ERICSSON

Atlas CLI End User Guide

Cloud Execution Environment

USER GUIDE

11/1553-CRA 119 1873/5 Uen A

Contents

1	**Introduction**	**1**
2	**cApp Create**	**2**
2.1	Create a cApp	2
2.2	Create a cApp with All Option Fields	4
3	**cApp Delete**	**6**
4	**cApp Update**	**7**
5	**cApp Show**	**9**
6	**cApp Show Template**	**11**
7	**List Available cApps**	**14**
8	**List Environment Files for a cApp**	**15**
9	**Show Environment File Content**	**16**
10	**cApp Export**	**17**
11	**Application Scaling**	**18**
11.1	Stack Scale Out	18
11.2	Stack Scale In	20
11.3	Stack Scaling List	21
12	**Deployment Wizard**	**23**
12.1	Deployment Wizard From CLI	23

1 Introduction

This user guide provides the syntax descriptions and examples of the commands used for managing Atlas by using the Command-Line Interface (CLI).

The CLI can be reached by using either of the following clients:

- OVFT client
- OpenStack client

  Note: The OpenStack client is planned to be the officially supported client on a long term basis.

Open Virtualization Format Translator (OVFT) Cloud Application (cApp) is an Ericsson service that provides catalog application handling. The templates stored in the Atlas database can be deployed by using the Heat Orchestration service. The following input formats are supported:

Open Virtualization Format (OVF):
The OVF package is processed to generate a corresponding HOT template and store the generated HOT in the Atlas database.

Heat Orchestration Template (HOT):
The input HOT template is getting stored in the Atlas database.

Topology and Orchestration Specification for Cloud Applications (TOSCA):
The input TOSCA template is getting stored in the Atlas database.

Note: Atlas includes Mistral, so application life cycle management is enriched with workflows and OpenStack specific actions. For further information, see *OpenStack End User Guide*.

The target group of this user guide consists of Atlas end users managing and uploading catalog cApps by using the CLI.

2 cApp Create

This section describes how to upload cApps.

2.1 Create a cApp

Syntax for OVFT client:

ovft capp-create --name ***<capp_name>*** ⇒
--file ***<path_to_the_capp>*** **--type** ***<ovf/hot/tosca>***

Syntax for OpenStack client:

openstack capp create --name ***<capp_name>*** ⇒
--file ***<path_to_the_capp>*** **--type** ***<ovf/hot/tosca>***

Examples:

```
root@atlas:~# ovft capp-create --type ovf --name test-capp --file ~/test.ova

+--------------+--------------------------------------+
| Property     | Value                                |
+--------------+--------------------------------------+
| created_at   | 2015-12-02T06:53:49.000000           |
| description  |                                      |
| fault        |                                      |
| id           | 4ebaef72-b8cc-4616-8b49-e5563fdaf0b3 |
| image_ids    |                                      |
| is_protected | False                                |
| is_public    | False                                |
| name         | test-capp                            |
| status       | Creating                             |
| type         | ovf                                  |
| updated_at   | 2015-12-02T06:53:50.000000           |
| user_id      | 9a89e98b6b764a37a83ba44a27161352     |
+--------------+--------------------------------------+
```

Example 1 Create a cApp with Type “ovf” by Using an OVFT Client

```
root@atlas:~# openstack capp create --type ovf --name test_capp --file ~/ test.ova

+--------------+--------------------------------------+
| Field        | Value                                |
+--------------+--------------------------------------+
| created_at   | 2015-12-02T06:56:53.000000           |
| description  |                                      |
| fault        |                                      |
| id           | d884e398-308b-4851-978f-4974e7eabcb3 |
| image_ids    |                                      |
| is_protected | False                                |
| is_public    | False                                |
| name         | test_capp                            |
| status       | Creating                             |
| type         | ovf                                  |
| updated_at   | 2015-12-02T06:56:55.000000           |
| user_id      | 9a89e98b6b764a37a83ba44a27161352     |
+--------------+--------------------------------------+
```

