

Oracle OpenWorld 2019

HOL1512

Infrastructure as Code: Oracle Linux,
Terraform, and Oracle Cloud Infrastructure

LAB GUIDE

Christophe Pauliat

Oracle Solution Center sales consultant, Oracle

Matthieu Bordonné

Oracle Solution Center sales consultant, Oracle

Simon Hayler

Senior Principal Technical Product Manager, Oracle

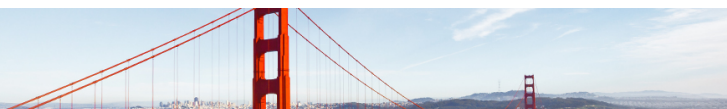


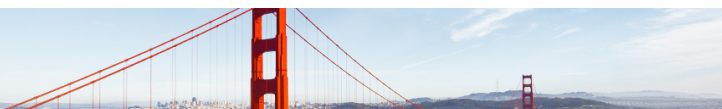
Table of Contents

1	INTRODUCTION	3
1.1	LAB OBJECTIVE	3
1.2	PREREQUISITE.....	3
1.3	SUMMARY OF STEPS	3
1.4	INFRASTRUCTURE DIAGRAM	4
2	RUNNING THE LAB.....	5
2.1	START THE ORACLE LINUX 7 VIRTUAL MACHINE	5
2.2	INSTALL TERRAFORM	6
2.3	GET THE TERRAFORM CONFIGURATION FILES FOR THIS LAB	7
2.4	CREATE A SSH KEY PAIR	8
2.5	CREATE AN API KEY PAIR	8
2.6	ADD THE PUBLIC API KEY IN THE OCI CONSOLE	9
2.7	CREATE A FILE FOR TERRAFORM VARIABLES	11
2.8	TAKE A LOOK AT THE TERRAFORM CONFIGURATION FILES	15
2.9	INITIALIZE TERRAFORM	16
2.10	SIMULATE THE PROVISIONING OF THE OCI INFRASTRUCTURE	17
2.11	PROVISION THE OCI INFRASTRUCTURE	18
2.12	CONNECT TO THE COMPUTE INSTANCE (SSH AND HTTP)	20
2.13	VIEW THE OCI OBJECTS JUST CREATED IN THE OCI CONSOLE.....	21
2.14	PATCH ORACLE LINUX KERNEL WITHOUT ANY SERVICE INTERRUPTION WITH KSPLICE (OPTIONAL)	24
2.15	DESTROY ALL THE OCI OBJECTS CREATED	27
3	APPENDIX A: TERRAFORM CONFIGURATION FILES AND SCRIPTS.....	28
4	APPENDIX B: DOCUMENTATION	33

Last update: September 2, 2019 (version 3)

Author : Christophe Pauliat

Special thanks to : Simon Hayler, Matthieu Bordonné



1 INTRODUCTION

1.1 Lab objective

In this hands on lab HOL1512, you will quickly provision an Oracle Linux compute instance in Oracle Cloud Infrastructure (OCI) using Terraform, a widely used open source tool for Infrastructure as Code.

You will also apply security fixes to Oracle Linux kernel without any service interruption with Ksplice.

This document can be found at <http://bit.ly/oow2019-hol1512-pdf>

1.2 Prerequisite

To run this lab, you need to have access to an Oracle Cloud Infrastructure (OCI) tenancy with enough privileges to create network, compute and block volume resources in a compartment.

If you don't already have such an access, you can request a free trial cloud account on <https://cloud.oracle.com/tryit>

This cloud account is **free** and valid for **30 days**.

It will come with **300 USD** free credits.

In this lab, we will use small and simple OCI objects for a short duration, so few credits will be consumed (typically less than 1 USD).

1.3 Summary of steps

In this lab, you will execute the following steps:

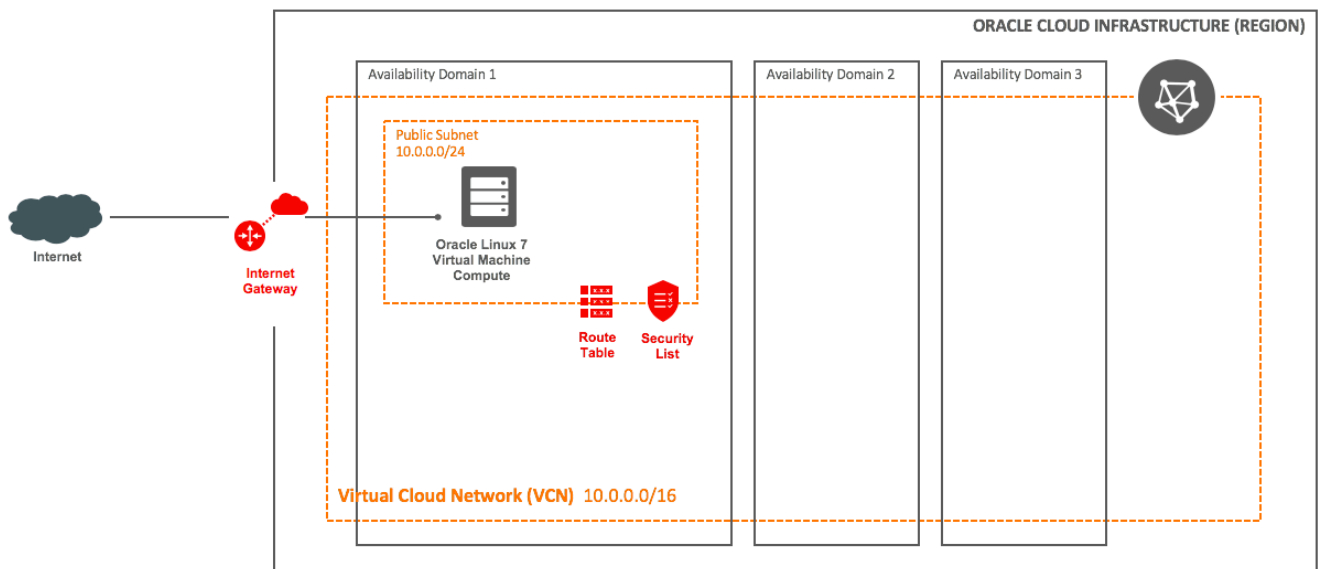
- Learn how to install Terraform in Oracle Linux 7
- Get the Terraform configuration files for this lab
- Create an SSH key pair
- Create an API key pair
- Connect to OCI console and add the API public key to your OCI user
- Update the credentials in the variables file to match your OCI tenancy and user
- Look at the Terraform configuration files
- Initialize Terraform for your project (terraform init)
- Simulate the provisioning of your infrastructure (network and compute instance) in OCI (terraform plan)
- Provision your infrastructure in OCI (terraform apply)
- Connect to the OCI compute instance (Oracle Linux 7) just created.
- Apply security patches to Oracle Linux kernel without any service interruption with Ksplice.
- Cleanup of the infrastructure (terraform destroy)

1.4 Infrastructure diagram

The diagram below shows the OCI (Oracle Cloud Infrastructure) resources that will be provisioned during this lab:

- A virtual cloud network (aka VCN)
- A route table
- A security list (firewall)
- An Internet gateway (gateway to access Internet from the instance and access the instance from Internet)
- A public subnet using the route table and the security list
- A compute instance (Oracle Linux 7 in a Virtual Machine shape)

Note: in OCI, you can also provision a compute instance on a bare metal server rather than a virtual machine.





2 RUNNING THE LAB



2.1 Start the Oracle Linux 7 virtual machine

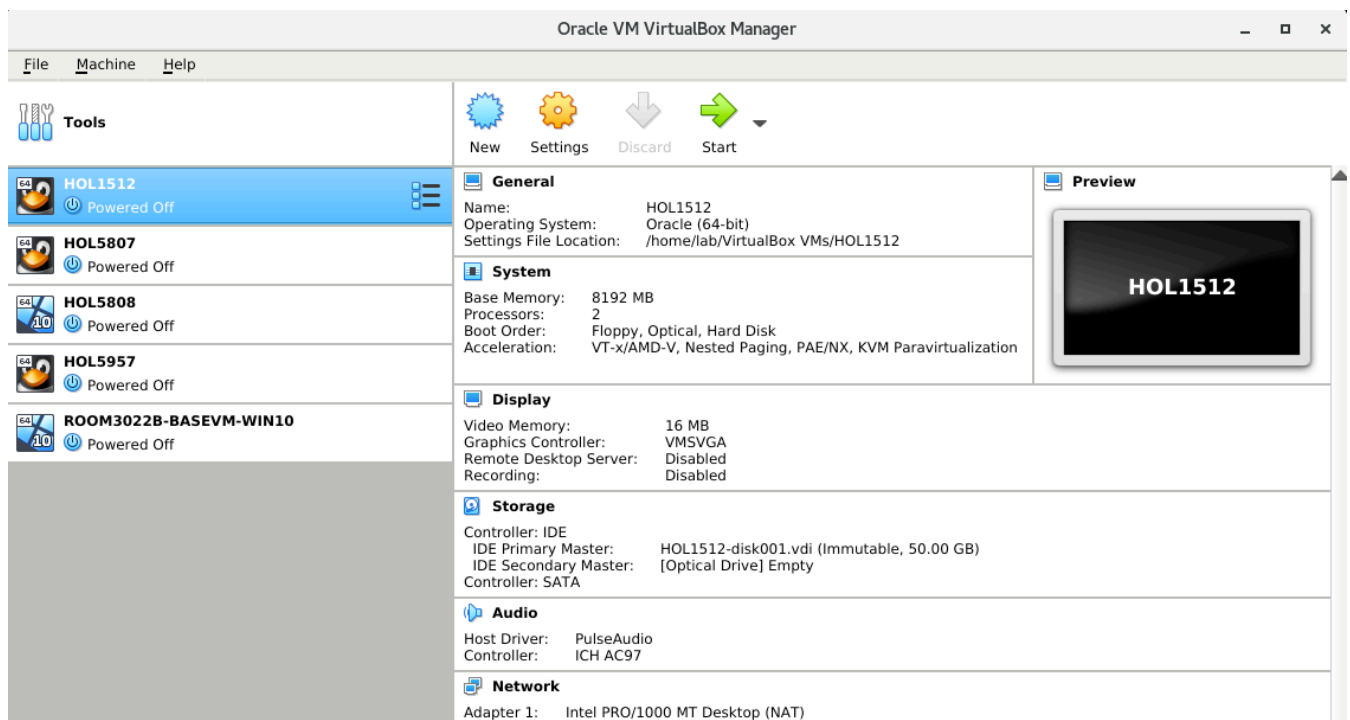
This hands on lab will be done on an Oracle Linux 7.x host.

