Package 'googleComputeEngineR'

September 16, 2017

Type Package

Version 0.2.0

Title R Interface with Google Compute Engine

Description Interact with the 'Google Compute Engine' API in R. Lets you create, start and stop instances in the 'Google Cloud'. Support for preconfigured instances, with templates for common R needs.

URL https://cloudyr.github.io/googleComputeEngineR/

BugReports https://github.com/cloudyr/googleComputeEngineR/issues

Depends R (>= 3.3.0)

Imports assertthat, future (>= 1.2.0), googleAuthR (>= 0.5.1), httr (>= 1.3.1), jsonlite (>= 1.1), utils

Suggests covr, devtools (>= 1.12.0), googleCloudStorageR, knitr, rmarkdown, testthat

License MIT + file LICENSE

LazyData true

VignetteBuilder knitr

RoxygenNote 6.0.1

NeedsCompilation no

Author Mark Edmondson [aut, cre], Scott Chamberlain [ctb], Winston Chang [ctb], Henrik Bengtsson [ctb], Jacki Novik [ctb]

Maintainer Mark Edmondson <r@sunholo.com>

Repository CRAN

Date/Publication 2017-09-16 16:25:37 UTC

R topics documented:

`as.cluster.gce_instance`

Create a future cluster for GCE objects

Description

S3 method for `as.cluster()` in the **future** package.

Usage

```
## S3 method for class 'gce_instance'
as.cluster(x, project = gce_get_global_project(),
  zone = gce_get_global_zone(), rshopts = ssh_options(x), ...,
  recursive = FALSE)
```

Arguments

`x`	The instance to make a future cluster
`project`	The GCE project
`zone`	The GCE zone
`rshopts`	Options for the SSH
`...`	Other arguments passed to makeDockerClusterPSOCK
`recursive`	Not used.

Details

Only works for r-base containers created via `gce_vm_template("r-base")` or for docker containers created using the `--net=host` argument flag

Value

A `cluster` object.

Examples

```
## Not run:
vm <- gce_vm("r-base", name = "future", predefined_type = "f1-micro")
plan(cluster, workers = vm)  ## equivalent to workers = as.cluster(vm)
x %<-% { Sys.getinfo() }
print(x)

## End(Not run)
```

`as.container` *Coerce an object into a container object.*

Description

Coerce an object into a container object.

Usage

```
as.container(x, host = localhost)
```

Arguments

`x`	An object to coerce
`host`	A docker host
	A container object represents a Docker container on a host.

Author(s)

Winston Change <winston@stdout.org>

`containers` *Get list of all containers on a host.*

Description

Get list of all containers on a host.

Usage

```
containers(host = localhost, ...)
```

Arguments

`host`	A host object.
`...`	Other arguments passed to the SSH command for the host

Author(s)

Winston Change <winston@stdout.org>

`container_logs` *Retrieve logs for a container.*

Description

Retrieve logs for a container.

Usage

```
container_logs(container, timestamps = FALSE, follow = FALSE)
```

Arguments

`container`	A container object
`timestamps`	Show timestamps.
`follow`	Follow log output as it is happening.

Author(s)

Winston Change <winston@stdout.org>

Examples

```
## Not run:
container_rm(con)

## End(Not run)
```

`container_rm` *Delete a container.*

Description

Delete a container.

Usage

```
container_rm(container, force = FALSE)
```

Arguments

`container`	A container object
`force`	Force removal of a running container.

Author(s)

Winston Change <winston@stdout.org>

Examples

```
## Not run:
container_rm(con)

## End(Not run)
```

---

container_running *Report whether a container is currently running.*

Description

Report whether a container is currently running.

Usage

```
container_running(container)
```

Arguments

container	A container object

Author(s)

Winston Change <winston@stdout.org>

Examples

```
## Not run:
container_running(con)

## End(Not run)
```

container_update_info *Update the information about a container.*

Description

This queries docker (on the host) for information about the container, and saves the returned information into a container object, which is returned. This does not use reference semantics, so if you want to store the updated information, you need to save the result.

Usage

```
container_update_info(container)
```

Arguments

container	A container object

Author(s)

Winston Change <winston@stdout.org>

Examples

```
## Not run:
con <- container_update_info(con)

## End(Not run)
```

docker_build *Build image on an instance from a local Dockerfile*

Description

Uploads a folder with a Dockerfile and supporting files to an instance and builds it

Usage

```
docker_build(host = localhost, dockerfolder, new_image,
  folder = "buildimage", wait = FALSE, ...)
```

Arguments

host	A host object.
dockerfolder	Local location of build directory including valid Dockerfile
new_image	Name of the new image
folder	Where on host to build dockerfile
wait	Whether to block R console until finished build
...	Other arguments passed to the SSH command for the host

Details

Dockerfiles are best practice when creating your own docker images, rather than logging into a Docker container, making changes and committing.

Value

A table of active images on the instance

See Also

Best practices for writing Dockerfiles

An example Dockerfile for rOpensci

General R Docker images found at rocker-org

Examples

```
## Not run:
docker_build(localhost, "/home/stuff/dockerfolder" ,"new_image", wait = TRUE)
docker_run(localhost, "new_image")

## End(Not run)
```

---

docker_cmd *Run a docker command on a host.*

Description

Run a docker command on a host.

Usage

```
docker_cmd(host, cmd = NULL, args = NULL, docker_opts = NULL,
  capture_text = FALSE, ...)
```

Arguments

host	A host object.
cmd	A docker command, such as "run" or "ps"
args	Arguments to pass to the docker command
docker_opts	Options to docker. These are things that come before the docker command, when run on the command line.
capture_text	If FALSE (the default), return the host object. This is useful for chaining functions. If TRUE, capture the text output from both stdout and stderr, and return that. Note that TRUE may not be available on all types of hosts.
...	Other arguments passed to the SSH command for the host

Author(s)

Winston Change <winston@stdout.org>

Examples

```
## Not run:
docker_cmd(localhost, "ps", "-a")

## End(Not run)
```

`docker_cmd.gce_instance` *Docker S3 method for use with harbor package*

Description

Docker S3 method for use with harbor package

Usage

```
## S3 method for class 'gce_instance'
docker_cmd(host, cmd = NULL, args = NULL,
  docker_opts = NULL, capture_text = FALSE, ...)
```

Arguments

`host`	The GCE instance
`cmd`	The command to pass to docker
`args`	arguments to the command
`docker_opts`	options for docker
`capture_text`	whether to return the output
`...`	other arguments passed to gce_ssh

Details

Instances launched in the `google-containers` image family automatically add your user to the docker group, but for others you will need to run `sudo usermod -a -G docker ${USER}` and log out and back in.

docker_inspect *Inspect one or more containers, given name(s) or ID(s).*

Description

Inspect one or more containers, given name(s) or ID(s).

Usage

```
docker_inspect(host = localhost, names = NULL, ...)
```

Arguments

host	A host object.
names	Names of the containers
...	Other arguments passed to the SSH command for the host

Value

A list of lists, where each sublist represents one container. This is the output of 'docker inspect' translated directly from raw JSON to an R object.

Author(s)

Winston Change <winston@stdout.org>

Examples

```
## Not run:
docker_run(localhost, "debian:testing", "echo foo", name = "harbor-test")
docker_inspect(localhost, "harbor-test")

## End(Not run)
```

docker_pull *Pull a docker image onto a host.*

Description

Pull a docker image onto a host.

Usage

```
docker_pull(host = localhost, image, ...)
```

Arguments

host	A host object.
image	The docker image to pull e.g. rocker/rstudio
...	Other arguments passed to the SSH command for the host

Value

The host object.

Author(s)

Winston Change <winston@stdout.org>

Examples

```
## Not run:
docker_pull(localhost, "debian:testing")

## End(Not run)
```

---

docker_run *Run a command in a new container on a host.*

Description

Run a command in a new container on a host.

Usage

```
docker_run(host = localhost, image = NULL, cmd = NULL, name = NULL,
  rm = FALSE, detach = FALSE, docker_opts = NULL, ...)
```

Arguments

host	An object representing the host where the container will be run.
image	The name or ID of a docker image.
cmd	A command to run in the container.
name	A name for the container. If none is provided, a random name will be used.
rm	If TRUE, remove the container after it finishes. This is incompatible with detach=TRUE.
detach	If TRUE, run the container in the background.
docker_opts	Options to docker. These are things that come before the docker command, when run on the command line.
...	Other arguments passed to the SSH command for the host

Value

A container object. When rm=TRUE, this function returns NULL instead of a container object, because the container no longer exists.

Author(s)

Winston Change <winston@stdout.org>

Examples

```
## Not run:
docker_run(localhost, "debian:testing", "echo foo")
#> foo

# Arguments will be concatenated
docker_run(localhost, "debian:testing", c("echo foo", "bar"))
#> foo bar

docker_run(localhost, "rocker/r-base", c("Rscript", "-e", "1+1"))
#> [1] 2

## End(Not run)
```

---

gce_attach_disk *Attaches a Disk resource to an instance.*

Description

Attaches a Disk resource to an instance.