Example 2 Create a cApp with Type OVF by using an OpenStack Client

```
root@atlas:~# ovft capp-create --name test --file ~/test.yaml --type hot

+--------------+--------------------------------------+
| Property     | Value                                |
+--------------+--------------------------------------+
| created_at   | 2015-12-02T06:59:59.000000           |
| description  |                                      |
| fault        |                                      |
| id           | b0c50a44-6f89-485b-8d10-c9b1eda309f3 |
| image_ids    |                                      |
| is_protected | False                                |
| is_public    | False                                |
| name         | test                                 |
| status       | Creating                             |
| type         | hot                                  |
| updated_at   | 2015-12-02T06:59:59.000000           |
| user_id      | 9a89e98b6b764a37a83ba44a27161352     |
+--------------+--------------------------------------+
```

Example 3 Create a cApp with Type HOT by Using an OVFT Client

```
root@atlas:~# openstack capp create --name test --file ~/test.yaml --type hot

+--------------+--------------------------------------+
| Field        | Value                                |
+--------------+--------------------------------------+
| created_at   | 2015-12-02T07:01:27.000000           |
| description  |                                      |
| fault        |                                      |
| id           | b2ccb378-ca2a-429f-9816-e0b9443d12d1 |
| image_ids    |                                      |
| is_protected | False                                |
| is_public    | False                                |
| name         | test                                 |
| status       | Creating                             |
| type         | hot                                  |
| updated_at   | 2015-12-02T07:01:27.000000           |
| user_id      | 9a89e98b6b764a37a83ba44a27161352     |
+--------------+--------------------------------------+
```

Example 4 Create a cApp with Type HOT by Using an OpenStack Client

```
root@atlas:~# ovft capp-create --name test --file ~/test.yaml --type tosca

+--------------+--------------------------------------+
| Property     | Value                                |
+--------------+--------------------------------------+
| created_at   | 2015-12-02T07:03:02.000000           |
| description  |                                      |
| fault        |                                      |
| id           | a9301219-1d8c-441e-bbd3-8da7fc8ad077 |
| image_ids    |                                      |
| is_protected | False                                |
| is_public    | False                                |
| name         | test                                 |
| status       | Creating                             |
| type         | tosca                                |
| updated_at   | 2015-12-02T07:03:02.000000           |
| user_id      | 9a89e98b6b764a37a83ba44a27161352     |
+--------------+--------------------------------------+
```

Example 5 Create a cApp with Type TOSCA by Using an OVFT Client

```
root@atlas:~# openstack capp create --name test --file ~/test.yaml --type tosca

+--------------+--------------------------------------+
| Field        | Value                                |
+--------------+--------------------------------------+
| created_at   | 2015-12-02T07:03:18.000000           |
| description  |                                      |
| fault        |                                      |
| id           | 66616389-b434-4e73-9a3e-5a5d369f0a76 |
| image_ids    |                                      |
| is_protected | False                                |
| is_public    | False                                |
| name         | test                                 |
| status       | Creating                             |
| type         | tosca                                |
| updated_at   | 2015-12-02T07:03:18.000000           |
| user_id      | 9a89e98b6b764a37a83ba44a27161352     |
+--------------+--------------------------------------+
```

Example 6 Create a cApp with Type TOSCA by Using an OpenStack Client

2.2 Create a cApp with All Option Fields

This section describes how to create a cApp with `desc`, `is-public`, and `is-protected` fields.

Syntax for OVFT client:

ovft capp-create --type ***<ovf/hot/tosca>*** ⇒
--name ***<capp_name>*** **--file** ***<path_to_the_capp>*** ⇒
--desc ***<capp_description>*** **--is-public** ***<true/false>*** ⇒
--is-protected ***<true/false>***

Syntax for OpenStack client:

openstack capp create --type ***<ovf/hot/tosca>*** ⇒
--name ***<capp_name>*** **--file** ***<path_to_the_capp>*** ⇒
--desc ***<capp_description>*** **--is-public** ***<true/false>*** ⇒
--is-protected ***<true/false>***

Examples:

```
root@atlas:~# ovft capp-create --name test --file test.ova --type ovf⇒
 --desc "Create test capp" --is-public true --is-protected true

+--------------+--------------------------------------+
| Property     | Value                                |
+--------------+--------------------------------------+
| created_at   | 2015-12-02T07:09:08.000000           |
| description  | Create test capp                     |
| fault        |                                      |
| id           | f6256648-7574-42f4-897d-068484a20d30 |
| image_ids    |                                      |
| is_protected | True                                 |
| is_public    | True                                 |
| name         | test                                 |
| status       | Creating                             |
| type         | ovf                                  |
| updated_at   | 2015-12-02T07:09:09.000000           |
| user_id      | 9a89e98b6b764a37a83ba44a27161352     |
+--------------+--------------------------------------+
```

