of 153 items selected Show hidden types

Tags will show up by default

Name	TYPE	RESOURCE GROUP	LOCATION	TAGS	LABEL (TAG)
cb_bilkattch-0e6c21d4	Snapshot		East US	1	
cb_bilkattch-0e6c21d4	Virtual network		East US	1	
cb_bilkattch-a3de573c	Virtual network		East US	1	Label: cb_bilkattch
cb_bilkattch-a3de573c	Snapshot		East US	1	
cb_bilkattch-a637f88b	Virtual network		East US	1	
cb_bilkattch-a637f88b	Snapshot		East US	1	
cb_bilkattch-a637f88b	Disk		East US	1	
cb_bilkattch-a637f88b	Disk		East US	1	
cb_bilkattch-a8998ce8	Virtual network		East US	1	
cb_bilkattch-a8998ce8	Snapshot		East US	1	
cb_crudimage-13c00b	Virtual network		East US	1	
cb_crudimage-a7b0cf	Virtual network		East US	1	
cb_crudimage-b89ad2	Virtual network		East US	1	
cb_crudimage-e3acbc	Virtual network		East US	1	
cb_inst_props-1230bc	Network security group		East US	2	
cb_inst_props-1230bc	Virtual network		East US	1	
cb_inst_props-230aa5	Network security group		East US	2	
cb_inst_props-230aa5	Virtual network		East US	1	
cb_inst_props-6735ca	Network security group		East US	2	
cb_inst_props-6735ca	Virtual network		East US	1	
cb_inst_props-df87d2	Network security group		East US	2	
cb_inst_props-df87d2	Virtual network		East US	1	
cb_instcrud-034f88c4	Virtual network		East US	1	
cb_instcrud-2f12a749	Virtual network		East US	1	
cb_instcrud-36b7e8ad	Virtual network		East US	1	
cb_instcrud-c2e992b7	Virtual network		East US	1	
cb_instmethods-5930	Network security group		East US	2	
cb_instmethods-5930	Route table		East US	1	
cb_instmethods-5930	Virtual network		East US	1	
cb_instmethods-617d7	Network security group		East US	2	
cb_instmethods-617d7	Route table		East US	1	
cb_instmethods-617d7	Virtual network		East US	1	
cb_instmethods-d90f8	Network security group		East US	2	
cb_instmethods-d90f8	Route table		East US	1	

Fig. 19: The CloudBridge *name* property always maps to the unchangeable resource name in Azure. The *label* property maps to the tag with key 'Label' for most resources in Azure. By default, this label will appear in the tags column, but can also be made into its own column, using the feature pointed out in the screenshot above.

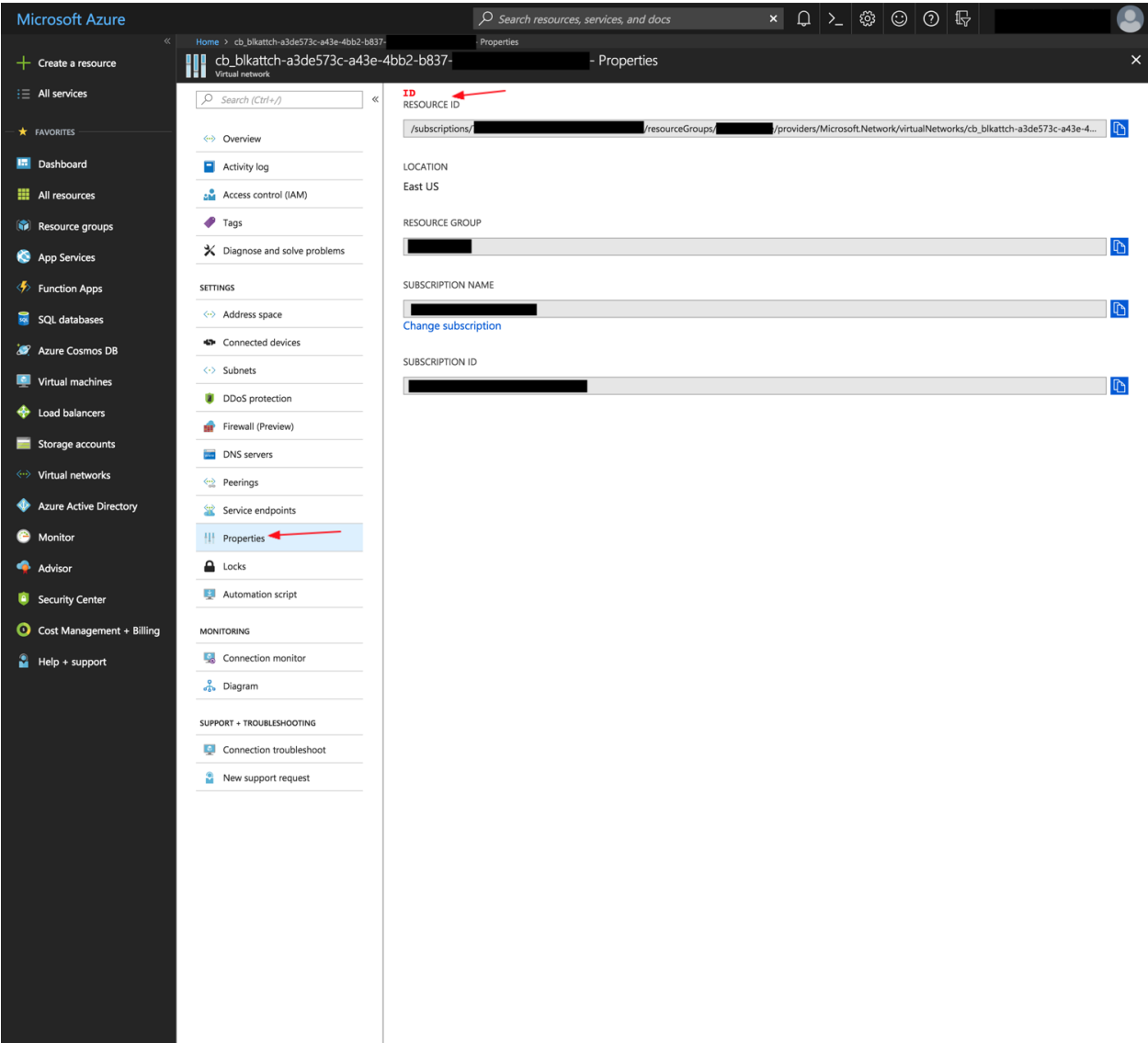


Fig. 20: The CloudBridge *ID* property most often maps to the Resource ID in Azure, which can be found under the properties tab within a resource. The above screenshot shows where to find a resource’s ID in Azure’s web portal.

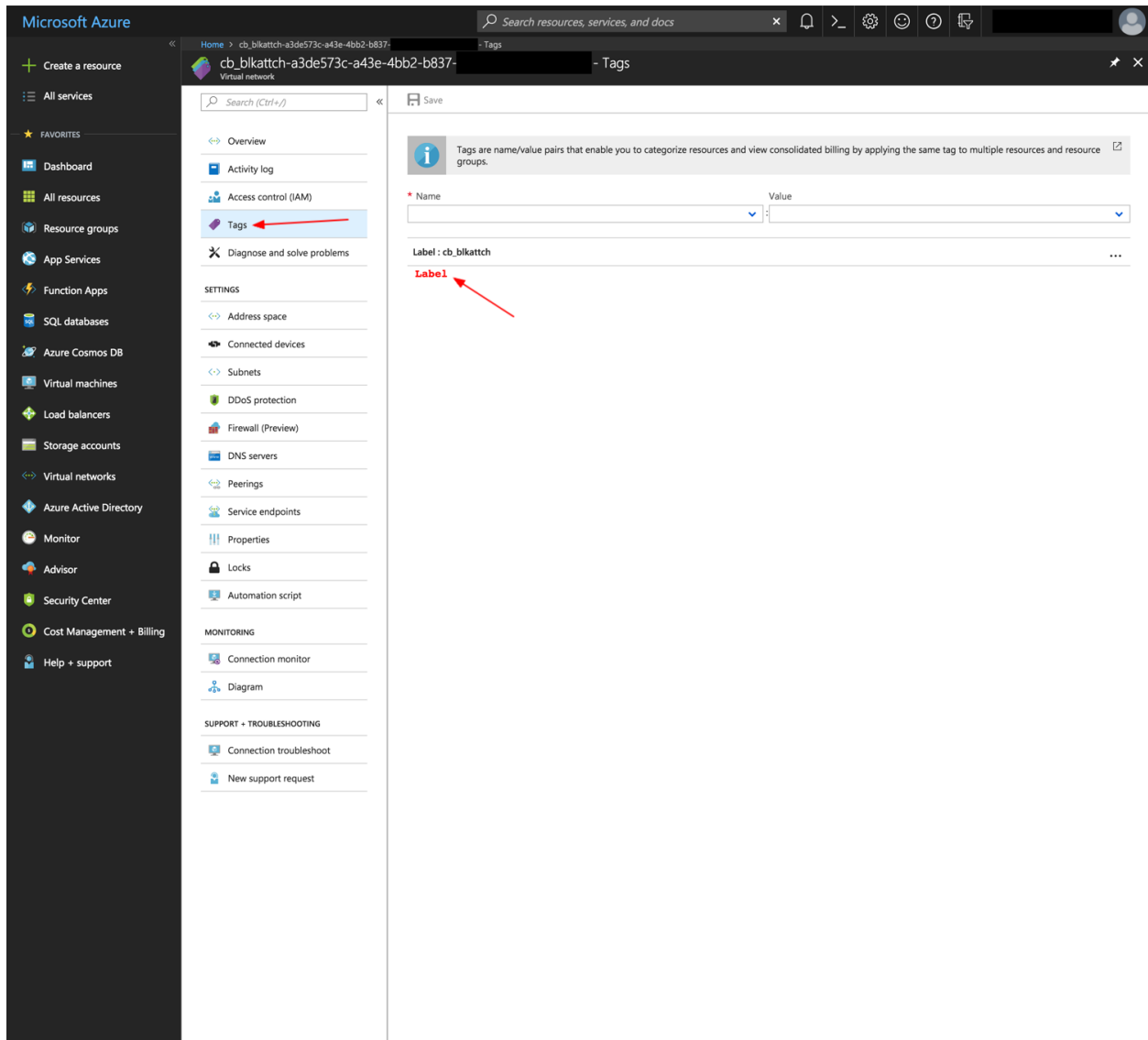


Fig. 21: The CloudBridge *label* property most often maps to the tag with key ‘Label’ in Azure, which can be found under the tags tab within a resource. The above screenshot shows where to find a resource’s label in Azure’s web portal.

(continued from previous page)

```
# The URN can also be used instead if it is already known
# When the latest version is desired, it can be retrieved with the
# keyword 'latest' in the URN without specifying a version
img = provider.compute.images.get(
    'Canonical:UbuntuServer:16.04.0-LTS:latest')
```

Given that these resources are not owned by the user, they can only be referenced and all setters will silently pass. CloudBridge properties *name* and *label* will map to the URN, while the *ID* will map to the public *ID*. It is also important to note that some of these resources are paid and required a plan to use, while others are free but likewise require accepting certain terms before being used. These plans and terms are passed and accepted silently by CloudBridge in order to keep the code cloud-independent. We therefore encourage using the [marketplace website](#) to view the images and plan details before using them in CloudBridge.

Additionally, Subnets are a particular resource in Azure because they are not simply found in the Resource Group like most resources, but are rather nested within a network. Moreover, Subnets do not support tags in Azure. However, they remain a labeled resource in CloudBridge, which was accomplished by creating Network tags holding Subnet labels in Azure. The below screenshots will show how to find Subnets and their labels in the Azure web portal.

Azure - Unlabeled Resources

Unlabeled Resource	Re-	Azure Resource Type	CB ID	CB Name	CB Label
AzureKeyPair		StorageAccount:Table	Name	Name	•
AzureBucket		StorageAccount:BlobContainer	Container	Name	•
AzureBucketObject		StorageAccount:BlobContainer:BlockBlob	Container:BlockBlob	Name	•

The resources listed above are unlabeled. They thus only have the *name* property in CloudBridge. These resources require a mandatory *name* parameter at creation, which will directly map to the unchangeable *name* property. Additionally, for these resources, the *ID* property also maps to the *name* in Azure, as these resources don't have an *ID* in the traditional sense and can be located simply by name. Finally, unlabeled resources support a *name* parameter for the *find* method in their corresponding services.

Azure - Special Unlabeled Resources

Unlabeled Resource	Re-	Azure Resource Type	CB ID	CB Name	CB Label
AzureFloatingIP		Public IP Address	ID	[public_ip]	•
AzureInternetGateway		None	cb-gateway-wrapper	cb-gateway-wrapper	•
AzureVMFirewallRule		Network Security Rules	ID	name	•

While these resources are similarly unlabeled, they do not follow the same general rules as the ones listed above.

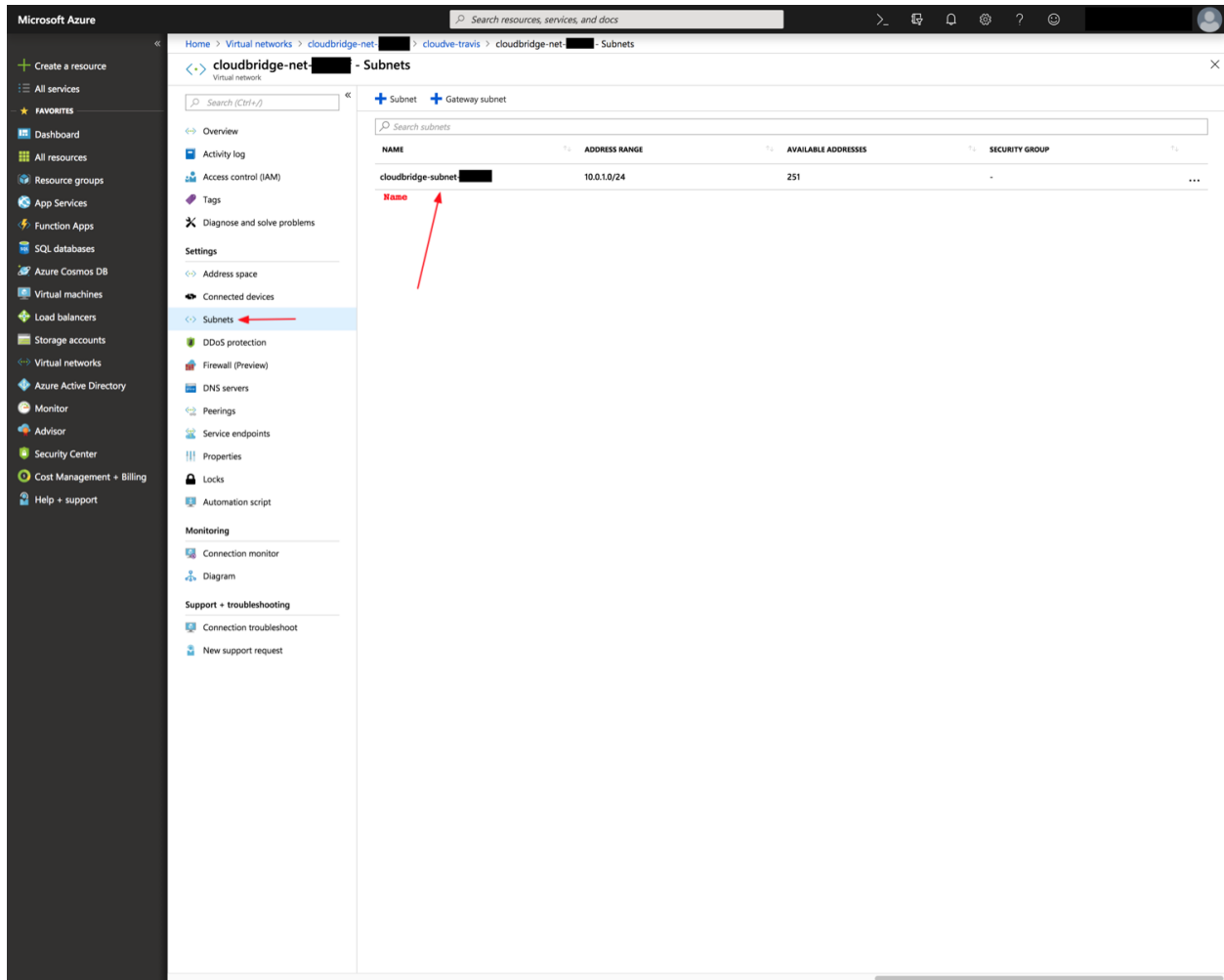


Fig. 22: The CloudBridge *name* property for Subnets corresponds to the unchangeable Resource Name in Azure. However, unlike other resources where the Azure Name maps directly to the *name* property alone, a Subnet's *name* property returns the Network's name and the Subnet's name, separated by a slash, thus having the format [network-Name]/[subnetName]. Subnets are additionally not found in the default resource list, but are rather nested within a Network, in the Subnets tab as shown above.