At Oracle OpenWorld 2019, the host will be an Oracle VM VirtualBox virtual Machine running on top of laptops, and you will use **lab** user (password is **lab2019**)

This virtual machine should be already started, so no action needed here. Go directly to section 2.3

a) If not yet done, start the virtual machine:

- Start Oracle VM VirtualBox Manager if not yet started by clicking icon 
- In this console, select the virtual machine named **HOL1512**, then click the icon  to start it



b) Once the virtual machine is ready, you will be able to work in it, using the graphical environment (Gnome).

c) Optionally, switch to “Full Screen” mode.

Note: the Oracle VM VirtualBox image for this VM can be downloaded from <http://bit.ly/oow2019-hol1512-ova>



2.2 Install Terraform

Terraform is already installed on your Oracle Linux 7 instance, so you can read this to learn how to install Terraform, but no action needed here. **Go directly to section 2.3**

Before you can use Terraform on OCI, you must first install Terraform in your Oracle Linux virtual machine.

On Oracle Linux 7.x, this is very easy as Oracle provides a **terraform** rpm

- Open a Linux Terminal (double click the Terminal icon on the desktop).
- Enable the **ol7_developer** yum channel (not enabled by default)

```
$ sudo yum-config-manager --enable ol7_developer
```
- Execute the following command to install the terraform package

```
$ sudo yum install -y terraform
```

Note: future upgrades of the Terraform for OCI

- To upgrade Terraform on Oracle Linux 7, just execute the following command:

```
$ sudo yum upgrade -y terraform
```

Note: Other platforms.

Terraform is available on many other platforms: other Linux distributions, MacOS, Microsoft Windows, Solaris, FreeBSD and OpenBSD.

See more details on:

- <https://www.terraform.io/downloads.html>
- <https://www.terraform.io/docs/providers/oci/index.html>
- <https://github.com/oracle/terraform-provider-oci>

2.3 Get the Terraform configuration files for this lab

The Terraform configuration files can be downloaded from Internet

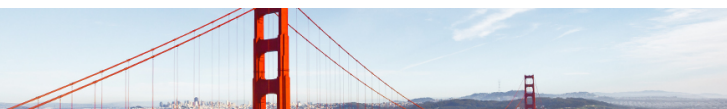
- Open the Firefox web browser, and click bookmark “**HOL1512 Lab files**” (alternatively type the following URL: <http://bit.ly/oow2019-hol1512-zip>)
- Select “**Save File**” and click **OK**
The file, named **OOW2019_HOL1512.zip**, will be saved in the **Downloads** folder (default location for downloaded files in Firefox)
- Open a File Manager window (Click icon **Home** on the Desktop), then go to **Downloads** folder.
You should see the file just downloaded.
- Double click the **OOW2019_HOL1512.zip** file to open **Archive Manager**
- Drag and drop folder **hol1512** from Archive Manager windows to File Manager windows (in **Downloads** folder) to extract the files.

After extraction, you should see the following files in the **~/Downloads/hol1512** folder

File name	Description
terraform.tfvars.TO_BE_MODIFIED	File containing Terraform variables (To be edited and renamed to <code>terraform.tfvars</code>)
01_auth.tf	Terraform configuration file: authentication to OCI tenancy
02_vcn.tf	Terraform configuration file: network objects (VCN, subnets...)
03_instance_ol7.tf	Terraform configuration file: Oracle Linux 7 compute instance
04_block_volume.tf	Terraform configuration file: block volume
versions.tf	Checks Terraform version is greater or equal to 0.12
userdata_bootstrap_ol7.sh	Post-provisioning script (cloud-init) for the compute instance
generate_ssh_keys.sh	Shell script to generate the SSH key pair
generate_api_keys.sh	Shell script to generate the API key pair
version.txt	Shows last update date

Note: Terraform 0.12

- As of August 2019, the current version of Terraform is 0.12.x
- Terraform 0.12 syntax is slightly different (simpler and stricter) from the syntax in previous versions, so you cannot use those files with Terraform 0.11 or older versions.



2.4 Create a SSH key pair

When provisioning a Linux compute instance in OCI, you need to provide a public SSH key. After provisioning, you will need the matching private SSH key to connect to the instance.

In a Linux terminal execute the provided script to generate the SSH key pair

```
[lab@localhost]$ cd ~/Downloads/hol1512
[lab@localhost]$ ./generate_ssh_keys.sh
```

This will create the 2 following files:

- **sshkey** : private SSH key
- **sshkey.pub** : public SSH key

2.5 Create an API key pair

To allow authentication from Terraform to the OCI tenancy, you need to:

- create an API key pair (public and private keys)
- add the public API key in the OCI console
- use the private API key in the Terraform configuration files

In the same Linux terminal execute the provided script to generate the API key pair

```
[lab@localhost]$ ./generate_api_keys.sh
```

This will create the 2 following files:

- **apikey.pem** : private API key
- **apikey_public.pem**: public API key

2.6 Add the public API key in the OCI console

- In the same Linux terminal, display the content of the **apikey_public.pem** file

```
[lab@localhost]$ cat apikey_public.pem
```
- Select the content of the file with your mouse, then right click in the terminal, and click **Copy**
- In the Firefox web browser, open one of the following URLs (depending on the region you subscribed to)
<https://console.us-ashburn-1.oraclecloud.com>
<https://console.us-phoenix-1.oraclecloud.com>
<https://console.eu-frankfurt-1.oraclecloud.com>
<https://console.uk-london-1.oraclecloud.com>
You can use the Firefox bookmarks in the “OCI Console” bookmark menu
- Connect to your OCI tenancy using your tenancy name (Cloud Tenant), your user name and your password. API keys are supported in both OCI local user (Oracle Cloud infrastructure on the right) or Oracle Identity Cloud Service federated user (Single Sign-On SSO on the left), so you can connect with any user (local or federated).

ORACLE Cloud Infrastructure

SIGN IN

Signing in to cloud tenant:
oscemea001
[Change tenant](#)

Single Sign-On (SSO)

We have detected that your tenancy has been federated to another Identity Provider.

Select your Identity Provider below.

IDENTITY PROVIDER
oscemea001

Continue

Oracle Cloud Infrastructure ⓘ


The login is uncommon for federated accounts. If you have questions, please review the [FAQ](#) or contact your tenancy administrator.

or

USER NAME
[input field]

PASSWORD
[input field]

Sign In [Forgot password?](#)

- As this is your first connection to OCI console from this host, you may be asked to enter “**Your choices Regarding Cookies on this Site**”. If so, answer **Yes** to all questions, then click **SUBMIT PREFERENCES**. Finally, click **Close**
- In the top right corner, click the user icon  then click “**User settings**”
- Click “**Add Public Key**”
- In the **PUBLIC KEY** field, right click, then click **Paste** to paste the content of the **apikey_public.pem** file, then click “**Add**”

- After adding the public API key, you should see a new “**Fingerprint**” on screen as shown in the example below. You will need this fingerprint in the next action.


MENU

ORACLE
Cloud Infrastructure

Search

eu-frankfurt-1

Identity » Users » User Details



ACTIVE

christophe.pauliat@oracle.com

Description: OSC

Create/Reset Password

Unblock

Delete

Apply Tag(s)

User Information

Tags

OCID: ...dedmpa

Show Copy

Status: Active

Created: Tue, 02 May 2017 20:22:06 GMT

Resources

API Keys (1)

Auth Tokens (2)

SMTP Credentials (0)

Amazon S3 Compatibility API Keys (2)

Groups (1)

API Keys

Displaying 1 API Keys

Add Public Key

PK

Fingerprint: f8:96:5c:65:fd:5d:63:0c:9f:3a:11:f3:4b:f6:31:92

Time Created: Wed, 05 Sep 2018 16:29:47 GMT

Copyright © 2019, Oracle and/or its affiliates. All rights reserved. Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners. 720041 A4

ORACLE®

10



2.7 Create a file for Terraform variables

To use your own OCI tenancy, you need to create the file `terraform.tfvars` with the right Terraform variables

- In the same Linux terminal, rename file `terraform.tfvars.TO_BE_MODIFIED` to `terraform.tfvars`

```
[lab@localhost]$ mv terraform.tfvars.TO_BE_MODIFIED terraform.tfvars
```
- Edit the `terraform.tfvars` file using your favorite text editor (for instance GEDIT or VI)


```
[lab@localhost]$ gedit terraform.tfvars
```