Example 7 Create a cApp with All Options by Using an OVFT Client

```
root@atlas:~# openstack capp create --name test --file test.ova --type ovf⇒
 --desc "Create test capp" --is-public true --is-protected true

+--------------+--------------------------------------+
| Field        | Value                                |
+--------------+--------------------------------------+
| created_at   | 2015-12-02T07:09:55.000000           |
| description  | Create test capp                     |
| fault        |                                      |
| id           | c8511b65-dd3c-429a-8687-ed21bc610168 |
| image_ids    |                                      |
| is_protected | True                                 |
| is_public    | True                                 |
| name         | test                                 |
| status       | Creating                             |
| type         | ovf                                  |
| updated_at   | 2015-12-02T07:09:57.000000           |
| user_id      | 9a89e98b6b764a37a83ba44a27161352     |
+--------------+--------------------------------------+
```

Example 8 Create a cApp with All Options by Using an OpenStack Client

3 cApp Delete

This section describes how to delete cApps.

Syntax for OVFT client:

ovft capp-delete ***<capp_name or capp_id>***

Syntax for OpenStack client:

openstack capp delete ***<capp_name or capp_id>***

Note: Do not use ***capp_name*** in these commands if there are multiple cApps available with the same name.

Examples:

```
root@atlas:~# ovft capp-delete c8511b65-dd3c-429a-8687-ed21bc610168
```

Example 9 Delete a cApp by cApp Name by Using an OVFT Client

```
root@atlas:~# openstack capp delete c8511b65-dd3c-429a-8687-ed21bc610168
```

Example 10 Delete a cApp by cApp Name by Using an OpenStack Client

4 cApp Update

This section describes how to update cApps.

Syntax for OVFT client:

```
ovft capp-update <existing_capp_name or existing_capp_id> --name <new_name>
```

Syntax for OpenStack client:

```
openstack capp update <existing_capp_name or existing_capp_id> --name <new_name>
```

Note: Do not use ***existing_capp_name*** in these commands if there are multiple cApps available with the same name.

The following arguments can be updated for an uploaded application:

--name <name> The name of the cApp

--file <file> A local file that contains disk cApp to be uploaded during the update. Alternatively, cApps can be passed to the client by using Standard Input (stdin).

--is-public {true,false}
Makes the cApp accessible to the public.

--is-protected {true,false}
Prevents the cApp from being deleted.

--desc <description>
Description of the cApp

Examples:

```
root@atlas:~# ovft capp-update c8511b65-dd3c-429a-8687-ed21bc610168 --name test1

+--------------+--------------------------------------+
| Property     | Value                                |
+--------------+--------------------------------------+
| created_at   | 2015-12-02T07:09:55.000000           |
| description  | Create test capp                     |
| fault        |                                      |
| id           | c8511b65-dd3c-429a-8687-ed21bc610168 |
| image_ids    | b97a9ac5-9f04-49ce-9475-caa29702d574 |
| is_protected | True                                 |
| is_public    | True                                 |
| name         | test1                                |
| status       | active                               |
| type         | ovf                                  |
| updated_at   | 2015-12-02T07:15:40.000000           |
| user_id      | 9a89e98b6b764a37a83ba44a27161352     |
+--------------+--------------------------------------+
```

Example 11 Update cApp by Using an OVFT Client

```
root@atlas:~# openstack capp update c8511b65-dd3c-429a-8687-ed21bc610168 --name test

+--------------+--------------------------------------+
| Field        | Value                                |
+--------------+--------------------------------------+
| created_at   | 2015-12-02T07:09:55.000000           |
| description  | Create test capp                     |
| fault        |                                      |
| id           | c8511b65-dd3c-429a-8687-ed21bc610168 |
| image_ids    | b97a9ac5-9f04-49ce-9475-caa29702d574 |
| is_protected | True                                 |
| is_public    | True                                 |
| name         | test                                 |
| status       | active                               |
| type         | ovf                                  |
| updated_at   | 2015-12-02T07:17:16.000000           |
| user_id      | 9a89e98b6b764a37a83ba44a27161352     |
+--------------+--------------------------------------+
```

Example 12 Update cApp by Using an OpenStack Client

5 cApp Show

This section describes how to display the metadata of cApps.