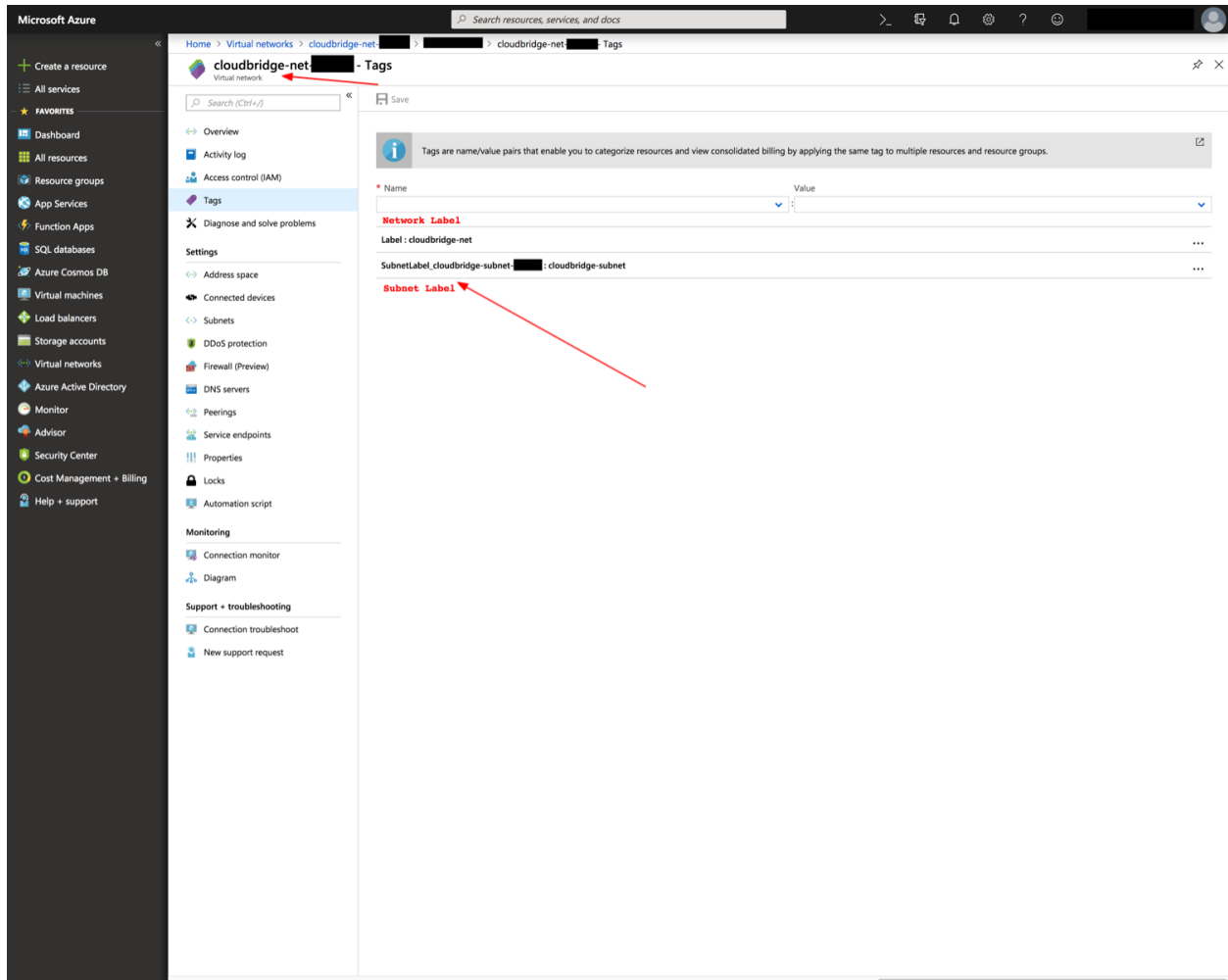


Fig. 23: The CloudBridge *label* property most often maps to the tag with key ‘Label’ in Azure, which can be found under the tags tab within a resource. However, given that Subnets can’t hold tags themselves, we set their tags in the Network with which they are associated. The tag name ‘Label’ thus corresponds to the Network’s label, while each contained Subnet will have a corresponding tag with the name ‘SubnetLabel_[subnetName]’.

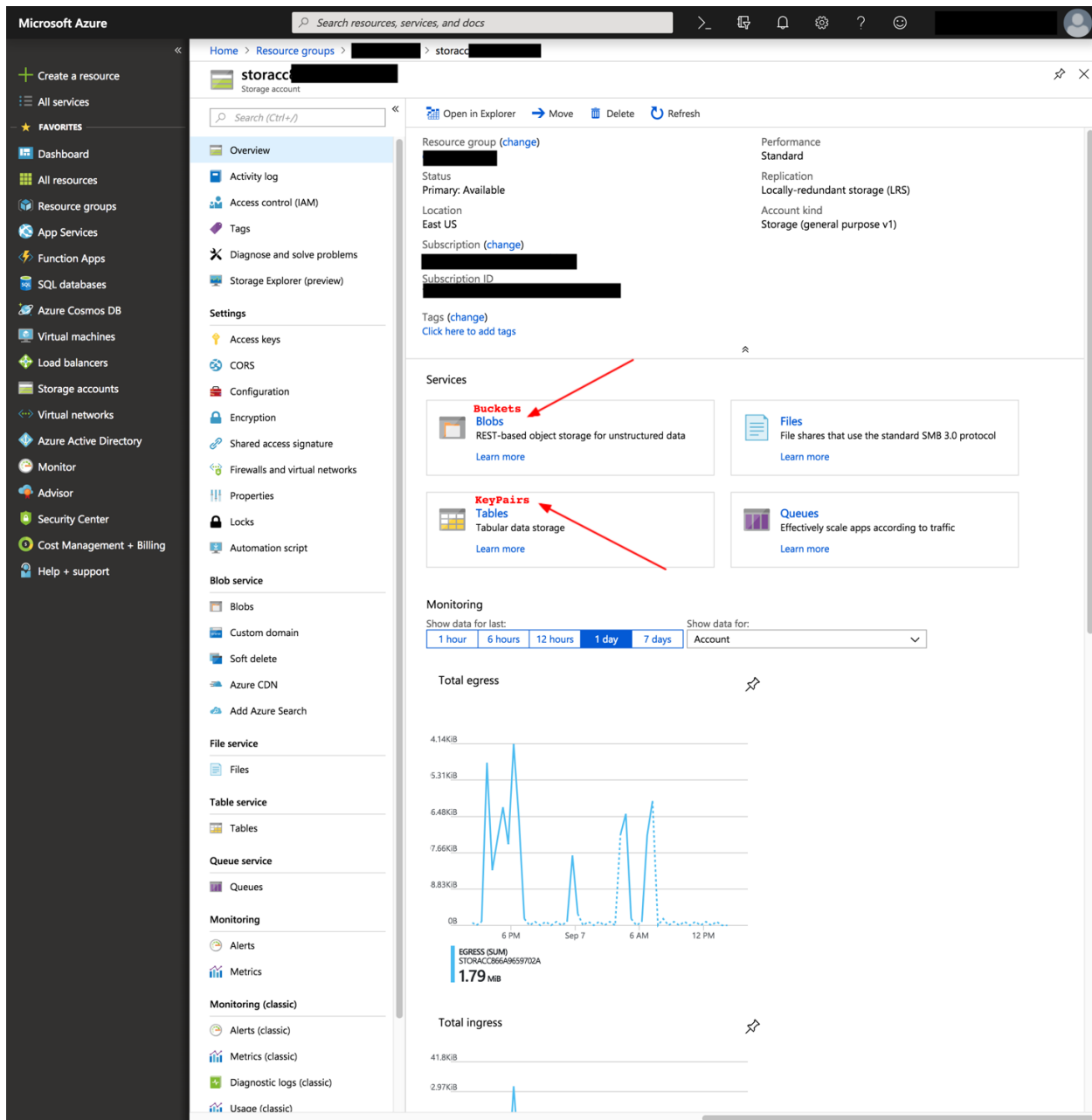


Fig. 24: Bucket and Key Pair objects are different than other resources in Azure, as they are not resources simply residing in a resource group, but are rather found in a storage account. As a result of this difference, these resources do not support labels, and cannot be seen on the default dashboard. In order to find these resources in the Azure web portal, one must head to the storage account containing them, and look in the *Blobs* and *Tables* services respectively for *Buckets* and *KeyPairs*.

Firstly, they differ by the fact that they take neither a *name* nor a *label* parameter at creation. Moreover, each of them has other special properties.

The FloatingIP resource has a traditional resource ID, but instead of a traditional name, its *name* property maps to its Public IP. Thus, the name seen in the Azure web portal will not map to the CloudBridge name, but will rather be auto-generated, while the Azure *IP Address* will map to CloudBridge name. Moreover, the corresponding *find* method for Floating IPs can thus help find a resource by *Public IP Address*, and the get method also accepts a ‘Public IP’ instead of an ‘ID’.

In terms of the gateway, one of the major discrepancies in Azure is the non-existence of an InternetGateway. In fact, Azure resources are exposed with no need for an Internet gateway. However, in order to keep resources consistent across providers, the CloudBridge Gateway resource exists regardless of provider. For Azure, the gateway object created through CloudBridge will not appear on the dashboard, but will rather be a cached CloudBridge-level wrapper object. For a succinct comparison between AWS Gateways and Azure, see [this answer](#).

Finally, Firewall Rules in Azure differ from traditional unlabeled resources by the fact that they do not take a *name* parameter at creation. These rules can be found within each Firewall (i.e. Security Group) in the Azure web portal, and will have an automatically generated *name* of the form ‘cb-rule-[int]’.

Detailed OpenStack Type and Resource Mappings

OpenStack - Labeled Resources

Labeled Resource	OS Resource Type	CB ID	CB Name	CB Label
OpenStackInstance	Instance	ID	ID	Name
OpenStackMachineImage	Image	ID	ID	Name
OpenStackNetwork	Network	ID	ID	Name
OpenStackSubnet	Subnet	ID	ID	Name
OpenStackRouter	Router	ID	ID	Name
OpenStackVolume	Volume	ID	ID	Name
OpenStackSnapshot	Snapshot	ID	ID	Name
OpenStackVMFirewall	Security Group	ID	ID	Name

The resources listed above are labeled, they thus have both the *name* and *label* properties in CloudBridge. These resources require a mandatory *label* parameter at creation. For all labeled resources, the *label* property in OpenStack maps to the Name attribute. However, unlike in Azure or AWS, no resource has an unchangeable name by which to identify it in our OpenStack implementation. The *name* property will therefore map to the ID, preserving its role as an unchangeable identifier even though not easily readable in this context. Finally, labeled resources support a *label* parameter for the *find* method in their corresponding services. The below screenshots will help map these properties to OpenStack objects in the web portal. Additionally, although OpenStack Security Groups are not associated with a specific network, such an association is done in CloudBridge, due to its necessity in AWS. As such, the VMFirewall creation method requires a *network* parameter and the association is accomplished in OpenStack through the description, by appending the following string to the user-provided description (if any) at creation: “[CB-AUTO-associated-network-id: associated_net_id]”

openstack demo admin

Project / Compute / Instances / cb-

Compute cb- Create Snapshot

Overview Log Console Action Log

Label	Name	ID	Status	Locked	Availability Zone	Created	Time Since Created	Host
	cb-	e8350863-ab3b-4e14-b27d-7b19cc778e8a	Active	False	nova	Oct. 16, 2018, 8:41 p.m.	1 month, 1 week	cloudbridge-devstack-18-1-22

Specs

Flavor Name	m1.tiny
Flavor ID	1
RAM	512MB
VCPUs	1 VCPU
Disk	1GB

IP Addresses

Cb-Instmethods-2e2c51	10.0.1.9
-----------------------	----------

Security Groups

Not available

Metadata

Key Name	None
Image Name	cirros-0.3.5-x86_64-disk
Image ID	17ee83

Volumes Attached

Volume	No volumes attached.
--------	----------------------

Fig. 25: The CloudBridge *name* and *ID* properties map to the unchangeable resource ID in OpenStack as resources do not allow for an unchangeable name. The *label* property maps to the ‘Name’ for all resources in OpenStack. By default, this label will appear in the first column.

OpenStack - Unlabeled Resources

Unlabeled Resource	OS Resource Type	CB ID	CB Name	CB Label
OpenStackKeyPair	Key Pair	Name	Name	•
OpenStackBucket	Object Store Container	Name	Name	•
OpenStackBucketObject	Object	Name	Name	•

The resources listed above are unlabeled. They thus only have the *name* property in CloudBridge. These resources require a mandatory *name* parameter at creation, which will directly map to the unchangeable *name* property. Additionally, for these resources, the *ID* property also maps to the *name* in OpenStack, as these resources don't have an *ID* in the traditional sense and can be identified by name. Finally, unlabeled resources support a *name* parameter for the *find* method in their corresponding services.

OpenStack - Special Unlabeled Resources

Unlabeled Resource	OS Resource Type	CB ID	CB Name	CB Label
OpenStackFloatingIP	Floating IP	ID	[public_ip]	•
OpenStackInternetGateway	Network <i>public</i>	ID	'public'	•
OpenStackVMFirewallRule	Security Group Rule	ID	Generated: [direction]- [protocol]- [from_port]- [to_port]-[cidr]- [fw_id]	•

While these resources are similarly unlabeled, they do not follow the same general rules as the ones listed before. Firstly, they differ by the fact that they take neither a *name* nor a *label* parameter at creation. Moreover, each of them has other special properties.

The FloatingIP resource has a traditional resource ID, but instead of a traditional name, its *name* property maps to its Public IP. Moreover, the corresponding *find* method for Floating IPs can thus help find a resource by *Public IP Address*.

In terms of the gateway in OpenStack, it maps to the network named 'public.' Thus, the internet gateway create method does not take a name parameter, and the *name* property will be 'public'.

Finally, Firewall Rules in OpenStack differ from traditional unlabeled resources by the fact that they do not take a *name* parameter at creation, and the *name* property is automatically generated from the rule's properties, as shown above. These rules can be found within each Firewall (i.e. Security Group) in the web portal, and will not have any name in the OpenStack dashboard.

4.3.12 Common Setup Issues

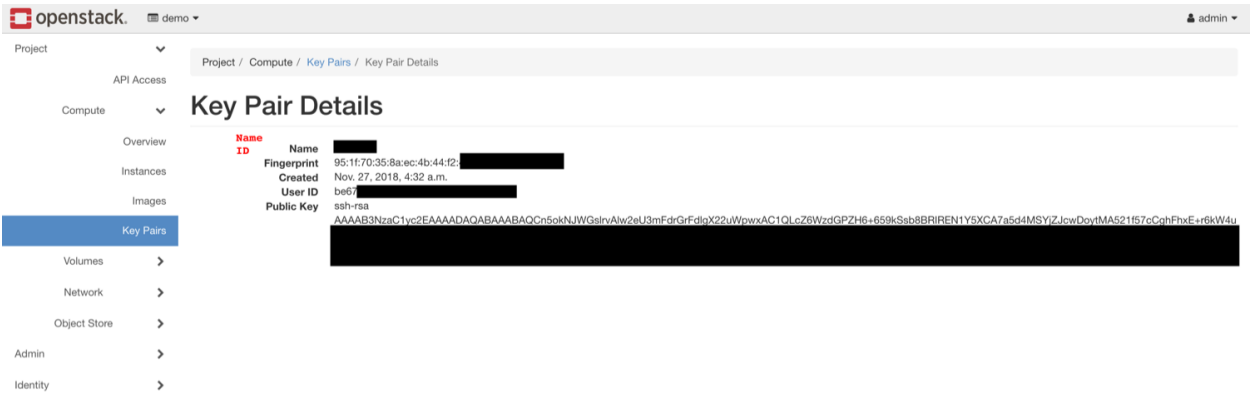


Fig. 26: KeyPairs and other unlabeled resources in OpenStack have *name* that is unique and unmodifiable. The *ID* will thus map to the *name* property when no other *ID* exists for that OpenStack resource.

macOS Issues

- If you are getting an error message like so: Authentication with cloud provider failed: [SSL: CERTIFICATE_VERIFY_FAILED] certificate verify failed (_ssl.c:749) then this indicates that you are probably using a newer version of Python on macOS. Starting with Python 3.6, the Python installer includes its own version of OpenSSL and it no longer uses the system trusted certificate keychains.

Python 3.6 includes a script that can install a bundle of root certificates from `certifi`. To install this bundle execute the following:

```
cd /Applications/Python\ 3.6/
sudo ./Install\ Certificates.command
```

For more information see [this StackOverflow answer](#) and the [Python 3.6 Release Notes](#).

4.4 Contributor Guide

This section has information on how to contribute to CloudBridge development, and a walkthrough of the process of getting started on developing a new CloudBridge Provider.

4.4.1 Design Goals

1. Create a cloud abstraction layer which minimises or eliminates the need for cloud specific special casing (i.e., Not require clients to write `if EC2 do x else if OPENSTACK do y.`)
2. Have a suite of conformance tests which are comprehensive enough that goal 1 can be achieved. This would also mean that clients need not manually test against each provider to make sure their application is compatible.
3. Opt for a minimum set of features that a cloud provider will support, instead of a lowest common denominator approach. This means that reasonably mature clouds like Amazon and OpenStack are used as the benchmark against which functionality & features are determined. Therefore, there is a definite expectation that the cloud infrastructure will support a compute service with support for images and snapshots and various machine sizes. The cloud infrastructure will very likely support block storage, although this is currently optional. It may optionally support object storage.
4. Make the CloudBridge layer as thin as possible without compromising goal 1. By wrapping the cloud provider's native SDK and doing the minimal work necessary to adapt the interface, we can achieve greater development speed and reliability since the native provider SDK is most likely to have both properties.