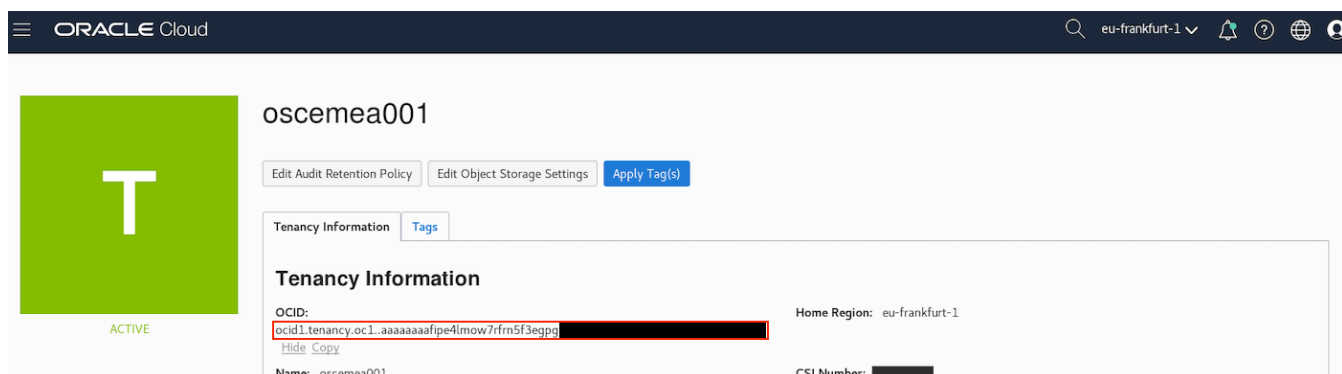
or

```
[lab@localhost]$ vi terraform.tfvars
```
- Modify the following lines

```
tenancy_ocid      = "ocid1.tenancy.oc1..aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa"
user_ocid         = "ocid1.user.oc1..aaaaaaaayaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa"
fingerprint       = "xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx"
compartment_ocid  = "ocid1.compartment.oc1..aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa"
private_key_path   = "apikey.pem"
region            = "us-phoenix-1"
```

You will find the needed information following the instructions below.


- **tenancy_ocid**
 - This is the unique identifier for your OCI Tenancy
 - To view it:
 - Click the user icon  (top right corner)
 - Click **Tenancy: <your tenancy>**
 - Click **Show** next to **OCID:** to display the tenancy OCID
 - Click **Copy** to copy the tenancy OCID. You can then paste it in the `terraform.tfvars` file

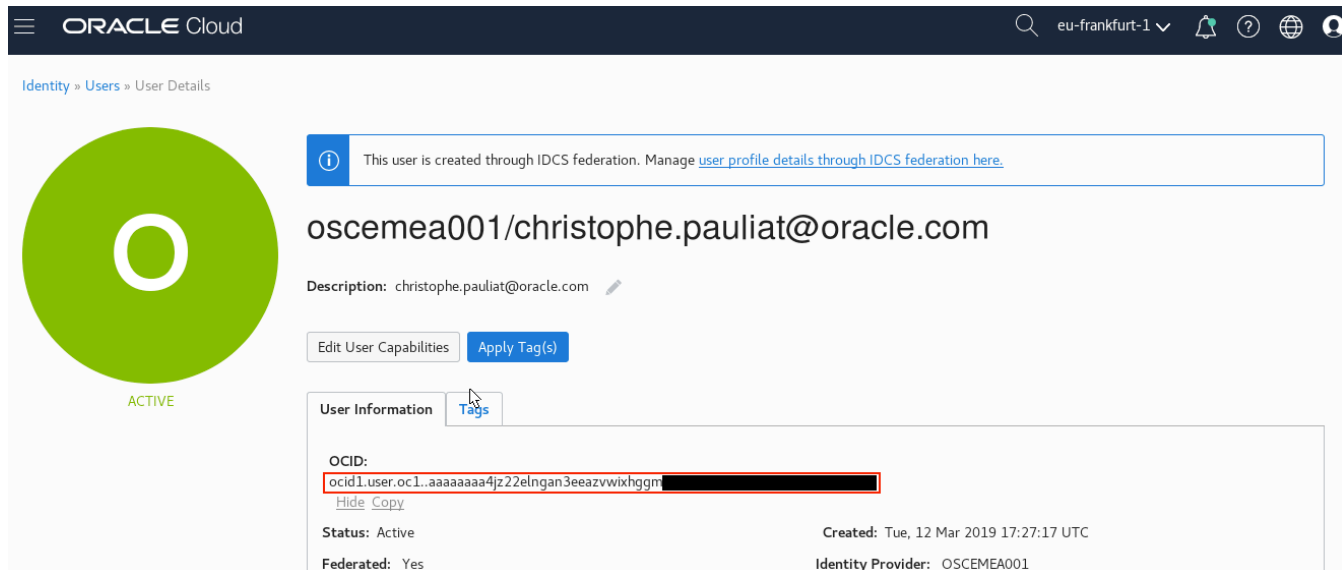


The screenshot shows the Oracle Cloud console interface. At the top, there's a navigation bar with the Oracle Cloud logo and a search bar. Below the navigation bar, there's a header for the tenancy 'oscemea001'. On the left, there's a green square with a white 'T' and the word 'ACTIVE' below it. To the right of the square, there are buttons for 'Edit Audit Retention Policy', 'Edit Object Storage Settings', and 'Apply Tag(s)'. Below these buttons, there's a 'Tenancy Information' section with a 'Tags' tab. The 'Tenancy Information' section shows the OCID: 'ocid1.tenancy.oc1..aaaaaaaafipe4lmow7rfrn5f3egpg' (highlighted in red) and the Name: 'oscemea001'. The Home Region is 'eu-frankfurt-1' and the CSI Number is partially visible.



• user_ocid

- This is the unique identifier of your user
- To view it:
 - Click the user icon  (top right corner)
 - Click **User Settings**
 - Click **Show** next to **OCID:** to display the user OCID
 - Click **Copy** to copy the user OCID. You can then paste it in the **terraform.tfvars** file



Identity » Users » User Details

ooscemea001/christophe.pauliat@oracle.com

Description: christophe.pauliat@oracle.com

ACTIVE

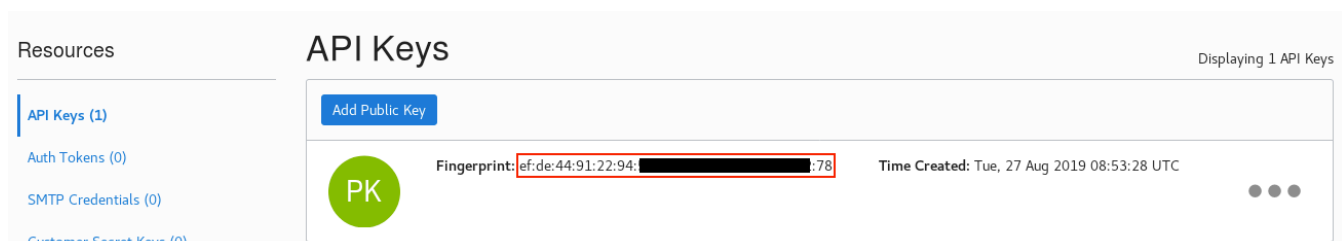
OCID: **ocid1.user.oc1..aaaaaaa4jz22elngan3eeazvwxhgmm**

Status: Active Created: Tue, 12 Mar 2019 17:27:17 UTC

Federated: Yes Identity Provider: OSCMEA001

• Fingerprint

- A fingerprint is created each time you add an API public key.
- Copy the fingerprint displayed in the previous action (select the content with the mouse, then right click, then click **Copy**)
- You can then paste it in the **terraform.tfvars** file



Resources

API Keys (1)

Auth Tokens (0)

SMTP Credentials (0)

Customer Secret Keys (0)

API Keys

Displaying 1 API Keys



Add Public Key

PK

Fingerprint: **ef:de:44:91:22:94:...**

Time Created: Tue, 27 Aug 2019 08:53:28 UTC

- **compartment_ocid**

- In OCI, you can use compartments to organize and isolate your cloud resources. For instance, you can create a compartment for production and another one for development and test. You can then create different OCI policies to give different privileges to different users.
- By default, only the root compartment exists.
- You can create sub-compartments (create a compartment in another compartment)
- It is not recommended to use the root compartment, so if you don't already have a compartment, you can create a new one (if your OCI user has enough privileges)
 - Click icon  (top left corner), then **Identity** (near bottom of the list), then **Compartments**
 - Click **Create Compartment**
 - Enter a **name** and **description**, then click **Create Compartment**
- To view the compartment OCID (if your OCI user has enough privileges)
 - Click icon  (top left corner), then **Identity**, then **Compartments**
 - Click the link in the **OCID** column for the compartment you want to use if it is a child of the root compartment, or first go to the parent compartment if it is different from the root compartment.
 - Click **Copy** to copy the compartment OCID. You can then paste it in the **terraform.tfvars** file

Resources

Child Compartments (8)
Work Requests (0)
Tag Defaults (0)

Tag Filters
add | clear
no tag filters applied

Child Compartments

Create Compartment

Name	Status	OCID	Authorized	Created	
adb_workshops		ocid1.compartment.oc1..aaaaaaaafbaqbcjg26xm4ux [redacted]rzja		Sun, 21 Jul 2019 08:05:19 GMT	⋮
cpauliat	● Active	...uirzja	Yes	Tue, 09 Jan 2018 09:50:12 GMT	⋮
fnworkshopmaster	● Active	...bpueaq	Yes	Tue, 20 Aug 2019 12:22:48 GMT	⋮
fnworkshopstudent	● Active	...teld6q	Yes	Tue, 20 Aug 2019 12:29:06 GMT	⋮

- If you are using your own Cloud trial account, your OCI user will have enough privileges.

- **private_key_path**

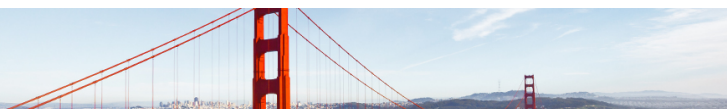
- This is the path to the API private key. It is already filled and no need to change it if you named your API private key **apikey.pem** and stored in the current directory as explained in this guide.

