CONTENTS

Preface ...................................................................................................................... 8

About this Guide ....................................................................................................... 8

Audience .................................................................................................................... 8

Structure of the Guide ............................................................................................... 8

Documentation References ....................................................................................... 9

1 Introduction ........................................................................................................... 10

2 General .................................................................................................................. 11

  2.1 Maestro CLI Admin Utility Purpose ................................................................. 11

  2.2 Connecting to Admin Utility ............................................................................. 11

  2.3 File Upload ........................................................................................................ 12

3 Maestro CLI Use for Project Management ............................................................ 13

  3.1 Refreshing Project Status .................................................................................. 13

  3.2 Migrating Instance to CSA ................................................................................ 13

  3.3 Setting Checkpoint Quota for Project ............................................................... 14

  3.4 Setting Volume Quota for Project .................................................................... 14

  3.5 Setting Instance Quota for Project ................................................................... 15

4 Using Admin Utility ................................................................................................ 16

  4.1 Basic Principles ................................................................................................ 16

  4.2 Maestro CLI Admin Utility Help ....................................................................... 16

  4.3 Command Execution ........................................................................................ 18

  4.4 Asynchronous Commands ............................................................................... 19

  4.5 Command Output ............................................................................................. 20

5 Command Groups .................................................................................................. 21

  5.1 General ............................................................................................................. 21

    5.1.1 SHOW ........................................................................................................ 21

    5.1.2 SUBSCRIPTION ....................................................................................... 21

    5.1.3 ACCOUNT ............................................................................................... 22

    5.1.4 SETTINGS ............................................................................................... 22
5.7.1 GOOGLE ........................................................................................................... 39
5.8 CSA, HP OO, OpenStack ..................................................................................... 40
5.8.1 CSA .................................................................................................................... 40
5.8.2 HPOO ................................................................................................................... 41
5.8.3 OPEN_STACK ....................................................................................................... 41
5.9 PaaS ......................................................................................................................... 43
5.9.1 PAAS ..................................................................................................................... 43
5.9.2 CHEF ..................................................................................................................... 43
5.9.3 DOCKER ............................................................................................................... 43
5.10 TEMP ...................................................................................................................... 44

6 Maestro CLI Admin Utility – Use Cases .................................................................... 45
6.1 AWS – Administration Cases ............................................................................... 45
6.1.1 AWS Zone Creation ........................................................................................... 45
  • Zone Creation ......................................................................................................... 45
  • Zone Virtual Profile Configuration ............................................................................ 46
  • Adding Machine Images to AWS Zone ...................................................................... 47
  • Setting Cost Center for AWS Zone ............................................................................ 47
6.1.2 Project Activation in AWS .................................................................................. 48
  • EC2 Instance Role Configuration ............................................................................. 49
  • SSO Configuration ................................................................................................... 49
  • SSO Roles Configuration ........................................................................................ 51
  • AWS Policy Management ......................................................................................... 51
  • IAM Role Group Configuration ................................................................................. 53
  • Security Groups Configuration .................................................................................. 54
  • Security Groups Backup .......................................................................................... 55
  • CloudTrail Service Activation .................................................................................. 56
6.1.3 Access to AWS .................................................................................................... 57
6.1.4 AWS Organizations ............................................................................................ 57
6.1.5 Reserved Instances .............................................................................................. 58
  • Displaying Reserved Instances ............................................................................... 58
• Modifying Reserved Instances ................................................................. 58
• Displaying Reserved Instance Offerings .................................................. 59
• Purchasing Reserved Instances .................................................................. 59

6.2 Microsoft Azure – Administration Cases .................................................. 61

6.2.1 Azure Zone Creation ............................................................................. 61
• Azure Enrolment Setup .............................................................................. 61
• Zone Creation .............................................................................................. 62
• Setting Cost Center for Azure Zone ............................................................. 62
• Adding Machine Images to Azure Zone ...................................................... 63

6.2.2 Activating a Project in Microsoft Azure .................................................. 63
• Project Activation ....................................................................................... 64
• Network Configuration ............................................................................... 64
• Configuration Check ................................................................................. 65

6.3 CSA – Administration cases ....................................................................... 66

6.3.1 CSA Zone Creation ............................................................................... 66
• CSA Zone Creation ..................................................................................... 66
• Orchestration Instance Assignment to CSA Zone ....................................... 67
• Setting Cost Center for CSA Zone ............................................................... 67
• Adding Shapes to CSA Zone .................................................................... 68

6.3.2 Reimporting Instances to CSA ............................................................... 69
• CSA Subscription Deletion ...................................................................... 69
• Instance Restoring to CSA ...................................................................... 70
• Subscription Synchronization .................................................................. 70

6.4 Google Cloud Platform – Administration Cases ........................................ 71

6.4.1 Google Account Configuration ............................................................... 71
6.4.2 Google Account Entity in Orchestrator Database .................................... 71

6.4.3 Adding Google Zones .......................................................................... 72
• Retrieving Google Zones ........................................................................ 73
• Adding Google Zones ............................................................................. 73
PREFACE

ABOUT THIS GUIDE

Maestro CLI Admin Utility Guide is the description of the Admin Utility console, the commands used by Cloud administrators and their syntax and purpose.

AUDIENCE

This guide is intended for the support and maintenance personnel performing configuration and setup tasks, maintenance works and assisting users with matters beyond the self-service scope.

STRUCTURE OF THE GUIDE

The guide consists of the following chapters:

- **Introduction** – the brief description of EPAM Orchestrator and its basic concept
- **General** – the description of Maestro CLI Admin Utility purpose and the instructions on connecting to the Maestro CLI Admin Utility for Windows and Linux operating systems
- **Maestro CLI Use for Project Management** – the description of admin commands existing in Maestro CLI
- **Using Admin Utility** – the description of the basic principles of Maestro CLI Admin Utility usage, the command string structure and the instructions on using the Maestro CLI Admin Utility help
- **Command Groups** – the list of command groups available in Maestro CLI Admin Utility together with the brief descriptions of commands within each group
- **Maestro CLI Admin Utility – Use Cases** – the description of several common cases of Maestro CLI Admin Utility usage with the command examples
- **Annex A** – Admin CLI Commands Usage in Different Virtualization Platforms – the reference table of commands used for project management in different virtualization platforms
- **Annex B** – Admin CLI Commands Requiring File Upload – the list of commands referring to the content of previously uploaded files and the description of the file content
- **Annex C** – Admin CLI Commands Sending Emails as the Result of Execution – the list of commands using email addresses as output for the data obtained as the result of the command execution
- **Annex D** – AWS-Related Collections in Database – the list of collections serving AWS platform with their descriptions
**DOCUMENTATION REFERENCES**

EPAM Orchestration is described in details in a number of documents, focused on different aspects of Orchestration usage, and on different types of users.

You can find these documents on our [Documentation](#) page.

The answers to the most frequently asked questions can be found on the [FAQ](#) page.

EPAM Cloud terms and conditions are described in our [EPAM Cloud Terms and Conditions](#). Please take a look at this document in order to avoid misunderstandings and conflicts that may arise during the service usage.

The terminology of EPAM Cloud and the related products can be found on the [Glossary](#) page.

Please email your comments and feedback to EPAM Cloud Consulting at SpecialEPM-CSUPConsulting@epam.com to help us provide you with documentation that is as clear, correct and readable as possible.
1 INTRODUCTION

Cloud computing is the computing model in which pooled resources and services are generally available via the Internet and accessible via self-service portals by dynamic assignment to multiple tenants. Cloud computing systems are characterized by high elasticity, that is, the ability to scale in or out according to the customers’ demand. Resource usage is charged on the pay-as-you-go basis, for which purpose cloud computing systems include monitoring, controlling and reporting functionality.

Cloud services made generally available form a public cloud. The same infrastructure deployed for a single enterprise only comprises a private cloud. Private clouds operate totally within their own secure environments. Cloud infrastructure having the features of both public and private cloud joined by a proprietary or standardized technology is described as hybrid cloud.

EPAM Cloud Orchestrator can be characterized as a hybrid cloud, because, in addition to the private cloud services, it supports integration with external cloud platforms.

According to Forrester’s Vendor Landscape: Private Cloud Software Solutions report, private cloud solutions fall into three major categories defined by their implementation method and the administration tools used: Cloud Platforms, combining physical and virtual resources into IaaS cloud environments, Standalone Cloud Management Tools, managing virtual resources on the basis of public and private cloud platforms, and Private Cloud Suites, combining the features of the two categories mentioned above.

EPAM Cloud Orchestrator belongs to Standard Cloud Management Tools which can be based on one of virtualization platforms (AWS, Microsoft Azure, HP OO, OpenStack or CSA) and performs cloud management, monitoring, account billing, access management and support.

According to the Private Cloud Software Reference Architecture described in the above-mentioned Forrester’s report, the Maestro CLI Admin Utility represents the Admin Portal implemented as a command-line interface. Together with other cloud management components, the Admin Portal forms the comprehensive Hybrid Cloud Management Solution.
2 GENERAL

For the purposes of project management in EPAM Cloud, a special tool, Maestro CLI Admin Utility, is used. In addition, certain project management operations are performed using the commands of Maestro CLI. This document describes options available both in the dedicated Admin Utility and in the Maestro CLI.

2.1 MAESTRO CLI ADMIN UTILITY PURPOSE

The Admin Utility is a tool allowing to monitor and maintain the Cloud infrastructure and projects hosted within, provide support and consulting on the Cloud projects operation and issues which may occur from time to time.

2.2 CONNECTING TO ADMIN UTILITY

Connection to the Admin Utility is performed via SSH. To set up your connection, generate a keypair with the `or2addkey` Maestro CLI Command or any other key generation tool. Once the keypair is generated, add your domain login (FirstName_LastName@epam.com) at the end of the public part of the keypair and send it to Level 3 Support Team, to register a personal account for you.

Connect to Admin Utility:

- Linux:

  ```
  ssh -i /path/to/your/private.key -p 2001 user@orchestration.epam.com
  ```

- Windows:

  1. Convert your Private Key:

     - Start PuTTYgen
     - Click Load. By default, PuTTYgen displays only files with the extension .ppk. To locate your .pem file, select the option to display files of all types.

        ![PuTTYgen screenshot]

        *Figure 1 – Locating Private Key*

     - Select your .pem file from the keypair which you specified when launching your instance then click Open. Click OK to dismiss the confirmation dialog box.
     - Click Save private key to save the key in the format acceptable by PuTTY. PuTTYgen displays a warning about saving the key without a passphrase. Click Yes.
     - Specify the same name for the key that you used for the keypair (for example, my-key-pair). PuTTY automatically adds the .ppk file extension.
2. Start PuTTY (use user@orchestration.epam.com and 2001 port). Add your private key in Connection>SSH>Auth.

### 2.3 FILE UPLOAD

Some commands use content of a file in their performance. In such cases, files have to be uploaded before the command execution. The files are uploaded using SCP, Secure Copy Protocol. SCP uses SSH in file transfer. To upload a file, use the following command:

```
scp -P port -i <path-to-keypair-pem-file> local_file_path host:filename
```

Files are to be uploaded outside the Maestro Admin Utility, that is, before logging in. When specifying the local path to the file, make sure you are using the **relative pathname** and not the absolute pathname, otherwise the file upload will fail.
3 MAESTRO CLI USE FOR PROJECT MANAGEMENT

Maestro Command Line Interface (CLI) is a tool used to send basic Orchestrator commands via the command line. Maestro CLI is widely used by the EPAM Cloud user community for virtual machine management.

For information on setting up Maestro CLI and the required components, logging in and basic usage guidelines, please refer to the Quick Start Guide.

Access to CLI commands is based on the system of permissions. The project management (administration) commands are available only to users with the ALL_SYSTEM_OPERATIONS permission and are hidden from all other users. Currently, the following Admin commands are available:

3.1 REFRESHING PROJECT STATUS

or2-refresh-projects (or2repl)

The command refreshes the status of the specified project or of all projects in the specified region.

<table>
<thead>
<tr>
<th>Parameter name</th>
<th>Description</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>--full</td>
<td>Show full command output instead of default basic one</td>
<td>No</td>
</tr>
<tr>
<td>-P, --plain-output</td>
<td>Use plain output instead of default table output</td>
<td>No</td>
</tr>
<tr>
<td>-json</td>
<td>Show command output in json format</td>
<td>No</td>
</tr>
<tr>
<td>-p, --project</td>
<td>Project abbreviation in UPSA</td>
<td>No</td>
</tr>
<tr>
<td>-r, --region</td>
<td>Virtualization region</td>
<td>Yes</td>
</tr>
<tr>
<td>--help</td>
<td>Display command help</td>
<td>No</td>
</tr>
</tbody>
</table>

Command example:

```
or2repl -p project -r region
```

3.2 MIGRATING INSTANCE TO CSA

or2-migrate-csa-instance (or2migcsains)

The command registers the specified instance in CSA.

<table>
<thead>
<tr>
<th>Parameter name</th>
<th>Description</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>--full</td>
<td>Show full command output instead of default basic one</td>
<td>No</td>
</tr>
<tr>
<td>-P, --plain-output</td>
<td>Use plain output instead of default table output</td>
<td>No</td>
</tr>
<tr>
<td>-json</td>
<td>Show command output in json format</td>
<td>No</td>
</tr>
<tr>
<td>-p, --project</td>
<td>Project abbreviation in UPSA</td>
<td>Yes</td>
</tr>
<tr>
<td>-r, --region</td>
<td>Virtualization region</td>
<td>Yes</td>
</tr>
<tr>
<td>-m, --image</td>
<td>Machine image</td>
<td>Yes</td>
</tr>
<tr>
<td>-I, --instance-name</td>
<td>Instance name</td>
<td>Yes</td>
</tr>
<tr>
<td>-g, --migration-date</td>
<td>Migration date in the yyyy-mm-dd'T'HH format</td>
<td>Yes</td>
</tr>
</tbody>
</table>
-s, --shape Instance type | Yes
--help Display command help | No

Command example:

```
or2refp -p project -r region -s shape -i instance_name -m image -g migration_date
```

### 3.3 SETTING CHECKPOINT QUOTA FOR PROJECT

**or2-set-project-checkpoint-quota (or2setpchq)**

The command defines the maximum number of checkpoints which can be created for the specified project and region.

<table>
<thead>
<tr>
<th>CLI Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parameter name</td>
</tr>
<tr>
<td>--full</td>
</tr>
<tr>
<td>-P, --plain-output</td>
</tr>
<tr>
<td>--json</td>
</tr>
<tr>
<td>-p, --project</td>
</tr>
<tr>
<td>-r, --region</td>
</tr>
<tr>
<td>-m, --maxCount</td>
</tr>
<tr>
<td>--help</td>
</tr>
</tbody>
</table>

Command example:

```
or2setpchq -p project -r region -m checkpoint_count
```

### 3.4 SETTING VOLUME QUOTA FOR PROJECT

**or2-set-project-volume-quota (or2setpvq)**

The command defines the maximum number and size of additional volumes created in the specified project and region within the specified time interval.

<table>
<thead>
<tr>
<th>CLI Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parameter name</td>
</tr>
<tr>
<td>--full</td>
</tr>
<tr>
<td>-P, --plain-output</td>
</tr>
<tr>
<td>--json</td>
</tr>
<tr>
<td>-p, --project</td>
</tr>
<tr>
<td>-r, --region</td>
</tr>
<tr>
<td>-c, --count</td>
</tr>
<tr>
<td>-s, --maxSize</td>
</tr>
<tr>
<td>-t, --time</td>
</tr>
<tr>
<td>--help</td>
</tr>
</tbody>
</table>
Command example:
```
or2setpvq -p project -r region -s max_size -c count -t time_interval
```

### 3.5 SETTING INSTANCE QUOTA FOR PROJECT

**or2-set-project-instance-quota (or2setpiq)**

The command defines the maximum number of instances created in the specified project and region within the specified time interval.

<table>
<thead>
<tr>
<th>CLI Parameters</th>
<th>Description</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>--full</td>
<td>Show full command output instead of default basic one</td>
<td>No</td>
</tr>
<tr>
<td>-P, --plain-output</td>
<td>Use plain output instead of default table output</td>
<td>No</td>
</tr>
<tr>
<td>--json</td>
<td>Show command output in json format</td>
<td>No</td>
</tr>
<tr>
<td>-p, --project</td>
<td>Project abbreviation in UPSA</td>
<td>Yes</td>
</tr>
<tr>
<td>-r, --region</td>
<td>Virtualization region</td>
<td>Yes</td>
</tr>
<tr>
<td>-c, --count</td>
<td>The number of instances which can be created within the specified time interval</td>
<td>Yes</td>
</tr>
<tr>
<td>-l, --time</td>
<td>Instance creation interval in hours</td>
<td>Yes</td>
</tr>
<tr>
<td>--help</td>
<td>Display command help</td>
<td>No</td>
</tr>
</tbody>
</table>

Command example:
```
or2setpiq -p project -r region -c count -t time_interval
```
4 USING ADMIN UTILITY

4.1 BASIC PRINCIPLES

Maestro CLI Admin Utility operates by executing commands sent via the command line. Each command consists of the group name, the command name and the arguments.

The **group name** defines the general area of the command application (e.g. ‘aws’ – commands related to AWS, ‘project’ – commands related to projects, etc.).

The **command name** is the actual command string defining the action to be performed (e.g. ‘delete_zone’, ‘grant_access’, etc.)

The **arguments** define the specific object of the command and/or the values to be set for it (e.g. `-p` – project abbreviation in UPSA, `-s` – shape name, etc.).

For example:

```
permission add_user -e email
```

In this example ‘permission’ is the group name of all permission-related commands, ‘add_user’ is the command name indicating that the command creates a new user and ‘-e email’ is the argument containing the email of the user to be created.

Please note that in case a parameter is specified incorrectly, the command will not return an error. All parameters specified before the incorrect one, will be applied. The incorrect parameter and others following it, will be skipped. In case the applied parameters are enough for the command execution, the command will be run.

4.2 MAESTRO CLI ADMIN UTILITY HELP

The correct format and the required arguments for each command can be found in the ‘Help’ topics. To get the complete list of all command groups available in the Maestro CLI Admin Utility, type ‘help’ in the command line:

```
help
```

![Figure 2 – Command groups](image-url)
This command returns the alphabetical list of all command groups with their brief description.

To see the commands included in each group, type the group name:

```
> permission
usage: permission COMMAND [ARGS]
The most commonly used permission commands are:
assign Assigns a simple user to a project
add_pmc_mapping Adds pmc role permission group mapping
get_perm_groups Describe permission groups
get_user_mapping Describe permission group mappings
del_pmc_mapping Removes pmc role permission group mapping
del_user_mapping Removes user permission group mapping
add_user_mapping Adds user permission group mapping
add_group Adds permission group
add_user Creates new Simple User
del_group Removes permission group
unassign Unassigns a simple user from a project
```

*Figure 3 – List of commands in a group*

The response will contain the list of all commands in their correct format and the brief explanation of their purpose and action. The ‘usage’ line shows the valid command syntax.

To get help for a particular command, type the complete command with the -h or --help parameter:

```
> permission add_group --help
usage: permission add_group [-g | --group] [-o | --operation] [-p | --project]
[-g | --group] * Permission group name.
[-o | --operation] * Operation name. For several operation repeat the parameter
[-p | --project] Project pmc code
```

*Figure 4 – Command help*

The response will contain the list of all possible arguments which can be used in the command. The mandatory arguments are marked with ‘*’.

Some commands require one of the optional parameters to be used in all cases. In this case, the command will return an error message if no optional parameter is specified. The error message will contain the prompt to use one of the optional parameters.

The ‘usage’ line shows the complete syntax of the command including the arguments. Some of the arguments have a short and a full form which have the same effect.

If an invalid command is sent, the response may indicate the missing or invalid parameter:

```
> show project
show project: Missing option [p, project]
```

*Figure 5 – Error message indicating missing parameter*

Boolean parameters with only ‘true’ or ‘false’ options are set to ‘false’ by default. To set them to ‘true’ only the argument without any value should be specified, otherwise the command will be rejected with the ‘command not found’ error message. For example, the `orch assign -z region -o OrchestratorID -a` command will be rejected if any value is sent for the -a parameter.
4.3 COMMAND EXECUTION

Some commands in Maestro CLI Admin Utility require the user’s reconfirmation of their intent to execute the command. When the user types the command string and presses ‘Enter’, the system responds with the following message: ‘Are you sure you want to perform the operation…? Type “y” or “n”’. The user has to confirm the operation by typing “y” or abort it by typing “n”.

Such reconfirmation is required, for example, for all ‘activate_project’ commands, the ‘billing lock’, ‘billing unlock’ commands, etc.