Usage

```
gce_attach_disk(instance, source = NULL, autoDelete = NULL, boot = NULL,
  deviceName = NULL, diskEncryptionKey = NULL, index = NULL,
  initializeParams = NULL, interface = NULL, licenses = NULL,
  mode = NULL, type = NULL, project = gce_get_global_project(),
  zone = gce_get_global_zone())
```

Arguments

instance	The instance name for this request
source	Specifies a valid partial or full URL to an existing Persistent Disk resource
autoDelete	Specifies whether the disk will be auto-deleted when the instance is deleted (but not when the disk is detached from the instance)
boot	Indicates that this is a boot disk
deviceName	Specifies a unique device name of your choice that is reflected into the /dev/disk/by-id/google-* tree of a Linux operating system running within the instance

`diskEncryptionKey`
: Encrypts or decrypts a disk using a customer-supplied encryption key

`index`
: Assigns a zero-based index to this disk, where 0 is reserved for the boot disk

`initializeParams`
: A gce_make_boot_disk object for creating boot disks. Cannot be used with `source` also defined.

`interface`
: Specifies the disk interface to use for attaching this disk, which is either SCSI or NVME

`licenses`
: [Output Only] Any valid publicly visible licenses

`mode`
: The mode in which to attach this disk, either READ_WRITE or READ_ONLY

`type`
: Specifies the type of the disk, either SCRATCH or PERSISTENT

`project`
: Project ID for this request

`zone`
: The name of the zone for this request

Details

Authentication scopes used by this function are:

- https://www.googleapis.com/auth/cloud-platform
- https://www.googleapis.com/auth/compute

See Also

Google Documentation

Other AttachedDisk functions: `AttachedDisk`

---

gce_auth *Authenticate this session*

Description

A wrapper for gar_auth and gar_auth_service

Usage

```
gce_auth(new_user = FALSE, no_auto = FALSE)
```

Arguments

new_user	If TRUE, reauthenticate via Google login screen
no_auto	Will ignore auto-authentication settings if TRUE If you have set the environment variable GCE_AUTH_FILE to a valid file location, the function will look there for authentication details. Otherwise it will look in the working directory for the '.httr-oauth' file, which if not present will trigger an authentication flow via Google login screen in your browser. If GCE_AUTH_FILE is specified, then gce_auth() will be called upon loading the package via library(googleComputeEngineR), meaning that calling this function yourself at the start of the session won't be necessary. GCE_AUTH_FILE can be either a token generated by gar_auth or service account JSON ending with file extension .json

Value

Invisibly, the token that has been saved to the session

---

gce_check_container *Check the docker logs of a container*

Description

Check the docker logs of a container

Usage

```
gce_check_container(instance, container)
```

Arguments

instance	The instance running docker
container	A running container to get logs of

Value

logs

gce_check_gpu *Check GPU installed ok*

Description

Check GPU installed ok

Usage

```
gce_check_gpu(vm)
```

Arguments

vm	The instance to check

Value

The NVIDIA-SMI output via ssh

See Also

https://cloud.google.com/compute/docs/gpus/add-gpus#verify-driver-install

Other GPU instances: gce_list_gpus, gce_vm_gpu

gce_check_ssh *Calls API for the current SSH settings for an instance*

Description

Calls API for the current SSH settings for an instance

Usage

```
gce_check_ssh(instance)
```

Arguments

instance	An instance to check

Value

A data.frame of SSH users and public keys

gce_delete_disk *Deletes the specified persistent disk.*

Description

Deleting a disk removes its data permanently and is irreversible.

Usage

```
gce_delete_disk(disk, project = gce_get_global_project(),
  zone = gce_get_global_zone())
```

Arguments

disk	Name of the persistent disk to delete
project	Project ID for this request
zone	The name of the zone for this request

Details

However, deleting a disk does not delete any snapshots previously made from the disk. You must separately delete snapshots.

Authentication scopes used by this function are:

- https://www.googleapis.com/auth/cloud-platform
- https://www.googleapis.com/auth/compute

See Also

Google Documentation

gce_delete_firewall_rule *Delete a firewall rule*

Description

Deletes a firewall rule of name specified

Usage

```
gce_delete_firewall_rule(name, project = gce_get_global_project())
```

Arguments

name	Name of the firewall rule
project	The Google Cloud project

See Also

API Documentation https://cloud.google.com/compute/docs/reference/latest/firewalls/delete

Other firewall functions: gce_get_firewall_rule, gce_list_firewall_rules, gce_make_firewall_rule, gce_make_firewall_webports

---

gce_delete_op *Deletes the specified Operations resource.*

Description

Deletes the specified Operations resource.

Usage

```
gce_delete_op(operation)
```

Arguments

operation	Name of the Operations resource to delete

Value

TRUE if successful

See Also

Google Documentation

`gce_delete_op.gce_global_operation`
Deletes the specified global Operations resource.

Description

Deletes the specified global Operations resource.

Usage

```
## S3 method for class 'gce_global_operation'
gce_delete_op(operation)
```

Arguments

`operation`	Name of the Operations resource to delete

Value

The deleted operation

See Also

Google Documentation

`gce_delete_op.gce_zone_operation`
Deletes the specified zone-specific Operations resource.

Description

Deletes the specified zone-specific Operations resource.

Usage

```
## S3 method for class 'gce_zone_operation'
gce_delete_op(operation)
```

Arguments

`operation`	Name of the Operations resource to delete

Value

The deleted operation

See Also

Google Documentation

`gce_delete_zone_op` *Deletes the specified zone-specific Operations resource.*

Description

Deletes the specified zone-specific Operations resource.

Usage

```
gce_delete_zone_op(operation, project = gce_get_global_project(),
  zone = gce_get_global_zone())
```

Arguments

`operation`	Name of the Operations resource to delete
`project`	Project ID for this request
`zone`	Name of the zone for this request

Details

Authentication scopes used by this function are:

- https://www.googleapis.com/auth/cloud-platform
- https://www.googleapis.com/auth/compute

Value

TRUE if successful

See Also

Google Documentation

gce_extract_projectzone

Extract zone and project from an instance object

Description

Extract zone and project from an instance object

Usage

```
gce_extract_projectzone(instance)
```

Arguments

instance	The instance

Value

A list of $project and $zone

gce_future_install_packages

Install R packages onto an instance's stopped docker image

Description

Install R packages onto an instance's stopped docker image

Usage

```
gce_future_install_packages(instance, docker_image, cran_packages = NULL,
  github_packages = NULL)
```

Arguments

instance	The instance running the container
docker_image	A docker image to install packages within.
cran_packages	A character vector of CRAN packages to be installed
github_packages	A character vector of devtools packages to be installed

Details

See the images on the instance via `docker_cmd(instance, "images")`

If using devtools github, will look for an auth token via `devtools::github_pat()`. This is an environment variable called `GITHUB_PAT`

Will start a container, install packages and then commit the container to an image of the same name via `docker commit -m "installed packages via gceR"`

Value

TRUE if successful

---

gce_get_disk *Returns a specified persistent disk.*

Description

Returns a specified persistent disk.

Usage

```
gce_get_disk(disk, project = gce_get_global_project(),
  zone = gce_get_global_zone())
```

Arguments

`disk`	Name of the persistent disk to return
`project`	Project ID for this request
`zone`	The name of the zone for this request

Details

Authentication scopes used by this function are:

- https://www.googleapis.com/auth/cloud-platform
- https://www.googleapis.com/auth/compute
- https://www.googleapis.com/auth/compute.readonly

See Also

Google Documentation

gce_get_external_ip *Get the external IP of an instance*

Description

Get the external IP of an instance

Usage

```
gce_get_external_ip(instance, verbose = TRUE, ...)
```

Arguments

instance	Name or instance object to find the external IP for
verbose	Give a user message about the IP
...	passed to gce_get_instance

This is a helper to extract the external IP of an instance

Value

The external IP

gce_get_firewall_rule *Get a firewall rule*

Description

Get a firewall rule of name specified

Usage

```
gce_get_firewall_rule(name, project = gce_get_global_project())
```

Arguments

name	Name of the firewall rule
project	The Google Cloud project

See Also

API Documentation https://cloud.google.com/compute/docs/reference/latest/firewalls/get

Other firewall functions: gce_delete_firewall_rule, gce_list_firewall_rules, gce_make_firewall_rule, gce_make_firewall_webports

gce_get_global_project *Get global project name*

Description

Project name set this session to use by default

Usage

```
gce_get_global_project()
```

Details

Set the project name via gce_global_project

Value

Project name

gce_get_global_zone *Get global zone name*

Description

zone name set this session to use by default

Usage

```
gce_get_global_zone()
```

Details

Set the zone name via gce_global_zone

Value

zone name

`gce_get_image` *Returns the specified image.*

Description

Returns the specified image.

Usage

```
gce_get_image(image_project, image)
```

Arguments

`image_project`	Project ID of where the image lies
`image`	Name of the image resource to return

Details

Authentication scopes used by this function are:

- https://www.googleapis.com/auth/cloud-platform
- https://www.googleapis.com/auth/compute
- https://www.googleapis.com/auth/compute.readonly

You may want to use gce_get_image_family instead to ensure the most up to date image is used.