- **region**

- There are currently 10 OCI commercial regions plus some government specific regions (other OCI regions are planned).
- Enter the commercial region you want to use from the following list (the April 2019 image for Oracle Linux 7 is only available on those 7 commercial regions, as the 3 other were built after April).
 - us-phoenix-1 for Phoenix (US West)
 - us-ashburn-1 for Ashburn (US East Coast)
 - ca-toronto-1 for Toronto (Canada)
 - eu-frankfurt-1 for Frankfurt (Europe)
 - uk-london-1 for London (Europe)
 - ap-tokyo-1 for Tokyo (Asia)
 - ap-seoul-1 for Seoul (Asia)
- For optimized performance (lowest latency) during Oracle OpenWorld labs, we recommend using the nearest region (**Phoenix**).
- Make sure to choose a region you subscribed to.

The screenshot shows the Oracle Cloud console interface. At the top, there's a navigation bar with the Oracle Cloud logo and a search bar. Below the navigation bar, the 'Regions' page is displayed. On the left, there's a sidebar with 'Resources' and 'Regions' links. The main content area shows a list of regions, each with a green circle containing an 'R' and the region name. The regions listed are: ap-mumbai-1, eu-frankfurt-1 (Home Region), sa-saopaulo-1, us-ashburn-1, and ap-seoul-1. A blue button labeled 'Subscribe To This Region' is visible next to the 'sa-saopaulo-1' region.

Region Name	Subscription Status
ap-mumbai-1	Not Subscribed
eu-frankfurt-1 (Home Region)	Not Subscribed
sa-saopaulo-1	Subscribe To This Region
us-ashburn-1	Not Subscribed
ap-seoul-1	Not Subscribed



2.8 Take a look at the Terraform configuration files

All configuration files are given in appendix A

As usual with Terraform:

- Objects are defined using the `resource` keyword.
- Outputs are defined using the `output` keyword
- Dynamic data is retrieved using the `data` keyword

In our example, we define the following OCI objects:

In `02_vcn.tf`

- A virtual cloud network (VCN)
- An Internet Gateway
- A route table
- A security list (firewall)
- A public subnet

In `03_instance_017.tf`

- An Oracle Linux 7.6 compute instance

In `04_block_volume.tf`

- A storage block volume
- A storage block volume attachment

The remaining file `01_auth.tf` is used for authentication and variables declaration.

Terraform does not care about the order in which the resources are declared, and automatically detect the dependencies in order to create the resources in the right order.

You can declare your resources in a single or multiple `.tf` files. (We use multiple files here for simplicity)

To simplify the configuration files and make them generic, it is recommended to use variables.
This is what we do here.

All the variables are stored in the **terraform.tfvars** file.

Note: OCI block volumes

- Block volumes uses SSD storage (network storage).
- You can get up to 25000 IOPS per volume with sub millisecond latencies
(<https://docs.cloud.oracle.com/iaas/Content/Block/Concepts/blockvolumeperformance.htm>)
- A block volume can have any size between 50GB and 32 TB
- You can attach up to 32 block volumes to a compute instance (max 32x32 = 1024 TB = 1PB)

2.9 Initialize Terraform

Before you can use Terraform in a specific folder, you need to initialize Terraform in this folder. Terraform will detect the required providers (“oci” and “local”) and automatically install them.

In a Linux terminal, execute the following commands

```
[lab@localhost]$ cd ~/Downloads/hol1512
[lab@localhost]$ terraform init
```

Notes:

- This will create a hidden directory named `.terraform`
- This needs to be done also after upgrading Terraform or the Terraform provider for OCI
- If you get an error, please verify the `terraform.tfvars` file you modified or ask a lab instructor for help.

You should see something similar to following output

```
Initializing the backend...

Initializing provider plugins...
- Checking for available provider plugins...
- Downloading plugin for provider "oci" (terraform-providers/oci) 3.39.0...
- Downloading plugin for provider "local" (terraform-providers/local) 1.3.0...

The following providers do not have any version constraints in configuration,
so the latest version was installed.

To prevent automatic upgrades to new major versions that may contain breaking
changes, it is recommended to add version = "..." constraints to the
corresponding provider blocks in configuration, with the constraint strings
suggested below.

* provider.local: version = "~> 1.3"

Terraform has been successfully initialized!

You may now begin working with Terraform. Try running "terraform plan" to see
any changes that are required for your infrastructure. All Terraform commands
should now work.

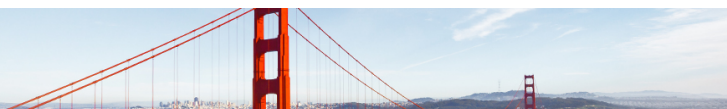
If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.
```

Note: working behind a HTTP proxy

During OpenWorld 2019, you will have direct Internet access from your laptop, so you can run `terraform` command. If you run this lab later at office behind a HTTP proxy, you need to set environments variables `http_proxy` and `https_proxy` so that Terraform can connect to OCI APIs.

On Linux or MacOS, execute the following commands in the terminal before executing `terraform init`

```
export http_proxy=http://<proxy-host>:<proxy-port>
export https_proxy=https://<proxy-host>:<proxy-port>
```

2.10 Simulate the provisioning of the OCI infrastructure

It is good practice to do a dry-run before actually provisioning the OCI infrastructure.

In the same Linux terminal, execute the following command

```
[lab@localhost]$ terraform plan
```

You should see something similar to following output

```
Refreshing Terraform state in-memory prior to plan...
The refreshed state will be used to calculate this plan, but will not be
persisted to local or remote state storage.

data.oci_identity_availability_domains.ADs: Refreshing state...
data.oci_core_images.ImageOCID-ol7: Refreshing state...

-----

An execution plan has been generated and is shown below.
Resource actions are indicated with the following symbols:
+ create

Terraform will perform the following actions:

# local_file.sshconfig will be created
+ resource "local_file" "sshconfig" {
  + content = (known after apply)
  + filename = "sshcfg"
  + id      = (known after apply)
}

# oci_core_instance.tf-oow2019-hol1512-ol7 will be created
...

Plan: 9 to add, 0 to change, 0 to destroy.

-----

Note: You didn't specify an "-out" parameter to save this plan, so Terraform
can't guarantee that exactly these actions will be performed if
"terraform apply" is subsequently run.
```

This tells you that Terraform has detected 9 new resources to add (provision), 0 to modify and 0 to delete.

You can also see the resources types and names (see below), and the details.

```
# local_file.sshconfig
# oci_core_instance.tf-oow2019-hol1512-ol7
# oci_core_internet_gateway.tf-oow2019-hol1512-ig
# oci_core_route_table.tf-oow2019-hol1512-rt
# oci_core_security_list.tf-oow2019-hol1512-subnet1-sl
# oci_core_subnet.tf-oow2019-hol1512-public-subnet1
# oci_core_virtual_network.tf-oow2019-hol1512-vcn
# oci_core_volume.tf-oow2019-hol1512-vol1
# oci_core_volume_attachment.tf-oow2019-hol1512-vol1
```

If you get an error, please verify the **terraform.tfvars** file you modified or ask a lab instructor for help.

2.11 Provision the OCI infrastructure

In the same Linux terminal, execute the following command

```
[lab@localhost]$ terraform apply
```

You will see again the list of OCI objects that will be provisioned or modified (identical to `terraform plan` output) and will need to enter “yes” to confirm that you want to apply the modifications.

Note: you can skip this interactive confirmation by using “`terraform apply --auto-approve`” instead.

The tasks should complete after 1 or 2 minutes (a few seconds to create the network objects and a 1 or 2 minutes to create the compute instance).

You should see something similar to the following output

```
data.oci_identity_availability_domains.ADs: Refreshing state...
data.oci_core_images.ImageOCID-ol7: Refreshing state...

An execution plan has been generated and is shown below.
Resource actions are indicated with the following symbols:
+ create

Terraform will perform the following actions:

...

Plan: 9 to add, 0 to change, 0 to destroy.

Do you want to perform these actions?
Terraform will perform the actions described above.
Only 'yes' will be accepted to approve.

Enter a value: yes

oci_core_volume.tf-oow2019-hol1512-vol1: Creating...
oci_core_virtual_network.tf-oow2019-hol1512-vcn: Creating...
...
oci_core_instance.tf-oow2019-hol1512-ol7: Still creating... [30s elapsed]
...
oci_core_instance.tf-oow2019-hol1512-ol7: Creation complete after 1m19s
[id=ocidl.instance.oc1.phx.abyhqljrbugiji34rfhqxislv2j pz45vm ygwbv6cukdgggfkfpmnrnaweq]
...
oci_core_volume_attachment.tf-oow2019-hol1512-vol1: Creation complete after 24s
[id=ocidl.volumeattachment.oc1.phx.abyhqljrxcm7ywl4xxgq3z6tyosa7zfy lq7a4q3nhjvcdwxaujvphchkv5a]

Apply complete! Resources: 9 added, 0 changed, 0 destroyed.

Outputs:

Connection =

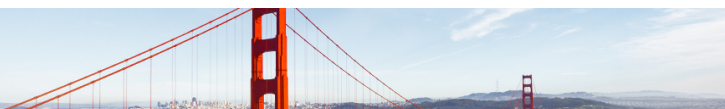
---- Using ssh command with all parameters
ssh -i sshkey opc@129.146.48.5

---- OR using the ssh alias contained in the sshcfg file (created in the local directory)
ssh -F sshcfg ol7

===== Web server URL = http://129.146.48.5
```

Finally, at the end of the `terraform apply` process, you should see the outputs that are defined in the `03_instance_ol7.tf` file. These outputs provide instructions on how to connect to the newly created compute instance.