Certain other commands require the particular instance to be stopped before the command can be executed. When the command string is entered, the following message is displayed: ‘The orchestrator instance should be stopped for performing this operation. Do you want to continue?’ If the user confirms the operation, the system checks whether the instance has been stopped and proceeds with the command execution. If the instance has not been stopped, the command is rejected with an error message.

Instance stopping is required, for example, for all ‘add_zone’ commands, the ‘zone delete’ command, the commands related to the RabbitMQ server configuration.

If you run a command immediately after stopping the instance, the system may still return the message prompting to stop it, as the instance status might not be updated yet. In this case allow up to 10 minutes after the instance stopping to run the command again.

However, to accelerate the process, the reconfirmations can be disabled by switching the system to the so-called ‘quiet’ mode. In the quiet mode the system does not require command reconfirmation before execution but executes it immediately. The ‘quiet’ mode is controlled by the ‘quiet on’ and ‘quiet off’ commands:

- quiet on Switches the ‘quiet’ mode on
- quiet off Switches the ‘quiet’ mode off

By default, the ‘quiet’ mode is disabled.

Certain commands requiring password for their execution cannot be run in the ‘quiet’ mode. The password is not specified as one of the mandatory parameters but is to be entered later, at the system prompt. If a command is sent in the ‘quiet’ mode, the following error message is displayed: “This command can't be running in quiet mode!”. The following commands cannot be used with the ‘quiet’ mode enabled:

- azure add_subscript
- azure_custom add_subscript
- aws add_user
- csa add_zone
- hpoo configvs
- open_stack add_zone
- open_stack notific_config
- permission add_user
- rabbit shovel
- rabbit create_upstream
- security add_nessus_conf
- settings set_upsa_config
- google add_temp_access_user
- zone orch_settings
4.4 ASYNCHRONOUS COMMANDS

When a command is executed in Maestro CLI Admin Utility, the CLI is unavailable until the command execution is complete. However, some commands requiring long time for execution (up to several hours) are performed in the background while the CLI can be used for other purposes. Such commands are called asynchronous commands.

When an asynchronous command is executed, its status can be retrieved by the ‘status get’ command. It shows the command progress or completion together with the data generated during the command execution. The command syntax is as follows:

```
status get --g command_group --n command_name
```

Each asynchronous command can be run only once simultaneously on the same Orchestrator node. If an asynchronous command has been sent by one of the users, other users cannot send the same command until the first command instance is completed.

Maestro CLI Admin Utility supports the following asynchronous commands:

- **arm check_config** Checks ARM configuration
- **arm config_project** Configures the project for using the ARM API
- **arm set_def_groups** Applies configuration of Azure security groups
- **aws check_config** Checks AWS configuration
- **aws config_sso** Configures AWS SSO
- **aws create_account** Creates an AWS account via the Organization API
- **aws export_billing_data** Sets up billing data export
- **aws_security set_def_groups** Applies configuration of AWS security groups
- **billing close_month** Closes the billing month.
- **billing health_check** Checks billing consistency.
- **billing send_units_reports** Sends business unit reports
- **integrity check** Checks data integrity.
- **csa get_capacity** Shows open, close, current values and blocked actions for all CSA regions
- **project clean_up** Marks instances and volumes as deleted, closes timelines and removes AWS instances usage profiles
- **radar aggregate** Aggregates Radar data for the specified month
- **timeline check_resource** Validates all timelines for a resource
- **zone delete** Marks the specified zone as inactive or deletes it together with all its references.
4.5 COMMAND OUTPUT

Execution of some commands results in generation of certain data. Such data is delivered to the user according to the command settings. The command output can be either the SSH console or the email of the user which is currently logged in.

The command output is defined by the `--target` parameter value which has to be set to `ssh_console` or `email`. The following commands support the target selection:

- chef get_nodes
- csa check_offerings
- integrity check
- pricing_policy get
- show all_zones
- show all_projects
- subscription show_default
- aws_security check_mfa
- aws_ri describe

At the same time, with certain commands the user can specify whether the generated data is to be delivered in the plain text format or in the HTML format. To obtain the command output in the HTML format, the `--html` parameter has to be sent in the command. The following commands support HTML output:

- chef get_nodes
- integrity check
- instance refresh_missing
- volumes refresh_missing
- show all_zones
- show all_projects
- subscription show_default

If no target selection is offered, the HTML file is delivered to the SSH console.

The `aws_security get_backup` and `billing health_check` commands always deliver data to the user's email in the HTML format. No output selection is supported.
5 COMMAND GROUPS

The commands implemented in Maestro CLI Admin Utility cover various scenarios and issues occurring in the everyday work of the Cloud Support Team. However, they can be classified under several categories according to their application and purpose.

5.1 GENERAL

The ‘General’ category of commands includes the commands related to the basic Orchestrator settings and functions.

5.1.1 SHOW

The ‘show’ group includes the commands used to display the specified items or lists of items.

We recommend starting your introduction to Maestro CLI Admin Utility with this group of commands, as they can give you the basic idea of how the Admin Utility works, how the command strings are built and how the responses are organized. At the same time, these commands return a lot of useful data about the objects and resources managed by EPAM Orchestrator, their parameters and value formats.

The following commands are available:

- **show project**
  Displays the configuration of the specified project
- **show settings**
  Displays settings of the whole Orchestration
- **show zone**
  Displays the configuration of the specified virtualization zone
- **show all_projects**
  Displays brief info of all projects. The list of requested projects may be filtered
- **show project_dls**
  Displays the list of ORG Cloud user emails included in the project distribution list
- **show all_regions**
  Displays brief info of all regions
- **show all_zones**
  Displays brief info of all zones. The list of requested zones may be filtered

For the arguments used with the commands of the ‘show’ group type `show [command_name] -h` in the command line.

5.1.2 SUBSCRIPTION

The ‘subscription’ group includes the commands related to configuration of notification and report subscriptions. The following commands are available:

- **subscription show_templates**
  Displays available subscription templates
- **subscription update_default**
  Updates the default subscriptions
- **subscription update_template**
  Updates the subscription template
• subscription show_default Displays the default subscriptions
For the arguments used with the commands of the 'subscription' group type subscription [command_name] -h in the command line.

5.1.3 ACCOUNT
The 'account' group includes the commands related to EPAM Orchestrator accounts. The following commands are available:

• account subscribe Creates subscription to the EO account for the specified email(s)
• account unsubscribe Stops subscription to the EO account for the specified email(s)
For the arguments used with the commands of the 'account' group type account [command_name] -h in the command line.

5.1.4 SETTINGS
The 'settings' group includes the commands related to the general system settings. The following commands are available:

• settings describe Describes orchestration settings
• settings get_test_emails Retrieves emails authorized for testing
• settings edit_test_emails Adds or removes email addresses of users having access to EPAM Orchestrator in the testing mode
• settings healthcheck_to Sets the health check timeout duration
• settings upsa Enables or disables UPSA integration
• settings set_upsa_config Sets UPSA client configuration
• settings epam_metrics Enables or disables EPAM metrics integration
• settings chef_aggregation Turns Chef statistics aggregation on and off
• settings manage_blacklist Adds user emails to or removes them from the blacklist
• settings add_key Adds SSH public key to be automatically uploaded to all Linux-based VMs in the Cloud, giving a support team the ability to get access to VMs for maintenance purposes
• settings get_key Retrieves the support engineers' SSH public key
• settings get_blacklist Retrieves the blacklisted user emails
• settings report_cache Enables or disables the Report Cache
For the arguments used with the commands of the 'settings' group type settings [command_name] -h in the command line.
5.1.5 ORCH

The ‘orch’ group includes the commands related to Orchestrator. The following commands are available:

- **orch assign**: Assigns or unassigns the Orchestrator instance from the specified zone
- **orch jenkins_service**: Switches the Jenkins service on or off for the specified Orchestrator instance
- **orch config_healthch**: Updates healthCheckSettings for the specified Orchestrator instance
- **orch integr_service**: Switches the Integration Service on or off for the specified Orchestrator instance
- **orch config_zabbix**: Configures the Zabbix service for the specified Orchestrator instance
- **orch dconfig**: Retrieves configuration settings from the local configuration and database
- **orch assign_cur**: Assigns the current Orchestrator instance or unassigns it from the specified zone
- **orch dis_recovery**: Sets/removes the disaster recovery flag
- **orch set_profile**: Sets a profile for the node
- **orch switch_mode**: Switches the Orchestrator mode between RUNNING, TESTING and MAINTENANCE

Use this command with care, as the Orchestrator mode settings affect entire Orchestrator performance

For the arguments used with the commands of the ‘orch’ group type **orch [command_name] -h** in the command line.

5.1.6 INIT

The ‘init’ group includes the commands related to Orchestrator initialization. The following commands are available:

- **init version**: Initializes the Version collection
- **init region**: Initializes the Regions collection
- **init settings**: Initializes the OrchestrationSettings collection
- **init config**: Initializes the OrchestratorConfig collection

The commands of the ‘init’ group type are used without arguments, as their action consists of initializing the specified collection.
5.1.7 INTEGRITY

The ‘integrity’ group includes the ‘integrity check’ command checking the data integrity. The command also includes the integrity check settings.

For the arguments used with the ‘integrity check’ command type **integrity check -h** in the command line.

5.1.8 CLI

The ‘cli’ group includes ‘cli notify’ command notifying the user about CLI update. The ‘cli notify’ command uses no arguments.
5.2 SECURITY

The ‘Security’ category includes the commands related to user account management, permission assignment and mapping, as well as the settings and configuration of Nessus security scanner.

5.2.1 PERMISSION

The ‘permission’ group includes the commands related to user and permission management. The following commands are available:

- `permission assign` Assigns a simple user to the specified project
- `permission unassign` Unassigns a simple user from the specified project
- `permission del_pmc_mapping` Removes mapping between a PMC role and a permission group
- `permission add_pmc_mapping` Adds mapping between a project role and a permission group
- `permission get_perm_groups` Retrieves the available permission groups
- `permission get_user_mapping` Retrieves the existing permission group mappings
- `permission add_user_mapping` Adds the user to the specified permission group
- `permission del_user_mapping` Removes permissions for the specified user
- `permission set_user_requestor` Sets a requestor for a simple user
- `permission add_user` Creates a new simple user
- `permission add_group` Adds a new permission group specifying the allowed operations
- `permission del_group` Removes the specified permission group

For the arguments used with the commands of the ‘permission’ group type permission [command_name] -h in the command line.

5.2.2 SECURITY

The ‘security’ group includes the commands related to the Nessus Server configuration and management. The following commands are available:

- `security sched_switcher` Enables or disables Nessus server scheduling
- `security upd_nessus_conf` Updates the Nessus Server configuration
- `security update_def_group` Adds ingress rule to a specific security group
- `security launch_switcher` Enables or disables immediate launch of the Nessus Server
- `security add_nessus_conf` Adds a new Nessus Server configuration
- `security def_serv_switch` Enables or disables default server mode for the Nessus Server
• **security dmz_switcher** Enables or disables DMZ scanning mode for the Nessus Server
• **security set_templates** Sets the policy templates for the Nessus Server
• **security vlan_switcher** Enables or disables VLAN scanning mode for the Nessus Server

For the arguments used with the commands of the ‘security’ group type `security [command_name] -h` in the command line.

### 5.2.3 USER

The ‘user’ group includes the commands related to user management. The following commands are available:

• **user refresh** Refreshes the status of the specified user in EPAM Cloud Orchestrator and activates personal projects if they are not activated
• **user describe** Retrieves the data of the specified user
• **user prolong_access_token** Prolongs user access which normally expires in a year after activation

For the arguments used with the commands of the ‘user’ group type `user [command_name] -h` in the command line.

### 5.2.4 NESSUS

The ‘nessus’ group includes the commands related to Nessus security scanning. The following commands are available:

• **nessus status** Shows the information about the current Nessus status (scans in progress, pending, etc.)
• **nessus desc_config** Describes Nessus configuration

For the arguments used with the commands of the ‘nessus’ group type `nessus [command_name] -h` in the command line.
5.3  INFRASTRUCTURE

The ‘Infrastructure’ category includes the commands related to different resources existing in EPAM Cloud and to the overall infrastructure of the system.

5.3.1 ZONE

The ‘zone’ group includes the commands related to zone management. The following commands are available:

- **zone delete**
  Marks the specified zone as inactive or deletes it together with all its references

- **zone conf_service**
  Enables or disables the specified service for the specified virtualization zone

- **zone set_default_vlan**
  Sets the default VLAN for personal projects in the specified zone (for HP OO zones only)

- **zone configure_nessus**
  Configures Nessus scanning for the specified zone

- **zone describe_locations**
  Retrieves the locations of the specified zone

- **zone get_default_vlan**
  Retrieves the data of default VLAN for personal projects

- **zone get_nonadmin_act**
  Retrieves the actions not available for users other than administrators

- **zone non_admins_act**
  Retrieves actions blocked for non-administrator users

- **zone manage_actions**
  Configures the list of actions not available for users other than administrators

- **zone set_virt_profile**
  Configures the shape mappings for the specified zone

- **zone orch-settings**
  Sets zone orchestration settings

- **zone set_location**
  Sets zone location

- **zone switch_mode**
  Switches the integrationMode on or off for the specified zone

- **zone getadmins**
  Retrieves the list of administrators for the specified zone

- **zone deprecate**
  Deprecates the specified zone. When a zone is deprecated, its functionality is limited. A deprecated zone is usually removed in the near future and migrates to a different region

- **zone add_location**
  Adds a physical location to the specified zone

- **zone manage_admins**
  Adds or removes users to the list of administrators of the specified zone

- **zone get_actions**
  Retrieves the list of actions blocked for the specified zone

- **zone change_status**
  Changes zone status
For the arguments used with the commands of the ‘zone’ group type `zone [command_name] -h` in the command line.
5.3.2 PROJECT

The ‘project’ group includes the commands related to project management. The following commands are available:

- **project delete** Deletes the specified project
- **project unlink** Removes the link between two projects
- **project link** Creates a link between two projects and disables quotas for the linked project
- **project set_default_vlan** Sets a default VLAN for the specified PMC project and zone (not to be used for CSA-type zones)
- **project set_custom_chef** Sets or unsets the project custom chef server
- **project set_ac_flag** Sets the auto-configuration flag for the project
- **project set_default_owner** Sets the default owner for the project (the PM, PC or primary contact from UPSA)
- **project set_ip_wl** Sets the white list of IP addresses for the project
- **project get_ip_wl** Retrieves the white list of IP addresses for the project
- **project del_ip_wl** Deletes the white list of IP addresses for the project
- **project activ_dl** Activates project DL in AWS
- **project del_dl** Removes project DL in AWS
- **project update_threshold** Updates the threshold value for the project in the specified zone
- **project set_type** Sets project type
- **project set_shapes** Sets the shapes allowed for the project
- **project set_quota** Sets the monthly project quotas and the quota notification plan
- **project set_personal_quota** Sets the quota level for the specified position
- **project check_billing_types** Checks the consistency of project billing types
- **project clean_up** Marks instances and volumes as deleted, closes timelines and removes AWS instance usage profiles
- **project deactivate** Deactivates the specified project

For the arguments used with the commands of the ‘project’ group type `project [command_name] -h` in the command line.

5.3.3 INSTANCE

The ‘instance’ group includes the commands related to instances. The following commands are available:
• **instance refresh_missing**  Finds and refreshes missing instances in the project
• **instance maintenance**  Sets the maintenance mode for the instance

For the arguments used with the commands of the ‘instance’ group type instance [command_name] -h in the command line.

### 5.3.4 VOLUMES

The ‘volumes’ group includes the commands related to storage volume management. The following commands are available:

• **volumes refresh_missing**  Finds and refreshes missing storage volumes for the specified project
• **volumes set_sys_disks**  Marks one of the disks as the ‘system disk’ in case of system disk missing

For the arguments used with the commands of the ‘volumes’ group type volumes [command_name] -h in the command line.
5.3.5 RESOURCES

The ‘resources’ group consists of the ‘resources change_ownership’ command which sets a different user as the project owner. This command is used when the project owner is dismissed or leaves the project while their resources remain. In this case the remaining resources can be moved to another user who should be assigned to the same project.

For the arguments used with the commands of the ‘resources’ group type resources [command_name] -h in the command line.

5.3.6 RABBIT

The ‘rabbit’ group includes the commands related to the RabbitMQ server configuration. The following commands are available:

- **rabbit clean** Removes redundant queues and exchanges from the RabbitMQ server
- **rabbit check** Checks the configuration of the local RabbitMQ server
- **rabbit config** Configures the local RabbitMQ server
- **rabbit show_upstreams** Describes existing federation upstream parameters
- **rabbit create_upstream** Create a new federation upstream parameter
- **rabbit federate** Federates all exchanges with other broker(s) defined in the upstream parameters
- **rabbit shovel** Creates the shovel configuration moving messages from the reply queue

For the arguments used with the commands of the ‘rabbit’ group type rabbit [command_name] -h in the command line.

5.3.7 RADAR

The ‘radar’ group includes the commands related to zones monitoring. The following commands are available:

- **radar aggregate** Aggregates the Radar data for the specified month
- **radar add_credits** Adds credit for the specified cloud usage to radar. Can be applied only to previous months

For the arguments used with the commands of the ‘radar’ group type radar [command_name] -h in the command line.
5.4 BILLING

The ‘Billing’ category includes the commands related to billing configuration and pricing policy in EPAM Cloud.

5.4.1 BILLING

The ‘billing’ group includes the commands related to billing in EPAM Cloud. The following commands are available:

- **billing lock**
  Locks billing for the period of non-billable works

- **billing reset**
  Resets billing for the specified period in the specified Cloud

- **billing unlock**
  Unlocks billing

- **billing set_adjustment**
  Adjusts billing for the specified project during the specified period

- **billing cbm_for_account**
  Closes the billing month for the specified account and generates a non-sendable report

- **billing describe_month**
  Returns billing data for the specified month

- **billing describe_business_units**
  Shows all business units

- **billing send_units_reports**
  Sends business unit reports

- **billing update_reports_config**
  Updates the configuration of business unit reports

- **billing set_cost_center**
  Assigns a cost center to the specified zone

- **billing add_services**
  Adds supported service(s)

- **billing get_services**
  Returns the supported service(s) data

- **billing close_month**
  Closes billing for the specified month

- **billing aws_china**
  Manages the billing report received from AWS China region

- **billing health_check**
  Checks billing consistency

For the arguments used with the commands of the ‘billing’ group type **billing [command_name] -h** in the command line.

5.4.2 PRICING_POLICY

The ‘pricing_policy’ group includes the commands related to the pricing policy. The following commands are available:

- **pricing_policy get**
  Retrieves the pricing policy details to the specified output (SSH console, file, or email)

- **pricing_policy update**
  Updates the pricing policy from the previously uploaded file
• **pricing_policy check** validates and shows the pricing policy changes

• **pricing_policy revert** Removes the open policy and the `useTo` field from the previous policy

For the arguments used with the commands of the ‘**pricing_policy**’ group type `pricing_policy [command_name] -h` in the command line.

### 5.4.3 TIMELINE

The ‘**timeline**’ group includes the commands related to instance billing timelines. The following commands are available:

• **timeline close** Closes all timelines for the specified project

• **timeline check_resource** Validates all timelines for the resource

For the arguments used with the commands of the ‘**timeline**’ group type `timeline [command_name] -h` in the command line.
5.5  **AWS**

The ‘AWS’ category includes the commands related to resource configuration and management on the AWS platform, as well as to the management of security groups in AWS.

