

Oracle Cloud Infrastructure

Study Guide

Infrastructure as Code

Release 04/2018

Walter Goerner



Version: 04/18

Walter Goerner
Director Cloud Business Development Oracle EMEA

ORACLE



ORACLE
CLOUD INFRASTRUCTURE

A woman with long brown hair and black-rimmed glasses is sitting at a wooden desk in a modern office. She is wearing a brown leather jacket over a blue and black patterned scarf. She is holding a black smartphone to her ear with her left hand and looking down at a document on the desk with her right hand. In the background, another person is sitting at a desk, and there are large windows letting in natural light.

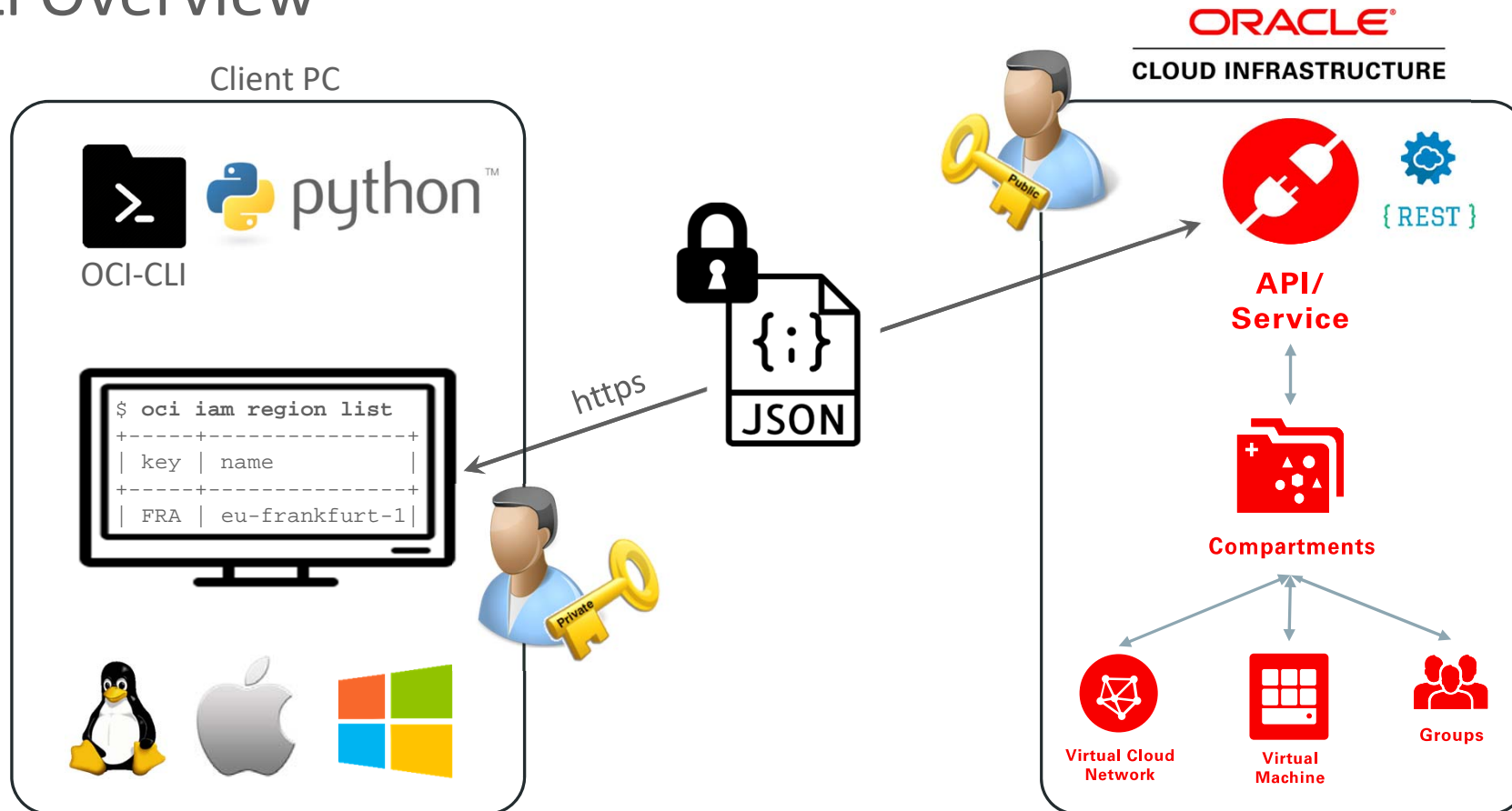
OCI Command Line Interface (CLI)

The OCI Command Line Interface (CLI)

In this chapter we will cover the following topics:

- OCI CLI overview
- JSON and REST basics
- OCI CLI Installation on Windows, Linux, and Mac
- Python installation on Windows machines
- Initial OCI CLI configuration
- Upgrading the OCI CLI
- First Time CLI setup process
- The `config` CLI configuration file
- The `oci_cli_rc` CLI configuration file
- Basic CLI command examples
- Working with environment variables in CLI
- Getting help for CLI commands and options
- Advanced CLI command examples
- Using JMESPath expressions in CLI
- Using JSON files in CLI

CLI Overview





JSON Basics

JSON (JavaScript Object Notation) is a lightweight data-interchange format. It is easy for humans to read and write. It is easy for machines to parse and generate and is based on a subset of the JavaScript Programming Language. These properties make JSON an ideal data-interchange language.

JSON is built on two structures:

1. A collection of name/value pairs.
2. An ordered list of values.

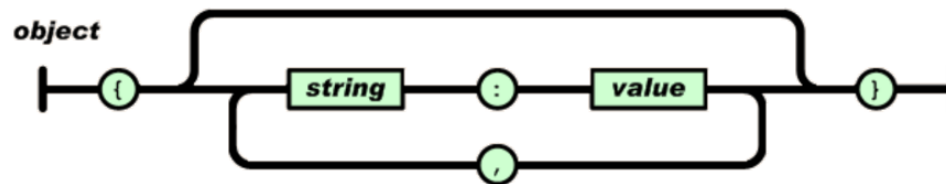
Example.JSON of a JSON record for an OCI security list:

```
{  
  "compartmentId": "ocidl.compartment.[...]",  
  "displayName": "POCSL1",  
  "egressSecurityRules": [  
    {  
      "destination": "0.0.0.0/0",  
      "tcpOptions": {  
        "destinationPortRange": null,  
        "sourcePortRange": null  
      }  
    }  
  ]  
}
```

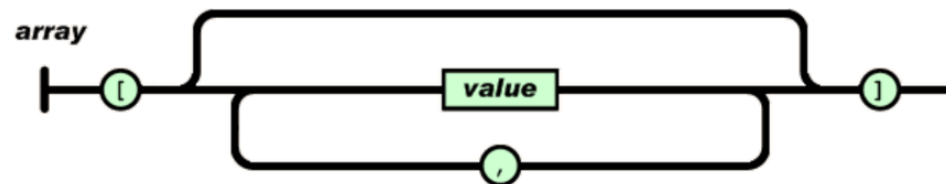


JSON Syntax

The basic JSON data structure is thus a name/value pair in the form of "name" : "value" enclosed by curly brackets { } also called an *object*. This can be depicted using a syntax graph as follows:



An *array* on the other hand is an ordered collection of values. An array begins with [(left bracket) and ends with] (right bracket). Values are separated by , (comma). The syntax graph for an array looks as follows:



{ REST } (REpresentational State Transfer) Basics

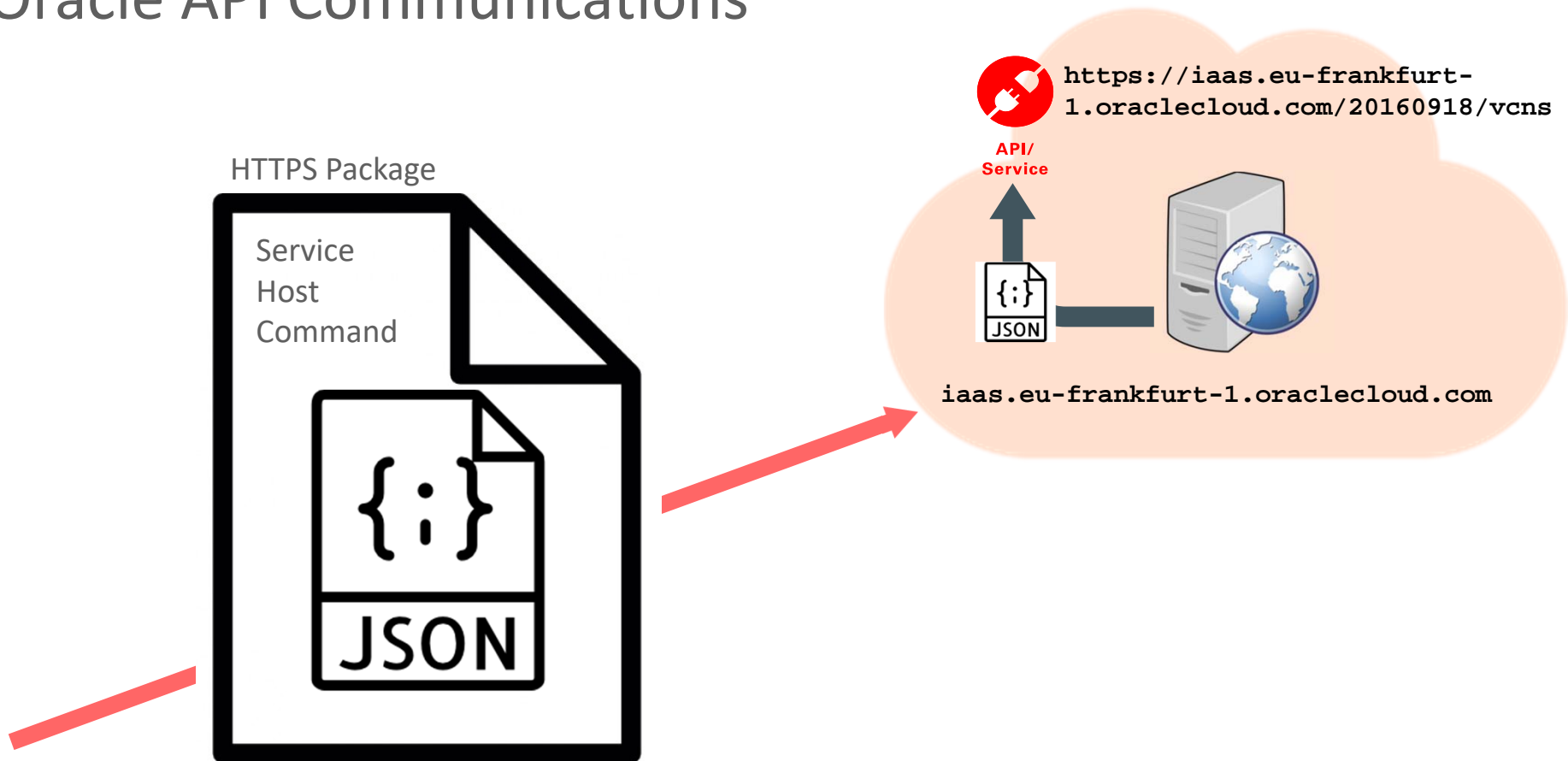
REpresentational State Transfer (REST), or RESTful, web services provide interoperability between computer systems on the Internet. REST-compliant web services allow the requesting systems to access and manipulate textual representations of web resources by using a uniform and predefined set of stateless operations.

In the Oracle API context, requests contain an endpoint and a JSON file defining the API request.

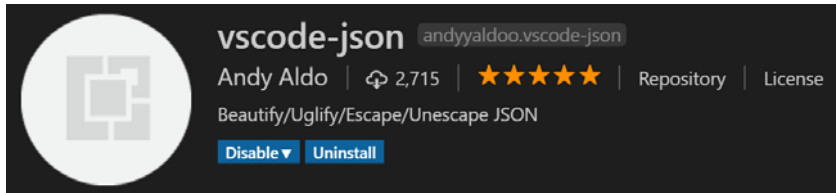
Example Request for Creating a VCN:

```
POST https://iaas.eu-frankfurt-1.oraclecloud.com/20160918/vcns
host: iaas.eu-frankfurt-1.oraclecloud.com
Content-Type: application/json
HTTP headers required for authentication
Other HTTP request headers per the HTTP spec
{
  "compartmentId": "ocid1.compartment.oc1..[...]",
  "displayName": "My Virtual Cloud Network",
  "cidrBlock": "172.16.0.0/16"
}
```

Oracle API Communications



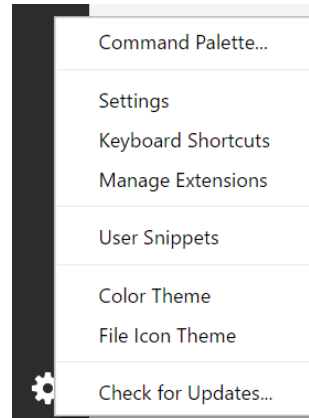
Visual Studio Code JSON Extension Exercise



The image shows the Visual Studio Code extension marketplace card for 'vscode-json' by Andy Aldo. The card includes the extension's icon, name, author, download count (2,715), a five-star rating, and links to the repository and license. It also features 'Disable' and 'Uninstall' buttons.