Please note the public IP assigned to the compute instance (129.146.48.5 in my example)



Note: Terraform State file

- After running “`terraform apply`” for the first time, a new file named **terraform.tfstate** is created.
- This file contains information about the OCI objects that were created by Terraform.
- In order to make modifications or delete objects, **Terraform needs this file, so it is very important keep it.**
- This file is updated each time you make modifications with `terraform apply` or `terraform destroy`.
- Before modifications, a backup copy is created (file **terraform.tfstate.backup**).
- To save those 2 files, it is possible to tell Terraform to store them automatically in an OCI object store location (not done here).

Note: SSH configuration file

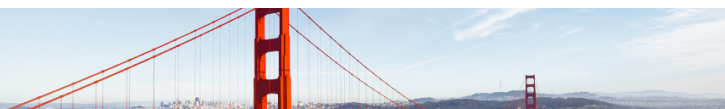
- To simplify SSH connection to the compute instance, a SSH configuration file named **sshcfg** is created in the local directory.
- The content of this file is shown below

```
Host ol7
  Hostname 129.146.48.5
  User opc
  IdentityFile sshkey
  #proxycommand corkscrew <proxy-host> <proxy-port> %h %p
```
- To use it, just add “**-F sshcfg**” to the ssh command as shown in the “`terraform apply`” output

Note: default SSH configuration file (OPTIONAL)

- Optionally, you can copy this file to the SSH configuration file default location and name (`~/.ssh/config`) or append the content to it if already existing, thus removing the need for “**-F sshcfg**” when using `ssh` command.
- If you want to do this, just execute the following command:

```
[lab@localhost]$ mkdir -m 700 -p ~/.ssh
[lab@localhost]$ cat sshcfg >> ~/.ssh/config
[lab@localhost]$ chmod 600 ~/.ssh/config
```



2.12 Connect to the compute instance (SSH and HTTP)

- Wait a few seconds for the boot of the compute instance and the post-provisioning script to complete.
- In the Linux terminal, execute the following SSH command to connect to the compute instance

```
[lab@localhost]$ mkdir -m 700 -p ~/.ssh      (not needed if executed before)
[lab@localhost]$ ssh -F sshcfg o17         (or "ssh o17" if you created the default SSH configuration file)
Are you sure you want to continue connecting (yes/no)? yes
```

Note: make sure to execute this command from the directory containing the **sshcfg** file

- You can execute the following commands to get information about CPU, memory and disks

```
[opc@hol1512 ~]$ cat /proc/cpuinfo      (you should see 2 processors/VCPUs)
[opc@hol1512 ~]$ cat /proc/meminfo      (you should see 15GB of RAM in line MemTotal)
[opc@hol1512 ~]$ lsblk                  (you should see 2 disks /dev/sda and /dev/sdb)
```

Note: Disk naming

When you use multiple block volumes, **/dev/sd<x>** names may change after reboot, so it is recommended to use LVM (Logical Volume Manager) or **/dev/oracleoci/oraclevd<x>** names (available on Oracle Linux) in **/etc/fstab** file to mount filesystems at boot.

- Note: Once connected with the **opc** user, you can execute root actions with the **sudo** command

In our example, we used a cloud-init post-provisioning script (**userdata_bootstrap_o17.sh**) to install and start a Web server, and authorize access from Internet (open port tcp/80 in Oracle Linux firewall service), so we can now check if this Web server is running and reachable.

- In your Firefox (new tab), open the Web server URL shown in the “**terraform apply**” (in my case, <http://129.146.48.5>)
If you don't remember it, you can run “**terraform output**” to display it again.

You should see the following message confirming the Web server is running.


Infrastructure as Code: Oracle Linux, Terraform, and Oracle Cloud Infrastructure [HOL1512]

The Web server is running.

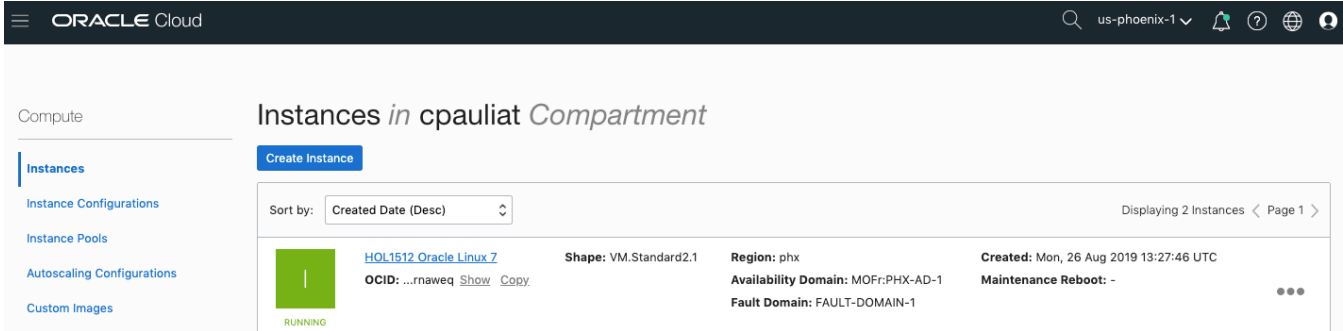
Note: connecting with SSH behind a HTTP proxy.

During OpenWorld 2019, you will have direct Internet access from your laptop, so you can use SSH as explained before. If you run this lab later at office behind a HTTP proxy, you cannot connect with SSH directly, so contact your network administrator to know what you can do in your company.

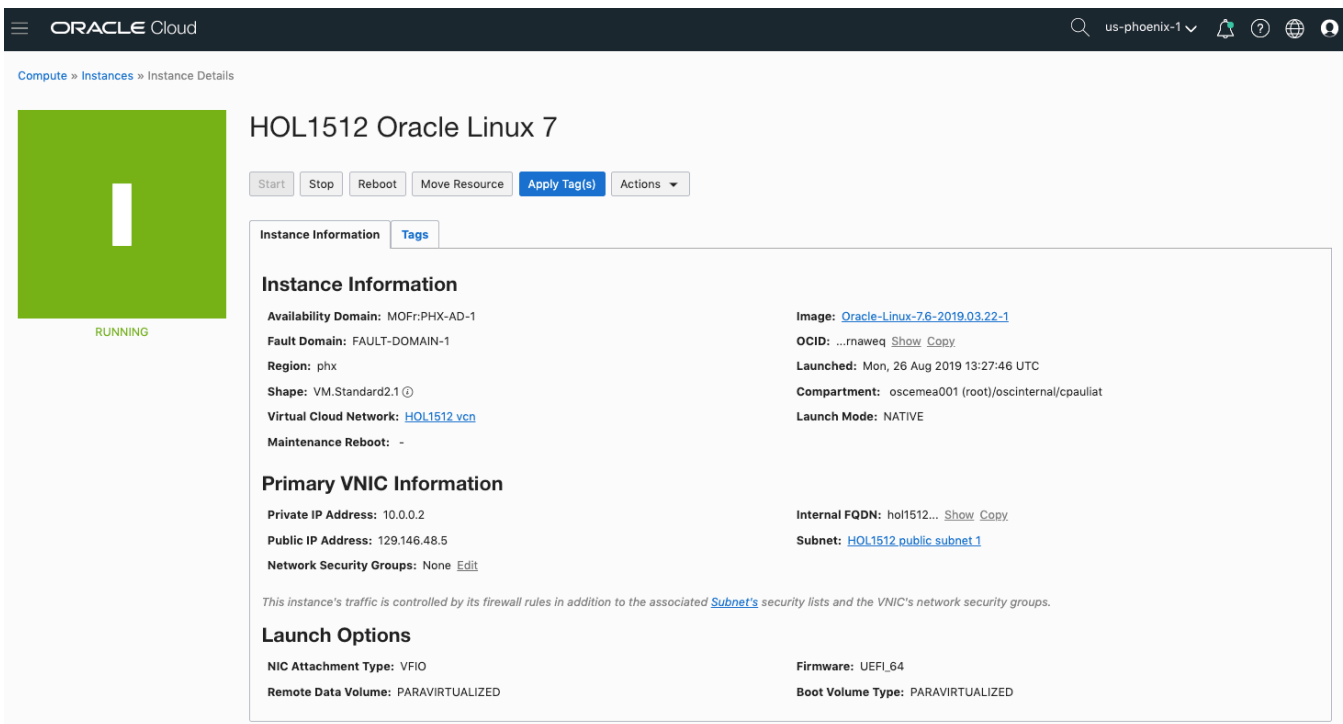
2.13 View the OCI objects just created in the OCI console

- Go back to your Firefox web browser tab showing the OCI console
- Click icon  (top left corner), then **Compute**, then **Instances**
- In the left panel, select the compartment you used

You should see the Oracle Linux 7 compute instance just created as shown below



- Click the compute instance name (“**HOL1512 Oracle Linux 7**”) to get more information






As you can see, some performance metrics (CPU, Memory, Disk, Network) are shown.

- Click **“Boot Volume”** on the left in Resources to see more details about the boot disk

Resources

Boot Volume

Displaying 1 Boot Volumes

 ATTACHED	HOL1512 Oracle Linux 7 (Boot Volume) OCID: ...dhooxa Show Copy Image: Oracle-Linux-7.6-2019.03.22-1	Size: 46.6 GB	Availability Domain: MOFr:PHX-AD-1	Attachment Type: PARAVIRTUALIZED	In-transit Encryption: Disabled	Created: Mon, 26 Aug 2019 13:27:50 UTC
---	---	---------------	------------------------------------	----------------------------------	---------------------------------	--


- Click **“Attached Block Volumes”** on the left in Resources to see more details about the second disk

Resources

Attached Block Volumes

Displaying 1 Attached Block Volumes

Attach Block Volume

 ATTACHED	HOL1512 volume1 OCID: ...z54y2q Show Copy Attachment OCID: ...unliua Show Copy	Attachment Type: paravirtualized Attachment Access: Read/Write	Size: 60.0 GB Device Path: /dev/oracleoci/oraclelevdb	In-transit Encryption: Disabled	Created: Mon, 26 Aug 2019 13:27:42 UTC Availability Domain: MOFr:PHX-AD-1
---	--	---	--	---------------------------------	--

Note: you can see the Device Path that was set to **/dev/oracleoci/oraclelevdb**. It is recommended to use this path instead of default **/dev/sdb** as it will be consistent across reboots.