Syntax for OVFT client:

ovft capp-show ***<capp_name or capp_id>***

Syntax for OpenStack client:

openstack capp show ***<capp_name or capp_id>***

Note: Do not use ***capp_name*** in these commands if there are multiple cApps available with the same name.

Examples:

```
root@atlas:~# ovft capp-show c8511b65-dd3c-429a-8687-ed21bc610168

+--------------+--------------------------------------+
| Property     | Value                                |
+--------------+--------------------------------------+
| created_at   | 2015-12-02T07:09:55.000000           |
| description  | Create test capp                     |
| fault        |                                      |
| id           | c8511b65-dd3c-429a-8687-ed21bc610168 |
| image_ids    | b97a9ac5-9f04-49ce-9475-caa29702d574 |
| is_protected | True                                 |
| is_public    | True                                 |
| name         | test                                 |
| status       | active                               |
| type         | ovf                                  |
| updated_at   | 2015-12-02T07:17:16.000000           |
| user_id      | 9a89e98b6b764a37a83ba44a27161352     |
+--------------+--------------------------------------+
```

Example 13 Show cApp by Using an OVFT Client

```
root@atlas:~# openstack capp show c8511b65-dd3c-429a-8687-ed21bc610168

+--------------+--------------------------------------+
| Field        | Value                                |
+--------------+--------------------------------------+
| created_at   | 2015-12-02T07:09:55.000000           |
| description  | Create test capp                     |
| fault        |                                      |
| id           | c8511b65-dd3c-429a-8687-ed21bc610168 |
| image_ids    | b97a9ac5-9f04-49ce-9475-caa29702d574 |
| is_protected | True                                 |
| is_public    | True                                 |
| name         | test                                 |
| status       | active                               |
| type         | ovf                                  |
| updated_at   | 2015-12-02T07:17:16.000000           |
| user_id      | 9a89e98b6b764a37a83ba44a27161352     |
+--------------+--------------------------------------+
```

Example 14 Show cApp by Using an OpenStack Client

6 cApp Show Template

This section describes how to display the stored HOT or TOSCA template of a cApp.

Syntax for OVFT client:

ovft capp-template-show ***<capp_name or capp_id>***

Syntax for OpenStack client:

openstack capp template show ***<capp_name or capp_id>***

Note: Do not use ***capp_name*** in these commands if there are multiple cApps available with the same name.

Examples:

```
root@atlas:~# ovft capp-template-show c8511b65-dd3c-429a-8687-ed21bc610168
description: stack template generated from OVF file
heat_template_version: '2013-05-23'
parameters:
  param_1:
    description: IP subnet
    label: BAT-OVF-ctrl_100 subnet
    type: string
  param_2:
    description: Gateway
    label: BAT-OVF-ctrl_100 gateway
    type: string
  param_3:
    description: Select fixedip within BAT-OVF-BGW-L3-3958_subnet
    label: Fixed IP defined for port vfr1_vnic2
    type: string
  param_4:
    description: Select fixedip within BAT-OVF-BGW-L3-3959_subnet
    label: Fixed IP defined for port vfr1_vnic3
    type: string
resources:
  BAT-A1-001:
    depends_on: delay_6
    properties:
      config_drive: 'True'
      flavor:
        get_resource: flavor_BAT-A1-001
      image: b97a9ac5-9f04-49ce-9475-caa29702d574
      metadata: {}
      name: BAT-A1-001
      networks:
      - port:
          get_resource: BAT-A1-001_existing_port_1
      - port:
          get_resource: port_46
      - port:
          get_resource: port_48
      scheduler_hints:
        different_host: []
        same_host:
        - get_resource: BAT-A2-001
        - get_resource: BAT-B1-001
        - get_resource: BAT-C1-001
    type: OS::Nova::Server
  BAT-A1-001_existing_port_1:
    properties:
      name: BAT-A1-001_existing_port_1
      network_id: 4df3ff29-a31e-464a-943b-291c778e8881
    type: OS::Neutron::Port
  BAT-A1-003:
    depends_on:
    - BAT-A2-001
    - BAT-A2-003
    - BAT-A2-005
    - BAT-A2-007
    properties:
      config_drive: 'True'
      flavor:
        get_resource: flavor_BAT-A1-003
      image: b97a9ac5-9f04-49ce-9475-caa29702d574
      metadata: {}
      name: BAT-A1-003
      networks:
      - port:
          get_resource: port_13
      - port:
          get_resource: port_15
      scheduler_hints:
        different_host: []
        same_host:
        - get_resource: BAT-A2-003
    type: OS::Nova::Server
```