5.5.1  **AWS**

The ‘aws’ group includes the commands related to the AWS platform. The following commands are available:

- `aws config_project` Configures the project with the specified UPSA abbreviation in AWS
- `aws get_cloudtrail` Displays the AWS CloudTrail configuration for the specified project
- `aws activ_cloudtrail` Activates and configures the AWS CloudTrail Service
- `aws deactiv_cloudtrail` Deactivates the AWS CloudTrail Service
- `aws activate_project` Activates and configures a project in AWS
- `aws check_account` Checks the AWS account associated with the specified project
- `aws create_account` Creates an AWS account via the Organization API
- `aws add_account` Creates a new AWS account
- `aws get_accounts` Retrieves the existing AWS accounts. **Note:** AWS accounts are retrieved per project which is a mandatory parameter for the command. However, when the project is set to ‘none’ (`-p none`) the command returns all empty accounts.
- `aws move_account` Moves a linked account to another paying account
- `aws del_account` Deletes a reachable account
- `aws add_zone` Creates and configures a new zone in the specified AWS region
- `aws add_az` Adds an availability zone to the specified AWS region
- `aws remove_az` Removes the availability zone from the specified AWS region
- `aws describe_az` Describes the configured availability zone
- `aws add_user` Adds an IAM user to the specified zone
- `aws delete_user` Deletes the specified IAM user
- `aws rename_user` Renames the IAM user
- `aws get_iam_entities` Retrieves the IAM roles
- `aws attach_policy` Attaches an IAM policy to the specified IAM entity in the database
- `aws detach_policy` Detaches the IAM policy from the specified IAM entity in the database
EPAM Cloud Orchestrator. Maestro CLI Admin Utility

- **aws get_policies** Describes IAM policies
- **aws save_policy** Saves the IAM policy to the database
- **aws up_group_policy** Uploads a group policy from the database to the specified group for the specified AWS accounts
- **aws up_man_policy** Uploads a managed policy to the specified AWS accounts
- **aws config_group** Configures the AWS permission group
- **aws add_group** Adds an IAM group to the specified project
- **aws describe_groups** Describes IAM groups for the specified project
- **aws add_image** Creates a new AWS image
- **aws del_image** Removes an AWS image
- **aws update_amis** Checks the current Windows images and updates the Windows image IDs
- **aws export_billing_data** Sets up billing data export
- **aws sso_manage_access** Manages access to AWS SSO, restricts access to particular roles
- **aws config_sso** Configures SSO for the specified project
- **aws sso_add_custom** Adds a custom SSO role for the specified user
- **aws sso_del_custom** Deletes the custom SSO role for the specified user
- **aws sso_get_custom** Retrieves custom SSO roles
- **aws check_config** Checks the AWS configuration
- **aws assoc_inst_prof** Associate instance profile with the specified project
- **aws set_def_role** Creates the default instance role for in the AWS account

For the arguments used with the commands of the ‘aws’ group type *aws [command_name] -h* in the command line.
5.5.2 AWS_SECURITY

The ‘aws_security’ group includes the commands related to the security in AWS. The following commands are available:

- **aws_security describe_backups**: Retrieves the backup configuration of the security groups for the specified project
- **aws_security set_def_groups**: Assigns default security groups to the specified project. If no groups are set, the project is automatically assigned 4 default AWS security groups
- **aws_security restore_groups**: Reverts to the backup configuration of the security groups. Please note that any groups deleted after the backup will be restored, while any groups added after the backup will be removed
- **aws_security save_groups**: Saves the current configuration of security groups
- **aws_security get_backup**: Sends the backup configuration of the security groups to email
- **aws_security check_mfa**: Finds users with no MFA configured and sends the list to the specified target

For the arguments used with the commands of the ‘aws_security’ group type **aws_security [command_name] -h** in the command line.

5.5.3 AWS_RI

The ‘aws_ri’ group includes the commands related to AWS Reserved Instances. The following commands are available:

- **aws_ri describe**: Describes reserved instances
- **aws_ri list_offerings**: Describes RI offerings
- **aws_ri buy**: Buys reserved instances
- **aws_ri modify**: Modifies reserved instances

For the arguments used with the commands of the ‘aws_ri’ group type **aws_ri [command_name] -h** in the command line.

5.5.4 TEMPLATE

The ‘template’ group includes the ‘template analyze’ command used to perform analysis of the CloudFormation template from the previously uploaded file. The command displays the human-readable template description in the response.

For the arguments used with the ‘template analyze’ command type **template analyze -h** in the command line.
5.6 Azure

The ‘azure’ category includes the commands related to resource configuration and management on the Microsoft Azure platform, as well as to the custom configuration of Azure for specific project requirements.

5.6.1 Azure

The ‘azure’ group includes the commands related to Microsoft Azure. The following commands are available:

- **azure add_zone_alias** Adds an alias to an existing Microsoft Azure zone (for example, northeurope for North Europe)
- **azure add_trusted_ip** Adds the NSG (Network Security Group) rule allowing inbound connection to the instances of the specified project from the specified IP
- **azure config_network** Creates Network Security Group and Virtual Network for the specified Azure project
- **azure subscript_pool** Describes the current state of subscriptions pool by the given enrolment
- **azure get_net_config** Retrieves the network configuration of the specified project(s)
- **azure del_trusted_ip** Removes the NSG (Network Security Group) rule allowing inbound connection to the instances of the specified project from the specified IP
- **azure activate_project** Activates a project in Microsoft Azure
- **azure add_zone** Adds a new zone in Microsoft Azure
- **azure add_image** Adds a new image in Microsoft Azure to the specified zone
- **azure add_subscript** Adds a new Microsoft Azure subscription
- **azure add_enrolment** Adds a new Microsoft Azure enrolment
- **azure del_subscript** Removes an existing Microsoft Azure subscription
- **azure grant_access** Grants access to the Microsoft Azure portal to the specified user within the specified project
- **azure revoke_access** Revokes access to the Microsoft Azure portal to the specified user within the specified project
- **azure share_credit** Shares credit among all projects of the specified enrolment in proportion to their workload
- **azure get_subscript** Retrieves the list of Azure subscriptions
- **azure switch_api** Switches the API mode (ARM/ASM) for the specified region

For the arguments used with the commands of the ‘azure’ group type **azure [command_name] -h** in the command line.
5.6.2 AZURE_CUSTOM

The ‘azure_custom’ group includes the commands related to custom half-manual configuration of Microsoft Azure for specific requests. The following commands are available:

- **azure_custom activate_project** Activates a project in Microsoft Azure skipping network configuration
- **azure_custom add_subscript** Adds a new Microsoft Azure subscription skipping network configuration

For the arguments used with the commands of the ‘azure_custom’ group type **azure_custom [command_name] -h** in the command line.

5.6.3 ARM

The ‘arm’ group includes the commands related to ARM (Azure Resource Management). The following commands are available:

- **arm config_project** Configures the project for using the ARM API
- **arm set_def_groups** Applies configuration for Azure security groups
- **arm shape_mapping** Configures shape mapping for a zone
- **arm add_image** Adds a machine image for use with the ARM API. The list of images is available in the Azure Portal marketplace or via CLI (azure help vm image)
- **arm init_lookup** Initially adds an ARM resource to EPAM Orchestrator
- **arm check_config** Checks the ARM configuration
5.7 GOOGLE

The ‘google’ category includes the commands related to resource configuration and management in Google Cloud Platform.

5.7.1 GOOGLE

The ‘google’ group includes the commands related to Google Cloud Platform. The following commands are available:

- **google add_temp_access_user**: Adds the user for providing temporary access
- **google change_password**: Changes the password of the user for providing temporary access
- **google add_account_system_username**: Adds a system username to a Google account
- **google list_temp_access_users**: Retrieves the list of users with temporary access
- **google list_iam_users**: Retrieves the list of IAM users within the project
- **google refresh_images**: Refreshes information about images (family, licenses, etc.)
- **google activate_project**: Activates a project in a Google zone
- **google add_zone**: Adds a new zone in Google Cloud
- **google add_image**: Adds a new image in Google Cloud to the specified zone
- **google list_accounts**: Retrieves the list of existing Google accounts
- **google list_images**: Retrieves the list of images in the specified project
- **google list_zones**: Retrieves the list of existing Google Cloud zones
- **google add_iam_user**: Adds a new IAM user to the project
- **google list_projects**: Retrieves the list of projects activated in Google Cloud
- **google edit_zone**: Edits Google zone settings
- **google setup_account**: Sets up a new Google account

For the arguments used with the commands of the ‘google’ group type `google [command_name] -h` in the command line.
5.8 CSA, HP OO, OPENSTACK

This category includes the commands related to resource management in CSA, HP OO and OpenStack virtualization platforms.

5.8.1 CSA

The ‘csa’ group includes the commands related to HP Cloud Services Automation. The following commands are available:

- **csa activate_project** Activates an HP Cloud Services Automation project in EPAM Cloud Orchestrator
- **csa fix_old_project** Changes the CSASubscription requestor (project) if the current project is inactive
- **csa restore_to_csa** Returns HP CSA subscriptions from EO to CSA
- **csa restore_missing** Restores missing EO instances existing in CSA
- **csa add_zone** Creates a new CSA zone
- **csa del_subscript** Deletes HP CSA subscriptions from CSA only
- **csa sync_from_csa** Synchronizes HP CSA subscription fields between CSA and EO
- **csa put_under_eo** Puts an existing HP CSA subscription under EO
- **csa add_ownership** Adds HP CSA ownership for the specified zone and instances
- **csa vlan_activate** Activates a new VLAN for the specified project in the specified zone
- **csa set_catalog** Sets catalog ID to the active HP CSA subscriptions in the specified zone
- **csa add_secondary_catalog** Adds a CSA secondary catalog
- **csa add_shape** Adds a new CSA shape
- **csa config_api** Set HP CSA API user name and catalog name
- **csa add_offering** Adds a new CSA offering
- **csa check_offerings** Retrieves the existing CSA offerings
- **csa get_capacity** Shows open, close and current values and blocked actions for all CSA regions
- **csa set_capacity** Sets open and close values for a single CSA region

For the arguments used with the commands of the ‘csa’ group type **csa [command_name] -h** in the command line.
5.8.2 HPOO

The ‘hpoo’ group includes the commands related to HP Operations Orchestration. The following commands are available:

- **hpoo activate_project**
  Activates an HPOO project in EPAM Cloud Orchestrator.

- **hpoo get_problem_inst**
  Lists instances in starting state or having no IP.

- **hpoo refresh_images**
  Refreshes the list of machine images in EPAM Cloud Orchestrator.

- **hpoo add_zone**
  Adds a new HPOO zone.

- **hpoo config_zone**
  Configures an HP OO zone.

- **hpoo vlan_activate**
  Activates a new VLAN for project.

- **hpoo configvs**
  Configures HP OO VSphere host name/username and password.

- **hpoo check_flows**
  Checks that flows are present and valid on HP OO.

- **hpoo config_flow**
  Configures HP OO flow.

For the arguments used with the commands of the ‘hpoo’ group type **hpoo [command_name] -h** in the command line.

5.8.3 OPEN_STACK

The ‘open_stack’ group includes the commands related to the OpenStack virtualization platform. The following commands are available:

- **open_stack activate_project**
  Activates a project in OpenStack.

- **open_stack activate_zones_personal_project**
  Activates a separate tenant for hosting personal project resources in the specified zone.

- **open_stack notific_switcher**
  Enables or disables notifications.

- **open_stack setup_networking**
  Sets up networking for all projects in the specified zone (for MANUAL networking mode only).

- **open_stack add_zone**
  Adds a new OpenStack zone.

- **open_stack edit_zone**
  Edits OpenStack zone settings.

- **open_stack get_zones**
  Retrieves the list of all active OpenStack zones.

- **open_stack add_image**
  Adds an image to the specified OpenStack zone.

- **open_stack get_images**
  Retrieves the list of all public images available in the zone.

- **open_stack delete_image**
  Deletes an image for the zone. The image is assigned the DELETED status but is not deleted from OpenStack.

- **open_stack set_image_id**
  Sets a new ID for an existing image.
• **open_stack add_shapes** Creates default shapes for an OpenStack zone
• **open_stack delete_shapes** Deletes shapes by flavor ID
• **open_stack get_shapes** Retrieves the list of all shapes available in the zone
• **open_stack get_default_shapes** Retrieves the list of default shape configurations
• **open_stack get_quotas** Sets quota for the specified project or for all projects in the specified zone
• **open_stack get_quotas** Retrieves the list of project quotas in the zone
• **open_stack get_hosts** Retrieves the list of all hosts in the OpenStack zone
• **open_stack create_recycle_bin** Creates a Recycle Bin for an OpenStack zone
• **open_stack describe_recycle_bin** Describes the Recycle Bin in an OpenStack zone
• **open-stack edit_recycle_bin** Edits the Recycle Bin properties for an OpenStack zone
• **open_stack restore_from_recycle_bin** Restores an instance from the Recycle Bin on OpenStack
• **open_stack remove_from_recycle_bin** Removes an instance from the Recycle Bin on OpenStack

For the arguments used with the commands of the `open_stack` group type **open_stack [command_name] -h** in the command line.
5.9 PAAS

The ‘PaaS’ category includes the commands related to various platform services available in EPAM Cloud.

5.9.1 PAAS

The ‘paas’ group includes the commands related to platform services management. The following commands are available:

- **paas register**: Registers a new custom platform service
- **paas delete**: Deletes the specified platform service
- **paas describe**: Retrieves the list of available platform services
- **paas restrict**: Restricts service usage for zone(s)
- **paas unrestrict**: Lifts restrictions for service usage for zone(s)
- **paas show_restricted**: Shows restricted services for zones, virtualization type or all

For the arguments used with the commands of the ‘paas’ group type **paas [command_name] -h** in the command line.

5.9.2 CHEF

The ‘chef’ group includes the commands related to the Chef service. The following commands are available:

- **chef cleanup**: Removes chef nodes (and clients) for the deleted instances
- **chef describe_server**: Retrieves the specified Chef server data
- **chef add_config**: Adds a new Chef server configuration
- **chef get_nodes**: Retrieves the list of chef nodes according to the specified filter
- **chef zone_unassign**: Unassigns the specified zone from the Chef server
- **chef zone_assign**: Assigns the specified zone to the specified Chef server
- **chef update_initscript**: Updates Chef initial scripts

For the arguments used with the commands of the ‘chef’ group type **chef [command_name] -h** in the command line.

5.9.3 DOCKER

The ‘docker’ group includes the commands related to the Docker Service. The following commands are available:

- **docker get_images**: Retrieves all Docker enterprise machine images
• **docker delete_image** Deletes the specified Docker enterprise machine image
• **docker add_image** Creates a new Docker enterprise machine image
• **docker del_repository** Deletes Docker enterprise repository defined by its search identifier
• **docker add_repository** Creates a new Docker enterprise repository
• **docker get_repositories** Retrieves all Docker enterprise repositories

For the arguments used with the commands of the ‘docker’ group type `docker [command_name] -h` in the command line.

**5.10 TEMP**

The ‘temp’ group includes the temporary commands. Please do not use them in the normal course of work.
6 MAESTRO CLI ADMIN UTILITY – USE CASES

6.1 AWS – ADMINISTRATION CASES

6.1.1 AWS Zone Creation

Virtualization zone creation in AWS consists of the following steps:

- **Create a New Zone**
  - `aws add_zone`
    - `--zone`
    - `--region`
    - `--availability-zone`
    - `--cf-endpoint`
    - `--ec-endpoint`
    - `--s-endpoint`
    - `--cw-endpoint`
    - `--assign`
    - `--disable-billing-mix-mode`
    - `--unreachable`

- **Configure Virtual Profile**
  - `zone set_virt_profile`
    - `--zone`
    - `--profile-name`
    - `--shape-mapping`

- **Add AMI**
  - `aws add_image`
    - `--zone`
    - `--image-id`
    - `--amiid`
    - `--description`
    - `--group`
    - `--virt-profile`
    - `--username`

- **Set Cost Center**
  - `billing set_cost_center`
    - `--zone`
    - `--cost-center-name`

![Figure 6 - AWS zone creation flow](image)

- **Zone Creation**

To create a new AWS zone, use the following command:

```bash
aws add_zone [arguments]
```

The `aws add_zone` command uses the following arguments:

<table>
<thead>
<tr>
<th>Command Arguments</th>
<th>Argument</th>
<th>Description</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>-z, --zone</td>
<td>Name of the virtualization zone to be created.</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>-r, --region</td>
<td>Code of AWS region in which the virtualization zone is to be created</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>-a, --availability-zone</td>
<td>AWS availability zone in which the virtualization zone is to be created</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>-l, --location</td>
<td>Location (e.g. North Europe)</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>-ar, --aws-region</td>
<td>AWS region code (e.g. eu-central-1)</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>-c, --cf-endpoint</td>
<td>CloudFormation endpoint. Required for reachable zones</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>-e, --ec-endpoint</td>
<td>EC2 endpoint. Required for reachable zones</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>-t, --ct-endpoint</td>
<td>CloudTrail endpoint. Required for reachable zones</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>-s, --s-endpoint</td>
<td>S3 endpoint. Required for reachable zones</td>
<td>No</td>
</tr>
</tbody>
</table>
Command Example:

```
aws add_zone --r us-east-1 --a us-east-1b --z zone_name --ar aws_region
```

### Zone Virtual Profile Configuration

A virtual profile contains the VM shape mapping between EPAM Cloud and AWS. Configuring a virtual profile for an AWS zone sets the shapes available for instance creation in such zone and ensures that the EPC shape selected for a VM corresponds to the correct shape in AWS.

To configure the zone virtual profile, use the following command:

```
zone set_virt_profile [arguments]
```

The `zone set_virt_profile` command uses the following arguments:

<table>
<thead>
<tr>
<th>Command Arguments</th>
<th>Description</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>-z, --zone</td>
<td>Virtualization zone</td>
<td>Yes</td>
</tr>
<tr>
<td>-p, --profile-name</td>
<td>Virtual profile name</td>
<td>Yes</td>
</tr>
<tr>
<td>-s, --shape-mapping</td>
<td>Shape mapping pair: epc_shape=aws_shape. Use &quot;=&quot; as delimiter. For several mappings repeat the parameter: -s epc_shape1=aws_shape1 -s epc_shape2=aws_shape2 -s epc_shapeN=aws_shapeN. If using Windows command line, encase the -s parameter in quotes i.e. &quot;epc_shape=aws_shape&quot;</td>
<td>Yes</td>
</tr>
</tbody>
</table>

Command Example:

```
zone set_virt_profile -z zone_name -p profile -s MICRO=t2.micro -s SMALL=t2.small -s MEDIUM=m3.medium
```

- **Adding Machine Images to AWS Zone**

To add machine images which will be available in the AWS zone, use the following command:

```bash
aws add_image [arguments]
```

The `aws add_image` command uses the following arguments:

<table>
<thead>
<tr>
<th>Command Arguments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Argument</td>
</tr>
<tr>
<td>---</td>
</tr>
<tr>
<td>-z, --zone</td>
</tr>
<tr>
<td>-i, --imageId</td>
</tr>
<tr>
<td>-a, --amiId</td>
</tr>
<tr>
<td>-d, --description</td>
</tr>
<tr>
<td>-g, --group</td>
</tr>
<tr>
<td>-v, --virt-profile</td>
</tr>
<tr>
<td>-u, --username</td>
</tr>
</tbody>
</table>

**Command Example:**

```bash
aws add_image -i W2012R2Std -a ami-******* -d 'Windows Server 2012 R2 Standard Edition' -z zone -g PUBLIC -v profile -u user
```

- **Setting Cost Center for AWS Zone**

For the correct billing of the Cloud services for the projects used in the AWS zone, a cost center has to be assigned to it. To assign a cost center to a new AWS zone, use the following command:

```bash
billing set_cost_center [arguments]
```

The `billing set_cost_center` command uses the following arguments:

<table>
<thead>
<tr>
<th>Command Arguments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Argument</td>
</tr>
<tr>
<td>---</td>
</tr>
<tr>
<td>-z, --zone</td>
</tr>
<tr>
<td>-c, --cost-center-name</td>
</tr>
</tbody>
</table>

**Command Example:**

```bash
billing set_cost_center -z zone -c cost_center
```
6.1.2 Project Activation in AWS

In AWS, each project is activated within its LINKED account, one account per project. The Level1.5 Team always has a pool of unreserved accounts to use for project activation. New accounts are created manually by the Level1.5 Team, with account creation sometimes taking up to 24 hours. With the introduction of AWS Organizations, the option of account creation via API has been implemented.

In addition to the LINKED account, project activation requires a PAYING account to enable consolidated billing of all LINKED accounts.

When accounts have been configured properly, the project can be activated. In AWS, a project can be activated in a standard way (by the `aws activate_project` Admin Utility command) or automatically. A project is activated automatically, if any costs exist for the project in a region where the project is not yet activated. This can happen, for example, when a project creates resources in a non-activated region via the AWS console.

The `aws activate_project` command uses the following arguments:

<table>
<thead>
<tr>
<th>Command Arguments</th>
<th>Argument</th>
<th>Description</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>-p, --project</td>
<td>Project abbreviation in UPSA</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>-s, --shape</td>
<td>Shape name. For several shapes, repeat the parameter</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>-z, --zone</td>
<td>Virtualization zone</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>--all</td>
<td>All zones (the project will be activated in all existing AWS zones except unreachable ones)</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>-f, --fake-project</td>
<td>Fake project flag (indicates a fake project, that is, the project not existing in UPSA; used for testing purposes)</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>-a, --auto-configuration-disabled</td>
<td>Flag disabling auto-configuration for the project</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>-c, --account</td>
<td>AWS account name</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>-n, --subnet-id</td>
<td>ID of AWS region subnet in which all resources of the project will be created</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>--skip-cloud-trail</td>
<td>Flag used to skip CloudTrail activation</td>
<td>No</td>
</tr>
</tbody>
</table>

Command Example:
```
aws activate_project -p project_code -s small -s medium -s large --all
```

When a project is activated with the `aws activate_project` command, the following actions are automatically performed on the AWS side:

1. Creation and configuration of the EC2_INSTANCE_ROLE IAM role:
   - Creation of the EC2_INSTANCE_ROLE IAM role and configuration of its permissions
   - Creation of the Instance Profile
   - Association of EC2_INSTANCE_ROLE with Instance Profile
2. SSO configuration:
   - Creation of SAML provider for the LINKED account and its upload to AWS
- Creation of roles and their permissions
- Creation of the account alias
3. Creation of a default group for IAM roles and their permissions configuration
4. Security Groups configuration
5. CloudTrail service configuration

Steps 4 and 5 are performed for each zone.

