

Let's talk about credentials

What is a Juju credential?

"Credentials" → username + password (auth. contents/material) ← cloud vendor

Use the `add-credential` command

Use provides an arbitrary *credential name*.

"Juju credential" → cloud + Juju user

Juju client → host (system user)

Versatile & lightweight, but to demonstrate...

- You could have two hosts with **different** credential names but with the **same** auth. content.
- You could have two hosts with **same** credential names but with **different** auth. content.

Local credentials

"Local" means local to a Juju client

Use the `credentials` command to list:

```
local-credentials:
  google:
    gandalf:
      auth-type: oauth2
      client-email: 524443925537-compute@developer.gserviceaccount.com
      client-id: "112674610382273922048"
      project-id: juju-gce-1725
```

Here, `gandalf` is the credential for cloud google and the currently logged in Juju user (not shown).

Notice it says "local credentials".

Default local credential

When a local credential is needed, it can be specified with:

- `--credential` option (explicit)
- default local credential (implicit)

A default local credential can be set in two ways:

1. `set-default-credential` command (manually)
2. `bootstrap` command (automatically)

Automatic method is applied when a sole local credential exists when a controller is created.

The `--credential` option always refers to a local credential.

Remote credentials

Juju cloud (OOB or added) → need to authenticate → controller

Juju client → local credential → upload  *remote credential*

A remote credential is needed when a model is created.

A model is created with:

- bootstrap command ('controller' and 'default' models)
- add-model command

The remote credential then gets related to the model.

A remote credential cannot be shared with another Juju user.

Remote credentials (continued)

Use the `show-credentials` command to list:

```
controller-credentials:  
  google:  
    gandalf:  
      content:  
        auth-type: oauth2  
        client-email: 524443925537-compute@developer.gserviceaccount.com  
        client-id: "112674610382273922048"  
        project-id: juju-gce-1725  
      models:  
        controller: admin  
        default: admin
```

Here, `gandalf` is a remote credential for cloud `google` and is now related to models `'controller'` and `'default'`.

Notice it says "controller credentials".

It's possible for a remote credential to not be related to a model (the model was removed or a private cloud may not require authentication).

Credential used when creating a controller

With the bootstrap command, a **local** credential is uploaded (becoming remote).

The **local** credential that gets used comes from (in this order):

1. `--credential` option
2. default local credential
3. sole existing one

So an error will occur when all this is true:

- `--credential` option was not used
- default local credential was not set
- there are multiple **local** credentials

Only local credentials are considered when creating a controller.

Credential used when adding a model

With the `add-model` command, the **remote** credential that gets used comes from (in this order):

1. `--credential` option (upload method, as in bootstrap)
2. sole existing one

So an error will occur when all this is true:

- `--credential` option was not used
- there are multiple **remote** credentials

There is no concept of a default remote credential.

What remote credential is related to a model?

Use the `show-model` command. For the 'default' model:

```
juju show-model default
```

```
default:
  name: admin/default
  short-name: default
  .
  .
  .
  credential:
    name: jlaurin
    owner: admin
    cloud: aws
```

Here,

- model is default
- remote credential is jlaurin
- credential owner is admin
- cloud name is aws.

A credential owner is the Juju user who uploaded the local credential.

What models are related to a remote credential?

Use the `show-credential` command.

For the `saruman` credential on the `google` cloud:

```
juju show-credential google saruman
```

```
controller-credentials:
  google:
    saruman:
      content:
      .
      .
      .
    models:
      isengard: admin
      orthanc: admin
```

Here, remote credential `saruman` is related to models `isengard` and `orthanc`.

The `show-credentials` command can be used to show this information for all credentials.

Managing credentials with multiple users

Add a user and grant 'add-model' permissions:

```
juju add-user frodo
juju grant frodo add-model
```

Once user 'frodo' has registered the controller (register command):

```
juju add-model shire
```

But this will throw an error due to no **remote** credential found:

```
ERROR detecting credentials for "google" cloud provider: gce credentials not found
```

Managing credentials with multiple users (continued)

If multiple local credentials are added, say 'potatoes' and 'beer', and the operation is re-attempted an error is once again emitted:

```
ERROR more than one credential is available. List credentials with:
```

```
juju credentials
```

```
and then run the add-model command again with the --credential option.
```

This time a **local** credential was not selected from among the multiple ones available.

The local credential would then be uploaded to become a **remote** credential.

The model can be added in this way:

```
juju add-model --credential potatoes shire
```

Dealing with expired credentials

If your cloud vendor has expired your account credentials (it happens!) workloads will continue to run but Juju will be unable to administer the cloud.

On the vendor side, update your password (dashboard, API).

On the Juju side, do one of the following:

1. update the existing **remote** credential [next slide]
2. create a new **local** credential and relate it to affected models [next slide]

An expired credential can be simulated by deactivating the account on the vendor side.

Dealing with expired credentials (continued)

Option 1: Update the existing remote credential

Use the `update-credential` command. The identically named **local** credential will be uploaded to the controller. So the local credential first needs to be changed.

The local credential contents are changed with the `add-credential` command, where the `--replace` option is needed because the credential is already existing.

```
juju add-credential --replace google -f credentials-gandalf-changed.yaml
juju update-credential google gandalf
```

The contents of credential `gandalf` was modified in file `credentials-gandalf-changed.yaml`.

In Juju v.2.6, a remote credential can be updated directly:

```
juju update-credential google gandalf -f credentials-gandalf-changed.yaml
```

Dealing with expired credentials (continued)

Option 2: Create a new credential and relate it to affected models

Use the `add-credential` command to create a **local** credential and the `set-credential` command to relate a **remote** credential to a model. Here, the affected models are 'shire' and 'rohan'.

```
juju add-credential google -f credentials-mithrandir.yaml
juju set-credential -m shire google mithrandir
juju set-credential -m rohan google mithrandir
```

The contents for credential `mithrandir` were added to file `credentials-mithrandir.yaml`.

Bonus: A credential targeted with the `set-credential` command will upload the identically-named **local** credential if it is not found remotely. [That's what happened here.](#)

Dealing with a reanimated cloud account

Very similar to the case of an expired credential, a reanimated cloud account is one that has been deactivated and then reactivated, usually on the cloud dashboard.

Juju has invalidated the credential internally. You do not need to replace the credential's contents. Just announce to the controller that the remote credential is a good one. The solution is as before: "update" the remote credential with the `update-credential` command.

Assuming that the cloud account is associated with remote credential `tharkun`, just do:

```
juju update-credential google tharkun
```