Example 15 Display HOT Template of a cApp by Using an OVFT Client

```
root@atlas:~# openstack capp template show c8511b65-dd3c-429a-8687-ed21bc610168
description: stack template generated from OVF file
heat_template_version: '2013-05-23'
parameters:
  param_1:
    description: IP subnet
    label: BAT-OVF-ctrl_100 subnet
    type: string
  param_2:
    description: Gateway
    label: BAT-OVF-ctrl_100 gateway
    type: string
  param_3:
    description: Select fixedip within BAT-OVF-BGW-L3-3958_subnet
    label: Fixed IP defined for port vfr1_vnic2
    type: string
  param_4:
    description: Select fixedip within BAT-OVF-BGW-L3-3959_subnet
    label: Fixed IP defined for port vfr1_vnic3
    type: string
resources:
  BAT-A1-001:
    depends_on: delay_6
    properties:
      config_drive: 'True'
      flavor:
        get_resource: flavor_BAT-A1-001
      image: b97a9ac5-9f04-49ce-9475-caa29702d574
      metadata: {}
      name: BAT-A1-001
      networks:
      - port:
          get_resource: BAT-A1-001_existing_port_1
      - port:
          get_resource: port_46
      - port:
          get_resource: port_48
      scheduler_hints:
        different_host: []
        same_host:
        - get_resource: BAT-A2-001
        - get_resource: BAT-B1-001
        - get_resource: BAT-C1-001
    type: OS::Nova::Server
```

Example 16 Display HOT Template of a cApp by Using an OpenStack Client

7 List Available cApps

This section describes how to list all the uploaded cApps.

Syntax for OVFT client: **`ovft capp-list`**

Syntax for OpenStack client: **`openstack capp list`**

Examples:

```
root@atlas:~# ovft capp-list
+--------------------------------------+------+--------+------+----------------------------+
| id                                   | name | status | type | created_at                 |
+--------------------------------------+------+--------+------+----------------------------+
| c8511b65-dd3c-429a-8687-ed21bc610168 | test | active | ovf  | 2015-12-02T07:09:55.000000 |
+--------------------------------------+------+--------+------+----------------------------+
```

Example 17 List Available cApps by Using an OVFT Client

```
root@atlas:~# openstack capp list
+--------------------------------------+-------+--------+----------------------------+
| id                                   | name  | status | created_at                 |
+--------------------------------------+-------+--------+----------------------------+
| 7a6478c2-18ff-4386-b07f-75f4c7c1b522 | tests | None   | 2015-12-02T07:21:56.000000 |
+--------------------------------------+-------+--------+----------------------------+
```

Example 18 List Available cApps by Using an OpenStack Client

8 List Environment Files for a cApp

This section describes how to list the environment files for an OVF type cApp.

Syntax for OVFT client:

ovft capp-file-list ***<capp_name or capp_id>***

Syntax for OpenStack client:

openstack capp file list ***<capp_name or capp_id>***

Note: Do not use ***capp_name*** in these commands if there are multiple cApps available with the same name.

Examples:

```
root@atlas:~# ovft capp-file-list test

+--------------------------------------+-------------+
| file_id                              | file_name   |
+--------------------------------------+-------------+
| 67a0c543-72e7-4951-ba36-419963797f06 | config2.xml |
| ba15be76-5168-4aa0-9900-97c4d1338763 | config1.xml |
+--------------------------------------+-------------+
```

Example 19 List Environment Files for a cApp by Using an OVFT Client

```
root@atlas:~# openstack capp file list test

+--------------------------------------+-------------+
| file_id                              | file_name   |
+--------------------------------------+-------------+
| 67a0c543-72e7-4951-ba36-419963797f06 | config2.xml |
| ba15be76-5168-4aa0-9900-97c4d1338763 | config1.xml |
+--------------------------------------+-------------+
```

Example 20 List Environment Files for a cApp by Using an OpenStack Client

9 Show Environment File Content

This section describes how to display the content of the environment file for an OVF type cApp.