See Also

Google Documentation

`gce_get_image_family` *Returns the latest image that is part of an image family and is not deprecated.*

Description

Returns the latest image that is part of an image family and is not deprecated.

Usage

```
gce_get_image_family(image_project, family)
```

Arguments

`image_project`	Project ID for this request
`family`	Name of the image family to search for

Details

Authentication scopes used by this function are:

- https://www.googleapis.com/auth/cloud-platform
- https://www.googleapis.com/auth/compute
- https://www.googleapis.com/auth/compute.readonly

See Also

Google Documentation

`gce_get_instance` *Returns the specified Instance resource.*

Description

Returns the specified Instance resource.

Usage

```
gce_get_instance(instance, project = gce_get_global_project(),
  zone = gce_get_global_zone())
```

Arguments

`instance`	Name of the instance resource
`project`	Project ID for this request, default as set by gce_get_global_project
`zone`	The name of the zone for this request, default as set by gce_get_global_zone

Details

Authentication scopes used by this function are:

- https://www.googleapis.com/auth/cloud-platform
- https://www.googleapis.com/auth/compute
- https://www.googleapis.com/auth/compute.readonly

See Also

Google Documentation

gce_get_machinetype *Returns the specified machine type.*

Description

Returns the specified machine type.

Usage

```
gce_get_machinetype(machineType, project = gce_get_global_project(),
  zone = gce_get_global_zone())
```

Arguments

machineType	Name of the machine type to return
project	Project ID for this request
zone	The name of the zone for this request

Details

Authentication scopes used by this function are:

- https://www.googleapis.com/auth/cloud-platform
- https://www.googleapis.com/auth/compute
- https://www.googleapis.com/auth/compute.readonly

See Also

Google Documentation

gce_get_metadata *Extract metadata from an instance object*

Description

Extract metadata from an instance object

Usage

```
gce_get_metadata(instance, key = NULL)
```

Arguments

instance	instance to get metadata from
key	optional metadata key to filter metadata result

Value

data.frame $key and $value of metadata or NULL

`gce_get_network` *Returns the specified network.*

Description

Returns the specified network.

Usage

```
gce_get_network(network, project = gce_get_global_project())
```

Arguments

`network`	Name of the network to return
`project`	Project ID for this request

Details

Authentication scopes used by this function are:

- https://www.googleapis.com/auth/cloud-platform
- https://www.googleapis.com/auth/compute
- https://www.googleapis.com/auth/compute.readonly

See Also

Google Documentation

`gce_get_op` *Retrieves the specified Operations resource.*

Description

s3 method dispatcher

Usage

```
gce_get_op(operation)
```

Arguments

`operation`	Name of the Operations resource to return

Details

S3 Methods for classes

- gce_get_op.gce_zone_operation
- gce_get_op.gce_global_operation
- gce_get_op.gce_region_operation

See Also

Google Documentation

---

gce_get_op.gce_global_operation

Retrieves the specified global Operations resource.

Description

Retrieves the specified global Operations resource.

Usage

```
## S3 method for class 'gce_global_operation'
gce_get_op(operation)
```

Arguments

operation	Name of the Operations resource to return

See Also

Google Documentation

---

gce_get_op.gce_zone_operation

Retrieves the specified zone-specific Operations resource.

Description

Retrieves the specified zone-specific Operations resource.

Usage

```
## S3 method for class 'gce_zone_operation'
gce_get_op(operation)
```

Arguments

operation	Name of the Operations resource to return

See Also

Google Documentation

`gce_get_project` *Returns the specified Project resource.*

Description

Returns the specified Project resource.

Usage

```
gce_get_project(project = gce_get_global_project())
```

Arguments

project	Project ID for this request

Details

Authentication scopes used by this function are:

- https://www.googleapis.com/auth/cloud-platform
- https://www.googleapis.com/auth/compute
- https://www.googleapis.com/auth/compute.readonly

See Also

Google Documentation

`gce_get_zone` *Returns the specified Zone resource. Get a list of available zones by making a list() request.*

Description

Returns the specified Zone resource. Get a list of available zones by making a list() request.

Usage

```
gce_get_zone(project, zone)
```

Arguments

`project`	Project ID for this request
`zone`	Name of the zone resource to return

Details

Authentication scopes used by this function are:

- https://www.googleapis.com/auth/cloud-platform
- https://www.googleapis.com/auth/compute
- https://www.googleapis.com/auth/compute.readonly

See Also

Google Documentation

`gce_get_zone_op` *Retrieves the specified zone-specific Operations resource.*

Description

Retrieves the specified zone-specific Operations resource.

Usage

```
gce_get_zone_op(operation, project = gce_get_global_project(),
  zone = gce_get_global_zone())
```

Arguments

`operation`	Name of the Operations resource to return
`project`	Project ID for this request
`zone`	Name of the zone for this request

Details

Authentication scopes used by this function are:

- https://www.googleapis.com/auth/cloud-platform
- https://www.googleapis.com/auth/compute
- https://www.googleapis.com/auth/compute.readonly

See Also

Google Documentation

`gce_global_project` *Set global project name*

Description

Set a project name used for this R session

Usage

```
gce_global_project(project = gce_get_global_project())
```

Arguments

`project`	project name you want this session to use by default, or a project object

Details

This sets a project to a global environment value so you don't need to supply the project argument to other API calls.

Value

The project name (invisibly)

gce_global_zone *Set global zone name*

Description

Set a zone name used for this R session

Usage

```
gce_global_zone(zone)
```

Arguments

zone	zone name you want this session to use by default, or a zone object

Details

This sets a zone to a global environment value so you don't need to supply the zone argument to other API calls.

Value

The zone name (invisibly)

gce_list_disks *Retrieves a list of persistent disks contained within the specified zone.*

Description

Retrieves a list of persistent disks contained within the specified zone.

Usage

```
gce_list_disks(filter = NULL, maxResults = NULL, pageToken = NULL,
  project = gce_get_global_project(), zone = gce_get_global_zone())
```

Arguments

filter	Sets a filter expression for filtering listed resources, in the form filter=expression
maxResults	The maximum number of results per page that should be returned
pageToken	Specifies a page token to use
project	Project ID for this request
zone	The name of the zone for this request

Details

Authentication scopes used by this function are:

- https://www.googleapis.com/auth/cloud-platform
- https://www.googleapis.com/auth/compute
- https://www.googleapis.com/auth/compute.readonly

See Also

Google Documentation

---

gce_list_disks_all *Retrieves an aggregated list of persistent disks across all zones.*

Description

Retrieves an aggregated list of persistent disks across all zones.

Usage

```
gce_list_disks_all(filter = NULL, maxResults = NULL, pageToken = NULL,
  project = gce_get_global_project())
```

Arguments

filter	Sets a filter expression for filtering listed resources, in the form filter=expression
maxResults	The maximum number of results per page that should be returned
pageToken	Specifies a page token to use
project	Project ID for this request

Details

Authentication scopes used by this function are:

- https://www.googleapis.com/auth/cloud-platform
- https://www.googleapis.com/auth/compute
- https://www.googleapis.com/auth/compute.readonly

See Also

Google Documentation

gce_list_firewall_rules *List firewall rules*

Description

Get a firewall rule of name specified

Usage

```
gce_list_firewall_rules(filter = NULL, maxResults = NULL,
  pageToken = NULL, project = gce_get_global_project())
```

Arguments

filter	Sets a filter expression for filtering listed resources, in the form filter=expression
maxResults	The maximum number of results per page that should be returned
pageToken	Specifies a page token to use
project	The Google Cloud project

See Also

API Documentation https://cloud.google.com/compute/docs/reference/latest/firewalls/list

Other firewall functions: gce_delete_firewall_rule, gce_get_firewall_rule, gce_make_firewall_rule, gce_make_firewall_webports

gce_list_gpus *Retrieves a list GPUs you can attach to an instance*

Description

Retrieves a list GPUs you can attach to an instance

Usage

```
gce_list_gpus(filter = NULL, maxResults = NULL, pageToken = NULL,
  project = gce_get_global_project(), zone = gce_get_global_zone())
```

Arguments

filter	Sets a filter expression for filtering listed resources, in the form filter=expression
maxResults	The maximum number of results per page that should be returned
pageToken	Specifies a page token to use
project	Project ID for this request
zone	The name of the zone for this request

Details

Authentication scopes used by this function are:

- https://www.googleapis.com/auth/cloud-platform
- https://www.googleapis.com/auth/compute
- https://www.googleapis.com/auth/compute.readonly

To filter you need a single string in the form `field_name eq|ne string` e.g. `gce_list_instances("status eq RUNNING")` where `eq` is 'equals' and `ne` is 'not-equals'.

See Also

Google Documentation

Other GPU instances: `gce_check_gpu`, `gce_vm_gpu`

---

gce_list_images *Retrieves the list of private images available to the specified project.*

Description

Retrieves the list of private images available to the specified project.