4.4.2 Design decisions

This document captures outcomes and, in some cases, the through process behind some of the design decisions that took place while architecting CloudBridge. It is intended as a reference.

Require zone parameter when creating a default subnet

Placement zone is required because it is an explicit application decision, even though ideally *default* would not require input. Before requiring it, the implementations would create a subnet in each availability zone and return the first one in the list. This could potentially return different values over time. Another factor influencing the decision was the example of creating a volume followed by creating an instance with

presumably the two needing to be in the same zone. By requiring the zone across the board, it is less likely to lead to a mismatch. (Related to [63](#).)

Resource identification, naming, and labeling

While it would be reasonable to expect that complex constructs like networking would be the most difficult to abstract away uniformly across providers, in retrospect, simple naming of objects has arguably been the most complex and convoluted to map consistently. CloudBridge has been through several iterations of naming and labeling, before finally settling on the current design. This section captures that history and design rationale.

First iteration In the early days, when CloudBridge supported only AWS and OpenStack, there were only two concepts, *id* and *name*. The *id* was straightforward enough, as it usually mapped to a unique identifier, auto-generated by the provider. The *name* generally mapped to a tag in the case of AWS, and a *name* field in the case of OpenStack. However, even then, there were inconsistencies within individual providers. For example, while AWS generally supported tags, it had a dedicated name field for machine images called *ami-name*. Furthermore, this name field could only be set at image creation time, and could not be changed thereafter. Similarly, AWS does not allow VM firewall (i.e., security group) names to be changed. Nevertheless, CloudBridge continued to use *id* and *name*, with the *name* being changeable for some resources, and read-only in others.

As CloudBridge evolved and support was added for Azure and GCP, things only became more complex. Some providers (e.g. Azure and GCP) used a user-provided value instead of an auto-generated value as an *id*, which would also be displayed in their respective dashboards as *Name*. This meant that they were treating their servers as individually named pets, instead of adopting the cattle model, should one be tempted to use that macabre [pets vs cattle](#) analogy. These user-provided names could not be changed after a resource had been created. Instead, these providers seemed to be gravitating toward the use of tags (or labels) to support arbitrary naming and name changes. Yet, not all resources support tags so CloudBridge could not rely solely on tags. Further, tags do not need to be unique across multiple resources, while names do (at least for some resources, such as vmfirewalls within a private network). Overall, consistency was challenging to achieve with resource naming. Therefore, it was decided that CloudBridge would continue to support resource renaming to the best extent possible and balance between the use of the resource name property and resource tags. However, because of the inconsistency in rename functionality across providers, using the rename capabilities within CloudBridge would lead to cloud-dependent code (Related to [131](#).) and therefore, the only option was to continue to stick a caveat emptor to resource renaming.

Second iteration However, it soon became apparent that this overloaded terminology was continuing to cause confusion. The *id* property in CloudBridge mapped to the unchangeable *name* property in Azure and GCP, and the *name* property in cloudbridge sometimes mapped to a *tag* in certain providers, and a *name* in other providers and they were sometimes read-only, sometimes writable. In an attempt to disambiguate these concepts, it was then decided that perhaps three concepts were needed - *id*, *display_id*, and *label*. The *id* would continue to refer to a unique identifier for a resource and be mapped accordingly to the underlying provider. The *display_id* would be a more user-friendly version of an *id*, suitable for display to an end-user and be unchangeable, but on rare occasions, not unique. For example, AWS *ami-name* was a *display_id* while the *ami-id* was an *id*. Similarly, an Azure resource name mapped to the *display_id*, since it was an unchangeable, user-friendly identifier. Finally, *label* was a changeable, user-assignable value that would be mapped often to a tag on the resource, or the name of the resource, should the name be changeable. This clearly disambiguated between unique identifiers, user-assignable values and read-only, user-friendly values. At object creation, all services would accept a *label* as an optional parameter. If provided, the *display_id* would sometimes be derived from the *label* by appending a uuid to the *label*, depending on the provider. At other times, it would simply map to an *id*.

Third iteration It soon became apparent that some resources like keypairs could not have a *label* at all, yet needed to be named during object creation. However, we could not use *display_id* for this purpose

became the `display_id`, by definition, is unchangeable. It could not be called *label* because the label, in contrast, was changeable. Therefore, it seemed like we were back to calling it *name* instead, introducing yet a fourth term. To simplify this, it was then decided that *display_id* and *name* would be collapsed together into one term and be called *name* instead. All resources would have an *id* and a *name*, and resources that support it would have a *label*. To make things even simpler and consistent, it was also decided that label would be made mandatory for all resources during object creation, and follow the same restrictions as name, which is to have a 3 character minimum. (This was to deal with an exception in OpenStack, where the label mapped to instance name, but could not be empty. Therefore, by making all labels mandatory and adhere to minimum length restrictions, we could make the overall conventions uniform across all resources and therefore easier to remember and enforce.)

TL;DR CloudBridge has three concepts when it comes to naming and identifying objects:

- *id* is a unique identifier for an object, always auto-generated;
- *name* is a read-only, user-friendly value which is suitable for display to the end-user;
- *label* is a user-assignable value that can be changed.

The *name* is often derived from the *label* but not always. Not all resources support *labels*. Some only accept *names*, which can be specified at object creation time (e.g. keypairs). Both *names* and *labels* adhere to the same restrictions - a minimum length of 3 which should be alphanumeric characters or dashes only. Names or labels should not begin or end with a dash, or have consecutive dashes.

Make providers single zone

Allowing each operation to specify its own zone led to various complications, such as the one detailed above. Ultimately, it led to an impasse with GCP, which tended to require the zone for almost every operation and some of our methods were not geared to do so. Therefore, by making the provider zone specific, we have removed a considerable amount of complexity from both the code, with no significant impact on usability, since operations generally tend to be confined to the same zone. Multi-zone operations now require multiple cloud provider instances.

4.4.3 Running tests

In the spirit of the library's *Design Goals*, the aim is to have thorough tests for the entire library. This page explains the testing philosophy and shows how to run the tests locally.

Testing philosophy

Our testing goals are to:

1. Write one set of tests that all provider implementations must pass.
2. Make that set of tests a 'conformance' test suite, which validates that each implementation correctly implements the CloudBridge specification.
3. Make the test suite comprehensive enough that a provider which passes all the tests can be used safely by an application with no additional testing. In other words, the CloudBridge specification and accompanying test suite must be comprehensive enough that no provider specific workarounds, code or testing is required.
4. For development, mock providers may be used to speed up the feedback cycle, but providers must also pass the full suite of tests when run against actual cloud infrastructure to ensure that we are not testing against an idealised or imagined environment.
5. Aim for 100% code coverage.

Running tests

To run the test suite locally:

1. Install `tox` with `pip install tox`
2. Export all environment variables listed in `tox.ini` (under `passenv`)
3. Run `tox` command

This will run all the tests for all the environments defined in file `tox.ini`.

Specific environment and infrastructure

If you'd like to run the tests on a specific environment only, say Python 2.7, against a specific infrastructure, say `aws`, use a command like this: `tox -e py27-aws`. The available provider names are listed in the `ProviderList` class (e.g., `aws` or `openstack`).

Specific test cases

You can run a specific test case, as follows: `tox -- tests/test_image_service.py:CloudImageServiceTestCase.test_create_and_list_imag`

It can also be restricted to a particular environment as follows: `tox -e "py27-aws" -- tests/test_cloud_factory.py:CloudFactoryTestCase`

See nosetest documentation for other parameters that can be passed in.

Using unittest directly

You can also run the tests against your active virtual environment directly with `python setup.py test`. You will need to set the `CB_TEST_PROVIDER` environment variable prior to running the tests, or they will default to `CB_TEST_PROVIDER=aws`.

You can also run a specific test case, as follows: `python setup.py test -s tests.test_cloud_factory.CloudFactoryTestCase`

Using a mock provider

Note that running the tests may create various cloud resources, for which you may incur costs. For the AWS cloud, there is also a mock provider ([moto](#)) that will simulate AWS resources. You can use `CB_TEST_PROVIDER=mock` to run tests against the mock provider only, which will provide faster feedback times.

Alternatively you can run the mock tests through `tox`. `tox -e "py27-mock"`

4.4.4 Provider Development Walkthrough

This guide will walk you through the basic process of developing a new provider for CloudBridge.

1. We start off by creating a new folder for the provider within the `cloudbridge/cloud/providers` folder. In this case: `gcp`. Further, install the native cloud provider Python library, here `pip install google-api-python-client==1.4.2` and a couple of its requirements `oauth2client==1.5.2` and `pycrypto==2.6.1`.

2. Add a `provider.py` file. This file will contain the main implementation of the cloud provider and will be the entry point that CloudBridge uses for all provider related services. You will need to subclass `BaseCloudProvider` and add a class variable named `PROVIDER_ID`.

```
from cloudbridge.base import BaseCloudProvider

class GCPCloudProvider(BaseCloudProvider):

    PROVIDER_ID = 'gcp'

    def __init__(self, config):
        super(GCPCloudProvider, self).__init__(config)
```

3. Add an `__init__.py` to the `cloudbridge/cloud/providers/gcp` folder and export the provider.

```
from .provider import GCPCloudProvider # noqa
```

Tip: You can view the code so far here: [commit 1](#)

4. Next, we need to register the provider with the factory. This only requires that you register the provider's ID in the `ProviderList`. Add GCP to the `ProviderList` class in `cloudbridge/cloud/factory.py`.

5. Run the test suite. We will get the tests passing on py27 first.

```
export CB_TEST_PROVIDER=gcp
tox -e py27
```

You should see the tests fail with the following message:

```
"TypeError: Can't instantiate abstract class GCPCloudProvider with abstract
methods storage, compute, security, network."
```

6. Therefore, our next step is to implement these methods. We can start off by implementing these methods in `provider.py` and raising a `NotImplementedError`.

```
@property
def compute(self):
    raise NotImplementedError(
        "GCPCloudProvider does not implement this service")

@property
def network(self):
    raise NotImplementedError(
        "GCPCloudProvider does not implement this service")

@property
def security(self):
    raise NotImplementedError(
        "GCPCloudProvider does not implement this service")

@property
def storage(self):
    raise NotImplementedError(
        "GCPCloudProvider does not implement this service")
```

Running the tests now will complain as much. We will next implement each Service in turn.

7. We will start with the compute service. Add a `services.py` file.

```
from cloudbridge.base.services import BaseSecurityService

class GCPSecurityService(BaseSecurityService):

    def __init__(self, provider):
        super(GCPSecurityService, self).__init__(provider)
```

8. We can now return this new service from the security property in `provider.py` as follows:

```
def __init__(self, config):
    super(GCPCloudProvider, self).__init__(config)
    self._security = GCPSecurityService(self)

@property
def security(self):
    return self._security
```

Tip: You can view the code so far here: [commit 2](#)

9. Run the tests, and the following message will cause all security service tests to fail:

```
"TypeError: Can't instantiate abstract class GCPSecurityService with abstract
methods key_pairs, security_groups."
```

The Abstract Base Classes are doing their job and flagging all methods that need to be implemented.

10. Since the security service simply provides organisational structure, and is a container for the `key_pairs` and `security_groups` services, we must next implement these services.

```
from cloudbridge.base.services import BaseKeyPairService
from cloudbridge.base.services import BaseSecurityGroupService
from cloudbridge.base.services import BaseSecurityService

class GCPSecurityService(BaseSecurityService):

    def __init__(self, provider):
        super(GCPSecurityService, self).__init__(provider)

        # Initialize provider services
        self._key_pairs = GCPKeyPairService(provider)
        self._security_groups = GCPSecurityGroupService(provider)

    @property
    def key_pairs(self):
        return self._key_pairs

    @property
    def security_groups(self):
        return self._security_groups

class GCPKeyPairService(BaseKeyPairService):
```

(continues on next page)

(continued from previous page)

```

def __init__(self, provider):
    super(GCPKeyPairService, self).__init__(provider)

class GCPSecurityGroupService(BaseSecurityGroupService):

    def __init__(self, provider):
        super(GCPSecurityGroupService, self).__init__(provider)

```

Tip: You can view the code so far here: [commit 3](#)

Once again, running the tests will complain of missing methods:

```

"TypeError: Can't instantiate abstract class GCPKeyPairService with abstract
methods create, find, get, list."

```

11. Keep implementing the methods till the security service works, and the tests pass.

Note: We start off by implementing the list keypairs method. Therefore, to obtain the keypair, we need to have a connection to the cloud provider. For this, we need to install the Google sdk, and thereafter, to obtain the desired connection via the sdk. While the design and structure of that connection is up to the implementor, a general design we have followed is to have the cloud connection globally available within the provider.

To add the sdk, we edit CloudBridge's main `setup.py` and list the dependencies.

```

gcp_reqs = ['google-api-python-client==1.4.2']
full_reqs = base_reqs + aws_reqs + openstack_reqs + gcp_reqs

```

We will also register the provider in `cloudbridge/cloud/factory.py`'s provider list.

```

class ProviderList(object):
    AWS = 'aws'
    OPENSTACK = 'openstack'
    ...
    GCP = 'gcp'

```

Tip: You can view the code so far here: [commit 4](#)

12. Thereafter, we create the actual connection through the sdk. In the case of GCP, we need a Compute API client object. We will make this connection available as a public property named `gcp_compute` in the provider. We will then lazily initialize this connection.

A full implementation of the KeyPair service can now be made in a provider specific manner.

Tip: You can view the code so far here: [commit 5](#)

4.4.5 Release Process

1. Make sure all tests pass.
2. Increment version number in `cloudbridge/__init__.py` as per [semver rules](#).
3. Freeze all library dependencies in `setup.py` and commit. The version numbers can be a range with the upper limit being the latest known working version, and the lowest being the last known working version.

In general, our strategy is to make provider sdk libraries fixed within relatively known compatibility ranges, so that we reduce the chances of breakage. If someone uses CloudBridge, presumably, they do not use the SDKs directly. For all other libraries, especially, general purpose libraries (e.g. `six`), our strategy is to make compatibility as broad and unrestricted as possible.

4. Add release notes to `CHANGELOG.rst`. Also add last commit hash to changelog. List of commits can be obtained using `git shortlog <last release hash>..HEAD`
5. Release to PyPi. (make sure you have run `pip install wheel twine`) First, test release with PyPI staging server as described in: <https://hynek.me/articles/sharing-your-labor-of-love-pypi-quick-and-dirty/>

Once tested, run:

```
# remove stale files or wheel might package them
rm -r build dist
python setup.py sdist bdist_wheel
twine upload -r pypi dist/cloudbridge-1.0.0*
```

6. Tag release and make a GitHub release.

```
git tag -a v2.0.0 -m "Release 2.0.0"
git push
git push --tags
```

7. Increment version number in `cloudbridge/__init__.py` to `version-dev` to indicate the development cycle, commit, and push the changes.

4.5 API reference

This section includes the API documentation for the reference interface.

4.5.1 Providers

CloudProvider

class `cloudbridge.interfaces.provider.CloudProvider` (*config*)

Base interface for a cloud provider

__init__ (*config*)

Create a new provider instance given a dictionary of configuration attributes.

Parameters *config* (*dict*) – A dictionary object containing provider initialization values.

Alternatively, this can be an iterable of key/value pairs (as tuples or other iterables of length two). See specific provider implementation for the required fields.

Return type *CloudProvider*

Returns a concrete provider instance

authenticate()

Checks whether a provider can be successfully authenticated with the configured settings. Clients are *not* required to call this method prior to accessing provider services, as most cloud connections are initialized lazily. The `authenticate()` method will return `True` if cloudbridge can establish a successful connection to the provider. It will raise an exception with the appropriate error details otherwise.

Example:

```
try:
    if provider.authenticate():
        print("Provider connection successful")
except ProviderConnectionException as e:
    print("Could not authenticate with provider: %s" % (e, ))
```

Return type `bool`

Returns `True` if authentication is successful.

clone(zone=None)

Create a clone of this provider. An optional `zone` parameter can be used to clone the provider to use a different zone. As each cloudbridge provider is restricted to a particular zone, this is useful when performing cross-zonal operations.

Example:

```
# list instances in all availability zones
all_instances = []
for zone in provider.compute.regions.current.zones:
    new_provider = provider.clone(zone=zone)
    all_instances.append(list(new_provider.compute.instances))
print(all_instances)
```

Parameters `zone` (*PlacementZone* object) – Changes the provider’s zone to the requested *AvailabilityZone*

Return type *CloudProvider*

Returns A clone of the *CloudProvider*, with zone changed to the requested zone.

compute

Provides access to all compute related services in this provider.

Example:

```
regions = provider.compute.regions.list()
vm_types = provider.compute.vm_types.list()
instances = provider.compute.instances.list()
images = provider.compute.images.list()

# Alternatively
for instance in provider.compute.instances:
    print(instance.name)
```

Return type *ComputeService*

Returns a *ComputeService* object

config

Returns the config object associated with this provider. This object is a subclass of `dict` and will contain the properties provided at initialization time, grouped under *cloud_properties* and *credentials* keys. In addition, it also contains extra provider-wide properties such as the default result limit for *list()* queries.

Example:

```
config = { 'aws_access_key' : '<my_key>' }
provider = factory.create_provider(ProviderList.AWS, config)
print(provider.config['credentials'].get('aws_access_key'))
print(provider.config.default_result_limit)
# change provider result limit
provider.config.default_result_limit = 100
```

Return type *Configuration*

Returns An object of class *Configuration*, which contains the values used to initialize the provider, as well as other global configuration properties.

dns

Provides access to all DNS related services.