Syntax for OVFT client:

ovft capp-file-show ***<capp_name or capp_id>***

Syntax for Openstack client:

openstack capp file show ***<capp_name or capp_id>***

Note: Do not use ***capp_name*** in these commands if there are multiple cApps available with the same name.

Examples:

```
root@atlas:~# ovft capp-file-show 67a0c543-72e7-4951-ba36-419963797f06

+--------------+--------------------------------------+
| Property     | Value                                |
+--------------+--------------------------------------+
| file_content | env                                  |
|              | file                                 |
|              | example                              |
|              |                                      |
| file_id      | 67a0c543-72e7-4951-ba36-419963797f06 |
| file_name    | config2.xml                          |
+--------------+--------------------------------------+
```

Example 21 Show Environment File Content by Using an OVFT Client

```
root@atlas:~# openstack capp file show 67a0c543-72e7-4951-ba36-419963797f06

+--------------+--------------------------------------+
| Field        | Value                                |
+--------------+--------------------------------------+
| file_content | env                                  |
|              | file                                 |
|              | example                              |
|              |                                      |
| file_id      | 67a0c543-72e7-4951-ba36-419963797f06 |
| file_name    | config2.xml                          |
+--------------+--------------------------------------+
```

Example 22 Show Environment File Content by Using an OpenStack Client

10 cApp Export

This section describes how to export cApps from the database to the Atlas file system.

Syntax for OVFT client:

```
ovft capp-export --file <local_file_for_saving_downloaded_capp_data> <capp_name or capp_id>
```

Syntax for OpenStack client:

```
openstack capp export --file <local_file_for_saving_downloaded_capp_data> <capp_name or capp_id>
```

Note: Do not use ***capp_name*** in these commands if there are multiple cApps available with the same name.

Examples:

```
root@atlas:~# ovft capp-export --file ~/check c8511b65-dd3c-429a-8687-ed21bc610168
```

Example 23 Export a cApp by Using an OVFT Client

```
root@atlas:~# openstack capp export --file ~/check c8511b65-dd3c-429a-8687-ed21bc610168
```

Example 24 Export a cApp by Using an OpenStack Client

11 Application Scaling

This section describes how to manually scale resources of a running stack.

11.1 Stack Scale Out

Stack-scale (--method out) creates all the resources mentioned in the scale out JavaScript Object Notation (JSON) file.

The Scale Out JSON file contains the following sections:

scaling_groups Contains the details of the resources.

resources Contains the list of resources to scale.

source Name of the server you want to scale

target Name of the server with which the source server gets scaled

network(Opt) Mac address or IP address for a particular network:

- **name** Name of network
- **mac_address(Opt)** Mac address for the target VM
- **ip_address(Opt)** Fixed IP address for the target VM

personality(Opt)
Personality section for the target VM

Note: `(Opt)` in the section names stands for `(Optional)`. The `Mac Address`, `Fixed IP address`, and `Personality` sections are optional.

The following example shows the content of the Scale Out JSON file.

```
root@atlas:~# cat scale_out_fixed_ip_mac.json
{ "scaling_groups": {
    "resources": [
        {
                "source" : "vm1",
                "target" : "vm1_scaled",
                "networks" : [
                    {
                        "name" : "N1",
                        "mac_address" : "44:40:44:40:44:33",
                        "ip_address" : "10.0.0.18"
                }],
                "personality": [
                {
                    "contents": "Something",
                    "path": "/etc/script.sh"
                }
            ]

        }
    ]
} }
```

Example 25 The Scale Out JSON File

Syntax for OVFT client:

```
ovft stack-scale --method out --file <scale_out_json_file> <stack_name ⇒
or stack_id>
```

Syntax for OpenStack client:

```
openstack stack scale --method out --file <scale_out_json_file> <stack_name ⇒
or stack_id>
```

Note: Do not use ***capp_name*** in these commands if there are multiple cApps available with the same name.