Usage

```
gce_list_images(image_project, filter = NULL, maxResults = NULL,
  pageToken = NULL)
```

Arguments

`image_project`	Project ID for this request
`filter`	Sets a filter expression for filtering listed resources, in the form filter=expression
`maxResults`	The maximum number of results per page that should be returned
`pageToken`	Specifies a page token to use

Details

Authentication scopes used by this function are:

- https://www.googleapis.com/auth/cloud-platform
- https://www.googleapis.com/auth/compute
- https://www.googleapis.com/auth/compute.readonly

If you want to get a list of publicly-available images, use this method to make a request to the respective image project, such as debian-cloud, windows-cloud or google-containers.

See Also

Google Documentation

`gce_list_instances` *Retrieves the list of instances contained within the specified zone.*

Description

Retrieves the list of instances contained within the specified zone.

Usage

```
gce_list_instances(filter = NULL, maxResults = NULL, pageToken = NULL,
  project = gce_get_global_project(), zone = gce_get_global_zone())
```

Arguments

`filter`	Sets a filter expression for filtering listed resources, in the form filter=expression
`maxResults`	The maximum number of results per page that should be returned
`pageToken`	Specifies a page token to use
`project`	Project ID for this request
`zone`	The name of the zone for this request

Details

Authentication scopes used by this function are:

- https://www.googleapis.com/auth/cloud-platform
- https://www.googleapis.com/auth/compute
- https://www.googleapis.com/auth/compute.readonly

To filter you need a single string in the form `field_name eq|ne string` e.g. `gce_list_instances("status eq RUNNING")` where `eq` is 'equals' and `ne` is 'not-equals'.

See Also

Google Documentation

`gce_list_machinetype` *Retrieves a list of machine types available to the specified project.*

Description

Retrieves a list of machine types available to the specified project.

Usage

```
gce_list_machinetype(filter = NULL, maxResults = NULL, pageToken = NULL,
  project = gce_get_global_project(), zone = gce_get_global_zone())
```

Arguments

`filter`	Sets a filter expression for filtering listed resources, in the form filter=expression
`maxResults`	The maximum number of results per page that should be returned
`pageToken`	Specifies a page token to use
`project`	Project ID for this request
`zone`	The name of the zone for this request

Details

Authentication scopes used by this function are:

- https://www.googleapis.com/auth/cloud-platform
- https://www.googleapis.com/auth/compute
- https://www.googleapis.com/auth/compute.readonly

See Also

Google Documentation

`gce_list_machinetype_all` *Retrieves an aggregated list of machine types from all zones.*

Description

Retrieves an aggregated list of machine types from all zones.

Usage

```
gce_list_machinetype_all(filter = NULL, maxResults = NULL,
  pageToken = NULL, project = gce_get_global_project())
```

Arguments

filter	Sets a filter expression for filtering listed resources, in the form filter=expression
maxResults	The maximum number of results per page that should be returned
pageToken	Specifies a page token to use
project	Project ID for this request

Details

Authentication scopes used by this function are:

- https://www.googleapis.com/auth/cloud-platform
- https://www.googleapis.com/auth/compute
- https://www.googleapis.com/auth/compute.readonly

See Also

Google Documentation

---

gce_list_networks *Retrieves the list of networks available to the specified project.*

Description

Retrieves the list of networks available to the specified project.

Usage

```
gce_list_networks(filter = NULL, maxResults = NULL, pageToken = NULL,
  project = gce_get_global_project())
```

Arguments

filter	Sets a filter expression for filtering listed resources, in the form filter=expression
maxResults	The maximum number of results per page that should be returned
pageToken	Specifies a page token to use
project	Project ID for this request

Details

Authentication scopes used by this function are:

- https://www.googleapis.com/auth/cloud-platform
- https://www.googleapis.com/auth/compute
- https://www.googleapis.com/auth/compute.readonly

See Also

Google Documentation

`gce_list_registry` *List the docker images you have on Google Container Registry*

Description

List the docker images you have on Google Container Registry

Usage

```
gce_list_registry(instance, container_url = "gcr.io",
  project = gce_get_global_project())
```

Arguments

`instance`	The VM to run within
`container_url`	The URL of where the container was saved
`project`	Project ID for this request, default as set by gce_get_global_project

Details

Currently needs to run on a Google VM, not locally

See Also

Other container registry functions: `gce_pull_registry`, `gce_push_registry`, `gce_tag_container`

Examples

```
## Not run:

  vm <- gce_vm("my_instance")
  gce_list_registry(vm)


## End(Not run)
```

`gce_list_zones` *Retrieves the list of Zone resources available to the specified project.*

Description

Retrieves the list of Zone resources available to the specified project.

Usage

```
gce_list_zones(project, filter = NULL, maxResults = NULL,
  pageToken = NULL)
```

Arguments

`project`	Project ID for this request
`filter`	Sets a filter expression for filtering listed resources, in the form filter=expression
`maxResults`	The maximum number of results per page that should be returned
`pageToken`	Specifies a page token to use

Details

Authentication scopes used by this function are:

- https://www.googleapis.com/auth/cloud-platform
- https://www.googleapis.com/auth/compute
- https://www.googleapis.com/auth/compute.readonly

See Also

Google Documentation

`gce_list_zone_op` *Retrieves a list of Operation resources contained within the specified zone.*

Description

Retrieves a list of Operation resources contained within the specified zone.

Usage

```
gce_list_zone_op(filter = NULL, maxResults = NULL, pageToken = NULL,
  project = gce_get_global_project(), zone = gce_get_global_zone())
```

Arguments

filter	Sets a filter expression for filtering listed resources, in the form filter=expression
maxResults	The maximum number of results per page that should be returned
pageToken	Specifies a page token to use
project	Project ID for this request
zone	Name of the zone for request

Details

Authentication scopes used by this function are:

- https://www.googleapis.com/auth/cloud-platform
- https://www.googleapis.com/auth/compute
- https://www.googleapis.com/auth/compute.readonly

See Also

Google Documentation

---

gce_make_boot_disk *Make a boot disk for attachment to an instance*

Description

Make a boot disk for attachment to an instance

Usage

```
gce_make_boot_disk(diskName = NULL, diskSizeGb = NULL, diskType = NULL,
  sourceImage = NULL, sourceImageEncryptionKey = NULL)
```

Arguments

diskName	Specifies the disk name
diskSizeGb	Specifies the size of the disk in base-2 GB
diskType	Specifies the disk type to use to create the instance
sourceImage	The source image used to create this disk
sourceImageEncryptionKey	The customer-supplied encryption key of the source image

Details

Specifies the parameters for a new disk that will be created alongside the new instance.

Use initialization parameters to create boot disks or local SSDs attached to the new instance.

This property is mutually exclusive with the source property; you can only define one or the other, but not both.

Value

AttachedDiskInitializeParams object

`gce_make_disk` *Creates a persistent disk in the specified project using the data in the request.*

Description

You can create a disk with a sourceImage, a sourceSnapshot, or create an empty 500 GB data disk by omitting all properties.

Usage

```
gce_make_disk(name, sourceImage = NULL, sizeGb = NULL, description = NULL,
  diskEncryptionKey = NULL, licenses = NULL, sourceSnapshot = NULL,
  sourceImageEncryptionKey = NULL, sourceSnapshotEncryptionKey = NULL,
  type = NULL, project = gce_get_global_project(),
  zone = gce_get_global_zone())
```

Arguments

`name`	Name of the resource
`sourceImage`	The source image used to create this disk
`sizeGb`	Size of the persistent disk, specified in GB
`description`	An optional description of this resource
`diskEncryptionKey`	Encrypts the disk using a customer-supplied encryption key
`licenses`	Any applicable publicly visible licenses
`sourceSnapshot`	The source snapshot used to create this disk
`sourceImageEncryptionKey`	The customer-supplied encryption key of the source image
`sourceSnapshotEncryptionKey`	The customer-supplied encryption key of the source snapshot
`type`	URL of the disk type resource describing which disk type to use to create the disk
`project`	Project ID for this request
`zone`	The name of the zone for this request

Details

You can also create a disk that is larger than the default size by specifying the sizeGb property.

Authentication scopes used by this function are:

- https://www.googleapis.com/auth/cloud-platform
- https://www.googleapis.com/auth/compute

Value

a zone operation

See Also

Google Documentation

`gce_make_firewall_rule` *Add one firewall rule to the network*

Description

Use this to create firewall rules to apply to the network settings. Most commonly this is to setup web access (port 80 and 443)

Usage

```
gce_make_firewall_rule(name, protocol, ports, sourceRanges = NULL,
  sourceTags = NULL, project = gce_get_global_project())
```

Arguments

`name`	Name of the firewall rule
`protocol`	Protocol such as `tcp`, `udp`, `icmp`, `esp`, `ah`, `sctp` or IP protocol number.
`ports`	Port numbers to open
`sourceRanges`	From where to accept connections. If `NULL` then will default to `0.0.0.0/0` (everywhere)
`sourceTags`	A list of instance tags this rule applies to. One or both of `sourceRanges` and `sourceTags` may be set.
`project`	The Google Cloud project

Value

A global operation object

sourceRanges and/or sourceTags

If both properties are set, an inbound connection is allowed if the range or the tag of the source matches the sourceRanges OR matches the sourceTags property; the connection does not need to match both properties.