Example:

```
if provider.has_service(CloudServiceType.DNS):
    print("Provider supports DNS services")
    dns_zones = provider.dns.host_zones.list()
    print(dns_zones)
```

Return type *DnsService*

Returns a DNS service object

has_service (*service_type*)

Checks whether this provider supports a given service.

Example:

```
if provider.has_service(CloudServiceType.BUCKET):
    print("Provider supports object store services")
    provider.storage.buckets.list()
```

Parameters *service_type* (*CloudServiceType*) – Type of service to check support for.

Return type `bool`

Returns `True` if the service type is supported.

middleware

Returns the middleware manager associated with this provider. The middleware manager can be used to add or remove middleware from cloudbridge. Refer to pyeventsystem documentation for more information on how the middleware manager works.

Return type *MiddlewareManager*

Returns An object of class *MiddlewareManager*, which can be used to add or remove middleware from cloudbridge.

networking

Provide access to all network related services in this provider.

Example:

```
networks = provider.networking.networks.list()
subnets = provider.networking.subnets.list()
routers = provider.networking.routers.list()
```

Return type *NetworkingService*

Returns a NetworkingService object

region_name

Returns the region that this provider is connected to. All provider operations will take place within this region.

Return type *str*

Returns a zone id

security

Provides access to key pair management and firewall control

Example:

```
keypairs = provider.security.keypairs.list()
vm_firewalls = provider.security.vm_firewalls.list()
```

Return type object of *SecurityService*

Returns a SecurityService object

storage

Provides access to storage related services in this provider. This includes the volume, snapshot and bucket services,

Example:

```
volumes = provider.storage.volumes.list()
snapshots = provider.storage.snapshots.list()
if provider.has_service(CloudServiceType.BUCKET):
    print("Provider supports object store services")
    print(provider.storage.buckets.list())
```

Return type *StorageService*

Returns a StorageService object

zone_name

Returns the placement zone that this provider is connected to. All provider operations will take place within this zone. Placement zone must be within the provider default region.

Return type *str*

Returns a zone id

4.5.2 Services

- *CloudService*
- *ComputeService*
- *InstanceService*
- *VolumeService*
- *SnapshotService*
- *StorageService*
- *ImageService*
- *NetworkingService*
- *NetworkService*
- *SubnetService*
- *FloatingIPService*
- *RouterService*
- *GatewayService*
- *BucketService*
- *SecurityService*
- *KeyPairService*
- *VMFirewallService*
- *VMTypeService*
- *RegionService*
- *DnsService*
- *DnsZoneService*
- *DnsRecordService*

CloudService

class `cloudbridge.interfaces.services.CloudService`

Base interface for any service supported by a provider. This interface has a provider property that can be used to access the provider associated with this service.

provider

Returns the provider instance associated with this service.

Return type *CloudProvider*

Returns a CloudProvider object

ComputeService

class cloudbridge.interfaces.services.**ComputeService**

The compute service interface is a collection of services that provides access to the underlying compute related services in a provider. For example, the compute.instances service can be used to launch a new instance, and the compute.images service can be used to list available machine images.

images

Provides access to all Image related services in this provider. (e.g. Glance in OpenStack)

Example:

```
# print all images
for image in provider.compute.images:
    print(image.id, image.name, image.label)

# print only first 50 images
for image in provider.compute.images.list(limit=50):
    print(image.id, image.name, image.label)

# find image by name
image = provider.compute.images.find(name='Ubuntu 16.04')[0]
print(image.id, image.name, image.label)
```

Return type *ImageService*

Returns an ImageService object

instances

Provides access to all Instance related services in this provider.

Example:

```
# launch a new instance
image = provider.compute.images.find(name='Ubuntu 16.04')[0]
size = provider.compute.vm_types.find(name='m1.small')[0]
instance = provider.compute.instances.create('Hello', image, size)
print(instance.id, instance.label)
```

Return type *InstanceService*

Returns an InstanceService object

regions

Provides access to all Region related services in this provider.

Example:

```
for region in provider.compute.regions:
    print("Region: ", region.name)
    for zone in region.zones:
        print("\tZone: ", zone.name)
```

Return type *RegionService*

Returns a RegionService object

vm_types

Provides access to all VM type related services in this provider.

Example:

```
# list all VM sizes
for vm_type in provider.compute.vm_types:
    print(vm_type.id, vm_type.name)

# find a specific size by name
vm_type = provider.compute.vm_types.find(name='m1.small')[0]
print(vm_type.vcpus)
```

Return type *VMTypeService*

Returns an VMTypeService object

InstanceService

class cloudbridge.interfaces.services.InstanceService

Provides access to instances in a provider, including creating, listing and deleting instances.

create(*label*, *image*, *vm_type*, *subnet*, *key_pair*=None, *vm_firewalls*=None, *user_data*=None, *launch_config*=None, ***kwargs*)

Creates a new virtual machine instance.

Parameters

- **label** (*str*) – The label of the virtual machine instance. The instance name will be derived from this label.
- **image** (*MachineImage* or *str*) – The MachineImage object or id to boot the virtual machine with
- **vm_type** (*VMType* or *str*) – The VMType or name, specifying the size of the instance to boot into
- **subnet** (*Subnet* or *str*) – The subnet object or a subnet string ID with which the instance should be associated. The subnet is a mandatory parameter, and must be provided when launching an instance.

Note: Older clouds (with classic networking), may not have proper subnet support and are not guaranteed to work. Some providers (e.g. OpenStack) support a null value but the behaviour is implementation specific.

- **key_pair** (*KeyPair* or *str*) – The KeyPair object or its id, to set for the instance.
- **vm_firewalls** (A list of VMFirewall objects or a list of *str* object IDs) – A list of VMFirewall objects or a list of VMFirewall IDs, which should be assigned to this instance.

The VM firewalls must be associated with the same network as the supplied subnet. Use `network.vm_firewalls` to retrieve a list of firewalls belonging to a network.

- **user_data** (*str*) – An extra userdata object which is compatible with the provider.
- **launch_config** (*LaunchConfig* object) – A LaunchConfig object which describes advanced launch configuration options for an instance. Currently, this includes only `block_device_mappings`. To construct a launch configuration object, call `provider.compute.instances.create_launch_config()`

Return type object of *Instance*

Returns an instance of Instance class

create_launch_config()

Creates a `LaunchConfig` object which can be used to set additional options when launching an instance, such as block device mappings and network interfaces.

Return type object of *LaunchConfig*

Returns an instance of a `LaunchConfig` class

find(kwargs)**

Searches for an instance by a given list of attributes.

Supported attributes: name, label

Parameters

- **name** (`str`) – The name to search for
- **label** (`str`) – The label to search for

Return type List of object of *Instance*

Returns A list of Instance objects matching the supplied attributes.

get(instance_id)

Returns an instance given its id. Returns None if the object does not exist.

Return type object of *Instance*

Returns an Instance object

list(limit=None, marker=None)

List available instances.

The returned results can be limited with limit and marker. If not specified, the limit defaults to a global default. See `list()` for more information on how to page through returned results.

example:

```
# List instances
instlist = provider.compute.instances.list()
for instance in instlist:
    print("Instance Data: {0}", instance)
```

Parameters

- **limit** (`int`) – The maximum number of objects to return. Note that the maximum is not guaranteed to be honoured, and a lower maximum may be enforced depending on the provider. In such a case, the returned `ResultList`'s `is_truncated` property can be used to determine whether more records are available.
- **marker** (`str`) – The marker is an opaque identifier used to assist in paging through very long lists of objects. It is returned on each invocation of the list method.

Return type `ResultList` of *Instance*

Returns A `ResultList` object containing a list of Instances

VolumeService

class `cloudbridge.interfaces.services.VolumeService`

Base interface for a Volume Service.

create (*label, size, snapshot=None, description=None*)

Creates a new volume.

Parameters

- **label** (`str`) – The label for the volume.
- **size** (`int`) – The size of the volume (in GB).
- **snapshot** (`str` or `Snapshot` object) – An optional reference to a snapshot from which this volume should be created.
- **description** (`str`) – An optional description that may be supported by some providers. Providers that do not support this property will return `None`.

Return type object of `Volume`

Returns a newly created Volume object.

delete (*volume*)

Delete an existing volume.

Parameters **volume** (`str` or `Volume`) – The object or ID of the volume to be deleted.

find (***kwargs*)

Searches for a volume by a given list of attributes.

Supported attributes: label

Return type object of `Volume`

Returns a Volume object or `None` if not found.

get (*volume_id*)

Returns a volume given its id.

Return type object of `Volume`

Returns a Volume object or `None` if the volume does not exist.

list (*limit=None, marker=None*)

List all volumes.

Return type list of `Volume`

Returns a list of Volume objects.

SnapshotService

class `cloudbridge.interfaces.services.SnapshotService`

Base interface for a Snapshot Service.

create (*label, volume, description=None*)

Creates a new snapshot off a volume.

Parameters

- **label** (`str`) – The label for the snapshot.
- **volume** (`str` or `Volume`) – The volume to create a snapshot of.

- **description** (str) – An optional description that may be supported by some providers. Providers that do not support this property will return None.

Return type object of *Snapshot*

Returns a newly created Snapshot object.

delete (snapshot)

Delete an existing snapshot.

Parameters **snapshot** (str or Snapshot) – The object or ID of the snapshot to be deleted.

find (**kwargs)

Searches for a snapshot by a given list of attributes.

Supported attributes: label

Return type list of *Snapshot*

Returns a Snapshot object or an empty list if none found.

get (snapshot_id)

Returns a snapshot given its id.

Return type object of *Snapshot*

Returns a Snapshot object or None if the snapshot does not exist.

list (limit=None, marker=None)

List all snapshots.

Return type list of *Snapshot*

Returns a list of Snapshot objects.

StorageService

class cloudbridge.interfaces.services.StorageService

The Storage Service interface provides access to block device services, such as volume and snapshot services, as well as object store services, such as buckets, in the provider.

buckets

Provides access to object storage services in this provider.

Example:

```
# print all buckets
for bucket in provider.storage.buckets:
    print(bucket.id, bucket.name)

# find bucket by name
bucket = provider.storage.buckets.find(name='my_bucket')[0]
print(bucket.id, bucket.name)
```

Return type *BucketService*

Returns a BucketService object

snapshots

Provides access to volume snapshots for this provider.

Example:

```
# print all snapshots
for snap in provider.storage.snapshots:
    print(snap.id, snap.name, snap.label)

# find snapshot by label
snap = provider.storage.snapshots.find(label='my_snap')[0]
print(snap.id, snap.name, snap.label)
```

Return type *SnapshotService*

Returns a SnapshotService object

volumes

Provides access to volumes (i.e., block storage) for this provider.

Example:

```
# print all volumes
for vol in provider.storage.volumes:
    print(vol.id, vol.name, vol.label)

# find volume by label
vol = provider.storage.volumes.find(label='my_vol')[0]
print(vol.id, vol.name, vol.label)
```

Return type *VolumeService*

Returns a VolumeService object

ImageService

class cloudbridge.interfaces.services.**ImageService**

Base interface for an Image Service

find (***kwargs*)

Searches for an image by a given list of attributes

Supported attributes: name, label

Return type object of Image

Returns an Image instance

get (*image_id*)

Returns an Image given its id. Returns None if the Image does not exist.

Return type object of Image

Returns an Image instance

list (*filter_by_owner=True, limit=None, marker=None*)

List all images.

Parameters **filter_by_owner** (bool) – If True, return only images owned by the current user. Else, return all public images available from the provider. Note that fetching all images may take a long time.

Return type list of Image

Returns list of image objects

NetworkingService

class cloudbridge.interfaces.services.**NetworkingService**

Base service interface for networking.

This service offers a collection of networking services that in turn provide access to networking resources.

networks

Provides access to all Network related services.

Return type *NetworkService*

Returns a Network service object

routers

Provides access to all Router related services.

Return type *RouterService*

Returns a Router service object

subnets

Provides access to all Subnet related services.

Return type *SubnetService*

Returns a Subnet service object

NetworkService

class cloudbridge.interfaces.services.**NetworkService**

Base interface for a Network Service.

create (*label*, *cidr_block*)

Create a new network.

Parameters

- **label** (*str*) – A label for the network.
- **cidr_block** (*str*) – The cidr block for this network. Some providers will respect this at the network level, while others will only respect it at subnet level. However, to write portable code, you should make sure that any subnets you create fall within this initially specified range. Note that the block size should be between a /16 netmask (65,536 IP addresses) and /28 netmask (16 IP addresses), e.g. 10.0.0.0/16.

Return type object of *Network*

Returns A Network object

delete (*network*)

Delete an existing Network.

Parameters **network** (*str* or *Network*) – The object or id of the network to be deleted.

find (***kwargs*)

Searches for a network by a given list of attributes.

Supported attributes: name, label

Return type List of object of *Network*

Returns A list of Network objects matching the supplied attributes.

get (*network_id*)

Returns a Network given its ID or None if not found.

Parameters **network_id** (*str*) – The ID of the network to retrieve.

Return type object of *Network*

Returns a Network object

list (*limit=None, marker=None*)

List all networks.

Return type list of *Network*

Returns list of Network objects

subnets

Provides access to subnets.

Example:

```
# Print all subnets
for s in provider.networking.subnets:
    print(s.id, s.name, s.label)

# Get subnet by ID
s = provider.networking.subnets.get('subnet-id')
print(s.id, s.name, s.label)
```

Return type *SubnetService*

Returns a SubnetService object

SubnetService

class cloudbridge.interfaces.services.**SubnetService**

Base interface for a Subnet Service.

create (*label, network, cidr_block*)

Create a new subnet within the supplied network.

Parameters

- **label** (*str*) – The subnet label.
- **network** (*Network* object or *str*) – Network object or ID under which to create the subnet.
- **cidr_block** (*str*) – CIDR block within the Network to assign to the subnet.

Return type object of *Subnet*

Returns A Subnet object

delete (*subnet*)

Delete an existing Subnet.

Parameters **subnet** (*Subnet* object or *str*) – Subnet object or ID of the subnet to delete.

find (***kwargs*)

Searches for a subnet by a given list of attributes.

Supported attributes: name, label

Return type List of object of *Subnet*

Returns A list of Subnet objects matching the supplied attributes.

get (*subnet_id*)

Returns a Subnet given its ID or None if not found.

Parameters **subnet_id** (*Network* object or *str*) – The ID of the subnet to retrieve.

Return type object of *Subnet*

Returns a Subnet object

get_or_create_default ()

Return a default subnet for the account or create one if not found. This provides a convenience method for obtaining a network if you are not particularly concerned with how the network is structured.

A default network is one marked as such by the provider or matches the default label used by this library (e.g., cloudbridge-net).

Return type object of *Subnet*

Returns A Subnet object

list (*network=None, limit=None, marker=None*)

List all subnets or filter them by the supplied network ID.

Parameters **network** (*str*) – Network object or ID with which to filter the subnets.

Return type list of *Subnet*

Returns list of Subnet objects

FloatingIPService

RouterService

class cloudbridge.interfaces.services.RouterService

Manage networking router actions and resources.

create (*label, network*)

Create a new router.

Parameters

- **label** (*str*) – A router label.
- **network** (*Network* object or *str*) – Network object or ID under which to create the router.

Return type object of *Router*

Returns A Router object

delete (*router*)

Delete an existing Router.

Parameters **router** (*Router* object or *str*) – Router object or ID of the router to delete.

find (***kwargs*)

Searches for a router by a given list of attributes.

Supported attributes: label

Return type List of object of *Router*

Returns A list of Router objects matching the supplied attributes.

get (*router_id*)

Returns a Router object given its ID.

Parameters **router_id** (*str*) – The ID of the router to retrieve.

Return type object of *Router* or None

Returns a Router object of None if not found.

list (*limit=None, marker=None*)

List all routers.

Return type list of *Router*

Returns list of Router objects

GatewayService

BucketService

class cloudbridge.interfaces.services.**BucketService**

The Bucket Service interface provides access to the underlying object storage capabilities of this provider. This service is optional and the `CloudProvider.has_service()` method should be used to verify its availability before using the service.

create (*name, location=None*)

Create a new bucket.

If a bucket with the specified name already exists, return a reference to that bucket.

Example:

```
bucket = provider.storage.buckets.create('my_bucket_name')
print(bucket.name)
```

Parameters

- **name** (*str*) – The name of this bucket.
- **location** (object of *Region*) – The region in which to place this bucket.

Returns a Bucket object

Return type object of *Bucket*

find (***kwargs*)

Searches for a bucket by a given list of attributes.

Supported attributes: name

Example:

```
buckets = provider.storage.buckets.find(name='my_bucket_name')
for bucket in buckets:
    print(bucket.id, bucket.name)
```

Return type *Bucket*

Returns a Bucket instance

get (*bucket_id*)

Returns a bucket given its ID. Returns `None` if the bucket does not exist. On some providers, such as AWS and OpenStack, the bucket id is the same as its name.

Example:

```
bucket = provider.storage.buckets.get('my_bucket_id')
print(bucket.id, bucket.name)
```

Return type *Bucket*

Returns a Bucket instance

list (*limit=None, marker=None*)

List all buckets.