There may be cases when one or several of the project configuration steps is not performed automatically. In such situations, the necessary actions can be performed manually. The instructions and related Admin Utility commands are described below.

- **EC2 Instance Role Configuration**

Applications running on AWS instances make requests to AWS. Such operations require authorization with access keys transferred to each such instance. This process can be unified by setting IAM roles for EC2 instances. The flow is as follows:

- Create IAM role
- Assign permissions to the IAM role
- Specify the role during the instance launch

The instance will request temporary access keys and use them for all requests permitted to the corresponding role. Roles are stored in the AWSRoles collection. The same document also stores the trusted policies defining that the ec2.amazonaws.com service can use this role and the actions permitted to the role (AttachVolume, CreateVolume, CreateTags, S3, etc.) Roles are assigned to instances in Instance Profiles.

If needed, default instance roles can be added to the project with the following command:

```
aws set_def_role [arguments]
```

The `aws set_def_role` command uses the following arguments:

<table>
<thead>
<tr>
<th>Command Arguments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Argument</td>
</tr>
<tr>
<td>-p, --project</td>
</tr>
</tbody>
</table>

**Command Example:**

```
aws set_def_role --project
```

For more information, see [IAM Roles for Amazon EC2](https://docs.aws.amazon.com/IAM/latest/dg/sso-roles.html) page in the official AWS documentation.

- **SSO Configuration**

There are four roles used for SSO. They are stored in the AwsIamEntities collection. A role is selected in accordance with the user’s project role (see the [User Permissions](https://docs.aws.amazon.com/IAM/latest/dg/sso-roles.html) page on Knowledge Base for the full matrix). The roles include the policies from the AwsIamPolicies.

When a **user has to be assigned certain special permissions** other than default, use the following command:

```
aws set_def_role --p project
```
The `aws sso_add_custom` command uses the following arguments:

<table>
<thead>
<tr>
<th>Command Arguments</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Argument</strong></td>
</tr>
<tr>
<td>-p, --project</td>
</tr>
<tr>
<td>-e, --email</td>
</tr>
<tr>
<td>-n, --name</td>
</tr>
</tbody>
</table>

Command Example:

```bash
cmd> aws sso_add_custom -p project -e Firstname_Lastname@epam.com -n role_name
```

To remove a certain SSO role, use the following command:

```bash
cmd> aws sso_del_custom [arguments]
```

The `aws sso_del_custom` command uses the following arguments:

<table>
<thead>
<tr>
<th>Command Arguments</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Argument</strong></td>
</tr>
<tr>
<td>-p, --project</td>
</tr>
<tr>
<td>-e, --email</td>
</tr>
<tr>
<td>-n, --name</td>
</tr>
</tbody>
</table>

Command Example:

```bash
cmd> aws sso_del_custom -p project -e Firstname_Lastname@epam.com -n role_name
```

To configure permissions for a certain user, use the following command:

```bash
cmd> aws sso_manage_access [arguments]
```

The `aws sso_manage_access` command uses the following arguments:

<table>
<thead>
<tr>
<th>Command Arguments</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Argument</strong></td>
</tr>
<tr>
<td>-p, --project</td>
</tr>
<tr>
<td>-e, --email</td>
</tr>
<tr>
<td>-i, --iam-entity-name</td>
</tr>
<tr>
<td>-a, --action</td>
</tr>
</tbody>
</table>
In this case, user's permissions are updated by replacing their role with one of the four available roles. If a user has several roles in several projects, they can choose one of these to authorize in AWS.

**SSO Roles Configuration**

For easier use, each AWS account used in EPAM Orchestrator has an alias which is a more human-friendly string than the AWS account number. For example, account number 9213429384 can have alias epm-cit2-234. Aliases are unique within the entire AWS. To **configure a project AWS account**, use the following command:

```bash
aws config_sso [arguments]
```

The `aws config_sso` command uses the following arguments:

<table>
<thead>
<tr>
<th>Command Arguments</th>
<th>Argument</th>
<th>Description</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>-p, --project</td>
<td>Project abbreviation in UPSA</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>--all</td>
<td>Applies for all projects</td>
<td>No</td>
</tr>
</tbody>
</table>

**Command Example:**

```bash
aws config_sso -p project
```

To **retrieve the list of available SSO roles**, use the following command:

```bash
aws get_iam_entities [arguments]
```

The `aws get_iam_entities` command uses the following arguments:

<table>
<thead>
<tr>
<th>Command Arguments</th>
<th>Argument</th>
<th>Description</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>-n, --name</td>
<td>AWS IAM entity name. If omitted, all entities with their general info will be returned</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>-t, --type</td>
<td>AWS IAM entity type. Allows filtering by type or describing detailed info by name. Allowed values: SSO_ROLE</td>
<td>No</td>
</tr>
</tbody>
</table>

**Command Example:**

```bash
aws get_iam_entities -t SSO_ROLE
```

**AWS Policy Management**

To **attach an AWS policy to an SSO role**, use the following command:

```bash
aws attach_policy [arguments]
```

**Command Example:**

```bash
aws attach_policy
```
The `aws attach_policy` command uses the following arguments:

<table>
<thead>
<tr>
<th>Command Arguments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Argument</td>
</tr>
<tr>
<td>-p, --policy</td>
</tr>
<tr>
<td>-n, --name</td>
</tr>
<tr>
<td>-t, --type</td>
</tr>
</tbody>
</table>

Command Example:
```
aws attach_policy -p policy_name -n entity_name -t SSO_ROLE
```

To detach an AWS policy from an SSO role, use the following command:
```
aws detach_policy [arguments]
```

The `aws detach_policy` command uses the following arguments:

<table>
<thead>
<tr>
<th>Command Arguments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Argument</td>
</tr>
<tr>
<td>-p, --policy</td>
</tr>
<tr>
<td>-n, --name</td>
</tr>
<tr>
<td>-t, --type</td>
</tr>
</tbody>
</table>

Command Example:
```
aws detach_policy -p policy_name -n entity_name -t SSO_ROLE
```

To retrieve the list of available policies, use the following command:
```
aws get_policies [arguments]
```

The `aws get_policies` command uses the following arguments:

<table>
<thead>
<tr>
<th>Command Arguments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Argument</td>
</tr>
<tr>
<td>-n, --name</td>
</tr>
</tbody>
</table>

Command Example:
```
aws get_policies -n policy_name
```
To add a new policy, use the following command:

```
aws save_policy [arguments]
```

The ‘aws save_policy’ command uses the following arguments:

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>-n, --name</td>
<td>AWS policy name</td>
<td>Yes</td>
</tr>
<tr>
<td>-f, --file</td>
<td>Path to the file containing the policy*</td>
<td>No</td>
</tr>
<tr>
<td>-t, --type</td>
<td>Policy type. Allowed values: [INLINE, MANAGED, MANAGED_CUSTOM, S3]</td>
<td>Yes</td>
</tr>
</tbody>
</table>

* Make sure that the file contains placeholders for accounts, bucket names, etc. For default AWS policies, no file upload is required.

Command Example:

```
aws save_policy -n policy_name -t INLINE -f path_to_file
```

**IAM Role Group Configuration**

The scope of actions allowed to IAM users can be defined by using IAM user groups. One group is created for each AWS account. The group contains the permissions described in the “orchestrator-default-admin-group” document in the AWSRoles collection. All IAM users are included in this group.

If necessary, a group policy can be uploaded using the following command:

```
aws up_group_policy [arguments]
```

The ‘aws up_group_policy’ command uses the following arguments:

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>-n, --name</td>
<td>AWS group name</td>
<td>No</td>
</tr>
<tr>
<td>-a, --account</td>
<td>AWS account for which the policy is to be uploaded. If omitted, the policy will be uploaded for all accounts</td>
<td>No</td>
</tr>
</tbody>
</table>

Command Example:

```
aws up_group_policy -n policy_name -a aws_account
```

To update an existing group, use the following command:

```
aws config_group [arguments]
```

The ‘aws config_group’ command uses the following arguments:
### Command Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>-p, --project</td>
<td>Project abbreviation in UPSA</td>
<td>No</td>
</tr>
<tr>
<td>-s, --scope</td>
<td>Scope. Allowed values: [DEFAULT, PROJECT]. Default value: DEFAULT</td>
<td>No</td>
</tr>
<tr>
<td>-n, --name</td>
<td>Group name</td>
<td>Yes</td>
</tr>
<tr>
<td>-l, --location</td>
<td>Group policy location</td>
<td>No</td>
</tr>
</tbody>
</table>

**Command Example:**

```
aws config_group -p project -n group_name
```

### Security Groups Configuration

Security groups are stored in the OrchestrationSettings collection. This collection contains 5 security groups. One of the security groups, default, is not used in AWS configuration.

To **configure security groups for AWS**, use the following command:

```
aws_security set_def_groups [arguments]
```

The `aws_security set_def_groups` command uses the following arguments:

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>-p, --project</td>
<td>Project abbreviation in UPSA</td>
<td>No</td>
</tr>
<tr>
<td>-z, --zone</td>
<td>Virtualization zone</td>
<td>No</td>
</tr>
<tr>
<td>--all-zones</td>
<td>Applicable for all group activated for the project</td>
<td>Yes</td>
</tr>
<tr>
<td>--all-projects</td>
<td>Applicable for all projects in all zones</td>
<td>No</td>
</tr>
<tr>
<td>-v, --vpc-id</td>
<td>VPC ID. If omitted, the default VPC will be used</td>
<td>No</td>
</tr>
</tbody>
</table>

**Command Example:**

```
aws_security set_def_groups -p project -n group_name
```

To **update the default security groups** in the database, use the following command:

```
security update_def_group [arguments]
```
The `security update_def_group` command uses the following arguments:

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>-g, --security-group-name</td>
<td>Security group name</td>
<td>Yes</td>
</tr>
<tr>
<td>-i, --ip-range</td>
<td>IPv4 CIDR range to add a new rule to the specified security group. For example: 74.11.192.96/27</td>
<td>Yes</td>
</tr>
<tr>
<td>-r, --remove</td>
<td>Flag used to remove an item instead of adding</td>
<td>No</td>
</tr>
</tbody>
</table>

Command Example:

```
security update_def_group -g group_name -i 74.11.192.96/27
```

Security groups are updated in the database and then applied to AWS. When an update operation is repeated, the existing groups and the correct rules are not deleted but are matched to the groups in the OrchestratorSettings collection. This is done to prevent incorrect configuration of resources using such groups.

- **Security Groups Backup**

Security groups are backed up by schedule or manually using the following command:

```
aws_security save_groups [arguments]
```

The `aws_security save_groups` command uses the following arguments:

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>-p, --project</td>
<td>Project abbreviation in UPSA</td>
<td>Yes</td>
</tr>
<tr>
<td>-l, --label</td>
<td>Restore groups by label</td>
<td>No</td>
</tr>
</tbody>
</table>

Command Example:

```
aws_security save_groups -p project
```

Also, backups are created automatically during security groups update.

To restore the security groups from backup, use the following command:

```
aws_security restore_groups [arguments]
```

The `aws_security restore_groups` command uses the following arguments:

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>-p, --project</td>
<td>Project abbreviation in UPSA</td>
<td>Yes</td>
</tr>
</tbody>
</table>
EPAM Cloud OrCHEstrator. Maestro CLI Admin Utility

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>-i, --backup-id</td>
<td>Backup ID to restore from</td>
<td>No</td>
</tr>
<tr>
<td>-d, --date</td>
<td>Date to restore from in the yyyy-MM-dd'T'HH format (UTC)</td>
<td>No</td>
</tr>
<tr>
<td>-l, --label</td>
<td>Restore groups by label</td>
<td>No</td>
</tr>
<tr>
<td>-z, --zone</td>
<td>Virtualization zone</td>
<td>No</td>
</tr>
</tbody>
</table>

Command Example:
```
aws_security restore_groups -p project -i backup_id
```

To view the existing backups for a project, use the following command:
```
aws_security describe_backups [arguments]
```

The `aws_security describe_backups` command uses the following arguments:

<table>
<thead>
<tr>
<th>Command Arguments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Argument</td>
</tr>
<tr>
<td>-p, --project</td>
</tr>
</tbody>
</table>

Command Example:
```
aws_security describe_backups -p project
```

- **CloudTrail Service Activation**

For each project, the CloudTrail service must be activated and configured in all zones. CloudTrail should be configured to direct all logs to the S3 bucket of the root account (currently, the PAYING epmc-clo account). For that purpose, the permissions for the new account are added to the policy of the parent PAYING account's S3 bucket. Afterwards, the child account can store logs in the parent account bucket.

If the CloudTrail service is not activated for a project, **activate** it using the following command:
```
aws activ_cloudtrail [arguments]
```

The `aws activ_cloudtrail` command uses the following arguments:

<table>
<thead>
<tr>
<th>Command Arguments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Argument</td>
</tr>
<tr>
<td>-p, --project</td>
</tr>
<tr>
<td>-z, --zone</td>
</tr>
<tr>
<td>-b, --bucket-name</td>
</tr>
<tr>
<td>-l, --log-file-prefix</td>
</tr>
</tbody>
</table>
Command Example:
```bash
aws activ_cloudtrail -p project -z zone
```

To **view the CloudTrail configuration** for a project, use the following command:
```bash
aws get_cloudtrail [arguments]
```

The **aws activ_cloudtrail** command uses the following arguments:

<table>
<thead>
<tr>
<th>Command Arguments</th>
<th>Description</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>-p, --project</td>
<td>Project abbreviation in UPSA</td>
<td>Yes</td>
</tr>
<tr>
<td>-z, --zone</td>
<td>Virtualization zone</td>
<td>No</td>
</tr>
</tbody>
</table>

Command Example:
```bash
aws get_cloudtrail -p project -z zone
```

### 6.1.3 Access to AWS

There are three methods of getting access to AWS:

- **Via AWS SSO.** In this case, the user is assigned one of the four roles stored in the AwsIamEntities collection.

- **Using the or2awsmc Maestro CLI command.** In this case, the user is assigned the permissions of the FEDERATED_USER_ROLE stored in the AWSRoles collection. If the user is a member of the EPM-CSUP project, such user is by default assigned administrator permissions according to the CLOUD_SUPPORT_ROLE stored in the AWSRoles collection. If the user belongs to the ALL_OPERATIONS user group in EPAM Orchestrator, such user can access the AWS console under any project.

- **Through the IAM user.** In this case, the user is subject to the restrictions of the default group for IAM users GROUP_ROLE stored in the AWSRoles collection.

### 6.1.4 AWS Organizations

AWS Organizations support multiple AWS accounts management on the basis of policies. The AWS Organizations service allows creating Organization Units and assign certain policies to them. AWS Organizations offer the following features:

- **Automatic account creation.** If new accounts are included in the existing Organization Units, their policies will be automatically applied to the new accounts.

- **Accounts can be joined into Organization Units on the billable/non-billable principle which allows monitoring costs.**

- **Reserved Instances can be bought for certain Organization Units, thus reducing the internal project costs.**
For more details on AWS Organizations, see the What is AWS Organizations? page in the AWS documentation.

### 6.1.5 Reserved Instances

Reserved instances allow significantly reducing the infrastructure costs. They are reserved at fixed prices for a period of one year or more. A reserved instance is assigned to a random VM of the specified type within an organization. When the VM is stopped, the reserved instance is transferred to another VM of the same type.

The following actions are supported for reserved instances:

- An instance reserved for a region can be modified to be reserved for an availability zone, and vice versa
- A reserved instance size can be changed (for Linux instances only). For example, one c2.micro instance can be replaced with two c2.nano instances.

#### Displaying Reserved Instances

To view the list of reserved instances, use the following command:

```
aws_ri describe [arguments]
```

The ‘aws_ri describe’ command uses the following arguments:

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>-f, --force-update</td>
<td>Update info from Amazon before retrieving the data. May take long time!</td>
<td>No</td>
</tr>
<tr>
<td>--target</td>
<td>Parameter defining where the output is to be displayed. Allowed values: [ssh_console, file, email]. Default: ssh_console</td>
<td>No</td>
</tr>
</tbody>
</table>

The command returns the list of all available reserved instances:

<table>
<thead>
<tr>
<th>RI ID</th>
<th>Project code</th>
<th>Zone</th>
<th>Start Date</th>
<th>End Date</th>
<th>Cost</th>
<th>Product description</th>
<th>Share</th>
</tr>
</thead>
<tbody>
<tr>
<td>r302002-4751-42c4-5bea-56b387973706</td>
<td></td>
<td></td>
<td>2013-10-28</td>
<td>2014-10-27</td>
<td>7</td>
<td>c3.large</td>
<td></td>
</tr>
<tr>
<td>r302007-4751-42c4-5bea-56b387973706</td>
<td></td>
<td></td>
<td>2013-10-28</td>
<td>2014-10-27</td>
<td>7</td>
<td>c3.large</td>
<td></td>
</tr>
<tr>
<td>r308000-4751-42c4-5bea-56b387973706</td>
<td></td>
<td></td>
<td>2013-10-28</td>
<td>2014-10-27</td>
<td>7</td>
<td>c3.large</td>
<td></td>
</tr>
</tbody>
</table>

#### Modifying Reserved Instances

To modify reserved instances, use the following command:

```
aws_ri modify [arguments]
```

The ‘aws_ri modify’ command uses the following arguments:

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>-i, --ri-id</td>
<td>ID of the reserved instance</td>
<td>Yes</td>
</tr>
<tr>
<td>-c, --target-configuration</td>
<td>Target reserved instance configuration consisting of the availability zone name, instance shape and count. Input</td>
<td>Yes</td>
</tr>
</tbody>
</table>
format: az:shape:count. For example: us-west-2a:t2.micro:4. To apply reserved instances at the REGION level, use 'all' for availability zone. To set several configurations, repeat the parameter.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>-z, --zone</td>
<td>Virtualization zone</td>
<td>Yes</td>
</tr>
</tbody>
</table>

**Command Example:**

```
aws_ri modify -i 3e26582b-4713-4c0c-983e-9a8f07fad59 -c all:m4.xlarge:6 -c all:m4.large:2 -z AWS-EUCENTRAL
```

This command changes 7 m4.xlarge reserved instances from the screenshot above to 6 m4.xlarge and 2 m4.large instances.

For more information on reserved instances modification, see the [Modifying Standard Reserved Instances](#) page in the AWS documentation.

- **Displaying Reserved Instance Offerings**

To view the list of reserved instances available for purchase, use the following command:

```
aws_ri list_offerings [arguments]
```

The `aws_ri list_offerings` command uses the following arguments:

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>-p, --project</td>
<td>Project abbreviation in UPSA</td>
<td>Yes</td>
</tr>
<tr>
<td>-z, --zone</td>
<td>Virtualization zone</td>
<td>Yes</td>
</tr>
<tr>
<td>-t, --instance-type</td>
<td>AWS instance type</td>
<td>Yes</td>
</tr>
<tr>
<td>-o, --os</td>
<td>Operating system. Allowed values: [linux, windows]</td>
<td>Yes</td>
</tr>
<tr>
<td>-s, --scope</td>
<td>Scope. Allowed values: [az, region]</td>
<td>Yes</td>
</tr>
<tr>
<td>--all</td>
<td>Add marketplace reserved instances to the result</td>
<td>No</td>
</tr>
</tbody>
</table>

**Command Example:**

```
aws_ri list_offerings -p project -z zone -t m4.xlarge -o linux -s region
```

The command output may contain reserved instances offered for sale by other users. Such instances can be purchased for less than one year.