See Also

API Documentation https://cloud.google.com/compute/docs/reference/latest/firewalls/insert

Other firewall functions: gce_delete_firewall_rule, gce_get_firewall_rule, gce_list_firewall_rules, gce_make_firewall_webports

Examples

```
## Not run:

  gce_make_firewall_rule("allow-http", protocol = "tcp", ports = 80)
  gce_make_firewall_rule("allow-https", protocol = "tcp", ports = 443)


## End(Not run)
```

---

gce_make_firewall_webports

Make HTTP and HTTPS firewall rules

Description

Do the common use case of opening HTTP and HTTPS ports

Usage

```
gce_make_firewall_webports(project = gce_get_global_project())
```

Arguments

project	The project the firewall will open for

Details

This will invoke gce_make_firewall_rule and look for the rules named `allow-http` and `allow-https`. If not present, it will create them.

Value

Vector of the firewall objects

See Also

Other firewall functions: gce_delete_firewall_rule, gce_get_firewall_rule, gce_list_firewall_rules, gce_make_firewall_rule

---

`gce_make_image_source_url`
Make initial disk image object

Description

Make initial disk image object

Usage

```
gce_make_image_source_url(image_project, image = NULL, family = NULL)
```

Arguments

`image_project`	Project ID of where the image lies
`image`	Name of the image resource to return
`family`	Name of the image family to search for

Value

The selfLink of the image object

`gce_make_machinetype_url`
Construct a machineType URL

Description

Construct a machineType URL

Usage

```
gce_make_machinetype_url(predefined_type = NULL, cpus = NULL,
  memory = NULL, zone = gce_get_global_zone())
```

Arguments

`predefined_type`	A predefined machine type from gce_list_machinetype
`cpus`	If not defining `predefined_type`, the number of CPUs
`memory`	If not defining `predefined_type`, amount of memory
`zone`	zone for URL

Details

cpus must be in multiples of 2 up to 32 memory must be in multiples of 256

Value

A url for use in instance creation

---

gce_metadata_env *Turn metadata into an environment argument*

Description

This turns instance metadata into an environment argument R (and other software) can see. Only works on a running instance.

Usage

```
gce_metadata_env(key)
```

Arguments

key	The metadata key. Pass "" to list the keys

Value

The metadata key value, if successful

---

gce_pull_registry *Load a previously saved private Google Container*

Description

Load a previously saved private Google Container

Usage

```
gce_pull_registry(instance, container_name, container_url = "gcr.io",
  pull_only = FALSE, project = gce_get_global_project(), ...)
```

Arguments

instance	The VM to run within
container_name	The name of the saved container
container_url	The URL of where the container was saved
pull_only	If TRUE, will not run the container, only pull to the VM
project	Project ID for this request, default as set by gce_get_global_project
...	Other arguments passed to docker_run or docker_pull After starting a VM, you can load the container again using this command. • For Shiny based containers, pass "-p 80:3838" to run it at the IP URL • For RStudio based containers, pass "-p 80:8787" to run it at the IP URL

Value

The instance

See Also

Other container registry functions: gce_list_registry, gce_push_registry, gce_tag_container

---

gce_push_registry *Push to Google Container Registry*

Description

Commit and save a running container or docker image to the Google Container Registry

Usage

```
gce_push_registry(instance, save_name, container_name = NULL,
  image_name = NULL, container_url = "gcr.io",
  project = gce_get_global_project(), wait = FALSE)
```

Arguments

instance	The VM to run within
save_name	The new name for the saved image
container_name	A running docker container. Can't be set if image_name is too.
image_name	A docker image on the instance. Can't be set if container_name is too.
container_url	The URL of where to save container

project	Project ID for this request, default as set by gce_get_global_project This will only work on the Google Container optimised containers of image_family google_containers. Otherwise you will need to get a container authentication yourself (for now) It will start the push but it may take a long time to finish, especially the first time, this function will return whilst waiting but don't turn off the VM until its finished.
wait	Will wait for operation to finish on the instance if TRUE

Value

The tag the image was tagged with on GCE

See Also

Other container registry functions: gce_list_registry, gce_pull_registry, gce_tag_container

---

gce_rstudio_adduser *Creates a user on an RStudio templated instance*

Description

RStudio has users based on unix user accounts

Usage

```
gce_rstudio_adduser(instance, username, password, admin = TRUE,
  container = "rstudio")
```

Arguments

instance	An instance with RStudio installed via gce_vm_template
username	The user to create
password	The user password
admin	Default TRUE - Will the user be able to install packages and other sudo tasks?
container	The rstudio container to add the user to

Value

The instance

gce_rstudio_password *Changes password for a user on RStudio container*

Description

RStudio has users based on unix user accounts

Usage

```
gce_rstudio_password(instance, username, password, container = "rstudio")
```

Arguments

instance	An instance with RStudio installed via gce_vm_template
username	The user to change the password for
password	The user password
container	The rstudio container to add the user to

Value

The instance

gce_schedule_docker *Schedule running a docker image upon a VM*

Description

Utility function to start a VM to run a docker container on a schedule. You will need to create and build the Dockerfile first.

Usage

```
gce_schedule_docker(docker_image, schedule = "53 4 * * *",
  vm = gce_vm_scheduler())
```

Arguments

docker_image	the hosted docker image to run on a schedule
schedule	The schedule you want to run via cron
vm	A VM object to schedule the script upon that you can SSH into

Details

You may need to run gce_vm_scheduler yourself first and then set up SSH details if not defaults, to pass to argument vm

You can create a Dockerfile with your R script installed by running it through containeRit::dockerfile. It also takes care of any dependencies.

It is recommended to create a script that is self contained in output and input, e.g. don't save files to the VM, instead upload or download any files from Google Cloud Storage via authentication via googleAuthR::gar_gce_auth() then downloading and uploading data using library(googleCloudStorageR) or similar.

Once the script is working locally, build it and upload to a repository so it can be reached via argument docker_image

You can build via Google cloud repository build triggers, in which case the name can be created via gce_tag_container or build via docker_build to build on another VM or locally, then push to a registry via gce_push_registry

Any Docker image can be run, it does not have to be an R one.

Value

The crontab schedule of the VM including your script

See Also

Other scheduler functions: gce_vm_scheduler

Examples

```
## Not run:
# create a Dockerfile of your script
if(!require(containeRit)){
  devtools::install_github("o2r-project/containerit")
  library(containeRit)
}

script <- system.file("schedulescripts", "schedule.R", package = "googleComputeEngineR")

## put the "schedule.R" script in the working directory
file.copy(script, getwd())


## it will run the script whilst making the dockerfile
container <- dockerfile("schedule.R",
                        copy = "script_dir",
                        cmd = CMD_Rscript("schedule.R"),
                        soft = TRUE)
write(container, file = "Dockerfile")

## upload created Dockerfile to GitHub,
  then use a Build Trigger to create Docker image "demoDockerScheduler"
```

```
## built trigger uses "demo-docker-scheduler" as must be lowercase

## After image is built:
## Create a VM to run the schedule
vm <- gce_vm_scheduler("my_scheduler")

## setup any SSH not on defaults
vm <- gce_vm_setup(vm, username = "mark")

## get the name of the just built Docker image that runs your script
docker_tag <- gce_tag_container("demo-docker-scheduler", project = "gcer-public")

## Schedule the docker_tag to run every day at 0453AM
gce_schedule_docker(docker_tag, schedule = "53 4 * * *", vm = vm)



## End(Not run)
```

`gce_set_machinetype` *Changes the machine type for a stopped instance to the machine type specified in the request.*

Description

Changes the machine type for a stopped instance to the machine type specified in the request.

Usage

```
gce_set_machinetype(predefined_type, cpus, memory, instance,
  project = gce_get_global_project(), zone = gce_get_global_zone())
```

Arguments

`predefined_type`	A predefined machine type from gce_list_machinetype
`cpus`	If not defining `predefined_type`, the number of CPUs
`memory`	If not defining `predefined_type`, amount of memory
`instance`	Name of the instance resource to change
`project`	Project ID for this request, default as set by gce_get_global_project
`zone`	The name of the zone for this request, default as set by gce_get_global_zone

Details

Authentication scopes used by this function are:

- https://www.googleapis.com/auth/cloud-platform
- https://www.googleapis.com/auth/compute

Value

A zone operation job

See Also

Google Documentation

`gce_set_metadata` *Sets metadata for the specified instance to the data included in the request.*

Description

Set, change and append metadata for an instance.

Usage

```
gce_set_metadata(metadata, instance, project = gce_get_global_project(),
  zone = gce_get_global_zone())
```

Arguments

`metadata`	A named list of metadata key/value pairs to assign to this instance
`instance`	Name of the instance scoping this request
`project`	Project ID for this request, default as set by gce_get_global_project
`zone`	The name of the zone for this request, default as set by gce_get_global_zone

Details

Authentication scopes used by this function are:

- https://www.googleapis.com/auth/cloud-platform
- https://www.googleapis.com/auth/compute

To append to existing metadata passed a named list.