Example:

```
buckets = provider.storage.buckets.find(name='my_bucket_name')
for bucket in buckets:
    print(bucket.id, bucket.name)
```

Return type *Bucket*

Returns list of bucket objects

SecurityService

class cloudbridge.interfaces.services.**SecurityService**

The security service interface can be used to access security related functions in the provider, such as firewall control and keypairs.

key_pairs

Provides access to key pairs for this provider.

Example:

```
# print all keypairs
for kp in provider.security.keypairs:
    print(kp.id, kp.name)

# find keypair by name
kp = provider.security.keypairs.find(name='my_key_pair')[0]
print(kp.id, kp.name)
```

Return type *KeyPairService*

Returns a KeyPairService object

vm_firewalls

Provides access to firewalls (security groups) for this provider.

Example:

```
# print all VM firewalls
for fw in provider.security.vm_firewalls:
    print(fw.id, fw.name)
```

(continues on next page)

(continued from previous page)

```
# find firewall by name
fw = provider.security.vm_firewalls.find(name='my_vm_fw')[0]
print(fw.id, fw.name)
```

Return type `VMFirewallService`

Returns a VMFirewallService object

KeyPairService

class `cloudbridge.interfaces.services.KeyPairService`

Base interface for key pairs.

create (*name*, *public_key_material=None*)

Create a new key pair or raise an exception if one already exists. If the *public_key_material* is provided, the material will be imported to create the new keypair. Otherwise, a new public and private key pair will be generated.

Parameters

- **name** (*str*) – The name of the key pair to be created.
- **public_key_material** (*str*) – The key-pair material to import in OpenSSH format.

Return type object of `KeyPair`

Returns A keypair instance or None.

delete (*key_pair*)

Delete an existing keypair.

Parameters **key_pair** (*str* or `KeyPair`) – The object or id of the key pair to be deleted.

Return type `bool`

Returns `True` if the key does not exist, `False` otherwise. Note that this implies that the key may not have been deleted by this method but instead has not existed at all.

find (***kwargs*)

Searches for a key pair by a given list of attributes.

Supported attributes: name

Return type object of `KeyPair`

Returns a KeyPair object

get (*key_pair_id*)

Return a KeyPair given its ID or None if not found.

On some providers, such as AWS and OpenStack, the KeyPair ID is the same as its name.

Example:

```
key_pair = provider.security.keypairs.get('my_key_pair_id')
print(key_pair.id, key_pair.name)
```

Return type `KeyPair`

Returns a KeyPair instance

list (*limit=None, marker=None*)

List all key pairs associated with this account.

Return type list of *KeyPair*

Returns list of KeyPair objects

VMFirewallService

class cloudbridge.interfaces.services.VMFirewallService

Base interface for VM firewalls.

create (*label, network, description=None*)

Create a new VMFirewall.

Parameters

- **label** (*str*) – The label for the new VM firewall.
- **network** (*str*) – Network ID under which to create the VM firewall.
- **description** (*str*) – The description of the new VM firewall.

Return type object of *VMFirewall*

Returns A VMFirewall instance or None if one was not created.

delete (*vm_firewall*)

Delete an existing VMFirewall.

Parameters **vm_firewall** (*str* or *VMFirewall*) – The object or VM firewall ID to be deleted.

find (***kwargs*)

Get VM firewalls associated with your account filtered by name.

Supported attributes: name

Parameters **name** (*str*) – The name of the VM firewall to retrieve.

Return type list of *VMFirewall*

Returns A list of VMFirewall objects or an empty list if none found.

get (*vm_firewall_id*)

Returns a VMFirewall given its ID. Returns None if the VMFirewall does not exist.

Example:

```
fw = provider.security.vm_firewalls.get('my_fw_id')
print(fw.id, fw.name)
```

Return type *VMFirewall*

Returns a VMFirewall instance

list (*limit=None, marker=None*)

List all VM firewalls associated with this account.

Return type list of *VMFirewall*

Returns list of VMFirewall objects

VMTypeService

class cloudbridge.interfaces.services.VMTypeService

find (**kwargs)

Searches for instances by a given list of attributes.

Supported attributes: name

Return type object of *VMType*

Returns an Instance object

get (vm_type_id)

Returns an VMType given its ID. Returns None if the VMType does not exist.

Example:

```
vm_type = provider.compute.vm_types.get('my_vm_type_id')
print(vm_type.id, vm_type.name)
```

Return type *VMType*

Returns an VMType instance

list (limit=None, marker=None)

List all VM types.

Return type list of *VMType*

Returns list of VMType objects

RegionService

class cloudbridge.interfaces.services.RegionService

Base interface for a Region service

current

Returns the current region that this provider is connected to.

If the current region cannot be discovered, return None.

Return type object of *Region*

Returns a Region instance or None

find (**kwargs)

Searches for a region by a given list of attributes.

Supported attributes: name

Return type object of *Region*

Returns a Region object

get (region_id)

Returns a region given its id. Returns None if the region does not exist.

Return type object of *Region*

Returns a Region instance

list (*limit=None, marker=None*)

List all regions.

Return type list of *Region*

Returns list of region objects

DnsService

class cloudbridge.interfaces.services.DnsService

Base service interface for DNS.

This service offers a collection of DNS services that in turn provide access to DNS resources.

host_zones

Provides access to all dns zones.

Return type *DnsZoneService*

Returns a Dns Host Zone service

DnsZoneService

class cloudbridge.interfaces.services.DnsZoneService

Manage DNS Zone actions and resources. This service is optional and the `CloudProvider.has_service()` method should be used to verify its availability before using the service.

create (*label, admin_email*)

Create a new host zone.

Parameters

- **label** (*str*) – A host zone label.
- **admin_email** (*str*) – Email address of this zone's administrator.

Return type object of *DnsZone*

Returns A DnsZone object

delete (*dns_zone*)

Delete an existing DnsHostZone.

Parameters **dns_zone** (*DnsZone* object or *str*) – DnsZone object or ID of the host zone to delete.

find (***kwargs*)

Searches for a host zone by a given list of attributes.

Supported attributes: label

Return type List of object of *DnsZone*

Returns A list of Dns Zone objects matching the supplied attributes.

get (*dns_zone_id*)

Returns a DnsZone object given its ID.

Parameters **dns_zone_id** (*str*) – The ID of the host zone to retrieve.

Return type object of *DnsZone* or None

Returns a DnsZone object of None if not found.

list (*limit=None, marker=None*)

List all host zones.

Return type list of *DnsZone*

Returns list of DnsZone objects

DnsRecordService

class cloudbridge.interfaces.services.DnsRecordService

The Dns Record Service interface provides access to the records belonging to a Dns Zone.

create (*dns_zone, name, type, data, ttl=None*)

Create a new record within a zone.

Parameters

- **name** (*str*) – The record name.
- **type** (*str*) – The DnsRecord type. (e.g. A, CNAME, MX etc)
- **data** (*int*) – The corresponding value for the record. The relevant values must be fully qualified (e.g. CNAMEs). If the trailing dot is omitted, it will be automatically added and thus assumed to be fully qualified.
- **data** – The ttl (in seconds) for this record.

Return type object of *DnsRecord*

Returns A DnsRecord object

get (*dns_zone, rec_id*)

Returns a record given its ID and the dns_zone containing it. Returns None if the record does not exist.

Return type *DnsRecord*

Returns a DnsRecord instance

list (*dns_zone, limit=None, marker=None*)

List all records within a dns zone.

Return type *DnsRecord*

Returns a DnsRecord instance

4.5.3 Resources

- *CloudServiceType*
- *CloudResource*
- *Configuration*
- *ObjectLifeCycleMixin*
- *PageableObjectMixin*
- *ResultList*
- *InstanceState*
- *Instance*

- *MachineImageState*
- *LaunchConfig*
- *MachineImage*
- *NetworkState*
- *Network*
- *SubnetState*
- *Subnet*
- *FloatingIP*
- *RouterState*
- *Router*
- *Gateway*
- *InternetGateway*
- *VolumeState*
- *Volume*
- *SnapshotState*
- *Snapshot*
- *KeyPair*
- *Region*
- *PlacementZone*
- *VMType*
- *VMFirewall*
- *VMFirewallRule*
- *TrafficDirection*
- *BucketObject*
- *Bucket*
- *DnsZone*
- *DnsRecord*

CloudServiceType

class cloudbridge.interfaces.resources.**CloudServiceType**

Defines possible service types that are offered by providers.

Providers can implement the `has_service` method and clients can check for the availability of a service with:

```
if provider.has_service(CloudServiceTypes.BUCKET):
    ...
```

CloudResource

class `cloudbridge.interfaces.resources.CloudResource`

Base interface for any Resource supported by a provider.

This interface has a `_provider` property that can be used to access the provider associated with the resource, which is only intended for use by subclasses. Every CloudBridge resource also has an `id`, a `name` and a `label` property. The `id` property is a unique identifier for the resource. The `name` is a more user-friendly version of an `id`, suitable for display to an end-user. However, it cannot be used in place of `id`. See `@name` documentation. The `label` property is a user-assignable identifier for the resource.

id

Get the resource identifier.

The `id` property is used to uniquely identify the resource, and is an opaque value which should not be interpreted by CloudBridge clients, and is a value meaningful to the underlying cloud provider.

Return type `str`

Returns ID for this resource as returned by the cloud middleware.

name

Get the name id for the resource.

The `name` property is typically a user-friendly id value for the resource. The `name` is different from the `id` property in the following ways:

1. The `name` property is often a more user-friendly value to display to the user than the `id` property.
2. The `name` may sometimes be the same as the `id`, but should never be used in place of the `id`.
3. The `id` is what will uniquely identify a resource, and will be used internally by CloudBridge for all get operations etc.
4. All resources have a `name`.
5. The `name` is read-only.
6. However, the `name` may not necessarily be unique, which is the reason why it should not be used for uniquely identifying a resource.

Example: The AWS machine image `name` maps to a CloudBridge `name`. It is not editable and is a user friendly name such as 'Ubuntu 18.04' and corresponds to the `ami-name`. It is distinct from the `ami-id`, which maps to CloudBridge's `id` property. The `ami-name` cannot be edited, and is set at creation time. It is not necessarily unique. In Azure, the machine image's `name` corresponds to CloudBridge's `name` property. In Azure, it also happens to be the same as the `id` property.

The `name` property and the `label` property share the same character restrictions. See `label`.

to_json()

Returns a JSON representation of the CloudResource object.

Configuration

class `cloudbridge.interfaces.resources.Configuration`

Represents a CloudBridge configuration object

debug_mode

A flag indicating whether CloudBridge is in debug mode.

Setting this to `True` will cause the underlying provider's debug output to be turned on. The flag can be toggled by sending in the `cb_debug` value via the config dictionary, or setting the `CB_DEBUG` environment variable.

Return type `bool`

Returns Whether debug mode is on.

default_result_limit

Get the default maximum number of results to return for a list method.

The default limit will be applied to most `list()` and `find()` methods whenever an explicit limit is not specified.

Return type `int`

Returns The maximum number of results to return

default_wait_interval

Get the default wait interval for `LifeCycleObjects`.

The default wait interval is applied in `wait_for()` and `wait_till_ready()` methods if no explicit interval is specified.

Return type `int`

Returns How frequently to poll the object's state.

default_wait_timeout

Get the default wait timeout for `LifeCycleObjects`.

The default wait timeout is applied in `wait_for()` and `wait_till_ready()` methods if no explicit timeout is specified.

Return type `int`

Returns The maximum length of time (in seconds) to wait for the object to change to desired state.

ObjectLifeCycleMixin

class `cloudbridge.interfaces.resources.ObjectLifeCycleMixin`

A mixin for an object with a defined life-cycle.

Object examples include an Instance, Volume, Image, or Snapshot. An object that supports `ObjectLifeCycleMixin` will always have a state, defining which point in its life cycle it is currently at.

It also defines a `wait_till_ready` operation, which indicates that the object is in a state in its life cycle where it is ready to be used by an end-user.

A `refresh` operation allows the object to synchronize its state with the service provider.

refresh()

Refresh this object's state and synchronize it with the provider.

state

Get the current state of this object.

Return type `str`

Returns The current state as a string.

wait_for (*target_states*, *terminal_states=None*, *timeout=None*, *interval=None*)

Wait for for an object to reach a one of desired target states.

If the object does not reach the desired state within the specified timeout, a `WaitStateException` will be raised. The optional `terminal_states` property can be used to specify an additional set of states which, should the object reach one, the object thereafter will not transition into the desired target state. Should this happen, a `WaitStateException` will be raised.

Example:

```
instance.wait_for(
    [InstanceState.DELETED, InstanceState.UNKNOWN],
    terminal_states=[InstanceState.ERROR])
```

Parameters

- **target_states** (list of states) – The list of target states to wait for.
- **terminal_states** (list of states) – A list of terminal states after which the object will not transition into a target state. A `WaitStateException` will be raised if the object transition into a terminal state.
- **timeout** (int) – The maximum length of time (in seconds) to wait for the object to changed to desired state. If no timeout is specified, the global `default_wait_timeout` defined in the provider config will apply.
- **interval** (int) – How frequently to poll the object's state (in seconds). If no interval is specified, the global `default_wait_interval` defined in the provider config will apply.

Return type True

Returns Returns True if successful. A `WaitStateException` exception may be thrown by the underlying service if the object cannot get into a ready state (e.g. if the object is in an error state).

wait_till_ready (*timeout*, *interval*)

Wait till the current object reaches its ready state.

An object's ready state is any state where the end-user can successfully interact with the object. Will throw a `WaitStateException` if the object is not ready within the specified timeout.

Parameters

- **timeout** (int) – The maximum length of time (in seconds) to wait for the object to become ready.
- **interval** (int) – How frequently to poll the object's ready state (in seconds).

Return type True

Returns Returns True if successful. A `WaitStateException` exception may be thrown by the underlying service if the object cannot get into a ready state (e.g. if the object is in an error state).

PageableObjectMixin

class cloudbridge.interfaces.resources.**PageableObjectMixin**

A marker interface for objects which support paged iteration.

The list of objects can be iterated over using the `list(limit, marker)` method.

list (*limit=None, marker=None*)

Returns a list of objects up to a maximum limit.

If a limit and marker are specified, the records will be fetched up to the limit starting from the marker onwards. The returned list is a list of class `ResultList`, which has extra properties like `is_truncated`, `supports_total` and `total_records` to provide extra information about record availability.

If limit is not specified, the limit will default to the underlying provider's default limit. Therefore, you need to check the `is_truncated` property to determine whether more records are available.

The total number of results can be determined through the `total_results` property. Not all providers will support returning the `total_results` property, so the `supports_total` property can be used to determine whether a total is supported.

To iterate through all the records, it will be easier to iterate directly through the instances using `__iter__` instead of calling the list method. The `__iter__` method will automatically call the list method to fetch a batch of records at a time.

Example:

```
# get first page of results
instlist = provider.compute.instances.list(limit=50)
for instance in instlist:
    print("Instance Data: {0}".format(instance))
if instlist.supports_total:
    print("Total results: {0}".format(instlist.total_results))
else:
    print("Total records unknown, "
          "but has more data?: {0}".format(instlist.is_truncated))

# Page to next set of results
if (instlist.is_truncated):
    instlist = provider.compute.instances.list(limit=100,
                                              marker=instlist.marker)

# Alternative: iterate through every available record
for instance in provider.compute.instances:
    print(instance)
```

ResultList

class `cloudbridge.interfaces.resources.ResultList`

Provide extra properties to aid with paging through a many results.

This is a wrapper class around a standard Python list class.

Example:

```
# get first page of results
rl = provider.compute.instances.list(limit=50)
for result in rl:
    print("Instance Data: {0}".format(result))
if rl.supports_total:
    print("Total results: {0}".format(rl.total_results))
else:
    print("Total records unknown, "
          "but has more data?: {0}".format(rl.is_truncated))
```

(continues on next page)

(continued from previous page)

```
# Page to next set of results
if (rl.is_truncated)
    rl = provider.compute.instances.list(limit=100,
                                         marker=rl.marker)
```

is_truncated

Indicate whether this result list has more results that can be paged.

marker

An opaque identifier used in paging through very long lists of objects.

This marker can be provided to the list method to get the next set of results.

supports_server_paging

Indicate whether this `ResultList` supports server side paging.

If server side paging is not supported, the result will use client side paging and the data property provides direct access to all available data.

supports_total

Indicate whether can obtain the total number of available results.

The `supports_total` property should be checked before accessing the `total_results` property. This is a provider-specific property.

total_results

Indicate the total number of results for a particular query.

The `supports_total` property should be used to check whether the provider supports returning the total number of results, before accessing this property, or the behavior is indeterminate.

InstanceState

```
class cloudbridge.interfaces.resources.InstanceState
```

Standard states for an instance.

Variables

- **UNKNOWN** – Instance state unknown.
- **PENDING** – Instance is pending
- **CONFIGURING** – Instance is being reconfigured in some way.
- **RUNNING** – Instance is running.
- **REBOOTING** – Instance is rebooting.
- **DELETED** – Instance is deleted. No further operations possible.
- **STOPPED** – Instance is stopped. Instance can be resumed.
- **ERROR** – Instance is in an error state. No further operations possible.

Instance

```
class cloudbridge.interfaces.resources.Instance
```

```
add_floating_ip(floating_ip)
```

Add a public IP address to this instance.

Parameters `floating_ip` (:class: `.FloatingIP` or floating IP ID) – The `FloatingIP` object to associate with the instance. Note that is not the actual public IP address but the CloudBridge object encapsulating the IP or the respective provider ID that identifies the address.

add_vm_firewall (*firewall*)

Add a VM firewall to this instance

Parameters `firewall` (:class: `.VMFirewall`) – The `VMFirewall` to associate with the instance.

create_image (*label*)

Create a new image based on this instance.