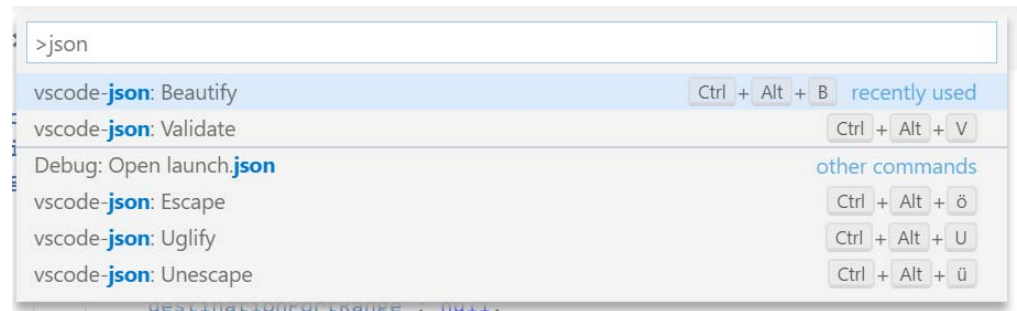
vscode-json andyaldoo.vscode-json
Andy Aldo | 2,715 | ★★★★★ | Repository | License
Beautify/Uglify/Escape/Unescape JSON
Disable Uninstall

```
1 {  
2   "compartmentId": "ocid1.compartment.[...]",  
3   "displayName": "POCSL1",  
4   "egressSecurityRules": [  
5     {  
6       "destination": "0.0.0.0/0",  
7       "tcpOptions": {  
8         "destinationPortRange": null,  
9         "sourcePortRange": null  
10      }  
11    }  
12  ]  
13 }
```



The image shows the 'Settings' menu in Visual Studio Code. The menu is open, displaying various settings categories such as 'Command Palette...', 'Settings', 'Keyboard Shortcuts', 'Manage Extensions', 'User Snippets', 'Color Theme', 'File Icon Theme', and 'Check for Updates...'. A gear icon is visible at the bottom left of the menu.





- Command Palette...
- Settings
- Keyboard Shortcuts
- Manage Extensions
- User Snippets
- Color Theme
- File Icon Theme
- Check for Updates...






The image shows the 'Command Palette' in Visual Studio Code. The search bar contains '>json'. The results list several commands related to JSON, including 'vscode-json: Beautify', 'vscode-json: Validate', 'Debug: Open launch.json', 'vscode-json: Escape', 'vscode-json: Uglify', and 'vscode-json: Unescape'. The 'vscode-json: Beautify' command is highlighted. The keyboard shortcuts for each command are also displayed.

Command	Keyboard Shortcut
vscode-json: Beautify	Ctrl + Alt + B
vscode-json: Validate	Ctrl + Alt + V
Debug: Open launch.json	
vscode-json: Escape	Ctrl + Alt + ö
vscode-json: Uglify	Ctrl + Alt + U
vscode-json: Unescape	Ctrl + Alt + ü

CLI Installation Requirements

Requirement	Description
	An Oracle Cloud Infrastructure account , i.e. a tenancy together with a compartment.
	An OCI user created in that account, in a group with a policy that grants the desired permissions for the OCI resources to be managed. This account user can be or yourself, another person, or a system that calls the API.
	A keypair used for signing API requests, with the public key uploaded to Oracle. Only the user calling the API should possess the private key.
	Python version 2.7.5 or 3.5 or later, running on Mac, Windows, or Linux. Note that if you use the CLI Installer and do not have Python on your machine, the Installer offers to automatically install Python for you.

CLI Installation on Windows – Steps

Step	Description
	Install the latest version of Python 3.x from the Python Software Foundation web-site (www.python.org). Make sure you pick the version suited for your PC architecture (32-bit is denoted as x86 and 64-bit as x86-64).
	Set the PowerShell execution policy to RemoteSigned for the installer script to be able to make the necessary modifications. This is especially necessary for adding the CLI auto-complete feature as part of the installation process.
	Invoke the PowerShell installer script from Oracle and step through the interactive installation process. This script together with the other CLI artifacts is maintained as a GitHub repository on https://github.com/oracle/oci-cli .

Python Installation on Windows

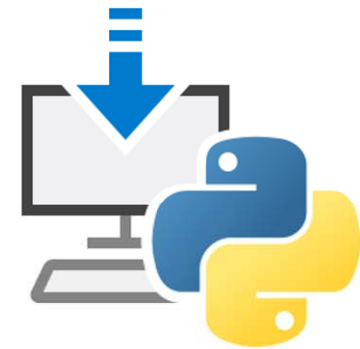
Download

Python source code and installers are available for download for all versions! Not sure which version to use? [Check here.](#)

Latest [Python 3.6.4](#) [Python 2.7.14](#)

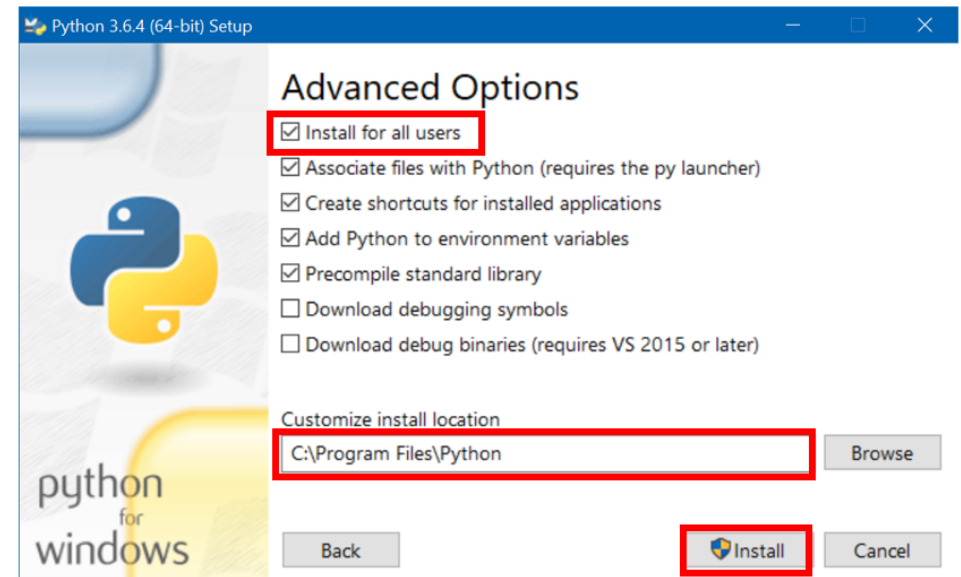
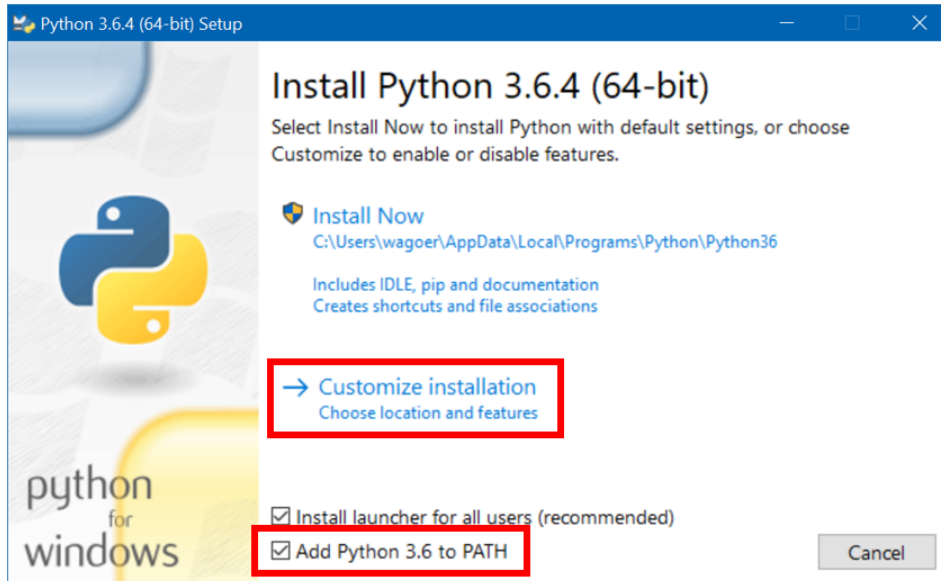