Examples:

```
root@atlas:~# ovft stack-scale --file scale_out.json --method out ⇒
scale_out_stack

+-----------+------------+
| Property  | Value      |
+-----------+------------+
| resources | vm1_scaled |
+-----------+------------+
```

Example 26 Scale Out a Stack by Using an OVFT Client

```
root@atlas:~# openstack stack scale --file scale_out.json --method out scale_out_stack

+-----------+------------+
| Field     | Value      |
+-----------+------------+
| resources | vm1_scaled |
+-----------+------------+
```

Example 27 Scale Out a Stack by Using an OpenStack Client

```
atlasadm@atlas:~$ heat resource-list  6866f864-3231-49ff-bdc4-99db84d69b3c
+---------------+--------------------------------------+
| resource_name | physical_resource_id                 |
+---------------+--------------------------------------+
| N1            | 7a8705b5-4e33-4155-9f03-38b07ad4e888 |
| N1_subnet     | 68b4dd6f-995d-4e5d-9653-53b2a1d0b533 |
| flavor_vm1    | cdc480c6-68df-4411-96ed-3a68622012f7 |
| vm1           | ba67ef98-a0e5-4565-bab3-69f30be1f86f |
| vm1_scaled    | bde6733b-677a-461b-a558-37ecf119943e |
+---------------+--------------------------------------+
+----------------------------+-----------------+----------------------+
| resource_type              | resource_status | updated_time         |
+----------------------------+-----------------+----------------------+
| OS::Neutron::Net           | CREATE_COMPLETE | 2015-12-02T07:49:04Z |
| OS::Neutron::Subnet        | CREATE_COMPLETE | 2015-12-02T07:49:04Z |
| Ericsson::Nova::Flavor     | CREATE_COMPLETE | 2015-12-02T07:49:04Z |
| OS::Nova::Server           | CREATE_COMPLETE | 2015-12-02T07:49:04Z |
| OS::Heat::AutoScalingGroup | CREATE_COMPLETE | 2015-12-02T07:49:54Z |
+----------------------------+-----------------+----------------------+
```

Example 28 Stack Resource List

11.2 Stack Scale In

Stack-scale (--method in) deletes all the scaled resources associated to a stack mentioned in the scale in JSON file.

The Scale In JSON file contains the following section:

groups List of groups you want to scale in

The following example shows the content of the Scale In JSON file.

```
root@atlas:~# cat scale_in.json
{
        "groups" : [
                "vm1_scaled"
        ]
}
```

Example 29 The Scale In JSON File

Syntax for OVFT client:

ovft stack-scale --method in ⇒
--file *<scale_in_json_file> <stack_name>*

Syntax for OpenStack client:

openstack stack scale --method in ⇒
--file *<scale_in_json_file> <stack_name>*

Examples:

```
root@atlas:~# ovft stack-scale --file scale_in.json --method in scale_out_stack

+-------------+-------+
| Property    | Value |
+-------------+-------+
| scaled_info | OK    |
+-------------+-------+
```

Example 30 Scale In a Stack by Using an OVFT Client

```
root@atlas:~# openstack stack scale --file scale_in.json --method in scale_out_stack

+-------------+-------+
| Field       | Value |
+-------------+-------+
| scaled_info | OK    |
+-------------+-------+
```

Example 31 Scale In a Stack by Using an OpenStack Client

11.3 Stack Scaling List

The stack-scaling-list lists all the scaling groups associated to a stack.

Syntax for OVFT client:

ovft stack-scaling-list *<stack_name>*

Syntax for OpenStack client:

openstack stack scaling list *<stack_name>*

Examples:

```
root@atlas:~# ovft stack-scaling-list scale_out_stack

+----------------+---------------+
| Property       | Value         |
+----------------+---------------+
| scaling groups | vm1_scaled    |
| servers        | vm1           |
+----------------+---------------+
```

Example 32 Stack Scaling List by Using an OVFT Client

```
root@atlas:~# openstack stack scaling list scale_out_stack

+----------------+-----------------+
| Field          | Value           |
+----------------+-----------------+
| scaling groups | vm1_scaled      |
| servers        | vm1             |
+----------------+-----------------+
```

Example 33 Stack Scaling List by Using an OpenStack Client

12 Deployment Wizard

Deployment wizard allows the user to provide customized resources while launching a stack. It also allows the user to provide extra properties for the resources existing in the application.

12.1 Deployment Wizard From CLI

Properties can be added through Deployment Wizard, to launch stack from cAPP.

The deployment wizard supports the following properties:

- User data
- Metadata
- File injection

  Note: As the maximum body size for HTTP requests is set to 112 KB in the default settings of the Nova API, larger environment files cannot be injected, even if the `injected_file_content_bytes quota` value is changed for the specific tenant. To increase the default value, set `osapi_max_request_body_size` in `/etc/nova/nova.conf` on all controllers and restart `nova-api`.