Return type :class: `.Image`

Returns an `Image` object

delete ()

Permanently delete this instance.

image_id

Get the image ID for this instance.

Return type `str`

Returns Image ID (i.e., AMI) this instance is using.

key_pair_id

Get the id of the key pair associated with this instance.

Return type `str`

Returns Id of the ssh key pair associated with this instance.

label

Get the resource label.

The label property is a user-defined, editable identifier for a resource. It will often correspond to a user editable resource label in the underlying cloud provider, or be simulated through tags/labels.

The label property adheres to the following restrictions:

- Must be at least 3 characters in length.
- Cannot be longer than 63 characters.
- May only contain ASCII characters comprising of lowercase letters, numeric characters, and dashes.
- Must begin with an alphanumeric character and end with one (i.e. cannot begin or end with a dash)

Some resources may not support labels, in which case, a `NotImplementedError` will be thrown.

Return type `str`

Returns Label for this resource as returned by the cloud middleware.

Raise `NotImplementedError` if this resource does not support labels.

private_ips

Get all the private IP addresses for this instance.

Return type `list`

Returns A list of private IP addresses associated with this instance.

public_ips

Get all the public IP addresses for this instance.

Return type `list`

Returns A list of public IP addresses associated with this instance.

reboot ()

Reboot this instance (using the cloud middleware API).

Return type `bool`

Returns `True` if the reboot was successful; `False` otherwise.

remove_floating_ip (*floating_ip*)

Remove a public IP address from this instance.

Parameters **floating_ip** (:class:`.FloatingIP` or floating IP ID) – The `FloatingIP` object to remove from the instance. Note that is not the actual public IP address but the `CloudBridge` object encapsulating the IP or the respective provider ID that identifies the address.

remove_vm_firewall (*firewall*)

Remove a VM firewall from this instance

Parameters **firewall** (`VMFirewall`) – The `VMFirewall` to associate with the instance.

subnet_id

Get the subnet ID where this instance is placed.

Return type `str`

Returns Subnet ID to which this instance is connected.

vm_firewall_ids

Get the IDs of the VM firewalls associated with this instance.

Return type `list` or :class:`str`

Returns A list of the `VMFirewall` IDs associated with this instance.

vm_firewalls

Get the firewalls (security groups) associated with this instance.

Return type `list` or `VMFirewall` objects

Returns A list of `VMFirewall` objects associated with this instance.

vm_type

Retrieve full VM type information for this instance.

Return type `VMType`

Returns VM type for this instance

vm_type_id

Get the VM type id for this instance.

This will typically be a string value like `'m1.large'`. On `OpenStack`, this may be a number or `UUID`. To get the full :class:`VMType` object, you can use the `instance.vm_type` property instead.

Return type `str`

Returns VM type id for this instance (e.g., `m1.large`)

zone_id

Get the placement zone ID where this instance is running.

Return type `str`

Returns Region/zone/placement where this instance is running.

MachineImageState

class cloudbridge.interfaces.resources.**MachineImageState**
Standard states for a machine image

Variables

- **UNKNOWN** – Image state unknown.
- **PENDING** – Image is pending
- **AVAILABLE** – Image is available
- **ERROR** – Image is in an error state. Not recoverable.

LaunchConfig

class cloudbridge.interfaces.resources.**LaunchConfig**
Represents an advanced launch configuration object.

This object can contain information such as `BlockDeviceMappings` configurations and other advanced options, which may be useful when launching an instance.

Example:

```
lc = provider.compute.instances.create_launch_config()
lc.add_block_device(...)

inst = provider.compute.instances.create(
    'MyVM', image, vm_type, subnet, launch_config=lc)
```

add_ephemeral_device()

Add a new ephemeral block device mapping to the boot configuration.

This can be used to add existing ephemeral devices to the instance (the total number of ephemeral devices available for a particular `VMType` can be determined by querying the `VMType` service). Note that on some providers, such as AWS, ephemeral devices must be added in as a device mapping at instance creation time and cannot be added afterwards.

Note that the device name, such as `/dev/sda1`, cannot be selected at present, since this tends to be provider and VM type specific. However, the order of device addition coupled with device type will generally determine naming order, with devices added first getting lower letters than instances added later.

Example:

```
lc = provider.compute.instances.create_launch_config()

# 1. Add all available ephemeral devices
vm_type = provider.compute.vm_types.find(name='m1.tiny')[0]
for i in range(vm_type.num_ephemeral_disks):
    lc.add_ephemeral_device()
```

add_volume_device (*source=None, is_root=None, size=None, delete_on_terminate=None*)

Add a new volume based block device mapping to the boot configuration.

The volume can be based on a snapshot, image, existing volume or be a blank new volume, and is specified by the `source` parameter.

The property `is_root` can be set to `True` to override any existing root device mappings. Otherwise, the default behavior is to add new block devices to the instance.

Note that the device name, such as `/dev/sda1`, cannot be selected at present since this tends to be provider and VM type specific. However, the order of device addition coupled with device type will generally determine naming order, with devices added first getting lower letters than instances added later (except when `is_root` is set).

Example:

```
lc = provider.compute.instances.create_launch_config()

# 1. Create and attach an empty volume of size 100GB
lc.add_volume_device(size=100, delete_on_terminate=True)

# 2. Create and attach a volume based on a snapshot
snap = provider.storage.snapshots.get('<my_snapshot_id>')
lc.add_volume_device(source=snap)

# 3. Create+attach a volume based on an image and set it as root
img = provider.compute.images.get('<my_image_id>')
lc.add_volume_device(source=img, size=100, is_root=True)
```

Parameters

- **source** (Volume, Snapshot, Image or None.) – The source `block_device` to add. If Volume, the volume will be attached directly to the instance. If Snapshot, a volume will be created based on the Snapshot and attached to the instance. If Image, a volume based on the Image will be attached to the instance. If None, the source is assumed to be a blank volume.
- **is_root** (bool) – Determines which device will serve as the root device. If more than one device is defined as root, an `InvalidConfigurationException` will be thrown.
- **size** (int) – The size of the volume to create. An implementation may ignore this parameter for certain sources like ‘Volume’.
- **delete_on_terminate** (bool) – Determines whether to delete or keep the volume on instance termination.

MachineImage

```
class cloudbridge.interfaces.resources.MachineImage
```

```
delete()
```

Delete this image.

Return type bool

Returns True if the operation succeeded.

```
description
```

Get the image description.

Return type str

Returns Description for this image as returned by the cloud middleware.

```
min_disk
```

Return the minimum size of the disk that’s required to boot this image.

Value returned is in gigabytes.

Return type `int`

Returns The minimum disk size needed by this image.

NetworkState

class `cloudbridge.interfaces.resources.NetworkState`

Standard states for a network.

Variables

- **UNKNOWN** – Network state unknown.
- **PENDING** – Network is being created.
- **AVAILABLE** – Network is available.
- **DOWN** – Network is not operational.
- **ERROR** – Network is in error state.

Network

class `cloudbridge.interfaces.resources.Network`

Represents a software-defined network, like the Virtual Private Cloud.

`cidr_block`

A CIDR block for this network.

Note: OpenStack does not define a CIDR block for networks.

Return type `str`

Returns A CIDR block string.

`delete()`

Delete this network.

Return type `bool`

Returns `True` if successful.

`external`

A flag to indicate if this network is capable of Internet-connectivity.

Return type `bool`

Returns `True` if the network can be connected to the Internet.

`gateways`

Provides access to the internet gateways attached to this network.

Return type `GatewaySubService`

Returns A `GatewaySubService` object

label

Get the resource label.

The label property is a user-defined, editable identifier for a resource. It will often correspond to a user editable resource label in the underlying cloud provider, or be simulated through tags/labels.

The label property adheres to the following restrictions:

- Must be at least 3 characters in length.
- Cannot be longer than 63 characters.
- May only contain ASCII characters comprising of lowercase letters, numeric characters, and dashes.
- Must begin with an alphanumeric character and end with one (i.e. cannot begin or end with a dash)

Some resources may not support labels, in which case, a `NotImplementedError` will be thrown.

Return type `str`

Returns Label for this resource as returned by the cloud middleware.

Raise `NotImplementedError` if this resource does not support labels.

state

The state of the network.

Return type `str`

Returns One of unknown, pending, available, down or error.

subnets

The associated subnets.

Return type list of *Subnet*

Returns List of subnets associated with this network.

SubnetState

class `cloudbridge.interfaces.resources.SubnetState`
Standard states for a subnet.

Variables

- **UNKNOWN** – Subnet state unknown.
- **PENDING** – Subnet is being created.
- **AVAILABLE** – Subnet is available.
- **DOWN** – Subnet is not operational.
- **ERROR** – Subnet is in error state.

Subnet

class `cloudbridge.interfaces.resources.Subnet`
Represents a subnet, as part of a Network.

cidr_block

A CIDR block for this subnet.

Return type `str`

Returns A CIDR block string.

delete()

Delete this subnet.

Return type `bool`

Returns `True` if successful.

label

Get the resource label.

The label property is a user-defined, editable identifier for a resource. It will often correspond to a user editable resource label in the underlying cloud provider, or be simulated through tags/labels.

The label property adheres to the following restrictions:

- Must be at least 3 characters in length.
- Cannot be longer than 63 characters.
- May only contain ASCII characters comprising of lowercase letters, numeric characters, and dashes.
- Must begin with an alphanumeric character and end with one (i.e. cannot begin or end with a dash)

Some resources may not support labels, in which case, a `NotImplementedError` will be thrown.

Return type `str`

Returns Label for this resource as returned by the cloud middleware.

Raise `NotImplementedError` if this resource does not support labels.

network

The parent network object associated with this this subnet.

Return type `Network`

Returns `Network` object

network_id

ID of the network associated with this this subnet.

Return type `str`

Returns Network ID.

zone

Placement zone of the subnet.

If the provider does not support subnet placement, return `None`.

Return type `PlacementZone` object

Returns Placement zone of the subnet, or `None` if not defined.

FloatingIP

class `cloudbridge.interfaces.resources.FloatingIP`

Represents a floating (i.e., static) IP address.

delete()

Delete this address.

Return type `bool`

Returns `True` if successful.

in_use

Whether the address is in use or not.

Return type `bool`

Returns `True` if the address is attached to an instance.

private_ip

Private IP address this address is attached to.

Return type `str`

Returns IP address or `None`.

public_ip

Public IP address.

Return type `str`

Returns IP address.

RouterState

```
class cloudbridge.interfaces.resources.RouterState
```

Standard states for a router.

Variables

- **UNKNOWN** – Router state unknown.
- **ATTACHED** – Router is attached to a network and should be operational.
- **DETACHED** – Router is detached from a network.

Router

```
class cloudbridge.interfaces.resources.Router
```

Represents a private network router.

This logical router is meant to roughly mimic the properties of a physical router. Therefore, attaching a subnet can be thought of as plugging in a network cable to enable routing to/from that subnet. Attaching a gateway can be thought of as plugging in an upstream link.

attach_gateway (*gateway*)

Attach a gateway to this router.

Parameters **gateway** (`Gateway`) – The Gateway to attach to this router.

Return type `bool`

Returns `True` if successful.

attach_subnet (*subnet*)

Attach this router to a subnet.

Parameters **subnet** (`Subnet` or `str`) – The subnet to which to attach this router.

Return type `bool`

Returns `True` if successful.

delete ()

Delete this router.

Return type `bool`

Returns `True` if successful.

detach_gateway (*gateway*)

Detach this router from a gateway.

Return type `bool`

Returns `True` if successful.

detach_subnet (*subnet*)

Detach this subnet from a network.

Parameters **subnet** (`Subnet` or `str`) – The subnet to detach from this router.

Return type `bool`

Returns `True` if successful.

label

Get the resource label.

The label property is a user-defined, editable identifier for a resource. It will often correspond to a user editable resource label in the underlying cloud provider, or be simulated through tags/labels.

The label property adheres to the following restrictions:

- Must be at least 3 characters in length.
- Cannot be longer than 63 characters.
- May only contain ASCII characters comprising of lowercase letters, numeric characters, and dashes.
- Must begin with an alphanumeric character and end with one (i.e. cannot begin or end with a dash)

Some resources may not support labels, in which case, a `NotImplementedError` will be thrown.

Return type `str`

Returns Label for this resource as returned by the cloud middleware.

Raise `NotImplementedError` if this resource does not support labels.

network_id

ID of the network to which the router is attached.

Return type `str`

Returns ID for the attached network or `None`.

state

Router state: attached or detached to a network.

Return type `str`

Returns `attached` or `detached`.

subnets

List of subnets attached to this router.

Return type list of `Subnet` objects

Returns A list of subnets associated with this router.

Gateway

class `cloudbridge.interfaces.resources.Gateway`

Represents a gateway resource.

delete()

Delete this gateway. On some providers, if the gateway is public/a singleton, this operation will do nothing.

floating_ips

Provides access to floating IPs connected to this internet gateway.

Return type `FloatingIPSubService`

Returns A `FloatingIPSubService` object

network_id

ID of the network to which the gateway is attached.

Return type `str`

Returns ID for the attached network or `None`.

InternetGateway

class `cloudbridge.interfaces.resources.InternetGateway`

Represents an Internet gateway resource.

VolumeState

class `cloudbridge.interfaces.resources.VolumeState`

Standard states for a volume

Variables

- **UNKNOWN** – Volume state unknown.
- **CREATING** – Volume is being created.
- **CONFIGURING** – Volume is being configured in some way.
- **AVAILABLE** – Volume is available and can be attached to an instance.
- **IN_USE** – Volume is attached and in-use.
- **DELETED** – Volume has been deleted. No further operations possible.
- **ERROR** – Volume is in an error state. No further operations possible.

Volume

class `cloudbridge.interfaces.resources.Volume`

Represents a block storage device (aka volume).

attach (*instance*, *device*)

Attach this volume to an instance.

Parameters

- **instance** (`str` or `Instance` object) – The ID of an instance or an `Instance` object to which this volume will be attached.

- **device** (`str`) – The device on the instance through which the volume will be exposed (e.g. `/dev/sdh`).

Return type `bool`

Returns `True` if successful.

attachments

Get attachment information for this volume.

Return type `AttachmentInfo`

Returns Returns an `AttachmentInfo` object.

create_snapshot (*label*, *description=None*)

Create a snapshot of this Volume.

Parameters

- **label** (`str`) – The label for this snapshot.
- **description** (`str`) – A description of the snapshot. Limited to 256 characters.

Return type `Snapshot`

Returns The created `Snapshot` object.

create_time

Get the creation data and time for this volume.

Return type `DateTime`

Returns Creation time for this volume as returned by the cloud middleware.

delete ()

Delete this volume.

Return type `bool`

Returns `True` if successful.

description

Get the volume description.

Some cloud providers may not support this property, and will return the volume label instead.

Return type `str`

Returns Description for this volume as returned by the cloud middleware.

detach (*force=False*)

Detach this volume from an instance.

Parameters **force** (`bool`) – Forces detachment if the previous detachment attempt did not occur cleanly. This option is supported on select clouds only. This option can lead to data loss or a corrupted file system. Use this option only as a last resort to detach a volume from a failed instance. The instance will not have an opportunity to flush file system caches nor file system meta data. If you use this option, you must perform file system check and repair procedures.

Return type `bool`

Returns `True` if successful.

label

Get the resource label.

The label property is a user-defined, editable identifier for a resource. It will often correspond to a user editable resource label in the underlying cloud provider, or be simulated through tags/labels.

The label property adheres to the following restrictions:

- Must be at least 3 characters in length.
- Cannot be longer than 63 characters.
- May only contain ASCII characters comprising of lowercase letters, numeric characters, and dashes.
- Must begin with an alphanumeric character and end with one (i.e. cannot begin or end with a dash)

Some resources may not support labels, in which case, a `NotImplementedError` will be thrown.

Return type `str`

Returns Label for this resource as returned by the cloud middleware.

Raise `NotImplementedError` if this resource does not support labels.

size

Get the volume size (in GB).

Return type `int`

Returns Size for this volume as returned by the cloud middleware.

source

If available, get the source that this volume is based on.

This can be a `Snapshot`, an `Image`, or `None` if no source.

Return type `Snapshot`, `Image`, or `None`

Returns `Snapshot` or `Image` source for this volume as returned by the cloud middleware.

zone_id

Get the placement zone id that this volume belongs to.

Return type `str`

Returns `PlacementZone` for this volume as returned by the cloud middleware.

SnapshotState

class `cloudbridge.interfaces.resources.SnapshotState`

Standard states for a snapshot

Variables

- **UNKNOWN** – Snapshot state unknown.
- **PENDING** – Snapshot is pending.
- **CONFIGURING** – Snapshot is being configured in some way.
- **AVAILABLE** – Snapshot has been completed and is ready for use.
- **ERROR** – Snapshot is in an error state. No further operations possible.

Snapshot

class `cloudbridge.interfaces.resources.Snapshot`

Represents a snapshot of a block storage device.

create_time

Get the creation data and time for this snapshot.

Return type `DateTime`

Returns Creation time for this snapshot as returned by the cloud middleware.

create_volume (*size=None, volume_type=None, iops=None*)

Create a new Volume from this Snapshot.