Windows help file	Windows
Windows x86-64 embeddable zip file	Windows
Windows x86-64 executable installer	Windows
Windows x86-64 web-based installer	Windows
Windows x86 embeddable zip file	Windows
Windows x86 executable installer	Windows
Windows x86 web-based installer	Windows



python-3.6.4-amd64.exe

Important Python Installation Options



Test Basic Python Functionality

```
PS C:\Users\Tux> python -version
Python 3.6.4
PS C:\Users\Tux> python
Python 3.6.4 (v3.6.4:d48eceb, Dec 19 2017, 06:54:40) [MSC v.1900 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> a=7
>>> b=4
>>>
>>> print('Sum : ', a+b)
Sum : 11
>>> print('Subtraction : ', a-b)
Subtraction : 3
>>> print('Multiplication : ', a*b)
Multiplication : 28
>>>
```

OCI-CLI GitHub Homepage

Command Line Interface for Oracle Cloud Infrastructure <https://cloud.oracle.com/cloud-infras...>

[bare-metal](#) [cloud](#) [infrastructure](#) [cli](#)

<https://github.com/oracle/oci-cli>

37 commits

23 branches

17 releases

6 contributors

Branch: master


New pull request

Create new file

Upload files

Find file

Clone or download

 mross22 Updating install scripts install.sh and install.ps1 to use latest ver... Latest commit 7191db5 a day ago

scripts	Updating install scripts install.sh and install.ps1 to use latest ver...	a day ago
src/oci_cli	Releasing version 2.4.21	2 days ago
tests	Releasing version 2.4.21	2 days ago
.gitignore	Releasing version 2.4.17	2 months ago
CHANGELOG.rst	Releasing version 2.4.21	2 days ago
CONTRIBUTING.rst	Releasing version 2.4.17	2 months ago
LICENSE.txt	Releasing version 2.4.16 (#42)	2 months ago
MANIFEST.in	Releasing version 2.4.12 (#28)	5 months ago
README.rst	Releasing version 2.4.16 (#42)	2 months ago
requirements.txt	Releasing version 2.4.21	2 days ago
setup.cfg	Releasing version 2.4.8 (#12)	7 months ago
setup.py	Releasing version 2.4.21	2 days ago
tox.ini	Releasing version 2.4.21	2 days ago
wercker.yml	Releasing version 2.4.14 (#34)	3 months ago

PowerShell Installation Script (Admin Shell)

```
PS C:\WINDOWS\system32> Set-ExecutionPolicy RemoteSigned
```

Execution Policy Change

The execution policy helps protect you from scripts that you do not trust. Changing the execution policy might expose you to the security risks described in the about_Execution_Policies help topic at <https://go.microsoft.com/fwlink/?LinkID=135170>. Do you want to change the execution policy?

[Y] Yes [A] Yes to All [N] No [L] No to All [S] Suspend [?] Help (default is "N"): Y

```
powershell -NoProfile -ExecutionPolicy Bypass -Command "iex ((New-Object System.Net.WebClient).DownloadString('https://raw.githubusercontent.com/oracle/oci-cli/master/scripts/install/install.ps1'))"
```

[Script from https://github.com/oracle/oci-cli](https://github.com/oracle/oci-cli)

Installation Script Output & Questions

```
==> In what directory would you like to place the install? (leave blank to use
'C:\Users\Tux\lib\oracle-cli'): <Enter>
-- Creating directory 'C:\Users\Tux\lib\oracle-cli'.
-- We will install at 'C:\Users\Tux\lib\oracle-cli'.

==> In what directory would you like to place the 'oci.exe' executable? (leave
blank to use 'C:\Users\Tux\bin'): <Enter>
-- Creating directory 'C:\Users\Tux\bin'.
-- The executable will be in 'C:\Users\Tux\bin'.
==> Modify PATH to include the CLI and enable tab completion in PowerShell now?
(Y/n): Y
--
-- ** Close and re-open PowerShell to reload changes to your PATH **
-- In order to run the autocomplete script, you may also need to set your PowerShell execution policy
to allow for running local scripts (as an Administrator run Set-ExecutionPolicy RemoteSigned in a
PowerShell prompt)
--
-- Installation successful.
-- Run the CLI with C:\Users\Tux\bin\oci.exe --help
```

Linux/Mac Prerequisites & Install Script

```
[tux@oraclelinux ~]$ sudo yum install gcc libffi-devel python-devel openssl-devel
[tux@oraclelinux ~]$ sudo easy_install pip
Reading https://pypi.python.org/simple/pip/
Best match: pip 9.0.1
Downloading
https://pypi.python.org/packages/11/b6/abcb525026a4be042b486df43905d6893fb04f05aac21c32c638
e939e447/pip-9.0.1.tar.gz#md5=35f01da33009719497f01a4ba69d63c9
Processing pip-9.0.1.tar.gz
[...]
```

```
bash -c "$(curl -L https://raw.githubusercontent.com/oracle/oci-
cli/master/scripts/install/install.sh)"
```

Script from <https://github.com/oracle/oci-cli>

OCI CLI Behind a Proxy Server



OCI CLI behind a proxy server

To use the OCI CLI (as well as the Python SDK and Terraform) behind a proxy server set the `HTTP_PROXY` and/or `HTTPS_PROXY` environment variables as shown below:

On Windows:

```
setx HTTP_PROXY http://<your proxy address here>
```

On Mac/Linux:

```
export HTTP_PROXY=http://<your proxy address here>
```

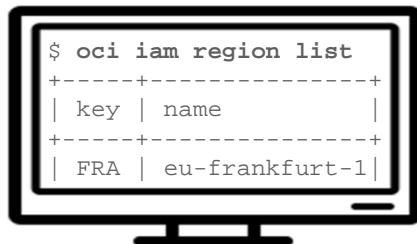
Upgrading the CLI

To upgrade an installed OCI CLI implementation use the following command line from an elevated PowerShell prompt or using Linux/Mac **sudo**:

```
PS C:\WINDOWS\system32> pip install oci-cli --upgrade
Collecting oci-cli
  Downloading oci_cli-2.4.17-py2.py3-none-any.whl (1.4MB)
    100% |████████████████████████████████████████| 1.4MB 930kB/s
Collecting oci==1.3.15 (from oci-cli)
  Downloading oci-1.3.15-py2.py3-none-any.whl (647kB)
    100% |████████████████████████████████████████| 655kB 1.7MB/s
```

CLI Configuration

Client PC



`~/.oci/config`




```
[DEFAULT]
user=ocidl.user.ocl...
fingerprint=16:7e:5f...
key_file=...oci\oci_api_key.pem
tenancy=ocidl.tenancy.ocl...
region=eu-frankfurt-1
```




`~/.oci/oci_cli_rc`

```
[OCI_CLI_COMMAND_ALIASES]
ls = list
rm = os.object.delete
[OCI_CLI_PARAM_ALIASES]
--ad = --availability-domain
--dn = --display-name
```

Setting up the Config File

Step	Description
	Identify and take note of the required Oracle Cloud IDs (OCIDs) for the initial setup process. OCIDs are required for the user account with adequate permissions to OCI resources as well as the tenancy where the user account resides.
 <code>oci setup config</code>	Initiate the initial configuration process with oci setup config using a command shell. This will generate the config file as well as a pair of public/private key files used for securely communicating with the Oracle cloud API service.
	Add the Public Key generated by the initial setup process to the User details in Oracle cloud identity and access management thus enabling secure communications using public/private key encryption.

Where to find the User and Tenancy OCID's



ACTIVE

[API User](#)
OCID: ...k6ejya [Show](#) [Copy](#)

Description: User Account for API calls


Created: Mon, 26 Mar 2018 13:52:50 GMT

...

Copy to Clipboard

TENANCY	REGION
	eu-frankfurt-1

OCID:
ocid1.tenancy.oc1..aaaaaaa75b4lz3hvodfw67q3d7dzrgnryogxlrnwjnljxle63vdl777q
[Hide](#) [Copy](#)

Name: 

Initiating the CLI Config – `oci setup config`

- Location and name of the config file
- Oracle Cloud ID (OCID) of the user account to be used
- Oracle Cloud ID (OCID) of the tenancy to be used
- OCI Region to be used
- Whether to generate API public/private keys

```
PS C:\Users\Tux> oci setup config
```

```
Enter a location for your config [C:\Users\<Name>\.oci\config]: <Enter>
```

```
Enter a user OCID: <Paste user OCID>
```

```
Enter a tenancy OCID: <Paste tenancy OCID>
```

```
Enter a region (e.g. eu-frankfurt-1, us-ashburn-1, us-phoenix-1): eu-frankfurt-1
```

```
Do you want to generate a new RSA key pair? (If you decline you will be asked to supply the path to an existing key.) [Y/n]: Y
```

```
Enter a directory for your keys to be created [C:\Users\<Name>\.oci]: <Enter>
```

```
Enter a name for your key [oci_api_key]: <Enter>
```

```
Public key written to: C:\Users\Tux\.oci\oci_api_key_public.pem
```

```
Enter a passphrase for your private key (empty for no passphrase): <Enter>
```

```
Private key written to: C:\Users\Tux\.oci\oci_api_key.pem
```


Adding the Public Key to the API_USER

API Keys

Add Public Key

Add Public Key

[help](#) [cancel](#)

Note: Public Keys must be in the PEM format.

PUBLIC KEY

Insert the contents of the
oci_api_key_public.pem file in here.

Add

PUBLIC KEY

```
-----BEGIN PUBLIC KEY-----
MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEA2Y04oGX7iPWkQBUI0J1
+eQynuEs8N6z8PfEsshgbl2ZGp3bKqHpQYuWJzqtW4gIhk3U1sVj1wNeaBxxCIX
Ls/GG0ZiaLYP31a/Pj02CMYsKWYTtXgtoIRKenPmgcTJQw3oNaSCiV4Bs894a8/A
ngZsw1KuRTb9KryqRDmhPB5rgOWppnNJzPahpC0HXBWspgX6AouhqcPp+rR1mKU4
vx5sLund5Fs5Vw7Df1omrPEEGKWS0bJP+RfjM48P9naBVp8ueXDSi9UMt1j7/DN
3w2u68fsGNhK2JPGJGZ7BBY9VXcFvWDKzFKLhT17eY7aYORHtsxVsXX61R0yFiP5
NwIDAQAB
-----END PUBLIC KEY-----
```



Fingerprint: f0:f6:da:45:15:eb:ed:73:00:71:6b:fd:01:22:1e:04

Time Created: Sun, 11 Feb 2018 22:29:09 GMT

Delete

ORACLE

Initial CLI Test

```
PS C:\Users\Tux> oci iam region list --output table
```

key	name
FRA	eu-frankfurt-1
IAD	us-ashburn-1
LHR	uk-london-1
PHX	us-phoenix-1

```
PS C:\Users\Tux> oci iam region list
```

```
{
  "data": [
    {
      "key": "FRA",
      "name": "eu-frankfurt-1"
    },
    {
      "key": "IAD",
      "name": "us-ashburn-1"
    },
    [...]
  ]
}
```

The config File



~/**.oci/config**

[DEFAULT]

```
user=ocidl.user.oc1..aaaaaaaaizoxjynjijsvkhzxe3yaw67g5fvjxontb  
xv2xokvexneuwsfo7a  
fingerprint=16:7e:5f:f2:64:3f:32:94:0f:d8:54:ab:07:d4:19:ba  
key_file=C:\Users\<Name>\.oci\oci_api_key.pem  
tenancy=ocidl.tenancy.oc1..aaaaaaaa75b4lz3hvodfw67q3d7dzrgnry  
ogxlrnwjnljqxle63vdlt777q  
region=eu-frankfurt-1
```

Entry	Description	Required?
User	OCID of the user calling the API.	Yes
Fingerprint	Fingerprint for the key pair being used.	Yes
key_file	Full path and filename of the private key. If you encrypted the key with a passphrase, you must also include the pass_phrase entry in the config file.	Yes
pass_phrase	Passphrase used for the key, if it is encrypted. Example: foobar	If key is encrypted
Tenancy	OCID of your tenancy.	Yes
Region	An Oracle Cloud Infrastructure region.	Yes

The oci_cli_rc File



`~/.oci/oci_cli_rc`

[OCI_CLI_SETTINGS]

```
# This defines the default profile in the config file
default_profile=DEFAULT
```

[DEFAULT]

```
# This defines default values to shorten typing
compartment-id = ocidl.compartment.ocl...
subnet-id = ocidl.subnet.ocl.phx.aaaaaaaaypsr...
```