- **Purchasing Reserved Instances**

To buy reserved instances from the list returned by the `aws_ri list_offerings` command, use the following command:
The `aws_ri buy` command uses the following arguments:

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>-p, --project</td>
<td>Project abbreviation in UPSA</td>
<td>Yes</td>
</tr>
<tr>
<td>-z, --zone</td>
<td>Virtualization zone</td>
<td>Yes</td>
</tr>
<tr>
<td>-i, --offering-id</td>
<td>Offering ID. Use the output of the <code>aws_ri list_offerings</code> command for possible options</td>
<td>Yes</td>
</tr>
<tr>
<td>-c, --count</td>
<td>Instance count</td>
<td>Yes</td>
</tr>
</tbody>
</table>

**Command Example:**

```
aws_ri buy -p project -z zone -i offering_id -c 5
```
6.2 MICROSOFT AZURE – ADMINISTRATION CASES

6.2.1 Azure Zone Creation

The typical Azure zone creation flow is as follows:

- **Setup Azure Enrollment**
  - `azure add_enrolment` command
  - `--enrolment-number`:
  - `--azure-image-name`:
  - `--bill-from`
  - `--tenant`
  - `--client-id`
  - `--client-key`

- **Create a New Zone**
  - `azure add_zone` command
  - `--zone`
  - `--location`
  - `--assign`:
  - `[--disable-billing-mix-mode]`

- **Set Cost Center**
  - `billing cost_center` command
  - `--zone`
  - `--cost-center-name`

- **Add Machine Images**
  - `azure add_image` command

![Figure 7 - Azure zone creation flow](image)

Each step is described in details below.

- **Azure Enrollment Setup**

Microsoft Azure provides its Cloud services on the basis of commitment under the Enterprise Agreement, the so-called enrolment. In order to create a zone in Azure and activate projects within such zone, the Azure enrolment details should be specified.

To add the Azure enrolment, use the following command:

```
azure add_enrolment [arguments]
```

The `azure add_enrolment` command uses the following arguments:

<table>
<thead>
<tr>
<th>Command Arguments</th>
<th>Description</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>-e, --enrolment-number</td>
<td>Enrolment ID</td>
<td>Yes</td>
</tr>
<tr>
<td>-a, --azure-image-name</td>
<td>Usage API access key received from the Enterprise Administrator</td>
<td>Yes</td>
</tr>
<tr>
<td>-b, --bill-from</td>
<td>The date to start billing from in yyyy-MM-dd'T'HH format</td>
<td>Yes</td>
</tr>
<tr>
<td>-t, --tenant</td>
<td>Tenant ID</td>
<td>Yes</td>
</tr>
<tr>
<td>-i, --client-id</td>
<td>Client ID</td>
<td>Yes</td>
</tr>
<tr>
<td>-k, --client-key</td>
<td>Client key</td>
<td>Yes</td>
</tr>
</tbody>
</table>

**Command Example:**

```
azure add_enrolment -e enrolment_number -a API_key -b 2016-04-01T00:00:00Z -t tenand_id -i client_id -k client_key
```
**Zone Creation**

To create a new Azure zone, use the following command:

```
azure add_zone [arguments]
```

The `azure add_zone` command uses the following arguments:

<table>
<thead>
<tr>
<th>Command Arguments</th>
<th>Description</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>-z, --zone</code></td>
<td>Name of the virtualization zone to be created. The zone name should contain the ‘AZURE’ pattern</td>
<td>Yes</td>
</tr>
<tr>
<td><code>-l, --location</code></td>
<td>Azure location</td>
<td>Yes</td>
</tr>
<tr>
<td><code>--assign</code></td>
<td>Assigns zone to the currently active node</td>
<td>No</td>
</tr>
<tr>
<td><code>--disable-billing-mix-mode</code></td>
<td>Defines whether the zone supports billing mode. If disabled, the Billing Engine shows costs based on EO audit only, otherwise EO audit will be integrated (mixed) with costs received from the cloud provider (e.g. in a form of CSV reports)</td>
<td>No</td>
</tr>
</tbody>
</table>

Command Example:

```
azure add_zone --assign -l "North Europe" -z AZURE-NEU
```

**Setting Cost Center for Azure Zone**

For the correct billing of the Cloud services for the projects used in the Azure zone, a cost center has to be assigned to it. To assign a cost center to a new Azure zone, use the following command:

```
billing set_cost_center [arguments]
```

The `billing cost_center` command uses the following arguments:

<table>
<thead>
<tr>
<th>Command Arguments</th>
<th>Description</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>-z, --zone</code></td>
<td>Name of zone</td>
<td>Yes</td>
</tr>
<tr>
<td><code>-c, --cost-center-name</code></td>
<td>Name of the cost center to be assigned to the zone</td>
<td>Yes</td>
</tr>
</tbody>
</table>

Command Example:

```
billing set_cost_center -z zone -c cost_center
```
Adding Machine Images to Azure Zone

To add machine images which will be available in the Azure zone, use the following command:

```
azure add_image [arguments]
```

The `azure add_image` command uses the following arguments:

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>-z, --zone</td>
<td>Name of virtualization zone to which the image is to be added</td>
<td>Yes</td>
</tr>
<tr>
<td>-i, --image-id</td>
<td>Image id (e.g. Ubuntu10.04_32-bit)</td>
<td>Yes</td>
</tr>
<tr>
<td>-a, --azure-image-name</td>
<td>Azure image name (e.g. 0c0083a6d9a24f2da9180e52cad83950__Zulu-1.7.0_55-0714-Win-GA)</td>
<td>Yes</td>
</tr>
<tr>
<td>-d, --description</td>
<td>VM image description</td>
<td>Yes</td>
</tr>
<tr>
<td>-g, --group</td>
<td>Image group. Valid values: PUBLIC, ENTERPRISE</td>
<td>Yes</td>
</tr>
<tr>
<td>-o, --os-type</td>
<td>Type of operating system. Valid values: WINDOWS, LINUX</td>
<td>Yes</td>
</tr>
<tr>
<td>-s, --size</td>
<td>Machine image size in GB</td>
<td>Yes</td>
</tr>
<tr>
<td>-u, --username</td>
<td>Default SSH user</td>
<td>Yes</td>
</tr>
</tbody>
</table>

Command Example:

```
azure add_image -i OracleLinux7_64-bit -a c290a6b031d841e09f2da759beb71f__Oracle-Linux-7 -d 'Oracle Linux 7 64-bit' -z zone -g PUBLIC -o LINUX -s 1 -u user
```

6.2.2 Activating a Project in Microsoft Azure

To activate a project in Azure, you need only the commands belonging to the `azure` group. The diagram below shows the typical flow for this case:

```
Activate Project
azure activate_project
  --project
  --shape
  [--zone]
  [--all]
  [--fake-project]
  [--auto-configuration-disabled]
  [--subscription-name]

Configure Network
azure config_network
  --project
  [--zone]

Check Configuration
azure get_net_config
  --project
```

Each step is described in details below.
### Project Activation

To activate a project in Microsoft Azure, use the following command:

```
azure activate_project [arguments]
```

The `azure activate_project` command uses the following arguments:

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>-p, --project</td>
<td>Project PMC code</td>
<td>Yes</td>
</tr>
<tr>
<td>-s, --shape</td>
<td>Shape name. For several shapes, repeat the parameter</td>
<td>Yes</td>
</tr>
<tr>
<td>-z, --zone</td>
<td>Virtualization zone</td>
<td>No</td>
</tr>
<tr>
<td>--all</td>
<td>All zones (the project will be activated in all existing Azure zones)</td>
<td>No</td>
</tr>
<tr>
<td>-f, --fake-project</td>
<td>Fake project flag (indicates a fake project, that is, the project not existing in UPSA; used for testing purposes)</td>
<td>No</td>
</tr>
<tr>
<td>-a, --auto-configuration-disabled</td>
<td>Flag disabling auto-configuration for the project</td>
<td>No</td>
</tr>
<tr>
<td>-u, --subscription-name</td>
<td>Azure subscription name</td>
<td>No</td>
</tr>
</tbody>
</table>

Command example:

```
azure activate_project -p project_code -s MICRO -s SMALL -s LARGE --all
```

### Network Configuration

When a project is activated, a network security group has to be configured for each zone in which the project is activated. The network security groups define the rules allowing or denying access instances in the virtual network.

To configure the network security groups for the project, use the following command:

```
azure config_network [arguments]
```

The `azure config_network` command uses the following arguments:

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>-p, --project</td>
<td>Project abbreviation in UPSA</td>
<td>Yes</td>
</tr>
<tr>
<td>-z, --zone</td>
<td>Virtualization zone. When no zones are specified, the network security groups will be configured for all zones in which the project has been activated.</td>
<td>No</td>
</tr>
</tbody>
</table>
The `azure config_network` command will create virtual networks and network security groups for all zones in which the project is activated and set the rules for them.

```
> azure config_network -p SAMPLE

Are you sure you want to perform the operation for instance SAMPLE? Type "y" or "n": Answer: y

Virtual Network 'VN_55ab5597e4b080f6aeb5eb23' for zone: 'East US' has been created.
Network Security Group 'NSG_55ab5597e4b080f6aeb5eb23' for zone: 'East US' has been created.
Virtual Network 'VN_55ab5597e4b080f6aeb5eb23' for zone: 'East US' has been created.
Network Security Group 'NSG_55ab5597e4b080f6aeb5eb23' (East US) has been added to subnet
Virtual Network 'VN_55ab5597e4b080f6aeb5eb23' for zone: 'East US' has been created.
Network Security Group 'NSG_55ab5597e4b080f6aeb5eb23' for zone: 'East US' has been created.
Virtual Network 'VN_55ab5597e4b080f6aeb5eb23' for zone: 'East US' has been created.
Network Security Group 'NSG_55ab5597e4b080f6aeb5eb23' (East US) has been added to subnet
Virtual Network 'VN_55ab5597e4b080f6aeb5eb23' for zone: 'East US' has been created.
Network Security Group 'NSG_55ab5597e4b080f6aeb5eb23' for zone: 'East US' has been created.
Virtual Network 'VN_55ab5597e4b080f6aeb5eb23' for zone: 'East US' has been created.
Network Security Group 'NSG_55ab5597e4b080f6aeb5eb23' (East US) has been added to subnet
Virtual Network 'VN_55ab5597e4b080f6aeb5eb23' for zone: 'East US' has been created.
Network Security Group 'NSG_55ab5597e4b080f6aeb5eb23' for zone: 'East US' has been created.
```

**Figure 8 – Network configuration**

- **Configuration Check**

To check the Azure network configuration of the project, use the following command:

```
> azure get_net_config -p project_code
```

The command returns the list of zones configured for the project and their status:

```
> azure get_net_config -p SAMPLE

<table>
<thead>
<tr>
<th>projectName</th>
<th>zoneName</th>
<th>configured</th>
</tr>
</thead>
<tbody>
<tr>
<td>SAMPLE</td>
<td>AZURE-JW</td>
<td>true</td>
</tr>
<tr>
<td>SAMPLE</td>
<td>AZURE-CUS</td>
<td>true</td>
</tr>
<tr>
<td>SAMPLE</td>
<td>AZURE-EUS</td>
<td>true</td>
</tr>
<tr>
<td>SAMPLE</td>
<td>AZURE-JE</td>
<td>true</td>
</tr>
<tr>
<td>SAMPLE</td>
<td>AZURE-NEU</td>
<td>true</td>
</tr>
<tr>
<td>SAMPLE</td>
<td>AZURE-NCUS</td>
<td>true</td>
</tr>
</tbody>
</table>
```

**Figure 9 – Configuration of Azure zones**
6.3 CSA – ADMINISTRATION CASES

6.3.1 CSA Zone Creation

A CSA virtualization zone is served by the CSA Portal and has to be configured with the CSA settings applicable to such CSA portal.

The typical CSA zone creation flow is as follows:

1. **Create a New Zone**
   
   ```
   csa add_zone
   --region
   --zone
   --csa-user
   --url
   --csp-user
   --organization
   --catalog
   [--location]
   [--hardware]
   ```

2. **Assign Orchestration Instance**
   
   ```
   orch assign
   --orch-id
   --zone
   [--unassign]
   [--billing]
   [--active]
   ```

3. **Set Cost Center**
   
   ```
   billing set_cost_center
   --zone
   --cost-center-name
   ```

4. **Add Shapes**
   
   ```
   csa add_shape
   --zone
   --shape
   --cpu
   --ram
   ```

*Figure 10 - CSA zone creation flow*

Each step is described in details below.

- **CSA Zone Creation**

To create a new CSA Orchestration zone, use the following command:

```
[csa_add_zone] [arguments]
```

The `csa add_zone` command uses the following arguments:

<table>
<thead>
<tr>
<th>Command Arguments</th>
<th>Description</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>-r, --region</code></td>
<td>Virtualization region in which the new zone is to be created</td>
<td>Yes</td>
</tr>
<tr>
<td><code>-z, --zone</code></td>
<td>Name of the virtualization zone to be created</td>
<td>Yes</td>
</tr>
<tr>
<td><code>-c, --csa-user</code></td>
<td>Name of the user to access the CSA portal</td>
<td>Yes</td>
</tr>
<tr>
<td><code>-u, --url</code></td>
<td>URL to the CSA portal which will manage this region</td>
<td>Yes</td>
</tr>
<tr>
<td><code>-s, --csp-user</code></td>
<td>Name of the CSP (Cloud Subscription Portal) user</td>
<td>Yes</td>
</tr>
<tr>
<td><code>-o, --organization</code></td>
<td>Name of the CSA Organization for this region</td>
<td>Yes</td>
</tr>
<tr>
<td><code>-a, --catalog</code></td>
<td>Name of the CSA Catalog for this region</td>
<td>Yes</td>
</tr>
<tr>
<td><code>-l, --location</code></td>
<td>Physical location of the new zone</td>
<td>No</td>
</tr>
<tr>
<td><code>--hardware</code></td>
<td>Flag setting the region as hardware</td>
<td>No</td>
</tr>
</tbody>
</table>

The `csa add_zone` command requires a password for execution. After the command is sent, the system prompts for the password. Therefore, this command cannot be sent in the 'quiet' mode.
**Orchestration Instance Assignment to CSA Zone**

When a new zone has been created, it has to be associated with an Orchestration instance for correct integration in the EPAM Orchestrator and proper service of the resources hosted in such zone.

To assign an Orchestration instance to the newly-created zone, use the following command:

```
orch assign [arguments]
```

The `orch assign` command uses the following arguments:

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>-o, --orch-id</td>
<td>Orchestrator instance ID</td>
<td>Yes</td>
</tr>
<tr>
<td>-z, --zone</td>
<td>Name of zone</td>
<td>Yes</td>
</tr>
<tr>
<td>-u, --unassign</td>
<td>Flag used to unassign a previously assigned zone</td>
<td>No</td>
</tr>
<tr>
<td>-b, --billing</td>
<td>Flag used to set the Orchestrator instance responsible for the zone billing</td>
<td>No</td>
</tr>
<tr>
<td>-a, --active</td>
<td>Flag used to set the Orchestrator instance as active</td>
<td>No</td>
</tr>
</tbody>
</table>

**Setting Cost Center for CSA Zone**

For the correct billing of the Cloud services for the projects used in the CSA zone, a cost center has to be assigned to it. To assign a cost center to a new CSA Orchestration zone, use the following command:

```
billing set_cost_center [arguments]
```

The `billing cost_center` command uses the following arguments:

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>-z, --zone</td>
<td>Name of zone</td>
<td>Yes</td>
</tr>
<tr>
<td>-c, --cost-center-name</td>
<td>Name of the cost center to be assigned to the zone</td>
<td>Yes</td>
</tr>
</tbody>
</table>
Command Example:

```
billing set_cost_center -z zone -c cost_center
```

- **Adding Shapes to CSA Zone**

A new zone is created with no VM shapes available in it. For the projects to be activated in a zone, shapes have to be added. Once shapes are added and configured, projects can be activated only with the shapes available in the zone. If a project requires a shape not available in the zone, the shape has to be added to the zone first.

To add a shape to a CSA zone, use the following command:

```
csa add_shape [arguments]
```

The `csa add_shape` command uses the following arguments:

<table>
<thead>
<tr>
<th>Command Arguments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Argument</td>
</tr>
<tr>
<td>-z, --zone</td>
</tr>
<tr>
<td>-s, --shape</td>
</tr>
<tr>
<td>-c, --cpu</td>
</tr>
<tr>
<td>-r, --ram</td>
</tr>
</tbody>
</table>

Command Example:

```
csa add_shape -z zone -s small -c 1 -r 1740
```

To add a shape to a particular project, use the `or2-set-shapes` Maestro CLI command.

### 6.3.1 Activating a Project in CSA

In CSA, projects are activated using just one Admin Utility command:

```
csa activate_project [arguments]
```

The `csa activate_project` command uses the following arguments:

<table>
<thead>
<tr>
<th>Command Arguments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Argument</td>
</tr>
<tr>
<td>-p, --project</td>
</tr>
<tr>
<td>-s, --shape</td>
</tr>
<tr>
<td>-z, --zone</td>
</tr>
</tbody>
</table>
### 6.3.2 Reimporting Instances to CSA

If CSA offerings have changes, instances have to be reimported to CSA, so that the offerings are properly updated and the updated data is applied. In such case, CSA subscriptions of instances are deleted and then restored again. After synchronization, the subscription data is updated.

The flow of instance reimporting is as follows:

- **Delete CSA Subscription**
  - Command: `csa del_subscript`  
  - Arguments:  
    - `-z, --zone` Virtualization zone  
    - `-i, --instance` Instance ID(s)  

- **Restore Instance to CSA**
  - Command: `csa restore_to_csa`  
  - Arguments:  
    - `-z, --zone` Virtualization zone  
    - `-i, --instance` Instance ID(s)  

- **Synchronize Subscriptions**
  - Command: `csa sync_from_csa`  
  - Arguments:  
    - `-z, --zone` Virtualization zone  
    - `-i, --instance` Instance ID(s)  

Each step is described in details below.

- **CSA Subscription Deletion**

  To delete the existing CSA subscriptions from instances in a certain zone, use the following command:

  ```
  csa del_subscript [arguments]
  ```

  The `csa del_subscript` command uses the following arguments:

<table>
<thead>
<tr>
<th>Command Arguments</th>
<th>Description</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>-z, --zone</code></td>
<td>Virtualization zone</td>
<td>Yes</td>
</tr>
<tr>
<td><code>-i, --instance</code></td>
<td>Instance ID(s)</td>
<td>Yes</td>
</tr>
</tbody>
</table>

  **Command Example:**

  ```
  csa del_subscript -z zone -i instance
  ```

---
• **Instance Restoring to CSA**

When the CSA subscriptions have been deleted, the instances have to be restored to CSA again for the updated subscriptions to apply. To restore instances to CSA, use the following command:

```
csa restore_to_csa [arguments]
```

The `csa restore_to_csa` command uses the following arguments:

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>-z, --zone</td>
<td>Virtualization zone</td>
<td>Yes</td>
</tr>
<tr>
<td>-i, --instance</td>
<td>Instance ID(s)</td>
<td>Yes</td>
</tr>
</tbody>
</table>

**Command Example:**

```
csa restore_to_csa -z zone -i instance
```

• **Subscription Synchronization**

After the subscriptions have been restored, their fields have to be synchronized between CSA and EPAM Orchestrator.

![Caution](warning_icon.png)

Before proceeding with synchronization, check that the updated subscriptions are active.

To synchronize CSA subscription fields, use the following command:

```
csa sync_from_csa [arguments]
```

The `csa sync_from_csa` command uses the following arguments:

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>-z, --zone</td>
<td>Virtualization zone</td>
<td>Yes</td>
</tr>
<tr>
<td>-i, --instance</td>
<td>Instance ID(s)</td>
<td>Yes</td>
</tr>
</tbody>
</table>

**Command Example:**

```
csa sync_from_csa -z zone -i instance
```
6.4 GOOGLE CLOUD PLATFORM – ADMINISTRATION CASES

This section describes the flows to be used in configuring infrastructure in the Google Cloud Platform as access to it.

6.4.1 Google Account Configuration

Google Cloud Platform is available for all users with Google accounts. To provide access to Google Cloud Platform, use the API Manager to allow access from your Google account.

When the access has been granted, create your first project on the Google console. This project will be the base project for all subsequent ones and the billing account, the API access permissions and IAM user administration will be associated with this project.

For your base project, different credentials (OAuth 2.0 Client IDs of the Other type) need to be created for two Google account entities in the database (see below). Before generating the credentials, create the OAuth Consent Screen (fill in only the required fields).

For the base project, use activate the following APIs required for Orchestrator operation using the API Manager on the Google console:

- Google Cloud Billing API
- Admin SDK
- Google Compute Engine API
- Google Cloud Resource Manager API

6.4.2 Google Account Entity in Orchestrator Database

To enable working with Google Cloud Platform, two account entities should be generated – for using the Compute API and for using the Admin Directory API (IAM user administration). Such organization allows separating the account management depending on the operation type. Each entity requires a separate set of credentials to be created in the base project on the Google console.

Important: do not create credentials for other projects in the account, it will block Google API requests performance by the Orchestrator under the project.