- Click icon  (top left corner), then **Networking**, then **Virtual Cloud Networks**

You should see the VCN that was created by Terraform

ORACLE Cloud

us-phoenix-1

Networking

Virtual Cloud Networks

Dynamic Routing Gateways

Customer-Premises Equipment

IPSec Connections

Load Balancers

FastConnect

Public IPs

DNS Zone Management

Traffic Management Steering

Policies

Virtual Cloud Networks in cpauliat Compartment

Create Virtual Cloud Network

Name	State	CIDR Block	Default Route Table	DNS Domain Name	Created
HOL1512 vcn	Available	10.0.0.0/16	Default Route Table for HOL1512 vcn	hol1512vcn.oraclevcn.com	Mon, Aug 26, 2019, 1:27:42 PM UTC

Showing 1 Item < Page 1 >

- Click the VCN name (“**HOL1512 vcn**”) to get more details

ORACLE Cloud

us-phoenix-1

Networking » Virtual Cloud Networks » Virtual Cloud Network Details

VCN

AVAILABLE

Move Resource

Add Tags

Terminate

VCN Information

Tags

CIDR Block: 10.0.0.0/16

Compartment: cpauliat

Created: Mon, Aug 26, 2019, 1:27:42 PM UTC

OCID: ..h25gfrq [Show](#) [Copy](#)

Default Route Table: [Default Route Table for HOL1512 vcn](#)

DNS Domain Name: hol1512vcn.oraclevcn.com

Resources

Subnets (1)

Route Tables (2)

Internet Gateways (1)

Dynamic Routing Gateways (0)

Network Security Groups (0)

Security Lists (2)

DHCP Options (1)

Local Peering Gateways (0)

NAT Gateways (0)

Service Gateways (0)

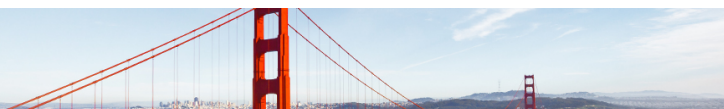
Subnets in cpauliat Compartment

Create Subnet

Name	State	CIDR Block	Subnet Access	Created
HOL1512 public subnet 1	Available	10.0.0.0/24	Public (MOF:PHX-AD-1)	Mon, Aug 26, 2019, 1:27:44 PM UTC

Showing 1 Item < Page 1 >

- Click the different resources in the left panel (**Subnets, Route Tables, Internet Gateway, Security Lists...**) to get more details about the resources that were created



2.14 Patch Oracle Linux kernel without any service interruption with Ksplice (optional)

We created the OCI compute instance using an old Oracle Linux 7.6 image (April 2019) to demo the Ksplice feature of Oracle Linux 7 (online patching of kernel). In real life, you usually want to provision new compute instances from the latest Oracle Linux 7 image.

On May 14, 2019, Intel disclosed some vulnerabilities so Oracle provided some fixes for this and included them in the next images, but obviously those fixes were not included in the April 2019 image, so we will have to manually install them.

May 14, 2019

Intel Processor MDS Vulnerabilities: CVE-2019-11091, CVE-2018-12126, CVE-2018-12130, and CVE-2018-12127



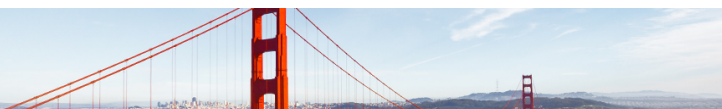
Eric Maurice
DIRECTOR OF SECURITY ASSURANCE

Today, [Intel disclosed](#) a new set of speculative execution side channel vulnerabilities, collectively referred to as "[Microarchitectural Data Sampling](#)" (MDS). These vulnerabilities affect a number of Intel processors and have received four distinct CVE identifiers to reflect how they impact the different microarchitectural structures of the affected Intel processors:

- CVE-2019-11091: Microarchitectural Data Sampling Uncacheable Memory (MDSUM)
- CVE-2018-12126: Microarchitectural Store Buffer Data Sampling (MSBDS)
- CVE-2018-12127: Microarchitectural Load Port Data Sampling (MLPDS)
- CVE-2018-12130: Microarchitectural Fill Buffer Data Sampling (MFBDS)

To install those fixes in our old compute instance, we could use "`sudo yum upgrade`" then reboot, but this would cause a service interruption.

Instead, we will use **Ksplice** to install those security fixes without service interruption.



ACTIONS:

1) Check the currently known cpu vulnerabilities and how they are mitigated in the current kernel

```
[opc@hol1512 cpu]$ cd /sys/devices/system/cpu
[opc@hol1512 cpu]$ grep . vulnerabilities/*
vulnerabilities/lltlf:Mitigation: PTE Inversion; VMX: conditional cache flushes, SMT vulnerable
vulnerabilities/meltdown:Mitigation: PTI
vulnerabilities/spec_store_bypass:Mitigation: Speculative Store Bypass disabled via prctl and seccomp
vulnerabilities/spectre_v1:Mitigation: __user pointer sanitization
vulnerabilities/spectre_v2:Mitigation: Basic IBRS, IBPB: conditional, IBRS_FW, STIBP: conditional
```

Notes:

- "mds" is unknown at this stage
- "spectre v1" is mitigated only by "__user pointer sanitization"

2) Check for available Ksplice patches using the "uptrack-show" command.

Note: For brevity we only look for Microarchitectural Data Sampling and Spectre related patches.

```
[opc@hol1512 cpu]$ sudo uptrack-show --available | egrep -i "spectre|Microarchitectural"
[rvydszk9] Spectre v2 bypass with EIBRS support.
[53rex1tm] Spectre v4 SSBD setting failure for KVM guests.
[p5s6ke5o] Missing hypervisor Spectre v4 mitigations with IBRS disabled.
[n8r0u4y1] CVE-2019-11091, CVE-2018-12126, CVE-2018-12130, CVE-2018-12127: Microarchitectural Data Sampling.
[b00305tt] Improved fix for Spectre v1: Bounds-check bypass in CD/DVD driver.
[o0tvhdwi] Improved fix for Spectre v1: Bounds-check bypass in Honeywell HMC6352 compass driver.
```

There is "Microarchitectural Data Sampling" patch available. Several other patches are also improving the "Spectre v1" mitigation.

3) Check the kernel versions

When using Ksplice, the usual Linux command **uname -r** shows the kernel version during last boot, and the **uptrack-uname -r** command shows the current/effective kernel version.

```
[opc@hol1512 cpu]$ uname -r
4.14.35-1844.4.5.el7uek.x86_64
```

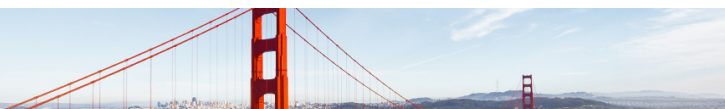
```
[opc@hol1512 cpu]$ uptrack-uname -r
4.14.35-1844.4.5.el7uek.x86_64
```

As we have not yet updated the kernel, we can see that the effective kernel version is identical to the version during last boot.

4) Live patch the kernel using the uptrack-upgrade command.

```
[opc@hol1512 cpu]$ sudo uptrack-upgrade
The following steps will be taken:
....
Install [k1lp4ywb] KPTI enablement for Ksplice.
Install [evhtxv4e] CVE-2019-11091, CVE-2018-12126, CVE-2018-12130, CVE-2018-12127:
Microarchitectural Data Sampling.
Install [h64yzfbo] Remote attack vector in TCP internal control sockets.
.....
Go ahead [y/N]? y

....
Installing [k1lp4ywb] KPTI enablement for Ksplice.
Installing [evhtxv4e] CVE-2019-11091, CVE-2018-12126, CVE-2018-12130, CVE-2018-12127:
Microarchitectural Data Sampling.
Installing [h64yzfbo] Remote attack vector in TCP internal control sockets.
.....
```



Your kernel is fully up to date.
Effective kernel version is 4.14.35-1902.4.8.el7uek

Note that once patching is finished, the "Effective kernel version" is slightly different.

5) Check the kernel versions

```
[opc@hol1512 cpu]$ uname -r
4.14.35-1844.4.5.el7uek.x86_64

[opc@hol1512 cpu]$ uptrack-uname -r
4.14.35-1902.4.8.el7uek.x86_64
```

As you can see, the effective kernel version is now different from last boot version (4.14.35-1902.4.8 vs 4.14.35-1844.4.5)

6) Check again the current known cpu vulnerabilities and how they are now mitigated in the effective kernel

```
[opc@hol1512 cpu]$ cd /sys/devices/system/cpu
[opc@hol1512 cpu]$ grep . vulnerabilities/*
vulnerabilities/lltlf:Mitigation: PTE Inversion; VMX: conditional cache flushes, SMT vulnerable
vulnerabilities/mds:Mitigation: Clear CPU buffers; SMT Host state unknown
vulnerabilities/meltdown:Mitigation: PTI
vulnerabilities/spec_store_bypass:Mitigation: Speculative Store Bypass disabled via prctl and seccomp
vulnerabilities/spectre_v1:Mitigation: usercopy/swapgs barriers and __user pointer sanitization
vulnerabilities/spectre_v2:Mitigation: Basic IBRS, IBPB: conditional, IBRS_FW, STIBP: conditional
```

Notes:

- "mds" is now known and mitigated.
- "spectre v1" mitigation was improved.