To change existing metadata pass a named list with the same key and modified value you will change.

To delete metadata pass an empty string `""` with the same key

See Also

Google Documentation

Other Metadata functions: `Metadata`

gce_shiny_addapp *Add Shiny app to a Shiny template instance*

Description

Add a local shiny app to a running Shiny VM installed via gce_vm_template via docker_build and gce_push_registry / gce_pull_registry.

Usage

```
gce_shiny_addapp(instance, app_image, dockerfolder = NULL)
```

Arguments

instance	The instance running Shiny
app_image	The name of the Docker image to create or use existing from Google Container Registry. Must be numbers, dashes or lowercase letters only.
dockerfolder	The folder location containing the Dockerfile and app dependencies

Details

To deploy a Shiny app, you first need to construct a Dockerfile which load the R packages and dependencies, as well as copying over the Shiny app in the same folder.

This function will take the Dockerfile, build it into a Docker image and upload it to Google Container Registry for use later.

If already created, then the function will download the app_image from Google Container Registry and start it on the instance provided.

Any existing Shiny Docker containers are stopped and removed, so if you want multiple apps put them in the same Dockerfile.

Value

The instance

Dockerfile

Example Dockerfile's are found in system.file("dockerfiles",package = "googleComputeEngineR")

The Dockerfile is in the same folder as your shiny app, which consists of a ui.R and server.R in a shiny subfolder. This is copied into the Dockerfile in the last line. Change the name of the subfolder to have that name appear in the final URL of the Shinyapp.

This is then run using the R commands below:

See Also

The vignette entry called Shiny App has examples and a walk through.

Examples

```
## Not run:

vm <- gce_vm("shiny-test",
             template = "shiny",
             predefined_type = "n1-standard-1")

vm <- vm_ssh_setup(vm)

app_dir <- system.file("dockerfiles","shiny-googleAuthRdemo",
                       package = "googleComputeEngineR")

gce_shiny_addapp(vm, app_image = "gceshinydemo", dockerfolder = app_dir)

# a new VM, it loads the Shiny docker image from before
gce_shiny_addapp(vm2, app_image = "gceshinydemo")


## End(Not run)
```

`gce_shiny_listapps` *List shiny apps on the instance*

Description

List shiny apps on the instance

Usage

```
gce_shiny_listapps(instance)
```

Arguments

`instance`	Instance with Shiny apps installed

Value

character vector

gce_shiny_logs *Get the latest shiny logs for a shinyapp*

Description

Get the latest shiny logs for a shinyapp

Usage

```
gce_shiny_logs(instance, shinyapp = NULL)
```

Arguments

instance	Instance with Shiny app installed
shinyapp	Name of shinyapp to see logs for. If NULL will return general shiny logs

Value

log printout

gce_ssh *Remotely execute ssh code, upload & download files.*

Description

Assumes that you have ssh & scp installed. If on Windows see website and examples for workarounds.

Usage

```
gce_ssh(instance, ..., key.pub = NULL, key.private = NULL, wait = TRUE,
  capture_text = "", username = Sys.info()[["user"]])

gce_ssh_upload(instance, local, remote, username = Sys.info()[["user"]],
  key.pub = NULL, key.private = NULL, verbose = FALSE, wait = TRUE)

gce_ssh_download(instance, remote, local, username = Sys.info()[["user"]],
  key.pub = NULL, key.private = NULL, verbose = FALSE,
  overwrite = FALSE, wait = TRUE)
```

Arguments

instance	Name of the instance of run ssh command upon
...	Shell commands to run. Multiple commands are combined with && so that execution will halt after the first failure.
key.pub	The filepath location of the public key
key.private	The filepath location of the private key
wait	Whether then SSH output should be waited for or run it asynchronously.
capture_text	Possible values are "", to the R console (the default), NULL or FALSE (discard output), TRUE (capture the output in a character vector) or a character string naming a file.
username	The username you used to generate the key-pair
local, remote	Local and remote paths.
verbose	If TRUE, will print command before executing it.
overwrite	If TRUE, will overwrite the local file if exists.

Details

Only works connecting to linux based instances.

On Windows you will need to install an ssh command line client - see examples for an example using RStudio's built in client.

You will need to generate a new SSH key-pair if you have not connected to the instance before via say the gcloud SDK.

To customise SSH connection see gce_ssh_setup

capture_text is passed to stdout and stderr of system2

Otherwise, instructions for generating SSH keys can be found here: https://cloud.google.com/compute/docs/instances/connecting-to-instance.

Uploads and downloads are recursive, so if you specify a directory, everything inside the directory will also be downloaded.

See Also

https://cloud.google.com/compute/docs/instances/connecting-to-instance

Other ssh functions: gce_ssh_addkeys, gce_ssh_browser, gce_ssh_setup

Examples

```
## Not run:


  vm <- gce_vm("my-instance")

  ## if you have already logged in via gcloud, the default keys will be used
  ## no need to run gce_ssh_addkeys
```

```
## run command on instance
gce_ssh(vm, "echo foo")
#> foo

## if running on Windows, use the RStudio default SSH client
## e.g. add C:\Program Files\RStudio\bin\msys-ssh-1000-18 to your PATH
## then run:
vm2 <- gce_vm("my-instance2")

## add SSH info to the VM object
## custom info
vm2 <- gce_ssh_setup(vm2,
                     username = "mark",
                     key.pub = "C://.ssh/id_rsa.pub",
                     key.private = "C://.ssh/id_rsa")

## run command on instance
gce_ssh(vm2, "echo foo")
#> foo



## End(Not run)
```

---

gce_ssh_addkeys *Add SSH details to a gce_instance*

Description

Add SSH details to a gce_instance

Usage

```
gce_ssh_addkeys(instance, key.pub = NULL, key.private = NULL,
  username = Sys.info()[["user"]], overwrite = FALSE)
```

Arguments

instance	The gce_instance
key.pub	filepath to public SSH key
key.private	filepath to the private SSK key
username	SSH username to login with
overwrite	Overwrite existing SSH details if they exist

Details

You will only need to run this yourself if you save your SSH keys somewhere other than `$HOME/.ssh/google_compute_engi` or use a different username than your local username as found in `Sys.info[["user"]]`, otherwise it will configure itself automatically the first time you use gce_ssh in an R session.

If key.pub is NULL then will look for default Google credentials at `file.path(Sys.getenv("HOME"), ".ssh", "google_c`

Value

The instance with SSH details included in $ssh

See Also

Other ssh functions: gce_ssh_browser, gce_ssh_setup, gce_ssh

Examples

```
## Not run:

  library(googleComputeEngineR)

  vm <- gce_vm("my-instance")

  ## if you have already logged in via gcloud, the default keys will be used
  ## no need to run gce_ssh_addkeys
  ## run command on instance
  gce_ssh(vm, "echo foo")


  ## if running on Windows, use the RStudio default SSH client
  ## e.g. add C:\Program Files\RStudio\bin\msys-ssh-1000-18 to your PATH
  ## then run:
  vm2 <- gce_vm("my-instance2")

  ## add SSH info to the VM object
  ## custom info
  vm <- gce_ssh_setup(vm,
                      username = "mark",
                      key.pub = "C://.ssh/id_rsa.pub",
                      key.private = "C://.ssh/id_rsa")

  ## run command on instance
  gce_ssh(vm, "echo foo")
  #> foo

  ## example to check logs of rstudio docker container
  gce_ssh(vm, "sudo journalctl -u rstudio")


## End(Not run)
```

gce_ssh_browser *Open a cloud SSH browser for an instance*

Description

This will open an SSH from the browser session if `getOption("browser")` is not NULL

Usage

```
gce_ssh_browser(instance)
```

Arguments

instance	the instance resource

Details

You will need to login the first time with an email that has access to the instance.

Value

Opens a browser window to the SSH session, returns the SSH URL.

See Also

https://cloud.google.com/compute/docs/ssh-in-browser

Other ssh functions: gce_ssh_addkeys, gce_ssh_setup, gce_ssh

gce_ssh_setup *Setup a SSH connection with GCE from a new SSH key-pair*

Description

Uploads ssh-keys to an instance

Usage

```
gce_ssh_setup(instance, key.pub = NULL, key.private = NULL,
  ssh_overwrite = FALSE, username = Sys.info()[["user"]])
```

Arguments

instance	Name of the instance of run ssh command upon
key.pub	The filepath location of the public key
key.private	The filepath location of the private key
ssh_overwrite	Will check if SSH settings already set and overwrite them if TRUE
username	The username you used to generate the key-pair

Details

This loads a public ssh-key to an instance's metadata. It does not use the project SSH-Keys, that may be set separately.

You will need to generate a new SSH key-pair if you have not connected to an instance before.

Instructions for this can be found here: https://cloud.google.com/compute/docs/instances/connecting-to-instance. Once you have generated run this function once to initiate setup.

If you have historically connected via gcloud or some other means, ssh keys may have been generated automatically.