To create a Google account entity in the Orchestrator database use the following command:

```
google setup_account [arguments]
```

The ‘google setup_account’ command uses the following arguments:

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>-u, --username</td>
<td>Account username (<a href="mailto:Name.Lastname@gmail.com">Name.Lastname@gmail.com</a>)</td>
<td>Yes</td>
</tr>
<tr>
<td>-i, --client-id</td>
<td>Client ID*</td>
<td>Yes</td>
</tr>
<tr>
<td>-p, --purpose</td>
<td>Purpose of the account. Allowed values: COMPUTE, ADMIN_DIRECTORY</td>
<td>Yes</td>
</tr>
<tr>
<td>-a, --admin-project-id</td>
<td>Admin project ID. Required for COMPUTE accounts</td>
<td>No</td>
</tr>
<tr>
<td>-b, --billing-account-id</td>
<td>Billing project ID. Required for COMPUTE accounts</td>
<td>No</td>
</tr>
<tr>
<td>-d, --billing-dataset-name</td>
<td>BigQuery billing dataset name</td>
<td>No</td>
</tr>
</tbody>
</table>

*For the -i (--client-id) parameter value, use the value from the credentials earlier generated in the base account.
Command Example:

google setup_account -u username -i client_id -p ADMIN_DIRECTORY

Figure 11 - Google account setup

The command is executed with simultaneous interactive operations in the browser. Enter the clientSecret from the generated credentials into the API Manager of the base project, then, after the clientSecret has been successfully validated, the Admin Utility console will display a link. Paste the link into your browser and copy the token which will be displayed. Paste the token into your Admin Utility console.

After the account creation, run the `mongo refresh_config` command in the DB utility, otherwise Orchestrator may work incorrectly.

How Google API authorization works:

- Authorization is performed by the accessToken issued by Google and valid for 1 hour
- The Orchestrator performs authorization by processing the 401 response code received for its request to the Google API
- When the 401 code is received, the Orchestrator sends a request for a token using the refreshToken and clientId. Note that there is a limit of 600 accessTokens to be used simultaneously in Google.
- refreshToken, clientId and accessToken are stored in the same document in the GoogleAccounts collection in the database (see https://developers.google.com/identity/protocols/OAuth2 for details).

### 6.4.3 Adding Google Zones

Google Cloud, unlike AWS, is project-centered, which causes certain specifics of Google projects and zones organization and processing by Orchestrator.

The zone configuration flow is as follows:
Each step is described in details below.

- **Retrieving Google Zones**

Before integrating Google zones into Orchestrator, run the following command to see the list of zones available via the Google console:

```
google list_zones
```

The ‘google list_zones’ command uses no arguments.

If a zone has already been added to Orchestrator, it will show its name from the Orchestrator database in the ‘name’ field. For zones not yet added to Orchestrator, the ‘name’ field will show “untracked”.

- **Adding Google Zones**

Run the following command to add the zone:

```
google add_zone [arguments]
```

The ‘google add_zone’ command uses the following arguments:

<table>
<thead>
<tr>
<th>Command Arguments</th>
<th>Description</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>-r, --region</td>
<td>Virtualization region*</td>
<td>Yes</td>
</tr>
<tr>
<td>-z, --zone</td>
<td>Virtualization zone**</td>
<td>Yes</td>
</tr>
<tr>
<td>-a, --account-id</td>
<td>Google account ID. Format example: account-91b5e7ec</td>
<td>Yes</td>
</tr>
<tr>
<td>-Z, --google-zone-name</td>
<td>Google zone name. Format example: us-central1-a. For the complete list of zones, visit the Regions and Zones page</td>
<td>Yes</td>
</tr>
<tr>
<td>-l, --location</td>
<td>Location (for example, North Europe)</td>
<td>No</td>
</tr>
<tr>
<td>--disable-billing-mix-mode</td>
<td>The flag defining whether the zone supports billing mode</td>
<td>No</td>
</tr>
<tr>
<td>-A, --aws-nearest-zone</td>
<td>AWS nearest zone (required for autoconfiguration)***</td>
<td>No</td>
</tr>
</tbody>
</table>

* The region must be added manually, as there is no Admin Utility command for that purpose. The region is to be added once before adding the first Google zone, and afterwards the EPAM-GOOGLE region will appear in the Orchestrator’s Regions collection in the database.

** The zone name to be used in Orchestrator

*** The nearest AWS zone is specified for the autoconfiguration data to be retrieved from the AWS S3 bucket in the zone nearest to the Google zone

Command Example:

```
google add_zone -r EPAM-GOOGLE -z zone_name -a google_account -Z us-central1-a
```

- **Editing Google Zones**

If you need to change or delete the AWS zone associated with the Google zone, use the ‘google edit_zone’ command:
The `google edit_zone` command uses the following arguments:

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>-z, --zone</td>
<td>Virtualization zone</td>
<td>Yes</td>
</tr>
<tr>
<td>-a, --aws-nearest-zone</td>
<td>AWS nearest zone (required for autoconfiguration). Send 'null' to remove the value.</td>
<td>Yes</td>
</tr>
</tbody>
</table>

Command Example:
```
[google edit_zone -z zone_name -a null]
```

### 6.4.4 Project Activation in Google Cloud

In Google Cloud, projects are activated by their abbreviations in UPSA. To activate a project, use the following command:
```
[google activate_project [arguments]
```

The `google activate_project` command uses the following arguments:

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>-p, --project</td>
<td>Project abbreviation in UPSA</td>
<td>Yes</td>
</tr>
<tr>
<td>-s, --shape</td>
<td>Name of the shape to be activated for the project. For several shapes, repeat the parameter: -s shape1, -s shape2, -s shapeN</td>
<td>Yes</td>
</tr>
<tr>
<td>-a, --auto-configuration-disabled</td>
<td>Flag defining that auto-configuration is disabled or enabled</td>
<td>No</td>
</tr>
<tr>
<td>-f, --fake-project</td>
<td>Fake project</td>
<td>No</td>
</tr>
<tr>
<td>-e, --existing-project-id</td>
<td>Existing Google project ID to use*</td>
<td>No</td>
</tr>
<tr>
<td>-z, --zone</td>
<td>Virtualization zone**</td>
<td>No</td>
</tr>
<tr>
<td>--all</td>
<td>All zones**</td>
<td>No</td>
</tr>
</tbody>
</table>

* The `existing-project-id` parameter is used to continue project activation when a project created in the Google console manually should be associated with the project representation in EPAM Orchestrator and UPSA

** Send either a specific zone or '--all'. If you activate a project with the '--all' flag, it will be activated in all available Google Cloud zones

Command Example:
```
[google activate_project -p project_id -s small -s mini --all]
```
Figure 12 - Project activation in Google Cloud

The command execution is interactive and requires activation of Google Compute Engine API for the newly-created project. This is done via the Google console and enables EPAM Orchestrator to send requests to Google Cloud.

The base project should not be associated with any UPSA project.

If during project activation, an error message saying that billing and Google Cloud Billing API are not activated for the base project, make sure you have completed all steps of base project configuration flow.

During activation, the following configuration actions are performed:

- the project is automatically connected to a Billing account common for the entire Google account
- a common network is created for the project allowing all project instances to access each other via an internal network
- necessary internal subnets are created (one for each Google availability zone)
- firewall rules for subnet IPs specified in securityGroups in the OrchestrationSettings collection in the database (orchestrator-default-firewall) are established.

6.4.5 Adding Images in Google Cloud

• Retrieving Google Public Images

In Google Cloud, public images are associated with public projects listed on the Images page. A separate project corresponds to each OS type. To view the images available for a specific project, use public project names and run the following Admin Utility command:

The ‘google list_images’ command uses the following arguments:

<table>
<thead>
<tr>
<th>Command Arguments</th>
<th>Description</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>-p, --project-id</td>
<td>Google project ID to retrieve images from. For example, centos-cloud or coreos-cloud</td>
<td>Yes</td>
</tr>
<tr>
<td>-d, --deprecated</td>
<td>Flag defining whether deprecated images are to be included</td>
<td>No</td>
</tr>
</tbody>
</table>
Command Example:

```
google list_images -p project_id
```

- **Adding Google Images**

To add an image, use the following command:

```
google add_image [arguments]
```

The ‘google add_image’ command uses the following arguments:

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>-i, --image-id</td>
<td>Image ID. For example, Ubuntu14.04_64-bit*</td>
<td>Yes</td>
</tr>
<tr>
<td>-N, --google-image-name</td>
<td>Google image name**</td>
<td>Yes</td>
</tr>
<tr>
<td>-P, --google-project-id</td>
<td>Google image project ID**</td>
<td>Yes</td>
</tr>
<tr>
<td>-d, --description</td>
<td>Image description</td>
<td>Yes</td>
</tr>
<tr>
<td>-t, --os-type</td>
<td>Type of the operating system (Windows, Linux)</td>
<td>Yes</td>
</tr>
<tr>
<td>-z, --zone</td>
<td>Virtualization zone</td>
<td>Yes</td>
</tr>
<tr>
<td>-g, --group</td>
<td>Image group (public, enterprise)</td>
<td>Yes</td>
</tr>
<tr>
<td>-u, --username</td>
<td>Default SSH user***</td>
<td>Yes</td>
</tr>
</tbody>
</table>

* Use image name corresponding to the common EPAM Orchestrator image mapping
** Specify Google image name and the ID of the public project from which the image will be retrieved
*** Specify the login under which the instance will be accessed with an SSH key

Command Example:

```
google add_image -i image_ID -N google_image_name -P google_project_id -d image_description -t os_type -z zone -g group -u username
```

### 6.4.6 Custom Image Creation in Google Cloud

Google Cloud creates custom images from system volume snapshots, therefore, storing a machine image with attached volumes requires a series of actions. Here we recommend creating tasks and subtasks as using task processing tools.

A task of creating an image from an instance with attached volumes consists of the following subtasks:

- Creation of a system volume snapshot
- Creation of a volume from the snapshot
- Creation of an image from the volume
- Deletion of the created volume
- Creation of snapshots of the attached volumes (can run simultaneously with system volume operations)

As the result, there is a project Google image and snapshots of the attached volumes.

Resource creation from such image also involves several subtasks:

- Instance run from the custom image
- Creation of volumes from the stored snapshots and their attachment to the launched instance

Such custom images can only be used in EPAM Orchestrator, because if an instance is launched from the custom image via the Google console, the instance will be started with no non-system volumes attached. The image size includes not only system volume data but also the data of the attached volume snapshots.

Note that the snapshots of attached volumes are part of the image and if either of them is deleted via the Google console, the snapshot size will be deducted from the image size and the instance launched from the image will not have the volume corresponding to the deleted snapshot.

Also, in Google Cloud resources belong to a project, therefore, creation of a custom project image causes duplicates of the Google MachineImage entity for all Google zones in the Orchestrator database (the MachineImages collection). Similarly, when an image is deleted, duplicates for all Google zones are deleted as well.

### 6.4.7 Public and Static IPs

By default, Google Cloud assigns public IPs to instances upon launch, however, these IPs may change with each start-stop operation. Google documentation refers to them as to “ephemeral” external IP addresses.

For cases when it is important that an instance keeps the same IP address, Google Cloud supports reserved static IPs.

In EPAM Orchestrator, a static IP is allocated with the following sequence of Maestro CLI commands:

1. Allocation of a static IP to the project and region:

```bash
or2alsip -p project -r region
```

2. Assignment of a static IP to the instance:

```bash
or2assip -p project -r region -i instance_id -a ip_address
```

Static IP assignment is performed as the following series of subtasks:

- Removal of the default public IP from the instance
- Waiting for the default public IP removal to complete
- Assignment of the configuration of public access to the instance with a static IP
- Waiting for the static IP assignment to complete

Due to the complexity of the flow, the command performance may take longer than with other cloud providers. Also, the probability of failure is higher.

The reverse process of a static IP disassociation is performed with the following Maestro CLI command:

```bash
or2dissip -p project -r region -a ip_address
```

IP disassociation is also a process involving several subtasks:

- Removal of the static IP from the instance
- Waiting for the static IP removal to complete
- Assignment of the configuration of public access to the instance with the default public IP
- Waiting for the default public IP assignment to complete
**6.4.8 Volumes in Google Cloud**

In Google Cloud, system volumes are created together with the corresponding instances and receive IDs fully matching those of the instance. At the same time, EPAM Orchestrator will show system volumes with have their unique IDs. For non-system volumes, the Google console will show names matching such volume IDs in EPAM Orchestrator.

Attach/detach volume operations are fully supported.

**6.4.9 Google IAM Users**

EPAM Orchestrator distinguishes two main types of users – temporary users and permanent IAM users. This system requires certain adaptation for Google Cloud, as Google has no such classification.

- **Temporary Users**

  Temporary access to Google Management Console is granted with the following command:

  ```bash
  or2goomc
  ```

  In this case, temporary access to the Google console is granted via a special user pool (GoogleAccountUsers collection in the database), the names of such users always start with SpecialEPM-CSUP*. When the or2goomc command is sent, EPAM Orchestrator searches for a free SpecialEPM-CSUP* user in the pool, changes its status to IN_USE and allocates a new password to be provided to the end user via email.

  All temporary access permissions are reset every day at 12:00 a.m.

  **To add a temporary user to the pool**, use the following Admin Utility command:

  ```bash
  [google add_temp_access_user [arguments]]
  ```

  The ‘google add_temp_access_user‘ command uses the following arguments:

<table>
<thead>
<tr>
<th>Command Arguments</th>
<th></th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>Argument</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>-u, --username</td>
<td>Email of the Google user</td>
<td>Yes</td>
</tr>
</tbody>
</table>

  **Command Example:**

  ```bash
  [google add_temp_access_user -u user_email]
  ```

  **To retrieve the list of all existing temporary users**, use the following command:

  ```bash
  [google list_temp_access_users]
  ```

  The ‘google list_temp_access_users‘ command uses no arguments.
• **Ordinary IAM Users**

The number of IAM users is limited to 100 per project.

To add a Google IAM user, use the following command:

```
google add_iam_user [arguments]
```

The `google add_iam_user` command uses the following arguments:

<table>
<thead>
<tr>
<th>Command Arguments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Argument</td>
</tr>
<tr>
<td>-p, --project</td>
</tr>
<tr>
<td>-e, --email</td>
</tr>
<tr>
<td>-r, --creation-reason</td>
</tr>
</tbody>
</table>

Command Example:

```
google add_iam_user -p project_ID -e user_email -r creation_reason
```

To retrieve the list of all existing ordinary IAM users, use the following command:

```
google list_iam_users [arguments]
```

The `google list_iam_users` command uses the following arguments:

<table>
<thead>
<tr>
<th>Command Arguments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Argument</td>
</tr>
<tr>
<td>-p, --project</td>
</tr>
</tbody>
</table>

Command Example:

```
google list_iam_users -p project_ID
```

Additionally, you can use the following Maestro CLI command to view the list of all ordinary IAM users:

```
or2iam [arguments]
```

The `google list_iam_users` command uses the following arguments:

<table>
<thead>
<tr>
<th>Command Arguments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Argument</td>
</tr>
<tr>
<td>-a, --action</td>
</tr>
<tr>
<td>-e, --email</td>
</tr>
<tr>
<td>-p, --project</td>
</tr>
<tr>
<td>--reason</td>
</tr>
<tr>
<td>-t, --type</td>
</tr>
</tbody>
</table>
System IAM Users

All system IAM users are stored in the GoogleAccounts collection in the Compute account (systemIamUserName field). The google list_iam-users Admin Utility command retrieves all IAM users, both ordinary and system, while or2iam Maestro CLI command lists only ordinary users.

System IAM uses are created with the ‘owner’ permissions while ordinary IAM users and temporary users have the ‘editor’ permissions which are narrower than ‘owner’.

To create a system IAM user, use the following command:

google add_account_system_username [arguments]

The ‘google list_iam_users’ command uses the following arguments:

<table>
<thead>
<tr>
<th>Command Arguments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Argument</td>
</tr>
<tr>
<td>-u, --username</td>
</tr>
</tbody>
</table>

Command Example:

google add_account_system_username -u username

All IAM users operations are synchronized with the Google console at 3:15 a.m. UTC.

6.4.10 Other

Init Scripts

In Google Cloud, the init script runs with each OS start, therefore, for Google Cloud special init scripts have been created and stored in the OrchestrationSettings collection of the database in the googleLinuxNativeScript and 'googleWindowsNativeScript' fields.

Interactive Operations

During interactive operations requiring simultaneous actions in the browser, Admin Utility may sometimes return invalid hyperlinks, especially, for API activation. We recommend searching for the correct hyperlinks in the API Manager if the Admin Utility returns an invalid link repeatedly.
6.5 OPENSTACK – ADMINISTRATION CASES

6.5.1 OpenStack Controller Hosts

OpenStack virtualizer is controller-based, with each controller corresponding to an OpenStack zone in EPAM Orchestrator. The controller by default contains an admin project (tenant) and is intended to create a network to host all instances launched by the controller (that is, within the corresponding EPAM Orchestrator zone). The controller also creates admin credentials (login/password) for the Orchestrator to access OpenStack API.

Direct access to the controller via the native UI, in addition to the login/password combination, may also require the domain which always has the “default” value.

6.5.2 OpenStack Hosts and Host Aggregates

Each controller has a number of hosts used to allocate resources to launched instances and created storage volumes. Each host is assigned an availability zone which is used when instances and system or attached volumes are created on the same host. There may be cases when a volume cannot be attached to a running instance if the host resources are insufficient. In this case, OpenStack prevents volume creation (contrary to CSA where the instance can move to a different host together with all related resources). Hosts may differ not only by capacity but also by the supported storage type (SSD/HDD).

Hosts are joined into aggregates by the supported storage type (SSD support information is included in the aggregate metadata). Aggregates are used to filter resource creation requests depending on the storage type. The filter also acts as load balancer distributing the load among the hosts within aggregates depending on the current utilization rate.

6.5.3 OpenStack Zone Management

The typical OpenStack zone creation flow is as follows:

![Diagram of OpenStack zone creation flow]

Each step is described in details below.
**Zone Creation**

An OpenStack zone is the controller’s entity in EPAM Orchestrator storing the controller data used by EPAM Orchestrator (link to the controller, access information, tenant information).

To create a new OpenStack zone, use the following command:

```
open_stack add_zone [arguments]
```

The `open_stack add_zone` command uses the following arguments:

<table>
<thead>
<tr>
<th>Command Arguments</th>
<th>Description</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>-r, --region</code></td>
<td>Virtualization region</td>
<td>Yes</td>
</tr>
<tr>
<td><code>-z, --zone</code></td>
<td>Name of the virtualization zone to be used in EPAM Orchestrator (case-sensitive)</td>
<td>Yes</td>
</tr>
<tr>
<td><code>-u, --auth-url</code></td>
<td>OpenStack authentication URL for domain authorization to resources. The authorization server is one of the OpenStack services and can be reached at the [http:&lt;controller_IP&gt;:5000/v1] endpoint</td>
<td>Yes</td>
</tr>
<tr>
<td><code>-l, --location</code></td>
<td>Location (e.g. North Europe)</td>
<td>No</td>
</tr>
<tr>
<td><code>-c, --counter</code></td>
<td>Instance start counter (used for instance name generation)</td>
<td>Yes</td>
</tr>
<tr>
<td><code>--assign</code></td>
<td>Assigns zone to the currently active node</td>
<td>No</td>
</tr>
<tr>
<td><code>-a, --admin-name</code></td>
<td>Admin name to be used for API calls</td>
<td>Yes</td>
</tr>
<tr>
<td><code>-t, --admin-tenant</code></td>
<td>Admin tenant</td>
<td>Yes</td>
</tr>
<tr>
<td><code>-m, --networking-mode</code></td>
<td>Networking mode. Allowed values: [AUTO, MANUAL]</td>
<td>Yes</td>
</tr>
<tr>
<td><code>--dns, --dns-server</code></td>
<td>DNS server on which VMs will be registered. Several DNS servers can be specified.</td>
<td>Yes</td>
</tr>
<tr>
<td><code>-n, --network-id</code></td>
<td>ID of the network created on the controller earlier</td>
<td>Yes</td>
</tr>
<tr>
<td><code>--rn, --region-name</code></td>
<td>OpenStack region name (to be used when a host serves several regions. EPAM Orchestrator zone entity will be associated with the specified region)</td>
<td>No</td>
</tr>
<tr>
<td><code>-d, --docker-only</code></td>
<td>Docker only (to be used when the zone is a dedicated zone for Docker/Kubernetes services deployed on CoreOS)</td>
<td>No</td>
</tr>
<tr>
<td><code>--mtp</code></td>
<td>Servicing host for the moveToProject command</td>
<td>No</td>
</tr>
</tbody>
</table>

The `open_stack add_zone` command requires a password for execution. After the command is sent, the system prompts for the password. Therefore, this command cannot be sent in the ‘quiet’ mode.