7) Log out from the OCI compute instance

```
[opc@hol1512 cpu]$ exit

[lab@localhost]$
```



2.15 Destroy all the OCI objects created

- In the Linux terminal, execute the following command

```
[lab@localhost]$ terraform destroy
```

Terraform will list the OCI objects that will be destroyed and ask for a confirmation before destroying them.

Note: you can skip this interactive confirmation by using “`terraform destroy --auto-approve`” instead.

```
oci_core_virtual_network.tf-oow2019-hol1512-vcn: Refreshing state...
[id=ocidl.vcn.oc1.phx.aaaaaaa5fi3d3sx4apdykcpt6kdn6qzaaex2qlvrpjhkjeoesj2zoh25gfg]
data.oci_core_images.ImageOCID-ol17: Refreshing state...
data.oci_identity_availability_domains.ADs: Refreshing state...
oci_core_internet_gateway.tf-oow2019-hol1512-ig: Refreshing state...
[id=ocidl.internetgateway.oc1.phx.aaaaaaaawpri66ckjffzlujuj2edp4kyrk2miot6det14acqhsqak327zdua]
oci_core_security_list.tf-oow2019-hol1512-subnet1-sl: Refreshing state...
[id=ocidl.securitylist.oc1.phx.aaaaaaa3lv5ddrky6ozhy7ofqyyu4uvlysdhfs37pgh24z2bplxydgkgk]
oci_core_route_table.tf-oow2019-hol1512-rt: Refreshing state...
[id=ocidl.routetable.oc1.phx.aaaaaaa6hmxjdhf6p4ztb5m2x6pt3wpsvabzsfdwlsniojncqpb6xxnw4oq]
oci_core_volume.tf-oow2019-hol1512-vol1: Refreshing state...
[id=ocidl.volume.oc1.phx.abyhqljrat6wff7njtkzuc04v4tjryxr2fb2ottumskgsrlgyjdnz54y2q]
oci_core_subnet.tf-oow2019-hol1512-public-subnet1: Refreshing state...
[id=ocidl.subnet.oc1.phx.aaaaaaaalxhlgtnonbzyju5ixb7qvmf35rpgapskq4wlchm3u74qromdrfrq]
oci_core_instance.tf-oow2019-hol1512-ol17: Refreshing state...
[id=ocidl.instance.oc1.phx.abyhqljrbruugiji34rfhqxislv2jz45vmygwbv6cukdgggfkffpmrnawe]
local_file.sshconfig: Refreshing state... [id=cc89dbc70ebf06f020d6bb60695c814af89faa36]
oci_core_volume_attachment.tf-oow2019-hol1512-vol1: Refreshing state...
[id=ocidl.volumeattachment.oc1.phx.abyhqljrj2cmhgzrxibukbv5uwgsauogsx44ntpsrwm12ildabyrunliua]
```

An execution plan has been generated and is shown below.
Resource actions are indicated with the following symbols:
- destroy

Terraform will perform the following actions:

...

Plan: 0 to add, 0 to change, 9 to destroy.

Do you really want to destroy all resources?
Terraform will destroy all your managed infrastructure, as shown above.
There is no undo. Only 'yes' will be accepted to confirm.

Enter a value: **yes**

```
oci_core_instance.tf-oow2019-hol1512-ol17: Destroying... (ID:
ocidl.instance.oc1.phx.abyhqljrwmancw3...agvctujjgfmhdgriwtr25gxurm4xi5pmmlnmha)
oci_core_instance.tf-oow2019-hol1512-ol17: Still destroying... (ID:
ocidl.instance.oc1.phx.abyhqljrwmancw3...agvctujjgfmhdgriwtr25gxurm4xi5pmmlnmha, 10s elapsed)
...
```

```
oci_core_instance.tf-oow2019-hol1512-ol17: Destruction complete after 1m28s
oci_core_subnet.tf-oow2019-hol1512-public-subnet1: Destroying... (ID:
ocidl.subnet.oc1.phx.aaaaaaaapjldsnbc4...esld6nw5vtpa6v7gen6tdrp7ywphs67pkjajka)
...
```

```
ocidl.vcn.oc1.phx.aaaaaaaabp5dxssh6ss4tffvr36s6wsk5caassfw77vzeegw2vhnfyfbvqlq]
oci_core_virtual_network.tf-oow2019-hol1512-vcn: Destruction complete after 1s
```

Destroy complete! Resources: 9 destroyed.

Congratulations !

You have successfully completed this lab.

We hope you enjoyed it.

3 APPENDIX A: TERRAFORM CONFIGURATION FILES AND SCRIPTS

The ZIP file contains the following files

File name	Description
terraform.tfvars.TO_BE_MODIFIED	File containing Terraform variables (To be edited and renamed to terraform.tfvars)
01_auth.tf	Terraform configuration file: authentication to OCI tenancy
02_vcn.tf	Terraform configuration file: network objects (VCN, subnets...)
03_instance_ol7.tf	Terraform configuration file: Oracle Linux 7 compute instance
04_block_volume.tf	Terraform configuration file: block volume
versions.tf	Checks Terraform version is greater or equal to 0.12
userdata_bootstrap_ol7.sh	Post-provisioning script (cloud-init) for the compute instance
generate_ssh_keys.sh	Shell script to generate the SSH key pair
generate_api_keys.sh	Shell script to generate the API key pair
version.txt	Shows last update date

terraform.tfvars.TO BE MODIFIED

```
# -- PLEASE UPDATE LINES BELOW TO MATCH YOUR OCI TENANT, COMPARTEMENT AND REGION
tenancy_ocid = "ocidl.tenancy.oc1..aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa"
user_ocid    = "ocidl.user.oc1..aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa"
fingerprint  = "xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx"
compartment_ocid = "ocidl.compartment.oc1..aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa"
private_key_path = "apikey.pem"
region       = "us-phoenix-1"
#region      = "us-ashburn-1"
#region      = "ap-seoul-1"
#region      = "ap-tokyo-1"
#region      = "ca-toronto-1"
#region      = "eu-frankfurt-1"
#region      = "uk-london-1"

# ---- Image OCID Oracle Linux 7.6 April 2019 (see https://docs.cloud.oracle.com/iaas/images/oraclelinux-7x)
image_ocid_map = { "ap-seoul-1"      = "ocidl.image.oc1.ap-seoul-1.aaaaaaa52jg3bmw4r72lhp6eciz6ql3r3kant3b6nwnqkcnmhtb3eaoehga",
                  "ap-tokyo-1"     = "ocidl.image.oc1.ap-tokyo-1.aaaaaaaairi7u3txkamxlw3kmw3dosbesrlm22vsh7yybhygzafd3awhlr5q",
                  "ca-toronto-1"   = "ocidl.image.oc1.ca-toronto-1.aaaaaaaq4fdhvvu5ylctfsjdvcfavcu233i4nkp2zbd6bowolvi6uf3cwq",
                  "eu-frankfurt-1" = "ocidl.image.oc1.eu-frankfurt-1.aaaaaaaaydd7mt37cpouxlg2zreiiy2xauebkcwv7g3ngwziisdjenc2sz5a",
                  "uk-london-1"    = "ocidl.image.oc1.uk-london-1.aaaaaaa3rebuz4fyupv2xgdsdhtoiu5aiun17pb76hfqqevoutnxznjn5ua",
                  "us-ashburn-1"   = "ocidl.image.oc1.iad.aaaaaaaajnvritem2k5gfkfq2hwfs4bid577u2jbrzla42wxo2qc77gwx",
                  "us-phoenix-1"   = "ocidl.image.oc1.phx.aaaaaaa6fy4rewjwz2kanrc47fnei5xhfsrprmei2fwfepff3drzmsexznpg" }

# ---- availability domain (1, 2 or 3)
AD = "1"

# ---- IP addresses
cidr_vcn      = "10.0.0.0/16"
cidr_subnet1  = "10.0.0.0/24"

# ---- Authorized public IPs ingress
authorized_ips = "0.0.0.0/0"

# ---- Compute instance
compute_instance_shape = "VM.Standard2.1"
BootStrapFile_ol7      = "userdata_bootstrap_ol7.sh"
ssh_public_key_file_ol7 = "sshkey.pub"
ssh_private_key_file_ol7 = "sshkey"
```



generate_ssh_keys.sh

```
#!/bin/bash

# ---- SSH key pair (will create files sshkey and sshkey.pub)
rm -f ./sshkey ./sshkey.pub
ssh-keygen -t rsa -P "" -f ./sshkey
```

generate_api_keys.sh

```
#!/bin/bash

# ---- API key pair (will create files apikey.pem and apikey_public.pem)
# ---- see doc on https://docs.cloud.oracle.com/iaas/Content/API/Concepts/apisigningkey.htm
openssl genrsa -out ./apikey.pem 2048
chmod 600 ./apikey.pem
openssl rsa -pubout -in ./apikey.pem -out ./apikey_public.pem
```

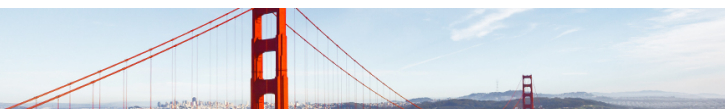
versions.tf

```
terraform {
  required_version = ">= 0.12"
}
```

01_auth.tf

```
# ---- use variables defined in terraform.tfvars file
variable "tenancy_ocid" {}
variable "user_ocid" {}
variable "fingerprint" {}
variable "private_key_path" {}
variable "compartment_ocid" {}
variable "region" {}
variable "AD" {}
variable "authorized_ips" {}
variable "cidr_vcn" {}
variable "cidr_subnet1" {}
variable "compute_instance_shape" {}
variable "BootStrapFile_ol7" {}
variable "ssh_public_key_file_ol7" {}
variable "ssh_private_key_file_ol7" {}
variable "image_ocid_map" { type="map" }

# ---- provider
provider "oci" {
  version      = ">= 3.27.0"
  region       = var.region
  tenancy_ocid = var.tenancy_ocid
  user_ocid    = var.user_ocid
  fingerprint  = var.fingerprint
  private_key_path = var.private_key_path
}
```