Parameters

- **size** (`int`) – The size of the new volume, in GiB (optional). Defaults to the size of the snapshot.
- **volume_type** (`str`) – The type of the volume (optional). Availability and valid values depend on the provider.
- **iops** (`int`) – The provisioned IOPs you want to associate with this volume (optional). Availability depends on the provider.

Return type `Volume`

Returns An instance of the created Volume.

delete ()

Delete this snapshot.

Return type `bool`

Returns `True` if successful.

description

Get the snapshot description.

Some cloud providers may not support this property, and will return the snapshot label instead.

Return type `str`

Returns Description for this snapshot as returned by the cloud middleware.

label

Get the resource label.

The label property is a user-defined, editable identifier for a resource. It will often correspond to a user editable resource label in the underlying cloud provider, or be simulated through tags/labels.

The label property adheres to the following restrictions:

- Must be at least 3 characters in length.
- Cannot be longer than 63 characters.
- May only contain ASCII characters comprising of lowercase letters, numeric characters, and dashes.
- Must begin with an alphanumeric character and end with one (i.e. cannot begin or end with a dash)

Some resources may not support labels, in which case, a `NotImplementedError` will be thrown.

Return type `str`

Returns Label for this resource as returned by the cloud middleware.

Raise `NotImplementedError` if this resource does not support labels.

size

Get the snapshot size (in GB).

Return type `int`

Returns Size for this snapshot as returned by the cloud middleware.

volume_id

Get the id of the volume that this snapshot is based on.

This method may return `None` if the source volume no longer exists.

Return type `int`

Returns Id of the volume that this snapshot is based on

KeyPair

class `cloudbridge.interfaces.resources.KeyPair`

Represents an ssh key pair.

delete()

Delete this key pair.

Return type `bool`

Returns `True` if successful.

material

Unencrypted private key.

Return type `str`

Returns Unencrypted private key or `None` if not available.

Region

class `cloudbridge.interfaces.resources.Region`

Represents a cloud region.

A cloud region is typically a separate geographic area and will contain at least one placement zone.

zones

Access information about placement zones within this region.

Return type `Iterable`

Returns `Iterable` of available placement zones in this region.

PlacementZone

class `cloudbridge.interfaces.resources.PlacementZone`

Represents a placement zone.

A placement zone is contained within a `Region`.

region_name

A region this placement zone is associated with.

Return type `str`

Returns The id of the region the zone is associated with.

VMType

class cloudbridge.interfaces.resources.VMType

A VM type object.

extra_data

A dictionary of extra data about this instance. May contain nested dictionaries, but all key value pairs are strings or integers.

Return type dict

Returns Extra attributes for this VM type.

family

The family/group that this VM type belongs to.

For example, General Purpose Instances or High-Memory Instances. If the provider does not support such a grouping, it may return None.

Return type str

Returns Name of the instance family or None.

num_ephemeral_disks

The total number of ephemeral disks on this VM type.

Return type int

Returns Number of ephemeral disks available.

ram

The amount of RAM (in GB) supported by this VM type.

Return type float

Returns Total RAM (in GB).

size_ephemeral_disks

The size of this VM types's total ephemeral storage (in GB).

Return type int

Returns Size of ephemeral disks (in GB).

size_root_disk

The size of this VM types's root disk (in GB).

Return type int

Returns Size of root disk (in GB).

size_total_disk

The total disk space available on this VM type (root_disk + ephemeral).

Return type int

Returns Size of total disk space (in GB).

vcpus

The number of VCPUs supported by this VM type.

Return type int

Returns Number of VCPUs.

VMFirewall

class `cloudbridge.interfaces.resources.VMFirewall`

Represents a firewall resource applied to virtual machines.

This is in contrast to a firewall for a network, for example.

description

Return the description of this VM firewall.

Return type `str`

Returns A description of this VM firewall.

label

Get the resource label.

The label property is a user-defined, editable identifier for a resource. It will often correspond to a user editable resource label in the underlying cloud provider, or be simulated through tags/labels.

The label property adheres to the following restrictions:

- Must be at least 3 characters in length.
- Cannot be longer than 63 characters.
- May only contain ASCII characters comprising of lowercase letters, numeric characters, and dashes.
- Must begin with an alphanumeric character and end with one (i.e. cannot begin or end with a dash)

Some resources may not support labels, in which case, a `NotImplementedError` will be thrown.

Return type `str`

Returns Label for this resource as returned by the cloud middleware.

Raise `NotImplementedError` if this resource does not support labels.

network_id

Network ID with which this VM firewall is associated.

Return type `str`

Returns Provider-supplied network ID or `None` is not available.

rules

Get access to the rules belonging to this VM firewall.

This object can be used for further operations on rules, such as get, list, create, etc.

Return type An object of `VMFirewallRuleSubService`

Returns A `VMFirewallRuleSubService` for further operations

VMFirewallRule

class `cloudbridge.interfaces.resources.VMFirewallRule`

Represents a VM firewall rule.

cidr

CIDR block this VM firewall is providing access to.

Return type `str`

Returns CIDR block.

delete()
Delete this rule.

direction
Direction of traffic to which this rule applies.
Either `TrafficDirection.INBOUND` or `TrafficDirection.OUTBOUND`.

Return type `str`
Returns Direction of traffic to which this rule applies.

from_port
Lowest port number opened as part of this rule.

Return type `int`
Returns Lowest port number or 0 if not set.

protocol
IP protocol used. Either `tcp` | `udp` | `icmp`.

Return type `str`
Returns Active protocol.

src_dest_fw
VM firewall given access permissions by this rule.

Return type `:class: .VMFirewall`
Returns The VM firewall granted access.

src_dest_fw_id
VM firewall id given access permissions by this rule.

Return type `str`
Returns The VM firewall granted access.

to_port
Highest port number opened as part of this rule.

Return type `int`
Returns Highest port number or 0 if not set.

TrafficDirection

class `cloudbridge.interfaces.resources.TrafficDirection`
Direction of data flow in a firewall.

BucketObject

class `cloudbridge.interfaces.resources.BucketObject`
Represents an object stored within a bucket.

delete()
Delete this object.

Return type `bool`
Returns `True` if successful.

generate_url (*expires_in*)

Generate a signed URL to this object.

A signed URL associated with an object gives time-limited read access to that specific object. Anyone in possession of the URL has the access granted by the URL.

Parameters **expires_in** (*int*) – Time to live of the generated URL in seconds.

Return type *str*

Returns A URL to access the object.

iter_content ()

Returns this object's content as an iterable.

Return type *Iterable*

Returns An iterable of the file contents

last_modified

Get the date and time this object was last modified.

Return type *str*

Returns Date and time formatted string *%Y-%m-%dT%H:%M:%S.%f*

name

Retrieve the name of the current object.

The bucket object name adheres to a naming requirement that is more relaxed than the naming requirement enforced across CloudBridge. More details are available here: <http://docs.aws.amazon.com/AmazonS3/latest/dev/UsingMetadata.html#object-key-guidelines>

Return type *str*

Returns Name for this object as returned by the cloud middleware.

refresh ()

Refresh this object's state and synchronize it with the underlying service provider.

save_content (*target_stream*)

Save this object and write its contents to the *target_stream*.

size

Get this object's size.

Return type *int*

Returns Size of this object in bytes.

upload (*source_stream*)

Set the contents of the object to the data read from the source stream.

Return type *bool*

Returns *True* if successful.

upload_from_file (*path*)

Store the contents of the file pointed by the "path" variable.

Parameters **path** (*str*) – Absolute path to the file to be uploaded to S3.

Bucket

class `cloudbridge.interfaces.resources.Bucket`

Represents a namespace for objects (eg, object store bucket or container).

delete (*delete_contents=False*)

Delete this bucket.

Parameters `delete_contents` (bool) – If True, all objects within the bucket will be deleted.

Return type bool

Returns True if successful.

name

Retrieve the name of the current bucket.

Return type str

Returns Name for this instance as returned by the cloud middleware.

objects

Get a container for the objects belonging to this Bucket.

This object can be used to iterate through bucket objects, as well as perform further operations on buckets, such as get, list, create, etc.

```
# Show all objects in bucket
print(list(bucket.objects))

# Find an object by name
print(bucket.objects.find(name='my_obj.txt'))

# Get first page of bucket list
print(bucket.objects.list())

# Create a new object within this bucket
obj = bucket.objects.create('my_obj.txt')
```

Return type BucketContainer

Returns A BucketContainer for further operations.

DnsZone

class `cloudbridge.interfaces.resources.DnsZone`

Represents a dns host zone.

A host zone represents a top level domain (e.g. cloudve.org) in which multiple dns records (e.g. A, CNAME, MX etc.) are contained.

admin_email

Email address of this zone's administrator. Some cloud providers do not support this field, and therefore, it may be stored in an extra field such as description or not supported at all. (This field is mandatory in OpenStack)

Returns Administrator's email as a string

delete ()

Delete this zone.

records

List of DNS records in this zone.

Return type list of *DnsRecord* objects

Returns A list of DnsRecords associated with this zone.

DnsRecord

class `cloudbridge.interfaces.resources.DnsRecord`

Represents a dns record.

A dns record belongs to a host zone and can contain records of various types such as A, CNAME, MX etc.

data

Dns Record data

Return type `str`

Returns A string containing this DNS record's data.

ttl

ttl for this record

Return type `int`

Returns The ttl (in seconds) for this record.

type

Dns Record type which could be A, CNAME, MX, AAAA, PTR

Return type `DnsRecordType`

Returns An enum representing the DNS record type.

zone_id

The containing zone for this dns record

Return type `str`

Returns The ID of the zone for this dns record

4.5.4 Exceptions

- *CloudBridgeBaseException*
- *WaitStateException*
- *InvalidConfigurationException*
- *ProviderConnectionException*
- *InvalidLabelException*
- *InvalidValueException*

CloudBridgeBaseException

class `cloudbridge.interfaces.exceptions.CloudBridgeBaseException`
Base class for all CloudBridge exceptions

WaitStateException

class `cloudbridge.interfaces.exceptions.WaitStateException`
Marker interface for object wait exceptions. Thrown when a timeout or errors occurs waiting for an object does not reach the expected state within a specified time limit.

InvalidConfigurationException

class `cloudbridge.interfaces.exceptions.InvalidConfigurationException`
Marker interface for invalid launch configurations. Thrown when a combination of parameters in a LaunchConfig object results in an illegal state.

ProviderConnectionException

class `cloudbridge.interfaces.exceptions.ProviderConnectionException`
Marker interface for connection errors to a cloud provider. Thrown when CloudBridge is unable to connect with a provider, for example, when credentials are incorrect, or connection settings are invalid.

InvalidLabelException

class `cloudbridge.interfaces.exceptions.InvalidLabelException` (*msg*)
Marker interface for any attempt to set an invalid label on a CloudBridge resource. An example would be setting uppercase letters, which are not allowed in a resource label. InvalidLabelExceptions inherit from, and are a special case of InvalidNameExceptions. At present, these restrictions are identical.

InvalidValueException

class `cloudbridge.interfaces.exceptions.InvalidValueException` (*param, value*)
Marker interface for any attempt to set an invalid value on a CloudBridge resource. An example would be setting an unrecognised value for the direction of a firewall rule other than TrafficDirection.INBOUND or TrafficDirection.OUTBOUND.

CHAPTER 5

Page index

- `genindex`

Symbols

`__init__()` (*cloudbridge.interfaces.provider.CloudProvider* method), 68

A

`add_ephemeral_device()` (*cloudbridge.interfaces.resources.LaunchConfig* method), 97

`add_floating_ip()` (*cloudbridge.interfaces.resources.Instance* method), 94

`add_vm_firewall()` (*cloudbridge.interfaces.resources.Instance* method), 95

`add_volume_device()` (*cloudbridge.interfaces.resources.LaunchConfig* method), 97

`admin_email` (*cloudbridge.interfaces.resources.DnsZone* attribute), 113

`attach()` (*cloudbridge.interfaces.resources.Volume* method), 104

`attach_gateway()` (*cloudbridge.interfaces.resources.Router* method), 102

`attach_subnet()` (*cloudbridge.interfaces.resources.Router* method), 102

`attachments` (*cloudbridge.interfaces.resources.Volume* attribute), 105

`authenticate()` (*cloudbridge.interfaces.provider.CloudProvider* method), 68

B

`Bucket` (class in *cloudbridge.interfaces.resources*), 113

`BucketObject` (class in *cloudbridge.interfaces.resources*), 111

`buckets` (*cloudbridge.interfaces.services.StorageService* attribute), 77

`BucketService` (class in *cloudbridge.interfaces.services*), 82

C

`cidr` (*cloudbridge.interfaces.resources.VMFirewallRule* attribute), 110

`cidr_block` (*cloudbridge.interfaces.resources.Network* attribute), 99

`cidr_block` (*cloudbridge.interfaces.resources.Subnet* attribute), 100

`clone()` (*cloudbridge.interfaces.provider.CloudProvider* method), 69

`CloudBridgeBaseException` (class in *cloudbridge.interfaces.exceptions*), 115

`CloudProvider` (class in *cloudbridge.interfaces.provider*), 68

`CloudResource` (class in *cloudbridge.interfaces.resources*), 90

`CloudService` (class in *cloudbridge.interfaces.services*), 72

`CloudServiceType` (class in *cloudbridge.interfaces.resources*), 89

`compute` (*cloudbridge.interfaces.provider.CloudProvider* attribute), 69

`ComputeService` (class in *cloudbridge.interfaces.services*), 73

`config` (*cloudbridge.interfaces.provider.CloudProvider* attribute), 69

`Configuration` (class in *cloudbridge.interfaces.resources*), 90

`create()` (*cloudbridge.interfaces.services.BucketService* method), 82

`create()` (*cloudbridge.interfaces.services.DnsRecordService* method), 88

`create()` (*cloudbridge.interfaces.services.DnsZoneService* method), 87

`create()` (*cloudbridge.interfaces.services.InstanceService* method), 74

`create()` (`cloudbridge.interfaces.services.KeyPairService` delete method), 84
`create()` (`cloudbridge.interfaces.services.NetworkService` delete method), 79
`create()` (`cloudbridge.interfaces.services.RouterService` delete method), 81
`create()` (`cloudbridge.interfaces.services.SnapshotService` delete method), 76
`create()` (`cloudbridge.interfaces.services.SubnetService` delete method), 80
`create()` (`cloudbridge.interfaces.services.VMFirewallService` delete method), 85
`create()` (`cloudbridge.interfaces.services.VolumeService` delete method), 76
`create_image()` (`cloudbridge.interfaces.resources.Instance` delete method), 95
`create_launch_config()` (`cloudbridge.interfaces.services.InstanceService` delete method), 75
`create_snapshot()` (`cloudbridge.interfaces.resources.Volume` delete method), 105
`create_time` (`cloudbridge.interfaces.resources.Snapshot` attribute), 107
`create_time` (`cloudbridge.interfaces.resources.Volume` attribute), 105
`create_volume()` (`cloudbridge.interfaces.resources.Snapshot` delete method), 107
`current` (`cloudbridge.interfaces.services.RegionService` attribute), 86