**Command Example:**

```
open_stack add_zone -z zone_name -u http://<server_hostname:port>/v2.0 -r region -l location -c 001 -a admin -t admin -m networking_mode -dns dns_server
```
The `mongo refresh_config` command of the DB Utility does not hide admin user credentials and URLs for authorization and additional servers (this does not refer to zones migrated from CSA).

- **Zone Editing**

To edit an OpenStack zone, use the following command:

```
open_stack edit_zone [arguments]
```

The `open_stack edit_zone` command uses the following arguments:

<table>
<thead>
<tr>
<th>Command Arguments</th>
<th>Argument</th>
<th>Description</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>-z, --zone</td>
<td>Name of the virtualization zone to be used in EPAM Orchestrator (case-sensitive)</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>-s, --strategy</td>
<td>Zone update strategy. Allowed values: [DESCRIBE, PUSH_NOTIFICATIONS, PUSH_NOTIFICATIONS_WITH_DESCRIBE]</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>-d, --disk-drive</td>
<td>Default storage type. Allowed values: [HDD, SSD]. We recommend using the type supported by the majority of hosts.</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>-r, --resource-placing-policy</td>
<td>Resource placing policy. Allowed values: [DEFAULT, SAME_HOST]. Defines whether instances should be placed on the same host as the volumes attached to them. The SAME_HOST value is the preferred setting.</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>-c, --create-volume-snapshots</td>
<td>Defines whether the operation of volume snapshot creation is supported</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>-t, --storage-threshold</td>
<td>Storage capacity threshold. Must be in the range of [0, 100]</td>
<td>No</td>
</tr>
</tbody>
</table>

**Command Example:**

```
open_stack edit_zone -z zone_name -s DESCRIBE -d SSD -r SAME_HOSTS -t 100
```

- **Retrieving the List of OpenStack Zones**

To retrieve the list of OpenStack zones existing in EPAM Orchestrator, use the following command:

```
open_stack get_zones
```

The `open_stack get_zones` command uses no arguments.

- **Orchestration Instance Assignment to OpenStack Zone**

When a new zone has been created, it has to be associated with an Orchestration instance for correct integration in the EPAM Orchestrator and proper service of the resources hosted in such zone.

To assign an Orchestration instance to the newly-created zone, use the following command:

```
orch assign [arguments]
```

The `orch assign` command uses the following arguments:
<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>-o, --orch-id</td>
<td>Orchestrator instance ID</td>
<td>Yes</td>
</tr>
<tr>
<td>-z, --zone</td>
<td>Name of zone</td>
<td>Yes</td>
</tr>
<tr>
<td>-u, --unassign</td>
<td>Flag used to unassign a previously assigned zone</td>
<td>No</td>
</tr>
<tr>
<td>-b, --billing</td>
<td>Flag used to set the Orchestrator instance responsible for the zone billing</td>
<td>No</td>
</tr>
<tr>
<td>-a, --active</td>
<td>Flag used to set the Orchestrator instance as active</td>
<td>No</td>
</tr>
</tbody>
</table>

Command Example:

```
orch assign -z zone_name -o instance_id -a
```

**Adding Shapes to OpenStack Zone**

A new zone is created with no VM shapes available in it. For the projects to be activated in a zone, shapes have to be added. Once shapes are added and configured, projects can be activated only with the shapes available in the zone. If a project requires a shape not available in the zone, the shape has to be added to the zone first.

OpenStack refers to shapes as “flavors” and distinguishes them not only by the CPU/RAM combination, but also by the storage type (SSD or HDD), OS type (Linux, Windows) and the system volume size.

Shape names are combined from several parameters, one of them always being the EPAM Orchestrator shape name. The name may also contain the volume size and the indicator of the storage type (ssd) or operating system (lin for Linux).

When an instance is launched with a certain shape, the flavor selection is influenced by the specified shape, image and the zone configuration. The image defines the operating system, and if the storage type is specified – the corresponding flavor will be used, otherwise, a default flavor will be selected in accordance with the zone configuration.

To add a shape to an OpenStack zone, use the following command:

```
open_stack add_shapes [arguments]
```

The `open_stack add_shapes` command uses the following arguments:

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>-z, --zone</td>
<td>Name of virtualization zone where shapes are to be added</td>
<td>Yes</td>
</tr>
<tr>
<td>-s, --shape</td>
<td>Shape name. For several shapes, repeat the parameter: -s SHAPE1 -s SHAPE2. If not specified, all available shapes will be added. Only the shapes not yet added to the zone will be added by the command</td>
<td>No</td>
</tr>
<tr>
<td>-d, --drive-type</td>
<td>Disk drive type. For several disk drives, repeat the parameter: -d SSD -d HDD. If not specified, all available drives will be added</td>
<td>No</td>
</tr>
</tbody>
</table>
-t, --os-type
  Operating System type (e.g. WINDOWS, LINUX). For several OS types, repeat the parameter. If not specified, all available OS types will be added. No

--see
  Print the shapes to be added No

Note that the --see flag blocks the shape addition operation and only prints the list of shapes to be added.

Command Example:

```
open_stack add_shapes -z zone -s small -t linux -d hdd
```

**Shape Management in OpenStack**

To view the default shapes existing in OpenStack, use the following command

```
open_stack get_default_shapes [arguments]
```

The `open_stack get_default_shapes` command uses no arguments.

To view the shapes available in a certain OpenStack zone, use the following command

```
open_stack get_shapes [arguments]
```

The `open_stack get_shapes` command uses the following arguments.

<table>
<thead>
<tr>
<th>Command Arguments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Argument</td>
</tr>
<tr>
<td>-z, --zone</td>
</tr>
</tbody>
</table>

To delete shapes from a certain OpenStack zone, use the following command

```
open_stack delete_shapes [arguments]
```

The `open_stack delete_shapes` command uses the following arguments.

<table>
<thead>
<tr>
<th>Command Arguments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Argument</td>
</tr>
<tr>
<td>-z, --zone</td>
</tr>
<tr>
<td>-f, --flavor</td>
</tr>
</tbody>
</table>

When flavors are updated by the OpenStack controller, their identifiers change which requires flavor updates in EPAM Orchestrator. This is done by means of the ‘revision’ field in the flavor collection in the EPAM Orchestrator database. The field is updated with the flavor update date; in this case, the flavor will still be used to describe the existing resources but will not be used to create new ones. For a new flavor identifier, the OpenStack controller generates a new document with the ‘revision’ field set to ‘latest’. All new instances launched by EPAM Orchestrator will use the flavors with ‘revision’ set to ‘latest’. 
• **Adding Machine Images to OpenStack Zone**

To add machine images which will be available in the OpenStack zone, use the following command:

```
open_stack add_image [arguments]
```

The `open_stack add_image` command uses the following arguments:

<table>
<thead>
<tr>
<th>Command Arguments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Argument</td>
</tr>
<tr>
<td>-z, --zone</td>
</tr>
<tr>
<td>-i, --image-id</td>
</tr>
<tr>
<td>-o, --open-stack-image-id</td>
</tr>
<tr>
<td>-d, --description</td>
</tr>
<tr>
<td>-g, --group</td>
</tr>
<tr>
<td>-u, --username</td>
</tr>
<tr>
<td>-t, --os-type</td>
</tr>
</tbody>
</table>

Command Example:

```
open_stack add_image -i Ubuntu14.04_64-bit -o openstack_image -d "Ubuntu14.04 64-bit LTS" -z zone_name -g PUBLIC -u user -t linux
```

• **Machine Image Management in OpenStack**

When an image is updated on the OpenStack controller, the image identifier changes. Identifiers synchronization between the OpenStack controller and EPAM Orchestrator is performed by schedule at 1:00 a.m. UTC. Identifiers can also be updated via jmx on the EPAM Orchestrator Web UI or with the following Admin Utility command:

```
open_stack set_image_id [arguments]
```

The `open_stack set_image_id` command uses the following arguments:

<table>
<thead>
<tr>
<th>Command Arguments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Argument</td>
</tr>
<tr>
<td>-z, --zone</td>
</tr>
<tr>
<td>-n, --name</td>
</tr>
<tr>
<td>-i, --id</td>
</tr>
</tbody>
</table>

Command Example:

```
open_stack set_image_id -n Ubuntu14.04_64-bit -z zone_name -i image_id
```
To retrieve the list of images existing in EPAM Orchestrator for the specified zone, use the following command:

```
open_stack get_images [arguments]
```

The `open_stack get_images` command uses the following arguments:

<table>
<thead>
<tr>
<th>Command Arguments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Argument</td>
</tr>
<tr>
<td>-z, --zone</td>
</tr>
</tbody>
</table>

**Command Example:**

```
open_stack get_images -z zone_name
```

To delete an image from a zone by its OpenStack ID, use the following command:

```
open_stack delete_image [arguments]
```

The `open_stack delete_image` command uses the following arguments:

<table>
<thead>
<tr>
<th>Command Arguments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Argument</td>
</tr>
<tr>
<td>-z, --zone</td>
</tr>
<tr>
<td>-i, --image-id</td>
</tr>
</tbody>
</table>

**Command Example:**

```
open_stack delete_image -z zone_name -i image_id
```

**Custom Image Management in OpenStack**

In OpenStack, an image does not store data of the attached volumes. Therefore, in EPAM Orchestrator, the MachineImages collection in the database stores the data of the snapshots of attached volumes together with the image data. The snapshot data is stored in the `volumeSnapshotInfoSet` field.

Note that the storage sequence influences the sequence of volume creation and attachment to the instance launched from the custom image.

Also, to enable creation of custom images from instances with attached volumes, run the `open_stack edit_zone` command with the `-c, --create-volume-snapshots` flag, as by default this option is disabled.

**Push Notifications Configuration**

OpenStack supports the Push Notifications functionality allowing EPAM Orchestrator to respond to changes and to perform synchronizations quicker.

Push notifications are sent via the RabbitMQ messenger service. The OpenStack controller sends messages about various events related to resource state changes to the RabbitMQ server. In its turn, EPAM
Orchestrator can monitor pre-defined message queues storing messages from the OpenStack controller and thus getting the change data immediately.

If push notifications are disabled, synchronization is performed by scheduled describe requests to the server. The request frequency depends on the resource (on the average, every 2-5 minutes).

Push notifications are configured with the following command:

```
open_stack notific_config [arguments]
```

The `open_stack notific_config` command uses the following arguments:

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>-z, --zone</td>
<td>Virtualization zone</td>
<td>Yes</td>
</tr>
<tr>
<td>--host</td>
<td>OpenStack Rabbit host*</td>
<td>No</td>
</tr>
<tr>
<td>-p, --port</td>
<td>OpenStack Rabbit port*</td>
<td>No</td>
</tr>
<tr>
<td>-m, --mport</td>
<td>OpenStack Rabbit Management port**</td>
<td>No</td>
</tr>
<tr>
<td>-v, --vhost</td>
<td>OpenStack Rabbit virtual host***</td>
<td>No</td>
</tr>
<tr>
<td>-u, --username</td>
<td>OpenStack Rabbit username under which EPAM Orchestrator will monitor messages on the RabbitMQ host</td>
<td>No</td>
</tr>
<tr>
<td>-r, --reply-timeout</td>
<td>Reply timeout (in milliseconds)****</td>
<td>No</td>
</tr>
<tr>
<td>-s, --shutdown-timeout</td>
<td>Shutdown timeout (in milliseconds)*****</td>
<td>No</td>
</tr>
<tr>
<td>-n, --min-threads</td>
<td>Minimum number of threads to monitor notifications******</td>
<td>No</td>
</tr>
<tr>
<td>-x, --max-threads</td>
<td>Maximum number of threads to monitor notifications******</td>
<td>No</td>
</tr>
<tr>
<td>--nova</td>
<td>Custom exchange name for Nova service*****</td>
<td>No</td>
</tr>
<tr>
<td>--cinder</td>
<td>Custom exchange name for Cinder service*****</td>
<td>No</td>
</tr>
<tr>
<td>--glance</td>
<td>Custom exchange name for Glance service*****</td>
<td>No</td>
</tr>
</tbody>
</table>

* RabbitMQ host and port for push notifications
** Currently not used as the management plugin is usually disabled on RabbitMQ server deployed together with the OpenStack controller
*** Usually the default virtual host is used ("/")
**** Not used
***** The shutdown timeout setting is used to terminate the amqp-listeners container
****** The recommended thread number is up to 30. The minimum setting defines the constant number of notification handling threads while the maximum setting limits their number
******* Custom exchange key names for the corresponding OpenStack services on RabbitMQ

The `open_stack notific_config` command requires a password for execution. After the command is sent, the system prompts for the password. Therefore, this command cannot be sent in the ‘quiet’ mode.
**Command Example:**

```bash
open_stack notic_config -z zone_name --host <server_hostname> -p 5672 -v "/" -u readonly -s 30000 -n 2 -x 6
```

Before enabling push notifications, run the `open_stack edit_zone` command with the `-s, --strategy` parameter set to "PUSH_NOTIFICATIONS_WITH_DESCRIBE", as this setting will not disable scheduled requests for resource synchronization with EPAM Orchestrator which is a more fault-tolerant configuration.

### Enabling Notifications

To enable the notification service for the OpenStack zone, use the following command:

```bash
open_stack notific_switcher [arguments]
```

The `open_stack notific_switcher` command uses the following arguments:

<table>
<thead>
<tr>
<th>Command Arguments</th>
<th>Argument</th>
<th>Description</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>-z, --zone</td>
<td>Virtualization zone</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>-e, --enable</td>
<td>Enable notifications</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>-d, --disable</td>
<td>Disable notifications</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>-q, --queues</td>
<td>Configure OS RabbitMQ queues</td>
<td>No</td>
<td></td>
</tr>
</tbody>
</table>

Before enabling or disabling the notification service, configure its settings using the `open_stack notific_config` command.

**Command Example:**

```bash
open_stack notific_switcher -e -q -z zone_name
```

### Pricing Policy Creation for OpenStack Zone

Billing of the Cloud services is based on a pricing policy.

To view an existing pricing policy, use the following command:

```bash
pricing_policy get [--target]
```

where the `--target` parameter indicates how the data is to be delivered. Valid values: `ssh_console, email`.

To update the pricing policy, use the following command:

```bash
pricing_policy update [arguments]
```
The 'pricing_policy update' command uses the following arguments:

<table>
<thead>
<tr>
<th>Command Arguments</th>
<th>Description</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>-f, --file</td>
<td>File name*</td>
<td>Yes</td>
</tr>
<tr>
<td>-w, --skip-warnings</td>
<td>Skip warnings</td>
<td>No</td>
</tr>
<tr>
<td>-c, --skip-changes</td>
<td>Skip changes</td>
<td>No</td>
</tr>
</tbody>
</table>

*This command requires file upload.

Command Example:
```
pricing_policy update -f file_name
```

- **Setting Cost Center for OpenStack Zone**

For the correct billing of the Cloud services for the projects used in the OpenStack zone, a cost center has to be assigned to it. To assign a cost center to a new OpenStack zone, use the following command:

```
billing set_cost_center [arguments]
```

The 'billing cost_center' command uses the following arguments:

<table>
<thead>
<tr>
<th>Command Arguments</th>
<th>Description</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>-z, --zone</td>
<td>Name of zone</td>
<td>Yes</td>
</tr>
<tr>
<td>-c, --cost-center-name</td>
<td>Name of the cost center to be assigned to the zone</td>
<td>Yes</td>
</tr>
</tbody>
</table>

Command Example:
```
billing set_cost_center -z zone -c cost_center
```

### 6.5.1 Project Activation in OpenStack

To activate a project in an OpenStack-based region, use the following command:

```
open_stack activate_project [arguments]
```

The 'open_stack activate_project' command uses the following arguments:

<table>
<thead>
<tr>
<th>Command Arguments</th>
<th>Description</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>-p, --project</td>
<td>Project abbreviation in UPSA</td>
<td>Yes</td>
</tr>
<tr>
<td>-s, --shape</td>
<td>Shape name. For several shapes, repeat the parameter: -s shape1 -s shape2 -s shapeN</td>
<td>Yes</td>
</tr>
<tr>
<td>-z, --zone</td>
<td>Virtualization zone</td>
<td>Yes</td>
</tr>
</tbody>
</table>
Fake project flag (indicates a fake project, that is, the project not existing in UPSA; used for testing purposes)  

Flag disabling auto-configuration for the project

Command Example:

```
open_stack activate_project -p project -z zone -s SMALL -s MEDIUM -s LARGE
```

After the command execution, projects are created on the OpenStack controller according to the pattern consisting of:

- The name of the node executing the command
- The name of the zone sent in the command
- The project name

During the command execution, default security groups are configured. For each project a separate security group is created with rules not limiting the access for the following protocols: udp, tcp, icmp.

If a project is activated in a zone with networking mode (`-m, --networking-mode`) set to 'MANUAL', a separate network with subnet 172.25.0.0/24 is created (the same subnet is used for all projects in zones with the MANUAL networking mode).

**Personal Projects in OpenStack**

To activate personal project support for a particular OpenStack zone, use the following command:

```
open_stack activate_zones_personal_project [arguments]
```

The `open_stack activate_zones_personal_project` command uses the following arguments:

<table>
<thead>
<tr>
<th>Command Arguments</th>
<th>Argument</th>
<th>Description</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>Argument</td>
<td>-z, --zone</td>
<td>Virtualization zone</td>
<td>Yes</td>
</tr>
</tbody>
</table>

As the result of the command execution, a project under the name “PERSONAL” is activated on the OpenStack controller. All resources launched under personal projects will be assigned to that project.

**6.5.2 OpenStack Networking**

When a zone is created, its networking mode (`-m, --networking-mode`) is set to one of the following values: AUTO or MANUAL. The mode is defined by the OpenStack controller.

The MANUAL mode is an older option which is currently in the process of deprecation. In the MANUAL mode, for each project a separate hard-coded network is created in the zone. When an instance is launched, a public static IP is generated and then assigned to the instance. Afterwards, the DNS name is registered. Private IP addresses are generated from the hard-coded network created for the project in the 172.25.0.0/24 subnet.

The AUTO mode is more advanced and widely used. Such controllers support only one default network for all projects activated on the controller. Private IP addresses for all instances launched on the controller are generated within that network. Public IP addresses are generated in EPAM subnets with IP addresses being public within the EPAM network.
6.5.3 DNS Name Creation in OpenStack

In OpenStack, each zone is created with specification of a DNS server (one or several). EPAM Orchestrator accesses such server(s) to register DNS names of instances launched in OpenStack zones.

Depending on the platform, requests to the DNS server can be made through one of the two utilities:

- **nsupdate** – an utility for Linux orchestrators. DNS name is added as follows:

```bash
nsupdate -g
> server <dns-server IP>
> zone epam.com
> update add <dns-name> <registration lifetime> A <instance IP>
> send
> quit
```

The same utility is used to delete a DNS name from the server:

```bash
nsupdate -g
> server <dns-server IP>
> zone epam.com
> update delete <dns-name> A
> send
> quit
```

- **Dnscmd** – a utility for Windows. DNS name is added as follows:

```bash
dnscmd ServerName /RecordAdd DNSZoneName HostName RecordType IPAddress
```

This utility does not support DNS name deletion, therefore, names are to be deleted manually.

Access to the DNS server is done via special tickets generated through the kinit utility.

The DNS name generation utility is selected by setting one of the profiles [nsupdate-dns-creation, dnscmd-dns-creation]. Also, set [dnscmd.location, nsupdate.location] in the ‘properties’ files for EPAM Orchestrator to discover the utilities.

6.5.4 OpenStack Metadata

Most of the resources created on OpenStack controllers have metadata. Instance metadata stores instance access information and the instance requestor’s identification data. Also, metadata logs instance migration from another project (mtp-action). The mtp-action parameter is set during the or2mtp Maestro CLI command, at the same time, the projectId field in the database used by the OpenStack controller is updated.

Instance flavors (shapes) also have metadata generated during flavor addition to EPAM Orchestrator. Both host metadata and flavor metadata are used to direct the resource creation request to the most suitable host.

System volumes are associated with the corresponding instances by the controller’s response to the instance launch. The response contains the availability zone in which the system volume will be created.
6.5.5 OpenStack Recycle Bin

The Recycle Bin feature allows restoring recently terminated resources. Recycle Bin is, in fact, a project activated on the OpenStack controller serving the entire zone.