02_vcn.tf

```
# ----- get the list of available ADs
data "oci_identity_availability_domains" "ADs" {
  compartment_id = var.tenancy_ocid
}

# ----- Create a new VCN
resource "oci_core_virtual_network" "tf-oow2019-hol1512-vcn" {
  cidr_block      = var.cidr_vcn
  compartment_id  = var.compartment_ocid
  display_name    = "HOL1512 vcn"
  dns_label       = "hol1512vcn"
}

# ----- Create a new Internet Gateway
resource "oci_core_internet_gateway" "tf-oow2019-hol1512-ig" {
  compartment_id = var.compartment_ocid
  display_name   = "HOL1512 ig"
  vcn_id         = oci_core_virtual_network.tf-oow2019-hol1512-vcn.id
}

# ----- Create a new Route Table
resource "oci_core_route_table" "tf-oow2019-hol1512-rt" {
  compartment_id = var.compartment_ocid
  vcn_id         = oci_core_virtual_network.tf-oow2019-hol1512-vcn.id
  display_name   = "HOL1512 route table"

  route_rules {
    destination      = "0.0.0.0/0"
    network_entity_id = oci_core_internet_gateway.tf-oow2019-hol1512-ig.id
  }
}

# ----- Create a new security list to be used in the new subnet
resource "oci_core_security_list" "tf-oow2019-hol1512-subnet1-sl" {
  compartment_id = var.compartment_ocid
  display_name   = "HOL1512 seclist"
  vcn_id         = oci_core_virtual_network.tf-oow2019-hol1512-vcn.id

  egress_security_rules {
    protocol = "all"
    destination = "0.0.0.0/0"
  }

  ingress_security_rules {
    protocol = "all"
    source   = var.cidr_vcn
  }

  ingress_security_rules {
    protocol = "6" # tcp
    source   = var.authorized_ips

    tcp_options {
      min = 22 # to allow SSH access to Linux instance
      max = 22
    }
  }

  ingress_security_rules {
    protocol = "6" # tcp
    source   = var.authorized_ips

    tcp_options {
      min = 80 # to allow HTTP access
      max = 80
    }
  }
}

# ----- Create a public subnet 1 in AD1 in the new VCN
resource "oci_core_subnet" "tf-oow2019-hol1512-public-subnet1" {
  availability_domain = data.oci_identity_availability_domains.ADs.availability_domains[var.AD - 1]["name"]
  cidr_block          = var.cidr_subnet1
  display_name        = "HOL1512 public subnet 1"
  dns_label           = "subnet1"
  compartment_id      = var.compartment_ocid
  vcn_id              = oci_core_virtual_network.tf-oow2019-hol1512-vcn.id
  route_table_id      = oci_core_route_table.tf-oow2019-hol1512-rt.id
  security_list_ids    = [oci_core_security_list.tf-oow2019-hol1512-subnet1-sl.id]
  dhcp_options_id     = oci_core_virtual_network.tf-oow2019-hol1512-vcn.default_dhcp_options_id
}
```



03_instance_ol7.tf

```
# ----- Get the OCID for the most recent for Oracle Linux 7.x disk image
data "oci_core_images" "ImageOCID-ol7" {
  compartment_id = var.compartment_ocid
  operating_system = "Oracle Linux"
  operating_system_version = "7.6"

  # filter to avoid Oracle Linux 7.x images for GPU
  filter {
    name = "display_name"
    values = ["^.*Oracle-Linux-7.6-[^G].*$"]
    regex = true
  }
}

# ----- Create a compute instance from the most recent Oracle Linux 7.x image
resource "oci_core_instance" "tf-oow2019-hol1512-ol7" {
  availability_domain = data.oci_identity_availability_domains.ADs.availability_domains[var.AD - 1]["name"]
  compartment_id = var.compartment_ocid
  display_name = "HOL1512 Oracle Linux 7"
  shape = var.compute_instance_shape
  preserve_boot_volume = "false"

  source_details {
    source_type = "image"

    # Uncomment the line below if you want to use the latest Oracle Linux 7.x image
    # source_id = lookup(data.oci_core_images.ImageOCID-ol7.images[0], "id")
    # In this lab, we use an old Oracle Linux 7.6 image (April 2019) to be able to apply online kernel security fixes with
    Ksplice
    source_id = lookup(var.image_ocid_map, var.region)
  }

  create_vnic_details {
    subnet_id = oci_core_subnet.tf-oow2019-hol1512-public-subnet1.id
    hostname_label = "hol1512"
    # If you want to force a specific private IP, uncomment the line below and update it
    # private_ip = "10.0.0.3"
  }

  metadata = {
    ssh_authorized_keys = file(var.ssh_public_key_file_ol7)
    user_data = base64encode(file(var.BootStrapFile_ol7))
  }
}

resource "local_file" "sshconfig" {
  content = <<EOF
Host ol7
  Hostname ${oci_core_instance.tf-oow2019-hol1512-ol7.public_ip}
  User opc
  IdentityFile ${var.ssh_private_key_file_ol7}
  #proxycommand corkscrew <proxy-host> <proxy-port> %h %p
EOF

  filename = "sshcfg"
}

# ----- Display the complete ssh command needed to connect to the instance
output "Connection" {
  value = <<EOF

  ---- Using ssh command with all parameters
  ssh -i ${var.ssh_private_key_file_ol7} opc@${oci_core_instance.tf-oow2019-hol1512-ol7.public_ip}

  ---- OR using the ssh alias contained in the sshcfg file (created in the local directory)
  ssh -F sshcfg ol7

  ===== Web server URL = http://${oci_core_instance.tf-oow2019-hol1512-ol7.public_ip}

EOF
}
```



04_block_volume.tf

```
# ----- Create a block volume
resource "oci_core_volume" "tf-oow2019-hol1512-vol1" {
  availability_domain = data.oci_identity_availability_domains.ADs.availability_domains[var.AD - 1]["name"]
  compartment_id      = var.compartment_ocid
  display_name        = "HOL1512_volume1"
  size_in_gbs         = "60"
}

# ----- Attach the new block volume to the ol7 compute instance after it is created
resource "oci_core_volume_attachment" "tf-oow2019-hol1512-vol1" {
  attachment_type = "paravirtualized"
  instance_id     = oci_core_instance.tf-oow2019-hol1512-ol7.id
  volume_id       = oci_core_volume.tf-oow2019-hol1512-vol1.id
  device          = "/dev/oracleoci/oraclelvd"
}
```

userdata_bootstrap_ol7.sh

```
#!/bin/bash

### ---- Send stdout, stderr to /var/log/messages/
exec 1> >(logger -s -t $(basename $0)) 2>&1

### ---- Install web server and start it on port 80
yum install -y httpd
systemctl start httpd
systemctl enable httpd

### ---- Create Welcome web page
cat > /var/www/html/index.html <<EOF
<!DOCTYPE html>
<html>
<head>
<meta http-equiv="content-type" content="text/html; charset=UTF-8">
<title>OOW2019 HOL1512</title>
</head>
<body>
<p>
<meta charset="utf-8">
<span style="color: rgb(1, 116, 142); font-family: 'Open Sans', Helvetica, Arial, sans-serif; font-size: 20px; font-style: normal; font-variant-ligatures: normal; font-variant-caps: normal; font-weight: 400; letter-spacing: normal; orphans: 2; text-align: left; text-indent: 0px; text-transform: none; white-space: normal; widows: 2; word-spacing: 0px; -webkit-text-stroke-width: 0px; background-color: rgb(255, 255, 255); text-decoration-style: initial; text-decoration-color: initial; display: inline !important; float: none;">
Infrastructure as Code: Oracle Linux, Terraform, and Oracle Cloud Infrastructure [HOL1512]</span>
&nbsp;</p><p>The Web server is running.</p><p><br></p>
</body>
</html>
EOF
chown apache /var/www/html/index.html
chmod 644 /var/www/html/index.html

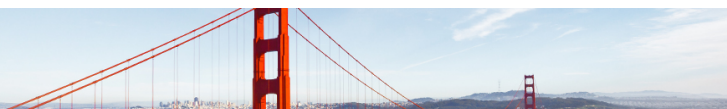
### ---- Open port tcp/80 in Linux Firewall
systemctl stop firewalld
sleep 5
firewall-offline-cmd --add-port=80/tcp
systemctl start firewalld
systemctl enable firewalld

### ---- Enable OCI utilities service
systemctl enable ocid.service
systemctl start ocid.service

### ---- Apply updates to Linux OS and reboot (disabled to save time during lab)
#yum update -y
#reboot
```

version.txt

last Update: August 28, 2019



4 APPENDIX B: DOCUMENTATION

- Oracle Cloud Infrastructure Documentation : <https://docs.cloud.oracle.com/iaas/Content/home.htm>
- Terraform main page (from Hashicorp) : <https://www.terraform.io/>
- Terraform provider for OCI:
 - Documentation : <https://www.terraform.io/docs/providers/oci/index.html>
 - Examples : <https://github.com/oracle/terraform-provider-oci/tree/master/docs/examples>
- Oracle Cloud Infrastructure blog : <https://blogs.oracle.com/cloud-infrastructure/>
- Oracle Linux blog : <https://blogs.oracle.com/linux/>
- Oracle Cloud Infrastructure utilities for Oracle Linux 7.x :
<https://blogs.oracle.com/wim/oci-utils-oracle-cloud-infrastructure-for-oracle-linux-package>