D

`data` (`cloudbridge.interfaces.resources.DnsRecord` attribute), 114
`debug_mode` (`cloudbridge.interfaces.resources.Configuration` attribute), 90
`default_result_limit` (`cloudbridge.interfaces.resources.Configuration` attribute), 91
`default_wait_interval` (`cloudbridge.interfaces.resources.Configuration` attribute), 91
`default_wait_timeout` (`cloudbridge.interfaces.resources.Configuration` attribute), 91
`delete()` (`cloudbridge.interfaces.resources.Bucket` delete method), 113
`delete()` (`cloudbridge.interfaces.resources.BucketObject` delete method), 111
`delete()` (`cloudbridge.interfaces.resources.DnsZone` delete method), 113
`delete()` (`cloudbridge.interfaces.resources.FloatingIP` delete method), 101
`delete()` (`cloudbridge.interfaces.resources.Gateway` delete method), 104
`delete()` (`cloudbridge.interfaces.resources.Instance` delete method), 95
`delete()` (`cloudbridge.interfaces.resources.KeyPair` delete method), 108
`delete()` (`cloudbridge.interfaces.resources.MachineImage` delete method), 98
`delete()` (`cloudbridge.interfaces.resources.Network` delete method), 99
`delete()` (`cloudbridge.interfaces.resources.Router` delete method), 102
`delete()` (`cloudbridge.interfaces.resources.Snapshot` delete method), 107
`delete()` (`cloudbridge.interfaces.resources.Subnet` delete method), 101
`delete()` (`cloudbridge.interfaces.resources.VMFirewallRule` delete method), 110
`delete()` (`cloudbridge.interfaces.resources.Volume` delete method), 105
`delete()` (`cloudbridge.interfaces.services.DnsZoneService` delete method), 87
`delete()` (`cloudbridge.interfaces.services.KeyPairService` delete method), 84
`delete()` (`cloudbridge.interfaces.services.NetworkService` delete method), 79
`delete()` (`cloudbridge.interfaces.services.RouterService` delete method), 81
`delete()` (`cloudbridge.interfaces.services.SnapshotService` delete method), 77
`delete()` (`cloudbridge.interfaces.services.SubnetService` delete method), 80
`delete()` (`cloudbridge.interfaces.services.VMFirewallService` delete method), 85
`delete()` (`cloudbridge.interfaces.services.VolumeService` delete method), 76
`description` (`cloudbridge.interfaces.resources.MachineImage` attribute), 98
`description` (`cloudbridge.interfaces.resources.Snapshot` attribute), 107
`description` (`cloudbridge.interfaces.resources.VMFirewall` attribute), 110
`description` (`cloudbridge.interfaces.resources.Volume` attribute), 105
`detach()` (`cloudbridge.interfaces.resources.Volume` delete method), 105

detach_gateway() (cloudbridge.interfaces.resources.Router method), 103

detach_subnet() (cloudbridge.interfaces.resources.Router method), 103

direction (cloudbridge.interfaces.resources.VMFirewallRule attribute), 111

dns (cloudbridge.interfaces.provider.CloudProvider attribute), 70

DnsRecord (class in cloudbridge.interfaces.resources), 114

DnsRecordService (class in cloudbridge.interfaces.services), 88

DnsService (class in cloudbridge.interfaces.services), 87

DnsZone (class in cloudbridge.interfaces.resources), 113

DnsZoneService (class in cloudbridge.interfaces.services), 87

E

external (cloudbridge.interfaces.resources.Network attribute), 99

extra_data (cloudbridge.interfaces.resources.VMType attribute), 109

F

family (cloudbridge.interfaces.resources.VMType attribute), 109

find() (cloudbridge.interfaces.services.BucketService method), 82

find() (cloudbridge.interfaces.services.DnsZoneService method), 87

find() (cloudbridge.interfaces.services.ImageService method), 78

find() (cloudbridge.interfaces.services.InstanceService method), 75

find() (cloudbridge.interfaces.services.KeyPairService method), 84

find() (cloudbridge.interfaces.services.NetworkService method), 79

find() (cloudbridge.interfaces.services.RegionService method), 86

find() (cloudbridge.interfaces.services.RouterService method), 81

find() (cloudbridge.interfaces.services.SnapshotService method), 77

find() (cloudbridge.interfaces.services.SubnetService method), 80

find() (cloudbridge.interfaces.services.VMFirewallService method), 85

find() (cloudbridge.interfaces.services.VMTypeService method), 86

find() (cloudbridge.interfaces.services.VolumeService method), 76

floating_ips (cloudbridge.interfaces.resources.Gateway attribute), 104

FloatingIP (class in cloudbridge.interfaces.resources), 101

from_port (cloudbridge.interfaces.resources.VMFirewallRule attribute), 111

G

Gateway (class in cloudbridge.interfaces.resources), 104

gateways (cloudbridge.interfaces.resources.Network attribute), 99

generate_url() (cloudbridge.interfaces.resources.BucketObject method), 111

get() (cloudbridge.interfaces.services.BucketService method), 82

get() (cloudbridge.interfaces.services.DnsRecordService method), 88

get() (cloudbridge.interfaces.services.DnsZoneService method), 87

get() (cloudbridge.interfaces.services.ImageService method), 78

get() (cloudbridge.interfaces.services.InstanceService method), 75

get() (cloudbridge.interfaces.services.KeyPairService method), 84

get() (cloudbridge.interfaces.services.NetworkService method), 79

get() (cloudbridge.interfaces.services.RegionService method), 86

get() (cloudbridge.interfaces.services.RouterService method), 82

get() (cloudbridge.interfaces.services.SnapshotService method), 77

get() (cloudbridge.interfaces.services.SubnetService method), 81

get() (cloudbridge.interfaces.services.VMFirewallService method), 85

get() (cloudbridge.interfaces.services.VMTypeService method), 86

get() (cloudbridge.interfaces.services.VolumeService method), 76

get_or_create_default() (cloudbridge.interfaces.services.SubnetService method), 81

has_service() (cloudbridge.interfaces.provider.CloudProvider method), 70

<code>host_zones</code> (<i>cloudbridge.interfaces.services.DnsService</i> label attribute), 87	<code>label</code> (<i>cloudbridge.interfaces.resources.Router</i> attribute), 103
I	<code>label</code> (<i>cloudbridge.interfaces.resources.Snapshot</i> attribute), 107
<code>id</code> (<i>cloudbridge.interfaces.resources.CloudResource</i> attribute), 90	<code>label</code> (<i>cloudbridge.interfaces.resources.Subnet</i> attribute), 101
<code>image_id</code> (<i>cloudbridge.interfaces.resources.Instance</i> attribute), 95	<code>label</code> (<i>cloudbridge.interfaces.resources.VMFirewall</i> attribute), 110
<code>images</code> (<i>cloudbridge.interfaces.services.ComputeService</i> label attribute), 73	<code>label</code> (<i>cloudbridge.interfaces.resources.Volume</i> attribute), 105
<code>ImageService</code> (class in <i>cloud-bridge.interfaces.services</i>), 78	<code>last_modified</code> (<i>cloud-bridge.interfaces.resources.BucketObject</i> attribute), 112
<code>in_use</code> (<i>cloudbridge.interfaces.resources.FloatingIP</i> attribute), 101	<code>LaunchConfig</code> (class in <i>cloud-bridge.interfaces.resources</i>), 97
<code>Instance</code> (class in <i>cloudbridge.interfaces.resources</i>), 94	<code>list()</code> (<i>cloudbridge.interfaces.resources.PageableObjectMixin</i> method), 92
<code>instances</code> (<i>cloudbridge.interfaces.services.ComputeService</i> attribute), 73	<code>list()</code> (<i>cloudbridge.interfaces.services.BucketService</i> method), 83
<code>InstanceService</code> (class in <i>cloud-bridge.interfaces.services</i>), 74	<code>list()</code> (<i>cloudbridge.interfaces.services.DnsRecordService</i> method), 88
<code>InstanceState</code> (class in <i>cloud-bridge.interfaces.resources</i>), 94	<code>list()</code> (<i>cloudbridge.interfaces.services.DnsZoneService</i> method), 87
<code>InternetGateway</code> (class in <i>cloud-bridge.interfaces.resources</i>), 104	<code>list()</code> (<i>cloudbridge.interfaces.services.ImageService</i> method), 78
<code>InvalidConfigurationException</code> (class in <i>cloudbridge.interfaces.exceptions</i>), 115	<code>list()</code> (<i>cloudbridge.interfaces.services.InstanceService</i> method), 75
<code>InvalidLabelException</code> (class in <i>cloud-bridge.interfaces.exceptions</i>), 115	<code>list()</code> (<i>cloudbridge.interfaces.services.KeyPairService</i> method), 85
<code>InvalidValueException</code> (class in <i>cloud-bridge.interfaces.exceptions</i>), 115	<code>list()</code> (<i>cloudbridge.interfaces.services.NetworkService</i> method), 80
<code>is_truncated</code> (<i>cloud-bridge.interfaces.resources.ResultList</i> attribute), 94	<code>list()</code> (<i>cloudbridge.interfaces.services.RegionService</i> method), 86
<code>iter_content()</code> (<i>cloud-bridge.interfaces.resources.BucketObject</i> method), 112	<code>list()</code> (<i>cloudbridge.interfaces.services.RouterService</i> method), 82
K	<code>list()</code> (<i>cloudbridge.interfaces.services.SnapshotService</i> method), 77
<code>key_pair_id</code> (<i>cloud-bridge.interfaces.resources.Instance</i> attribute), 95	<code>list()</code> (<i>cloudbridge.interfaces.services.SubnetService</i> method), 81
<code>key_pairs</code> (<i>cloudbridge.interfaces.services.SecurityService</i> attribute), 83	<code>list()</code> (<i>cloudbridge.interfaces.services.VMFirewallService</i> method), 85
<code>KeyPair</code> (class in <i>cloudbridge.interfaces.resources</i>), 108	<code>list()</code> (<i>cloudbridge.interfaces.services.VMTypeService</i> method), 86
<code>KeyPairService</code> (class in <i>cloud-bridge.interfaces.services</i>), 84	<code>list()</code> (<i>cloudbridge.interfaces.services.VolumeService</i> method), 76
L	M
<code>label</code> (<i>cloudbridge.interfaces.resources.Instance</i> attribute), 95	<code>MachineImage</code> (class in <i>cloud-bridge.interfaces.resources</i>), 98
<code>label</code> (<i>cloudbridge.interfaces.resources.Network</i> attribute), 99	<code>MachineImageState</code> (class in <i>cloud-bridge.interfaces.resources</i>), 97
	<code>marker</code> (<i>cloudbridge.interfaces.resources.ResultList</i> attribute), 94

- material (cloudbridge.interfaces.resources.KeyPair attribute), 108
- middleware (cloudbridge.interfaces.provider.CloudProvider attribute), 70
- min_disk (cloudbridge.interfaces.resources.MachineImage attribute), 98
- ## N
- name (cloudbridge.interfaces.resources.Bucket attribute), 113
- name (cloudbridge.interfaces.resources.BucketObject attribute), 112
- name (cloudbridge.interfaces.resources.CloudResource attribute), 90
- Network (class in cloudbridge.interfaces.resources), 99
- network (cloudbridge.interfaces.resources.Subnet attribute), 101
- network_id (cloudbridge.interfaces.resources.Gateway attribute), 104
- network_id (cloudbridge.interfaces.resources.Router attribute), 103
- network_id (cloudbridge.interfaces.resources.Subnet attribute), 101
- network_id (cloudbridge.interfaces.resources.VMFirewall attribute), 110
- networking (cloudbridge.interfaces.provider.CloudProvider attribute), 70
- NetworkingService (class in cloudbridge.interfaces.services), 79
- networks (cloudbridge.interfaces.services.NetworkingService attribute), 79
- NetworkService (class in cloudbridge.interfaces.services), 79
- NetworkState (class in cloudbridge.interfaces.resources), 99
- num_ephemeral_disks (cloudbridge.interfaces.resources.VMType attribute), 109
- ## O
- ObjectLifeCycleMixin (class in cloudbridge.interfaces.resources), 91
- objects (cloudbridge.interfaces.resources.Bucket attribute), 113
- ## P
- PageableObjectMixin (class in cloudbridge.interfaces.resources), 92
- PlacementZone (class in cloudbridge.interfaces.resources), 108
- private_ip (cloudbridge.interfaces.resources.FloatingIP attribute), 102
- private_ips (cloudbridge.interfaces.resources.Instance attribute), 95
- protocol (cloudbridge.interfaces.resources.VMFirewallRule attribute), 111
- provider (cloudbridge.interfaces.services.CloudService attribute), 72
- ProviderConnectionException (class in cloudbridge.interfaces.exceptions), 115
- public_ip (cloudbridge.interfaces.resources.FloatingIP attribute), 102
- public_ips (cloudbridge.interfaces.resources.Instance attribute), 95
- ## R
- ram (cloudbridge.interfaces.resources.VMType attribute), 109
- reboot () (cloudbridge.interfaces.resources.Instance method), 96
- records (cloudbridge.interfaces.resources.DnsZone attribute), 113
- refresh () (cloudbridge.interfaces.resources.BucketObject method), 112
- refresh () (cloudbridge.interfaces.resources.ObjectLifeCycleMixin method), 91
- Region (class in cloudbridge.interfaces.resources), 108
- region_name (cloudbridge.interfaces.provider.CloudProvider attribute), 71
- region_name (cloudbridge.interfaces.resources.PlacementZone attribute), 108
- regions (cloudbridge.interfaces.services.ComputeService attribute), 73
- RegionService (class in cloudbridge.interfaces.services), 86
- remove_floating_ip () (cloudbridge.interfaces.resources.Instance method), 96
- remove_vm_firewall () (cloudbridge.interfaces.resources.Instance method), 96
- ResultList (class in cloudbridge.interfaces.resources), 93
- Router (class in cloudbridge.interfaces.resources), 102
- routers (cloudbridge.interfaces.services.NetworkingService attribute), 79
- RouterService (class in cloudbridge.interfaces.services), 81
- RouterState (class in cloudbridge.interfaces.resources), 102
- rules (cloudbridge.interfaces.resources.VMFirewall attribute), 110

S

- `save_content()` (`cloudbridge.interfaces.resources.BucketObject` method), 112
- `security` (`cloudbridge.interfaces.provider.CloudProvider` attribute), 71
- `SecurityService` (class in `cloudbridge.interfaces.services`), 83
- `size` (`cloudbridge.interfaces.resources.BucketObject` attribute), 112
- `size` (`cloudbridge.interfaces.resources.Snapshot` attribute), 108
- `size` (`cloudbridge.interfaces.resources.Volume` attribute), 106
- `size_ephemeral_disks` (`cloudbridge.interfaces.resources.VMType` attribute), 109
- `size_root_disk` (`cloudbridge.interfaces.resources.VMType` attribute), 109
- `size_total_disk` (`cloudbridge.interfaces.resources.VMType` attribute), 109
- `Snapshot` (class in `cloudbridge.interfaces.resources`), 107
- `snapshots` (`cloudbridge.interfaces.services.StorageService` attribute), 77
- `SnapshotService` (class in `cloudbridge.interfaces.services`), 76
- `SnapshotState` (class in `cloudbridge.interfaces.resources`), 106
- `source` (`cloudbridge.interfaces.resources.Volume` attribute), 106
- `src_dest_fw` (`cloudbridge.interfaces.resources.VMFirewallRule` attribute), 111
- `src_dest_fw_id` (`cloudbridge.interfaces.resources.VMFirewallRule` attribute), 111
- `state` (`cloudbridge.interfaces.resources.Network` attribute), 100
- `state` (`cloudbridge.interfaces.resources.ObjectLifeCycleMixin` attribute), 91
- `state` (`cloudbridge.interfaces.resources.Router` attribute), 103
- `storage` (`cloudbridge.interfaces.provider.CloudProvider` attribute), 71
- `StorageService` (class in `cloudbridge.interfaces.services`), 77
- `Subnet` (class in `cloudbridge.interfaces.resources`), 100
- `subnet_id` (`cloudbridge.interfaces.resources.Instance` attribute), 96
- `subnets` (`cloudbridge.interfaces.resources.Network` attribute), 100
- `subnets` (`cloudbridge.interfaces.resources.Router` attribute), 103
- `subnets` (`cloudbridge.interfaces.services.NetworkingService` attribute), 79
- `subnets` (`cloudbridge.interfaces.services.NetworkService` attribute), 80
- `SubnetService` (class in `cloudbridge.interfaces.services`), 80
- `SubnetState` (class in `cloudbridge.interfaces.resources`), 100
- `supports_server_paging` (`cloudbridge.interfaces.resources.ResultList` attribute), 94
- `supports_total` (`cloudbridge.interfaces.resources.ResultList` attribute), 94

T

- `to_json()` (`cloudbridge.interfaces.resources.CloudResource` method), 90
- `to_port` (`cloudbridge.interfaces.resources.VMFirewallRule` attribute), 111
- `total_results` (`cloudbridge.interfaces.resources.ResultList` attribute), 94
- `TrafficDirection` (class in `cloudbridge.interfaces.resources`), 111
- `ttl` (`cloudbridge.interfaces.resources.DnsRecord` attribute), 114
- `type` (`cloudbridge.interfaces.resources.DnsRecord` attribute), 114

U

- `upload()` (`cloudbridge.interfaces.resources.BucketObject` method), 112
- `upload_from_file()` (`cloudbridge.interfaces.resources.BucketObject` method), 112

V

- `vcpus` (`cloudbridge.interfaces.resources.VMType` attribute), 109
- `vm_firewall_ids` (`cloudbridge.interfaces.resources.Instance` attribute), 96
- `vm_firewalls` (`cloudbridge.interfaces.resources.Instance` attribute), 96
- `vm_firewalls` (`cloudbridge.interfaces.services.SecurityService` attribute), 83
- `vm_type` (`cloudbridge.interfaces.resources.Instance` attribute), 96

[vm_type_id \(cloudbridge.interfaces.resources.Instance attribute\), 96](#)
[vm_types \(cloudbridge.interfaces.services.ComputeService attribute\), 73](#)
[VMFirewall \(class in cloud-bridge.interfaces.resources\), 110](#)
[VMFirewallRule \(class in cloud-bridge.interfaces.resources\), 110](#)
[VMFirewallService \(class in cloud-bridge.interfaces.services\), 85](#)
[VMType \(class in cloudbridge.interfaces.resources\), 109](#)
[VMTypeService \(class in cloud-bridge.interfaces.services\), 86](#)
[Volume \(class in cloudbridge.interfaces.resources\), 104](#)
[volume_id \(cloudbridge.interfaces.resources.Snapshot attribute\), 108](#)
[volumes \(cloudbridge.interfaces.services.StorageService attribute\), 78](#)
[VolumeService \(class in cloud-bridge.interfaces.services\), 76](#)
[VolumeState \(class in cloud-bridge.interfaces.resources\), 104](#)

W

[wait_for\(\) \(cloudbridge.interfaces.resources.ObjectLifeCycleMixin method\), 91](#)
[wait_till_ready\(\) \(cloud-bridge.interfaces.resources.ObjectLifeCycleMixin method\), 92](#)
[WaitStateException \(class in cloud-bridge.interfaces.exceptions\), 115](#)

Z

[zone \(cloudbridge.interfaces.resources.Subnet attribute\), 101](#)
[zone_id \(cloudbridge.interfaces.resources.DnsRecord attribute\), 114](#)
[zone_id \(cloudbridge.interfaces.resources.Instance attribute\), 96](#)
[zone_id \(cloudbridge.interfaces.resources.Volume attribute\), 106](#)
[zone_name \(cloudbridge.interfaces.provider.CloudProvider attribute\), 71](#)
[zones \(cloudbridge.interfaces.resources.Region attribute\), 108](#)