- **Recycle Bin Creation**

To create a Recycle Bin, use the following command:

```
open_stack create_recycle_bin [arguments]
```

The `open_stack create_recycle_bin` command uses the following arguments:

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>-z, --zone</td>
<td>Virtualization zone</td>
<td>Yes</td>
</tr>
<tr>
<td>-t, --ttl</td>
<td>Minimum instance TTL (lifetime) in hours. Default: 24</td>
<td>No</td>
</tr>
<tr>
<td>-d, --days</td>
<td>Number of days for the instance to remain in the Recycle Bin. Default: 7</td>
<td>No</td>
</tr>
</tbody>
</table>

Command Example:

```
open_stack create_recycle_bin -z zone -t 24 -d 7
```

- **Recycle Bin Management**

To edit a Recycle Bin, use the following command:

```
open_stack edit_recycle_bin [arguments]
```

The `open_stack edit_recycle_bin` command uses the following arguments:

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>-z, --zone</td>
<td>Virtualization zone</td>
<td>Yes</td>
</tr>
<tr>
<td>-t, --ttl</td>
<td>Minimum instance TTL (lifetime) in hours. Default: 24</td>
<td>No</td>
</tr>
<tr>
<td>-d, --days</td>
<td>Number of days for the instance to remain in the Recycle Bin. Default: 7</td>
<td>No</td>
</tr>
</tbody>
</table>

Command Example:

```
open_stack edit_recycle_bin -z zone -t 48 -d 5
```

The `open_stack edit_recycle_bin` command allows updating the `-t, --ttl` and `-d, --days` parameters. The `-z, --zone` parameter is sent to identify the zone in which the Recycle Bin has to be modified.

To describe a Recycle Bin, use the following command:

```
open_stack describe_recycle_bin [arguments]
```

The `open_stack describe_recycle_bin` command uses the following arguments:
EPAM Cloud Orchestration. Maestro CLI Admin Utility

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>-z, --zone</td>
<td>Virtualization zone</td>
<td>Yes</td>
</tr>
</tbody>
</table>

Command Example:

```
open_stack describe_recycle_bin -z zone
```

The command returns the Recycle Bin settings in the selected zone and lists the resources currently stored in it.

### Management of Resources in Recycle Bin

To restore an instance from the Recycle Bin, use the following command:

```
open_stack restore_from_recycle_bin [arguments]
```

The `open_stack restore_from_recycle_bin` command uses the following arguments:

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>-z, --zone</td>
<td>Virtualization zone</td>
<td>Yes</td>
</tr>
<tr>
<td>-s, --server-id</td>
<td>ID of the instance to be restored. Instance IDs to be used are instance IDs on the OpenStack controller.</td>
<td>Yes</td>
</tr>
<tr>
<td>-p, --project</td>
<td>Project abbreviation in UPSA*</td>
<td>No</td>
</tr>
</tbody>
</table>

* The -p, --project parameter allows restoring the instance in a different project activated in the same zone.

Command Example:

```
open_stack restore_from_recycle_bin -z zone -s instance_id
```

To remove an instance from the Recycle Bin, use the following command:

```
open_stack remove_from_recycle_bin [arguments]
```

The `open_stack remove_from_recycle_bin` command uses the following arguments:

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>-z, --zone</td>
<td>Virtualization zone</td>
<td>Yes</td>
</tr>
<tr>
<td>-s, --server-id</td>
<td>ID of the instance to be restored. Instance IDs to be used are instance IDs on the OpenStack controller.</td>
<td>Yes</td>
</tr>
</tbody>
</table>

Command Example:

```
open_stack remove_from_recycle_bin -z zone -s instance_id
```

To terminate an instance without placing it into the Recycle Bin, it has to be terminated via the Maestro CLI with the --permanently option:
6.5.6 OpenStack Instance State

OpenStack controller and EPAM Orchestrator determine instance state differently. To resolve this difference, EPAM Orchestrator supports special mapping using the following three parameters of the controller:

- Instance state
- Current task running on the instance
- Power state

Combinations of these parameters are mapped to EPAM Orchestrator instance states.

Sometimes, OpenStack controller may produce new combinations. In this case, after two-sided consultations with OpenStack support and approval of the updates, the new combination is added to the EPAM Orchestrator mapping which is part of the code.

6.5.7 Other

- **Volume Errors**

  In case of a volume error on the OpenStack controller, EPAM Orchestrator updates the volume state to ‘error’, because the controller returns an empty response (unlike the response on the instance which contains the reason) not allowing to identify the failure reason. In this case, the Level 1.5 Team will be notified.

- **Shape Change on OpenStack**

  Instance shape is changed with the or2chshape (or2-change-shape) Maestro CLI command. In OpenStack regions, this command execution consists of the following subtasks:

  - Validation of the shape change request
  - Shape change confirmation
  - Waiting for the corresponding flavor replacement

6.6 SIMPLE USER CONFIGURATION

Is cases when there is a necessity to provide access to a user whose credentials cannot be obtained from UPSA for some reason (for example, for a customer representative), a simple user should be created in Orchestrator

The user configuration flow is as follows:
Each step is described in details below:

### 6.6.1 User Creation

To create a new user in EPAM Cloud, use the following command:

```
permission add_user [arguments]
```

The `permission add_user` command uses the following arguments:

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>-e, --email</td>
<td>User's email</td>
<td>Yes</td>
</tr>
<tr>
<td>-u, --username</td>
<td>User's full name</td>
<td>Yes</td>
</tr>
<tr>
<td>-l, --login</td>
<td>User's login</td>
<td>Yes</td>
</tr>
<tr>
<td>-r, --requestor</td>
<td>Email of the requestor of the simple user account</td>
<td>No</td>
</tr>
</tbody>
</table>

**Command Example:**

```
permission add_user -e external_user_email@external.com -u Firstname Lastname -l FirstName_LastName@epam.com
```

### 6.6.2 User Assignment to Project

After the simple user has been created, it should be assigned to a project with the following command:

```
permission assign [arguments]
```
The `permission assign` command uses the following arguments:

<table>
<thead>
<tr>
<th>Command Arguments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Argument</td>
</tr>
<tr>
<td>-e, --email</td>
</tr>
<tr>
<td>-p, --project</td>
</tr>
</tbody>
</table>

**Command Example:**

```
permission assign -e external_user_email@external.com -p project
```

### 6.6.3 Permission Assignment

A simple user should be assigned one or more permission groups defining their access to the Orchestrator functions:

```
permission add_user_mapping [arguments]
```

The `permission add_user_mapping` command uses the following arguments:

<table>
<thead>
<tr>
<th>Command Arguments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Argument</td>
</tr>
<tr>
<td>-e, --email</td>
</tr>
<tr>
<td>-p, --project</td>
</tr>
<tr>
<td>-g, --group</td>
</tr>
</tbody>
</table>

**Command Example:**

```
permission add_user_mapping -p project -e external_user_email@external.com -g permission_group
```

If the `--project` parameter is not specified, the user will be assigned permissions applicable to the entire EPAM Cloud.

### 6.6.4 Permission Update

If the user’s permissions have to be updated according to any changes in their project role, the existing permission groups have to be deleted using the following command:

```
permission del_user_mapping [arguments]
```
The `permission del_user_mapping` command uses the following arguments:

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>-e, --email</td>
<td>User’s email</td>
<td>Yes</td>
</tr>
<tr>
<td>-p, --project</td>
<td>Project abbreviation in UPSA</td>
<td>No</td>
</tr>
</tbody>
</table>

**Command Example:**

```
permission del_user_mapping -p project -e external_user_email@external.com
```

The `permission del_user_mapping` command deletes all permission groups assigned to the user within the project. After all permission groups have been deleted, run the `permission add_user_mapping` command to assign new permission groups.

If you try to add permission groups without deleting the existing user mapping, the command will return an error. Make sure you run the `permission del_user_mapping` command first.

User permissions will be refreshed within 30 minutes, and afterwards the credentials (login, username and CLI password) can be passed on to the external user. The user will be able to access Cloud UI with their login and domain password and Maestro CLI with their login and the password created by the support team.
6.7  MULTIPLE NESSUS SERVER CONFIGURATION

Nessus security scanners are a part of the EPAM Cloud enterprise security solution intended to scan instances for vulnerabilities. Currently, two Nessus scanners are used – one in the EPAM Cloud and one in the Amazon Cloud. The Nessus scanner in the EPAM Cloud acts as the internal scanner for instances within EPAM Cloud and the external one for instances in Amazon Cloud. The Nessus scanner in Amazon Cloud is exactly the opposite – the internal scanner for Amazon instances and the external one for those in EPAM Cloud.

Each Nessus server has to be configured with the scanning settings. Nessus server configuration includes defining the templates, that is, sets of rules by which the virtual machines will be scanned. Also, for each virtualization zone in which security scanning is performed, a Nessus scanner has to be configured and defined as internal or external.

The flow of Nessus server configuration for a zone and further scanning start is as follows:

Each step is described in details below.

6.7.1 Adding New Nessus Server Configuration

To add a new Nessus server configuration, use the following command:

```
security add_nessus_conf [arguments]
```

The `security add_nessus_conf` command uses the following arguments:

<table>
<thead>
<tr>
<th>Command Arguments</th>
<th>Description</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>--host</td>
<td>Nessus server host</td>
<td>Yes</td>
</tr>
<tr>
<td>-x, --path</td>
<td>Application Context path</td>
<td>Yes</td>
</tr>
<tr>
<td>-u, --username</td>
<td>User name</td>
<td>Yes</td>
</tr>
<tr>
<td>-s, --default-template</td>
<td>Default template (set of scanning rules)</td>
<td>Yes</td>
</tr>
<tr>
<td>-n, --scan-name</td>
<td>Default Scan name</td>
<td>Yes</td>
</tr>
<tr>
<td>-l, --launch-schedule</td>
<td>Scan launch schedule (possible values: ONETIME, ON_DEMAND, DAILY, WEEKLY, MONTHLY, YEARLY)</td>
<td>Yes</td>
</tr>
<tr>
<td>-l, --simultaneous-scans</td>
<td>Simultaneous scans limit for this Nessus server</td>
<td>Yes</td>
</tr>
</tbody>
</table>
The ‘security add_nessus_cong’ command requires a password for execution. After the command is sent, the system prompts for the password. Therefore, this command cannot be sent in the ‘quiet’ mode.

Command Example:

```
security add_nessus_cong --host -x sample_path -u sample_username -s
\template_id -n scan_name -l on_demand -e -i -d -v -z
```

### 6.7.2 Setting Nessus Access to Zone

To configure Nessus scanning for a zone and to define the Nessus scanner as internal or external in respect of the zone, use the following command:

```
zone configure_nessus [arguments]
```

The ‘zone configure_nessus’ command uses the following arguments:

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>-z, --zone</td>
<td>Virtualization zone</td>
<td>Yes</td>
</tr>
<tr>
<td>-d, --delay</td>
<td>Delay between scans (min)</td>
<td>No</td>
</tr>
<tr>
<td>-t, --max-tasks</td>
<td>Maximum number of scan tasks per hour</td>
<td>No</td>
</tr>
<tr>
<td>-e, --external</td>
<td>Nessus ID if the Nessus server acts as external</td>
<td>No</td>
</tr>
<tr>
<td>-p, --ping</td>
<td>Enable ping before scanning</td>
<td>No</td>
</tr>
<tr>
<td>-i, --internal</td>
<td>File with configuration for internal VLAN/DMZ scan if the</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>Nessus server acts as internal*</td>
<td></td>
</tr>
</tbody>
</table>

*Such configuration requires file upload*
6.7.3 Adding Templates to Nessus Server

Nessus scanning is performed on the basis of certain sets of rules, the so-called templates. To add templates to a Nessus server, use the following command:

```
security set_templates [arguments]
```

The `security set_templates` command uses the following arguments:

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>--host</td>
<td>Nessus server host</td>
<td>Yes</td>
</tr>
<tr>
<td>-x, --path</td>
<td>Application Context path</td>
<td>Yes</td>
</tr>
<tr>
<td>-f, --file</td>
<td>JSON file containing the Nessus template*</td>
<td>Yes</td>
</tr>
</tbody>
</table>

*Such configuration requires file upload

Command Example:

```
security set_templates --host -x sample_path -f sample_file
```

When a Nessus scanner has been properly configured, scanning can be started either on schedule or manually using the `or2-security-check` Maestro CLI command.
**ANNEX A – ADMIN CLI COMMANDS USAGE IN DIFFERENT VIRTUALIZATION PLATFORMS**

<table>
<thead>
<tr>
<th>Command</th>
<th>aws</th>
<th>azure</th>
<th>csa</th>
<th>hpoo</th>
<th>open_stack</th>
</tr>
</thead>
<tbody>
<tr>
<td>add_image</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>add_zone</td>
<td>X</td>
<td>X</td>
<td></td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>add_shape</td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>add_shapes</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>add_group</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>config_group</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>describe_groups</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>add_account</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>check_account</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>get_accounts</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>activate_project</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>config_project</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>active_project_dl</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>del_project_dl</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>add_subscript</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>del_subscript</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>get_subscript</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>subscript_pool</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>add_enrolment</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>grant_access</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>revoke_access</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>share_credit</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>add_trusted_ip</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>del_trusted_ip</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>add_zone_alias</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>config_network</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>get_net_config</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>add_ownership</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>active_cloudtrail</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>get_cloudtrail</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>put_under_eo</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>set_catalog</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>Command</td>
<td>1</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>--------------------------</td>
<td>---</td>
<td>---</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>sync_from_csa</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>restore_to_csa</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>add_offering</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>check_offerings</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>vlan_activate</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>config_api</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>restore_missing</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>check_flows</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>configvs</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>refresh_images</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>get_problem_inst</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>notific_config</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>notific_switcher</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## ANNEX B – ADMIN CLI COMMANDS REQUIRING FILE UPLOAD

<table>
<thead>
<tr>
<th>Command</th>
<th>File</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>settings add_key</strong></td>
<td>SSH public key</td>
</tr>
<tr>
<td><strong>csa put_under_eo</strong></td>
<td>File containing commands. File format: -z zone -i instance -o offeringName -s shape. Only one command per line is allowed</td>
</tr>
<tr>
<td><strong>pricing_policy update</strong></td>
<td>File containing the pricing policy</td>
</tr>
<tr>
<td><strong>pricing_policy check</strong></td>
<td>File containing the pricing policy</td>
</tr>
<tr>
<td><strong>show project_dls</strong></td>
<td>File containing the list of ORG Cloud User emails (copied from Microsoft Outlook)</td>
</tr>
<tr>
<td><strong>template analyze</strong></td>
<td>File containing the CloudFormation template to be analyzed</td>
</tr>
<tr>
<td><strong>security set_templates</strong></td>
<td>JSON file containing the Nessus template</td>
</tr>
<tr>
<td><strong>zone configure_nessus</strong></td>
<td>File with configuration for internal VLAN/DMZ scan if the Nessus server acts as internal</td>
</tr>
<tr>
<td><strong>aws up_man_policy</strong></td>
<td>File containing the AWS policy</td>
</tr>
<tr>
<td><strong>aws save_policy</strong></td>
<td>File containing the AWS policy</td>
</tr>
<tr>
<td><strong>billing aws_china</strong></td>
<td>Previously uploaded CSV file containing the report</td>
</tr>
<tr>
<td><strong>chef add_config</strong></td>
<td>The command requires uploading three files:</td>
</tr>
<tr>
<td></td>
<td>- Validation pem file</td>
</tr>
<tr>
<td></td>
<td>- Authentication file</td>
</tr>
<tr>
<td></td>
<td>- Chef server certification file</td>
</tr>
</tbody>
</table>

## ANNEX C – ADMIN CLI COMMANDS SENDING EMAILS AS THE RESULT OF EXECUTION

<table>
<thead>
<tr>
<th>Command</th>
<th>Email content</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>billing health_check</strong></td>
<td>Billing consistency report</td>
</tr>
<tr>
<td><strong>billing aws_china</strong></td>
<td></td>
</tr>
<tr>
<td><strong>chef describe_server</strong></td>
<td>Chef server data</td>
</tr>
<tr>
<td><strong>chef get_nodes</strong></td>
<td>Data of Chef nodes and existing EO instances</td>
</tr>
</tbody>
</table>
ANNEX D – AWS-RELATED COLLECTIONS IN DATABASE

<table>
<thead>
<tr>
<th>Collection</th>
<th>Content</th>
</tr>
</thead>
<tbody>
<tr>
<td>AWSAccounts</td>
<td>Contains all AWS accounts. This collection stores both PAYING account and associated LINKED accounts. Each account has the account, accessKey and secretKey fields. They are used by EPAM Orchestrator to make requests to AWS within a project.</td>
</tr>
<tr>
<td>AWSRoles</td>
<td>Currently contains 4 documents each describing a particular role/group:</td>
</tr>
<tr>
<td></td>
<td>EC2_INSTANCE_ROLE – IAM role for Amazon EC2</td>
</tr>
<tr>
<td></td>
<td>GROUP_ROLE – default role used for all project IAM users. This group is created during project activation with the corresponding policies. All IAM users belong to this group</td>
</tr>
<tr>
<td></td>
<td>FEDERATED_USER_ROLE – used when a user accesses AWS via the or2awsmc Maestro CLI command</td>
</tr>
<tr>
<td></td>
<td>CLOUD_SUPPORT_ROLE – used when a user being a EPM-CSUP project member accesses AWS via the or2awsmc Maestro CLI command. This role allows all actions</td>
</tr>
<tr>
<td>AwsslamEntities</td>
<td>Contains roles for SSO configuration. Users accessing AWS via AWS SSO are assigned roles according to their project roles</td>
</tr>
<tr>
<td>Table Name</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>AwslamPolicies</td>
<td>Contains AWS policies for roles/groups/services, etc. For example, SSO roles from the AwslamEntities collection use policies from AwslamPolicies</td>
</tr>
<tr>
<td>AwslamUsers</td>
<td>Contains all IAM users created both via EPAM Orchestrator and AWS. Users data is synchronized once every week</td>
</tr>
<tr>
<td>AwsSSOUserMappings</td>
<td>Contains specific permission settings for users of AWS SSO</td>
</tr>
<tr>
<td>AwslamEntityProhibitionMapping</td>
<td>Contains restrictions for specific users within specific SSO roles</td>
</tr>
<tr>
<td>AwsSecurityGroupsBackup</td>
<td>Scheduled backups of security groups. The collection contains links to security group files</td>
</tr>
</tbody>
</table>
## TABLE OF FIGURES

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Locating Private Key</td>
<td>11</td>
</tr>
<tr>
<td>2</td>
<td>Command groups</td>
<td>16</td>
</tr>
<tr>
<td>3</td>
<td>List of commands in a group</td>
<td>17</td>
</tr>
<tr>
<td>4</td>
<td>Command help</td>
<td>17</td>
</tr>
<tr>
<td>5</td>
<td>Error message indicating missing parameter</td>
<td>17</td>
</tr>
<tr>
<td>6</td>
<td>AWS zone creation flow</td>
<td>45</td>
</tr>
<tr>
<td>7</td>
<td>Azure zone creation flow</td>
<td>61</td>
</tr>
<tr>
<td>8</td>
<td>Network configuration</td>
<td>65</td>
</tr>
<tr>
<td>9</td>
<td>Configuration of Azure zones</td>
<td>65</td>
</tr>
<tr>
<td>10</td>
<td>CSA zone creation flow</td>
<td>66</td>
</tr>
<tr>
<td>11</td>
<td>Google account setup</td>
<td>72</td>
</tr>
<tr>
<td>12</td>
<td>Project activation in Google Cloud</td>
<td>75</td>
</tr>
<tr>
<td>13</td>
<td>OpenStack zone creation</td>
<td>81</td>
</tr>
</tbody>
</table>
# VERSION HISTORY

<table>
<thead>
<tr>
<th>Version</th>
<th>Date</th>
<th>Summary</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.0.2</td>
<td>March 20, 2018</td>
<td>Added a ‘user prolong_access_token’ command to prolong simple user account access expiration</td>
</tr>
<tr>
<td>2.0.1</td>
<td>November 30, 2017</td>
<td>Information about MSQ3 removed</td>
</tr>
<tr>
<td>2.0</td>
<td>September 9, 2017</td>
<td>Document revised, Use Cases section rearranged with use cases grouped by virtualizer</td>
</tr>
<tr>
<td>1.0.4</td>
<td>January 11, 2017</td>
<td>Section describing admin Maestro CLI commands added</td>
</tr>
<tr>
<td>1.0.3</td>
<td>December 16, 2016</td>
<td>Classification changed from Confidential to Public, approved by Dzmitry Pliushch</td>
</tr>
<tr>
<td>1.0.2</td>
<td>November, 4, 2016</td>
<td>Added aws_security check_mfa command description</td>
</tr>
<tr>
<td>1.0.1</td>
<td>September 3, 2016</td>
<td>Added delete_user to aws group. Added incorrect parameters warning to the Basic Principles section.</td>
</tr>
<tr>
<td>1.0</td>
<td>April 10, 2016</td>
<td>First published</td>
</tr>
</tbody>
</table>