[OCI_CLI_COMMAND_ALIASES]

```
# This lets you use "ls" instead of "list" for any CLI command
ls = list
```

[OCI_CLI_PARAM_ALIASES]

```
# This defines option aliases to shorten typing
--ad = --availability-domain
--dn = --display-name
```

[OCI_CLI_CANNED_QUERIES]

```
# to filter or manipulate output, you can define named queries
get_id_and_display_name_from_list=data[*].{id: id, "display-name":
"display-name"}
```

Generate oci_cli_rc File

```
PS C:\Users\Tux> oci setup oci-cli-rc  
Predefined queries written under section OCI_CLI_CANNED_QUERIES  
Command aliases written under section OCI_CLI_COMMAND_ALIASES
```

Lab 1

Installing and Configuring the OCI-CLI Tool

ORACLE

Copyright © 2018 Oracle and/or its affiliates. All rights reserved. | Oracle Confidential – Internal/Restricted/Highly Restricted

30

Lab 1: Setting up and Configuring the OCI-CLI Tool

Objectives

The objective of this lab is to prepare for, install, and configure the OCI CLI tool on your computer. Processes and procedures have been provided for both Windows PCs as well as Mac / Oracle Linux 7.4 machines, you are free to pick and choose your preferred OS for installation. The lab is considered complete when you are able to access the Oracle cloud API service to display the regions available in OCI, this requires a config file to be present and correctly configured, an `oci_cli_rc` file is not required at this point.

Main tasks for this lab

- Task 1: Install all the necessary prerequisites for your OS
- Task 2: Install OCI-CLI using the provided installation script
- Task 3: Configure OCI-CLI with your OCI security credentials
- Task 4: Perform a basic functionality test to make sure OCI-CLI works properly

Estimated Time: 45 minutes

OCI Basic Command Line Syntax

oci `<service>` `<type>` `<action>` `<options>`



Example: **oci** `iam` `region` `list`

- `iam` is the `<service>`
- `region` is the resource `<type>`
- `list` is the `<action>`, and
- the rest of the command string consists of `<options>`, none in this case

Using Variables for OCI

```
export COMP = 'ocid1.compartment.oc1..aaaaaaaaal3gzijdlieqeyg35nz5zxil26astxxh[...]'
export IMG = 'ocid1.image.oc1.phx.aaaaaaaaaqtj4qjxihpl4mboabsa27mrpusygv6gurp[...]'
export SN = 'ocid1.subnet.oc1.phx.aaaaaaaaypsr25bzjyjyn6xwgkcrgxd3dbhiha6lodz[...]'
```

Using these variables, the above example becomes way more readable as

```
PS C:\Users\Tux> oci compute instance launch
  --availability-domain "EMIr:PHX-AD-1"
  --compartment-id $COMP
  --shape "VM.Standard1.1"
  --display-name "instance 1"
  --image-id $IMG
  --subnet-id $SN
```

In Windows PowerShell the (easiest) syntax for setting and evaluating variables is:

```
PS C:\Users\Tux> Set-Variable TEN "ocid1.tenancy.oc1..aaaaaaaaa75b41z3hvodfw67q"
PS C:\Users\Tux> $TEN
ocid1.tenancy.oc1..aaaaaaaaa75b41z3hvodfw67q
```

Getting Help for OCI

```
oci --help
oci compute instance -h
oci compute instance launch -?
```

There is also a dump of the complete inline help from **oci** on GitHub, if you are interested look at https://github.com/oracle/oci-cli/blob/master/tests/output/inline_help_dump.txt. See below for some output from **oci help**:

```
PS C:\Users\Tux> oci compute instance launch -?
```

```
launch
*****
```

```
Description
=====
```

Creates a new instance in the specified compartment and the specified Availability Domain. For general information about instances, see Overview of the Compute Service.

For information about access control and compartments, see Overview of the IAM Service.

OCI Examples (1)

(1) Set Variables

To make our life easier let's start with defining a couple of variables to avoid having to type in OCID's repeatedly (use the **set-variable** command in Windows PowerShell).

```
TEN  = 'Tenancy OCID' (your tenancy)
US   = 'User OCID' (for the API_User account)
COMP = 'Compartment OCID' (pick any compartment)
```

(2) List Compartments in a Tenancy

Similar to listing the available regions, however, a compartment is part of a tenancy's scope so we need to add the tenancy OCID as an option. Let's use help this time to get some hints:

```
PS C:\Users\<Name> oci iam compartment list -?
Description
=====
```

OCI Examples (2)

```
PS C:\Users\<Name> oci iam compartment list -c $TEN
{
  "data": [
    {
      "compartment-id":
"ocid1.tenancy.oc1..aaaaaaa75b4lz3hvodfw67q3d7dzrgnryogxlrnwjnljqxle63vdl1t777q",
```

(3) List Users in a Tenancy/Compartment

This is again similar to listing the compartments in a tenancy so by now you probably get the idea.

```
PS C:\Users\<Name> oci iam user list -c $TEN
{
  "data": [
    {
      "compartment-id": "ocid1.tenancy.oc1..aaaaaaaabzwjukrknofmqfti5mepdses4p7iwau[...]",
```

OCI Examples (3)

(4) List Available Compute Images in a Compartment as a Table

This time we want to display the available compute images in a compartment as a neatly formatted table. Based on the previous examples our first try would be as follows:

```
PS C:\Users\Tux> oci compute image list -c $TEN --output table
```

```
+-----+-----+-----+-----+-----+
+-----+-----+-----+-----+-----+
+-----+-----+-----+-----+-----+
+-----+-----+-----+-----+-----+
+-----+-----+-----+-----+-----+
+-----+-----+-----+-----+-----+
| base-image-id | compartment-id | create-image-allowed | defined-tags | display-name |
| freeform-tags | id | launch-mode | launch-
options | lifecycle-
state | operating-system | operating-system-version | time-created |
```

OCI Examples (4)

The way to do this in CLI is to use the `--query` option together with a list of columns to be displayed using JMESPath syntax. **JMESPath** is a complete query language for JSON, we will just be using simple forms for filtering column IDs for a table (more details on JMESPath can be found at <http://jmespath.org>). Writing correct JMESPath syntax is still a bit of a challenge in the beginning, but let's walk thru this step by step.