These will be looked for and used if found, at `file.path(Sys.getenv("HOME"), ".ssh", "google_compute_engine.pub`

Value

TRUE if successful

See Also

https://cloud.google.com/compute/docs/instances/adding-removing-ssh-keys

Other ssh functions: gce_ssh_addkeys, gce_ssh_browser, gce_ssh

Examples

```
## Not run:

  library(googleComputeEngineR)

  vm <- gce_vm("my-instance")

  ## if you have already logged in via gcloud, the default keys will be used
  ## no need to run gce_ssh_addkeys
  ## run command on instance
  gce_ssh(vm, "echo foo")


  ## if running on Windows, use the RStudio default SSH client
  ## e.g. add C:\Program Files\RStudio\bin\msys-ssh-1000-18 to your PATH
  ## then run:
  vm2 <- gce_vm("my-instance2")

  ## add SSH info to the VM object
  ## custom info
  vm <- gce_ssh_setup(vm,
                      username = "mark",
                      key.pub = "C://.ssh/id_rsa.pub",
                      key.private = "C://.ssh/id_rsa")

  ## run command on instance
  gce_ssh(vm, "echo foo")
  #> foo
```

```
  ## example to check logs of rstudio docker container
  gce_ssh(vm, "sudo journalctl -u rstudio")


## End(Not run)
```

`gce_tag_container` *Return a container tag for Google Container Registry*

Description

Return a container tag for Google Container Registry

Usage

```
gce_tag_container(container_name, project = gce_get_global_project(),
  container_url = "gcr.io")
```

Arguments

`container_name`	A running docker container. Can't be set if `image_name` is too.
`project`	Project ID for this request, default as set by gce_get_global_project This will only work on the Google Container optimised containers of image_family google_containers. Otherwise you will need to get a container authentication yourself (for now) It will start the push but it may take a long time to finish, especially the first time, this function will return whilst waiting but don't turn off the VM until its finished.
`container_url`	The URL of where to save container

Value

A tag for use in Google Container Registry

See Also

Other container registry functions: `gce_list_registry`, `gce_pull_registry`, `gce_push_registry`

gce_vm *Create or fetch a virtual machine*

Description

Pass in the instance name to fetch its object, or create the instance via gce_vm_create.

Usage

```
gce_vm(name, ..., project = gce_get_global_project(),
  zone = gce_get_global_zone(), open_webports = TRUE)
```

Arguments

name	The name of the instance
...	Arguments passed on to gce_vm_create
	image_project Project ID of where the image lies
	image Name of the image resource to return
	image_family Name of the image family to search for
	disk_source Specifies a valid URL to an existing Persistent Disk resource.
	network The name of the network interface
	externalIP An external IP you have previously reserved, leave NULL to have one assigned or "none" for no external access.
	project Project ID for this request
	zone The name of the zone for this request
	dry_run whether to just create the request JSON
	auth_email If it includes '@' then assume the email, otherwise an environment file var that includes the email
	disk_size_gb If not NULL, override default size of the boot disk (size in GB)
	use_beta If set to TRUE will use the beta version of the API. Should not be used for production purposes.
	acceleratorCount [BETA] Number of GPUs to add to instance
	acceleratorType [BETA] Name of GPU to add, see gce_list_gpus
	name The name of the resource, provided by the client when initially creating the resource
	canIpForward Allows this instance to send and receive packets with non-matching destination or source IPs
	description An optional description of this resource
	metadata A named list of metadata key/value pairs assigned to this instance
	scheduling Scheduling options for this instance, such as preemptible instances
	serviceAccounts A list of service accounts, with their specified scopes, authorized for this instance
	tags A list of tags to apply to this instance

	predefined_type A predefined machine type from gce_list_machinetype **cpus** If not defining `predefined_type`, the number of CPUs **memory** If not defining `predefined_type`, amount of memory
`project`	Project ID for this request
`zone`	The name of the zone for this request
`open_webports`	If TRUE, will open firewall ports 80 and 443 if not open already

Details

Will get or create the instance as specified. Will wait for instance to be created if necessary.

Make sure the instance is big enough to handle what you need, for instance the default `f1-micro` will hang the instance when trying to install large R libraries.

Value

A `gce_instance` object

Creation logic

You need these parameters defined to call the right function for creation. Check the function definitions for more details.

If the VM name exists but is not running, it start the VM and return the VM object

If the VM is running, it will return the VM object

If you specify the argument `template` it will call gce_vm_template

If you specify one of `file` or `cloud_init` it will call gce_vm_container

Otherwise it will call gce_vm_create

Examples

```
## Not run:

library(googleComputeEngineR)
## auto auth, project and zone pre-set
## list your VMs in the project/zone

the_list <- gce_list_instances()

## start an existing instance
vm <- gce_vm("markdev")

## for rstudio, you also need to specify a username and password to login
vm <- gce_vm(template = "rstudio",
             name = "rstudio-server",
             username = "mark", password = "mark1234")

## specify your own cloud-init file and pass it into gce_vm_container()
vm <- gce_vm(cloud_init = "example.yml",
```

```
              name = "test-container",
              predefined_type = "f1-micro")

## specify disk size at creation
vm <- gce_vm('my-image3', disk_size_gb = 20)



## End(Not run)
```

`gce_vm_container` *Launch a container-VM image*

Description

This lets you specify docker images when creating the VM. These are a special class of Google instances that are setup for running Docker containers.

Usage

```
gce_vm_container(file = NULL, cloud_init = NULL,
  image_family = "cos-stable", ...)
```

Arguments

`file`	file location of a cloud-init file. One of `file` or `cloud_init` must be supplied
`cloud_init`	contents of a cloud-init file, for example read via `readChar(file, nchars = 32768)`
`image_family`	An image-family. It must come from the `google-containers` family.
`...`	Other arguments passed to gce_vm_create

Details

`file` expects a filepath to a https://cloudinit.readthedocs.io/en/latest/topics/format.html configuration file.

`image_project` will be ignored if set, overriden to `cos-cloud`. If you want to set it then use the gce_vm_create function directly that this function wraps with some defaults.

Value

A zone operation

See Also

https://cloud.google.com/container-optimized-os/docs/how-to/create-configure-instance

gce_vm_create *Creates an instance resource in the specified project using the data included in the request.*

Description

Creates an instance resource in the specified project using the data included in the request.

Usage

```
gce_vm_create(name, predefined_type = "f1-micro",
  image_project = "debian-cloud", image_family = "debian-8", cpus = NULL,
  memory = NULL, image = "", disk_source = NULL, network = "default",
  externalIP = NULL, canIpForward = NULL, description = NULL,
  metadata = NULL, scheduling = NULL, serviceAccounts = NULL,
  tags = NULL, auth_email = "GCE_AUTH_FILE",
  project = gce_get_global_project(), zone = gce_get_global_zone(),
  dry_run = FALSE, disk_size_gb = NULL, use_beta = FALSE,
  acceleratorCount = NULL, acceleratorType = "nvidia-tesla-k80")
```

Arguments

name	The name of the resource, provided by the client when initially creating the resource
predefined_type	A predefined machine type from gce_list_machinetype
image_project	Project ID of where the image lies
image_family	Name of the image family to search for
cpus	If not defining predefined_type, the number of CPUs
memory	If not defining predefined_type, amount of memory
image	Name of the image resource to return
disk_source	Specifies a valid URL to an existing Persistent Disk resource.
network	The name of the network interface
externalIP	An external IP you have previously reserved, leave NULL to have one assigned or "none" for no external access.
canIpForward	Allows this instance to send and receive packets with non-matching destination or source IPs
description	An optional description of this resource
metadata	A named list of metadata key/value pairs assigned to this instance
scheduling	Scheduling options for this instance, such as preemptible instances
serviceAccounts	A list of service accounts, with their specified scopes, authorized for this instance

tags	A list of tags to apply to this instance
auth_email	If it includes '@' then assume the email, otherwise an environment file var that includes the email
project	Project ID for this request
zone	The name of the zone for this request
dry_run	whether to just create the request JSON
disk_size_gb	If not NULL, override default size of the boot disk (size in GB)
use_beta	If set to TRUE will use the beta version of the API. Should not be used for production purposes.
acceleratorCount	[BETA] Number of GPUs to add to instance
acceleratorType	[BETA] Name of GPU to add, see gce_list_gpus

Details

Authentication scopes used by this function are:

- https://www.googleapis.com/auth/cloud-platform
- https://www.googleapis.com/auth/compute

cpus must be in multiples of 2 up to 32 memory must be in multiples of 256

One of image or image_family must be supplied

To create an instance you need to specify:

- Name
- Project [if not default]
- Zone [if not default]
- Machine type - either a predefined type or custom CPU and memory
- Network - usually default, specifies open ports etc.
- Image - a source image containing the operating system

You can add metadata to the server such as startup-script and shutdown-script. Details available here: https://cloud.google.com/compute/docs/storing-retrieving-metadata

If you want to not have an external IP then modify the instance afterwards

Value

A zone operation, or if the name already exists the VM object from gce_get_instance

Preemptible VMS

You can set preemptible VMs by passing this in the scheduling arguments scheduling = list(preemptible = TRUE)

This creates a VM that may be shut down prematurely by Google - you will need to sort out how to save state if that happens in a shutdown script etc. However, these are much cheaper.