- Availability zone
- `extra_specs` for flavor, or use of existing flavor

This section describes how to add above mentioned properties into an existing cApp, to generate a new HOT template that can be used to launch stack with the added properties.

Syntax for OVFT client:

ovft capp-template-personalize -e *<input env file>* -o *<output file name>* *<capp_id>*

```
root@atlas:~# ovft capp-template-personalize -e personalize-env.yaml -o output.yaml output capp_id
```

Example 34 Deployment Wizard Command

```
root@atlas:~# cat personalize-env.yaml
resource_registry:
  resources:
    Controller_node1:
      properties:
        availability_zone: "nova"
        personality:
          /home/files/enivronmentfile.xml:
            get_file: /tmp/env.xml
        user_data:
          str_replace:
             template: |
               #!/bin/bash
               chmod +x $path_to_setup_file
               $path_to_setup_file --interface
               $path_to_setup_file --copy
               $path_to_setup_file --setup_remote
             params:
               $path_to_setup_file: /home/atlasadm/user_data
        metadata:
           app_ip: IP of the application
    type: OS::Nova::Server
    flavor_Controller_node1:
      properties:
        extra_specs:
          "quota:disk_read_bytes_sec": "10240000"
      type: Ericsson::Nova::Flavor
```

Example 35 Input environment File Format

The output (`output.yaml`) will be generated with the properties mentioned in Example 35 updated/added:

```
root@atlas:~# cat output.yaml
description: stack template generated from OVF file
heat_template_version: '2013-05-23'
parameters:
  param_1: {description: Segmentation ID of network, label: demo segmentation ID,
    type: string}
  param_2: {description: IP subnet, label: demo subnet, type: string}
  param_3: {description: Gateway, label: demo gateway, type: string}
resources:
  Controller_node1:
    properties:
      availability_zone: nova
      config_drive: 'True'
      flavor: {get_resource: flavor_Controller_node1}
      image: 22603a05-9efe-4d9f-a92c-53a8bd49ff06
      metadata: {app_ip: IP of the application}
      name: Controller node1
      networks:
      - network: {get_resource: demo}
      personality: {/home/files/enivronmentfile.xml: 'This is environment

          ', setup/cfg.xml: "<instance name=\"cirros\">\n  <image name=\"cirros_img\"\
>cirros_img</image>\n  <flavor name=\"m1.tiny\">m1.tiny</flavor>\n  <property\
          \ key='username' value='$username'/>\n</instance>\n", setup/resources.xml: "<instance\
          \ name=\"cirros\">\n  <image name=\"cirros_img\">cirros_img</image>\n  <flavor\
          \ name=\"m1.tiny\">m1.tiny</flavor>\n  <property key='username' value='$username'/>\n\
</instance>\n"}
user_data:
        str_replace:
          params: {$path_to_setup_file: /home/atlasadm/user_data}
          template: '#!/bin/bash

            chmod +x $path_to_setup_file

            $path_to_setup_file --interface

            $path_to_setup_file --copy

            $path_to_setup_file --setup_remote
```

```
          '
    type: OS::Nova::Server
  Payload_node1:
    properties:
      config_drive: 'True'
      flavor: {get_resource: flavor_Payload_node1}
      image: 3f9c0250-f741-435a-b318-0a8e8fa3e6b0
      metadata: {}
      name: Payload node1
      networks:
      - network: {get_resource: demo}
    type: OS::Nova::Server
  demo:
    properties:
      name: demo
      value_specs:
        provider:network_type: vlan
        provider:physical_network: default
        provider:segmentation_id: {get_param: param_1}
        router:external: true
    type: OS::Neutron::Net
  demo_subnet:
    properties:
      cidr: {get_param: param_2}
      enable_dhcp: true
      gateway_ip: {get_param: param_3}
      ip_version: '4'
      name: demo_subnet
      network_id: {get_resource: demo}
    type: OS::Neutron::Subnet
  flavor_Controller_node1:
properties:
      disk: 1
      extra_specs: {'quota:disk_read_bytes_sec': '10240000'}
      ram: 1024
      vcpus: 1
    type: Ericsson::Nova::Flavor
  flavor_Payload_node1:
    properties: {disk: 1, ram: 1024, vcpus: 1}
    type: Ericsson::Nova::Flavor
```

Example 36 Output Template Generated