Assume we want just a table with the display names as we would see them in the web console. Looking at the previous output, the column to show is thus called display-name. The JMESPath query to do this looks as follows:

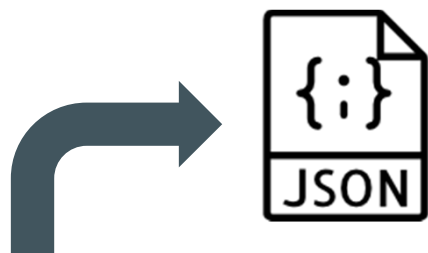
```
--query 'data [*].{"Image Name": "display-name"}'
```

Attention Windows users:

Due to differences in the way Windows parses command lines every double quote " needs to be preceded (escaped) by a backslash \. The above JMESPath query thus translates in Windows as:

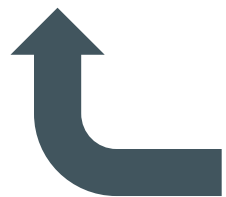
```
--query 'data [*].{"\Image Name\: \"display-name\""}'
```

Using JSON Files in OCI-CLI



```
--generate-full-command-json-input
```

```
oci <service> <type> <action> <options>
```



```
--from-json file: <filename>
```

OCI Examples (5)

```
PS C:\Users\<Name> cd $HOME
PS C:\Users\<Name> oci compute instance launch
--generate-full-command-json-input > launch-instance.json
```

```
{
  "availabilityDomain": "string",
  "compartmentId": "string",
  "displayName": "string",
  "imageId": "string",
  "shape": "string",
  "subnetId": "string"
}
```

```
PS C:\Users\<Name> oci compute instance launch
--from-json file://~/launch-instance.json
```


OCI CLI Exercises

OCI CLI Exercises

- **Exercise 1:** List all the users in your compartment
- **Exercise 2:** List all the availability domains in your region
- **Exercise 3:** Create a VCN in your compartment
- **Exercise 4:** Delete the VCN you just created
- **Exercise 5:** Create a User and Group and add that user to the group

Estimated Time: 30 minutes

Lab 2

Using OCI-CLI to Provision a POC Environment

ORACLE

Copyright © 2018 Oracle and/or its affiliates. All rights reserved. | Oracle Confidential – Internal/Restricted/Highly Restricted

42

Lab 2: Using OCI-CLI to Provision a POC Environment

Objectives

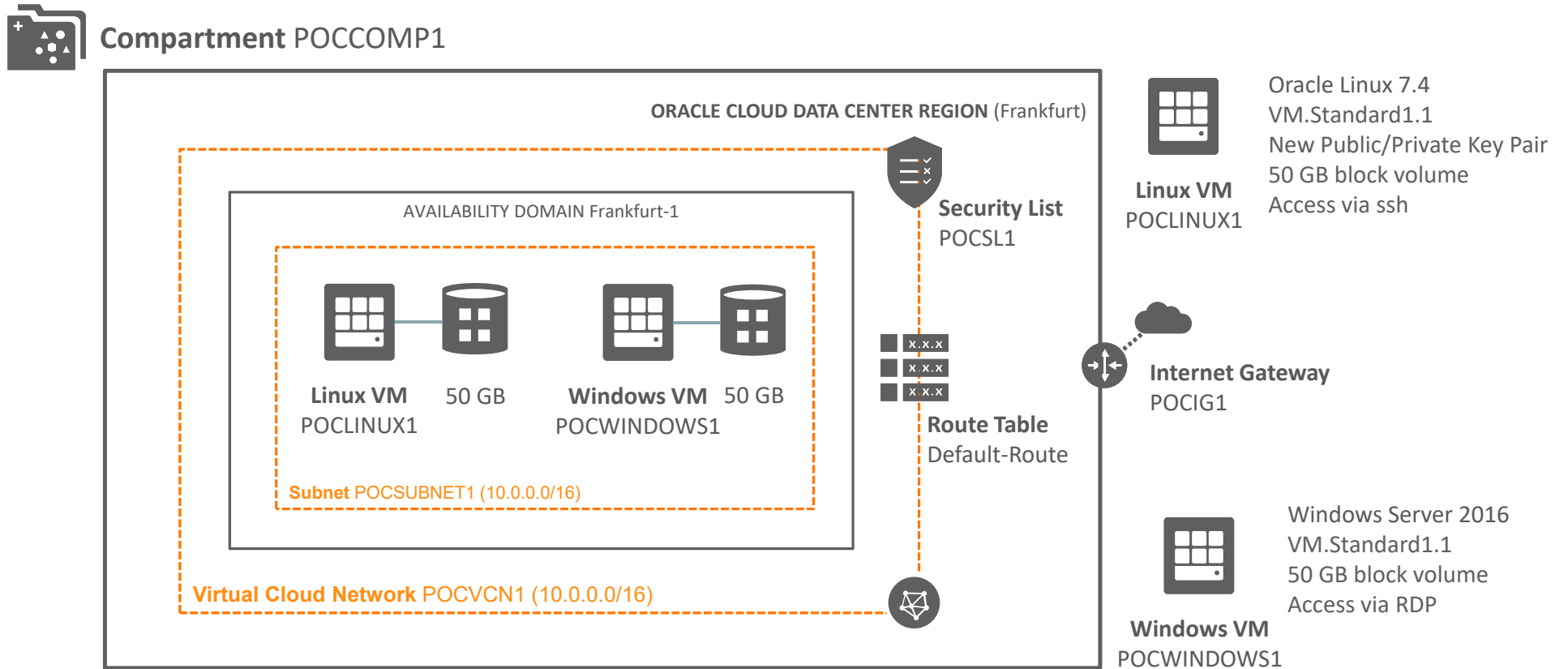
The objectives of this lab are to leverage the OCI-CLI tool to provision, setup, and configure a POC environment running in OCI consisting of all the resources necessary to get a Linux and a Windows VM running and operational for a POC. The lab is considered complete when you are able to connect to the Linux and Windows VM's successfully. At the end of this lab all the resources originally provisioned can be destroyed.

Main tasks for this lab

- Task 1: Elaborate a POC environment in OCI based on the POC scenario
- Task 2: Identify POC cloud resources and provision those with OCI-CLI
- Task 3: Connect to the Linux and Windows VM's to assure functionality
- Task 4: Destroy all the POC OCI resources provisioned before (optional)

Estimated Time: 90 minutes

Lab 2: POC Environment



POC Environment Analysis

1. As a first step, analyze the POC environment and identify all the resources you will need to implement via OCI-CLI. Also think about the sequence of provisioning for these resources:

Nr.	Resource/Task	Description

Lab 2 Guidance (1)

1. **Public/Private Key Pair:** Refer to the appendix for detailed guidance on creating public/private key pairs, feel free to reuse already existing keys you have created earlier.
2. **Create the Compartment POCCOMP1:** Let's start with creating the compartment POCCOMP1 as per the POC diagram. Help: `oci iam compartment create -h`. The general syntax for creating a compartment looks as follows:

```
oci iam compartment create
  -c <root_compartment_id>
  --name "<compartment_name>"
  --description "<friendly_description>"
```

3. **Create the Virtual Cloud Network POCVCN1:** Next steps is to create the VCN in the newly created compartment. Help: `oci network vcn create -h`. The general syntax for creating a VCN looks as follows:

```
oci network vcn create
  -c <compartment_id>
  --display-name "<friendly_name>"
  --dns-label <dns_name>
  --cidr-block "<0.0.0.0/0>"
```