See Also

Google Documentation

`gce_vm_delete` *Deletes the specified Instance resource.*

Description

Deletes the specified Instance resource.

Usage

```
gce_vm_delete(instance, project = gce_get_global_project(),
  zone = gce_get_global_zone())
```

Arguments

`instance`	Name of the instance resource, or an instance object e.g. from gce_get_instance
`project`	Project ID for this request, default as set by gce_get_global_project
`zone`	The name of the zone for this request, default as set by gce_get_global_zone

Details

Authentication scopes used by this function are:

- https://www.googleapis.com/auth/cloud-platform
- https://www.googleapis.com/auth/compute

See Also

Google Documentation

`gce_vm_gpu` *Launch a GPU enabled instance*

Description

Helper function that fills in some defaults passed to gce_vm

Usage

```
gce_vm_gpu(...)
```

Arguments

`...`	arguments passed to gce_vm

Details

If not specified, this function will enter defaults to get a GPU instance up and running.

- use_beta: TRUE
- acceleratorCount: 1
- acceleratorType: "nvidia-tesla-k80"
- scheduling: list(onHostMaintenance = "terminate", automaticRestart = TRUE)
- image_project: "centos-cloud"
- image_family: "centos-7"
- predefined_type: "n1-standard-1"
- metadata: the contents of the the startup script in system.file("startupscripts", "centos7cuda

Value

A VM object

See Also

Other GPU instances: gce_check_gpu, gce_list_gpus

---

gce_vm_logs *Open browser to the serial console output for a VM*

Description

Saves a few clicks

Usage

```
gce_vm_logs(instance, open_browser = TRUE,
  project = gce_get_global_project(), zone = gce_get_global_zone())
```

Arguments

instance	The VM to see serial console output for
open_browser	Whether to return a URL or open the browser
project	Project ID for this request, default as set by gce_get_global_project
zone	The name of the zone for this request, default as set by gce_get_global_zone

Value

a URL

gce_vm_reset *Performs a hard reset on the instance.*

Description

Performs a hard reset on the instance.

Usage

```
gce_vm_reset(instance, project = gce_get_global_project(),
  zone = gce_get_global_zone())
```

Arguments

instance	Name of the instance resource, or an instance object e.g. from gce_get_instance
project	Project ID for this request, default as set by gce_get_global_project
zone	The name of the zone for this request, default as set by gce_get_global_zone

Details

Authentication scopes used by this function are:

- https://www.googleapis.com/auth/cloud-platform
- https://www.googleapis.com/auth/compute

See Also

Google Documentation

gce_vm_scheduler *Create or start a scheduler VM*

Description

This starts up a VM with cron and docker installed that can be used to schedule scripts

Usage

```
gce_vm_scheduler(vm_name = "scheduler", ...)
```

Arguments

vm_name	The name of the VM scheduler to create or return
...	Arguments passed on to gce_vm
	name The name of the instance
	open_webports If TRUE, will open firewall ports 80 and 443 if not open already
	project Project ID for this request
	zone The name of the zone for this request

Value

A VM object

See Also

Other scheduler functions: gce_schedule_docker

---

gce_vm_start *Starts an instance that was stopped using the using the stop method.*

Description

Starts an instance that was stopped using the using the stop method.

Usage

```
gce_vm_start(instance, project = gce_get_global_project(),
  zone = gce_get_global_zone())
```

Arguments

instance	Name of the instance resource, or an instance object e.g. from gce_get_instance
project	Project ID for this request, default as set by gce_get_global_project
zone	The name of the zone for this request, default as set by gce_get_global_zone

Details

Authentication scopes used by this function are:

- https://www.googleapis.com/auth/cloud-platform
- https://www.googleapis.com/auth/compute

Value

An Operation object with pending status

See Also

Google Documentation

`gce_vm_stop` *Stops a running instance, shutting it down cleanly, and allows you to restart the instance at a later time.*

Description

Stops a running instance, shutting it down cleanly, and allows you to restart the instance at a later time.

Usage

```
gce_vm_stop(instance, project = gce_get_global_project(),
  zone = gce_get_global_zone())
```

Arguments

`instance`	Name of the instance resource, or an instance object e.g. from gce_get_instance
`project`	Project ID for this request, default as set by gce_get_global_project
`zone`	The name of the zone for this request, default as set by gce_get_global_zone

Details

Authentication scopes used by this function are:

- https://www.googleapis.com/auth/cloud-platform
- https://www.googleapis.com/auth/compute

Stopped instances do not incur per-minute, virtual machine usage charges while they are stopped, but any resources that the virtual machine is using, such as persistent disks and static IP addresses, will continue to be charged until they are deleted.

See Also

Google Documentation

`gce_vm_template` *Create a template container VM*

Description

This lets you specify templates for the VM you want to launch It passes the template on to gce_vm_container

Usage

```
gce_vm_template(template = c("rstudio", "shiny", "opencpu", "r-base",
  "example", "dynamic", "ropensci"), username = NULL, password = NULL,
  dynamic_image = NULL, image_family = "cos-stable", ...)
```

Arguments

`template`	The template available
`username`	username if needed (RStudio)
`password`	password if needed (RStudio)
`dynamic_image`	Supply an alternative to the default Docker image here to download
`image_family`	An image-family. It must come from the `cos-cloud` family.
`...`	Arguments passed on to `gce_vm_container` **file** file location of a cloud-init file. One of `file` or `cloud_init` must be supplied **cloud_init** contents of a cloud-init file, for example read via `readChar(file, nchars = 32768)` **image_family** An image-family. It must come from the `google-containers` family.

Details

Templates available are:

- rstudio An RStudio server docker image with tidyverse and devtools
- shiny A Shiny docker image
- opencpu An OpenCPU docker image
- r_base Latest version of R stable
- ropensci RStudio and tidyverse with all ropensci packages on CRAN and Github
- example A non-R test container running busybox
- dynamic Supply your own docker image to download such as `rocker/verse`

For `dynamic` templates you will need to launch the docker image with any ports you want opened, other settings etc. via docker_run.

Use `dynamic_image` to override the default rocker images e.g. `rocker/shiny` for shiny, etc.

Value

The VM object

Examples

```
## Not run:

 library(googleComputeEngineR)

 ## make instance using R-base
 vm <- gce_vm_template("r-base", predefined_type = "f1-micro", name = "rbase")

 ## run an R function on the instance within the R-base docker image
 docker_run(vm, "rocker/r-base", c("Rscript", "-e", "1+1"), user = "mark")
 #> [1] 2



## End(Not run)
```

`gce_wait` *Wait for an operation to finish*

Description

Will periodically check an operation until its status is `DONE`

Usage

```
gce_wait(operation, wait = 3, verbose = TRUE, timeout_tries = 50)

gce_check_zone_op(operation, wait = 3, verbose = TRUE)
```

Arguments

`operation`	The operation object
`wait`	Time in seconds between checks, default 3 seconds.
`verbose`	Whether to give user feedback
`timeout_tries`	Number of times to wait

Value

The completed job object, invisibly

get_dockerfolder *Get Dockerfolder of templates*

Description

This gets the folder location of available Dockerfile examples

Usage

```
get_dockerfolder(dockerfile_folder)
```

Arguments

dockerfile_folder	The folder containing Dockerfile

Value

file location

get_template_file *Show the cloud-config template files*

Description

Show the cloud-config template files

Usage

```
get_template_file(template)
```

Arguments

template	This returns the file location of template files for use in gce_vm_template

Details

Templates available are:

- rstudio An RStudio server docker image with the tidyverse installed
- shiny A Shiny docker image
- opencpu An OpenCPU docker image
- r_base Latest version of R stable
- example A non-R test container running busybox
- dynamic Supply your own docker image to download such as rocker/verse

Value

file location

`is.gce_global_operation` *Check if is a gce_global_operation*

Description

Check if is a gce_global_operation

Usage

```
is.gce_global_operation(x)
```

Arguments

`x`	The object to test if class `gce_global_operation`

Value

TRUE or FALSE

`is.gce_instance` *Check if is gce_instance*

Description

Check if is gce_instance

Usage

```
is.gce_instance(x)
```

Arguments

`x`	The object to test if class `gce_instance`

Value

TRUE or FALSE

is.gce_region_operation *Check if is a gce_region_operation*

Description

Check if is a gce_region_operation

Usage

```
is.gce_region_operation(x)
```

Arguments

x	The object to test if class gce_region_operation

Value

TRUE or FALSE

is.gce_zone_operation *Check if is a gce_zone_operation*

Description

Check if is a gce_zone_operation

Usage

```
is.gce_zone_operation(x)
```

Arguments

x	The object to test if class gce_zone_operation

Value

TRUE or FALSE

`localhost` *An object representing the current computer that R is running on.*

Description

An object representing the current computer that R is running on.

Usage

```
localhost
```

Format

An object of class `localhost` (inherits from `host`) of length 0.

Index