Lab 2 Guidance (2)

- 4. Create the Security List POCSL1:** When you create a VCN, a default security list is created for you which allows port 22 (ssh) inbound traffic. However, the Windows instance also requires inbound traffic enabled for port 3389 (rdp). The preferred approach is creating a second list that addresses the Windows port requirement. You will use the `--security-list-ids` option to associate both security lists with the subnet when you create it. Help: `oci network security-list create -h`. The syntax for creating a security list for port 3389 looks as follows:

```
oci network security-list create -c <compartment_id> --egress-security-rules
'[{ "destination": "<0.0.0.0/0>", "protocol": "<6>", "isStateless": <false>,
"tcpOptions": { "destinationPortRange": <null>,
"sourcePortRange": <null> } } ]'
--ingress-security-rules '[{ "source": "<0.0.0.0/0>", "protocol": "<6>",
"isStateless": <false>, "tcpOptions": { "destinationPortRange": { "max":
<3389>, "min": <3389> }, "sourcePortRange": <null> } } ]' --vcn-id <vcn_id>
--display-name <rule_name>
```

Lab 2 Guidance (3)

5. **Create the Subnet POCSN1:** In this next step, you have to provide the OCIDs for the default security list and the new POCSL1 security list. Help: `oci iam availability-domain list -h`, `oci network subnet create -h`.

```
oci network subnet create --vcn-id <vcn_id> -c <compartment_id>
--availability-domain "<availability_domain_name>" --display-name <display_name>
--dns-label "<dns_label>" --cidr-block "<10.0.0.0/16>"
--security-list-ids `["<security_list_id>","<security_list_id>"]`
```

6. **Create the Internet Gateway POCIG1:** For resources deployed in our subnet POCSN1 to be able to communicate with the Internet, an Internet Gateway is required. Help: `oci network internet-gateway create -h`.
7. **Add a Rule to the Route Table:** For VMs running in our subnet to be able to utilize the Internet Gateway we need to define IP routing so that Internet traffic is routed thru the gateway. When you create a VCN, a route table is created automatically. Before you can add a rule to the route table, you need to know the OCID for the route table. Help: `oci network route-table list -h`, `oci network route-table update -h`.

Lab 2 Guidance (4)

8. Launch the Linux VM: The general command for launching a VM instance is defined as follows:

```
oci compute instance launch
  --availability-domain "<availability_domain_name>"
  --compartment-id "<compartment_id>"
  --shape "<shape_name>"
  --display-name "<display_name>"
  --ssh-authorized-keys-file "<path_to_public_key_file>"
  --image-id "<image_id>"
  --subnet-id "<subnet_id>"
```

9. Launch the Windows VM: Launching the Windows VM is obviously very similar to launching the Linux VM; obviously for Windows, no public/private keys are needed as we will connect to the machine via RDP.

10. Connect to your VMs: To test whether all our configuration tasks were successful, the next step is to connect to the VMs and test basic OS functionality

Lab 2 Guidance (5)

11. Create and Connect Block Volumes: The last task for setting up our POC environment is to setup block volumes of 50GB and attach those to the VMs. Help: `oci bv volume create -h`. The general syntax for creating a block volume is as follows:

```
oci bv volume create
  --availability-domain "<availability_domain_name>"
  --compartment-id "<compartment_id>"
  --size-in-mbs <bv_size>
  --display-name "<volume_display_name>"
```

➤ Task 3: Connect to the Linux and Windows VMs to assure functionality

Connect to both VMs and assure functionality by testing e. g. the following functionality:

- Network/Internet access
- OS updates
- Availability of attached block volumes (remember that you also need to configure the OS to make use of the iSCSI block volumes)

➤ Task 4: Destroy the compute and storage POC OCI resources provisioned

When you're finished setting up the POC environment as described in this lab you should terminate the compute and storage resources to avoid incurring cost.



Review Questions

OCI Command Line Interface (CLI)

Review Questions for OCI CLI (1)

1. Which of the following statements regarding the OCI CLI is true
 - E. The CLI is written in Java.
 - F. The CLI is written in Python.
 - G. The CLI uses JSON as its underlying data-interchange format.
 - H. The CLI uses XML as its underlying data-interchange format.

2. How does the OCI CLI execute commands to manipulate OCI resources?
 - E. The CLI directly manipulates OCI cloud resources.
 - F. The CLI generates calls to the OCI API service.
 - G. The CLI generates jobs for the OCI job service.
 - H. The CLI uses terraform for resource manipulation.

3. Which of the following is a valid JSON record?
 - A. `{"displayName": "POCVCN1", "cidrBlock": "172.16.0.0/16"}`
 - B. `<displayName>POCVCN1</displayName> <cidrBlock>172.16.0.0/16</cidrBlock>`
 - C. `<displayName="POCVCN1"/> <cidrBlock="172.16.0.0/16"/>`
 - D. None of the above is a valid JSON record.

Review Questions for OCI CLI (2)

4. Which of the following is a required prerequisite for installing OCI CLI?
 - A. Java SDK 8.0 or later
 - B. Python 2.7.5 or 3.5 or later
 - C. A keypair for ssh access to OCI API
 - D. A valid Oracle DB license (Standard Edition 2 or greater)

5. The instruction to start the initial OCI CLI configuration process is
 - A. `oci setup oci-cli-rc`
 - B. `oci-cli init`
 - C. `oci setup config`
 - D. `oci.exe --quickstart`

Review Questions for OCI CLI (3)

6. Which of the following configuration files are used exclusively for OCI CLI?
- A. `config`
 - B. `.bashrc`
 - C. `Init.d`
 - D. `oci_cli_rc`
7. The basic OCI CLI command line syntax is defined as:
- A. `oci <service> <type> <action> <options>`
 - B. `oci <JSON command file> <JSON arguments file>`
 - C. `oci <commandlet> <options>`
 - D. `oci <OCID> <command> <options>`
8. What is the OCI CLI command to display the available compute images in a compartment with the OCID of the compartment stored in the variable COMP?
- A. `oci list_compute_images $COMP`
 - B. `oci compute image list -c $COMP`
 - C. `oci --display "compute images" --compartment $COMP`
 - D. `oci list-images "compute" use-compartment $COMP`

Review Questions for OCI CLI (4)

9. Which of the following statements regarding JMESPath in connection with the OCI CLI is true?
- A. JMESPath is used to encrypt communications between the API Service and the OCI CLI.
 - B. JMESPath is a complete query language for JSON.
 - C. JMESPath is used to describe Oracle cloud resources as XML resources.
 - D. JMESPath allows programmatic access to Oracle cloud via the Python language.
10. With which two OCI CLI options can JSON files be generated by and used for the OCI CLI?
- A. `--generate-full-command-json-input`
 - B. `--from-json`
 - C. `--query 'data [JSON]'`
 - D. `--raw-output`

ORACLE®

Integrated Cloud

Applications & Platform Services

ORACLE®