Instructions to schedule turn On/Off an OCI instance on Linux

Introduction

In this step by step guide, we will have a look how we can leverage on OCI CLI to automate the turning On/Off of OCI instances, as well as the scaling of said instance. For this guide, we will be using Autonomous Data Warehouse as an example. Any other instances will work the same way. For the scheduling, we will be running it on a Linux machine.

Why want to turn On/Off an OCI instance?

The reason of having to automatically turn On/Off instances is because of the reason that you would want to save cost, as well as resources. Instances can be turned off when it is not in use in order to manage your OCI cost and be turned on again when it is needed.

With automation, users of the instances would not need to keep reaching out to OCI Admins to turn on or off the instances. Admins will also decrease their workload by setting up the automation.

Prerequisites:

1. A Linux instance, either locally or on cloud
2. An OCI instance
3. Download the scripts provided. (There will be 2 scripts)

In this guide, you will learn how setup OCI CLI, how to use the tagging feature on OCI, as well as how to set up a Cron job to automate scheduling in Linux

Install OCI CLI Software on Linux

CLI is the utility to upload files into OCI object storage. This section documents the steps to install the CLI software. Following steps are the step by step method to install CLI on a Linux machine.

1. To run the Installer Script, run the command bash -c "$(curl -L https://raw.githubusercontent.com/oracle/oci-cli/master/scripts/install/install.sh)" in your Linux machine

```
[opc@ipxebastion ~]$ bash -c "$(curl -L https://raw.githubusercontent.com/oracle/oci-cli/master/scripts/install/install.sh)"
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100  4253  100  4253    0     0  19371      0 --:--:-- --:--:-- --:--:-- 19420
Downloading Oracle Cloud Infrastructure CLI install script from https://raw.githubusercontent.com/oracle/oci-cli/v2.4.13/scrip
ts/install/install.py to /tmp/oci_cli_install_tmp_EbpE.
##################################################################### 100.0%
Running install script.
-- Verifying Python version.
-- Python version 2.7.5 okay.
-- Verifying native dependencies.
-- Executing: 'rpm -q gcc libffi-devel python-devel openssl-devel'
-- One or more of the following native dependencies are not currently installed and may be required.
"sudo yum install -y gcc libffi-devel python-devel openssl-devel"

===> Missing native dependencies. Continue and install the following dependencies: gcc, libffi-devel, python-devel, openssl-de
vel? (Y/n): Y
```

To note some fields needed to be entered when running the script:

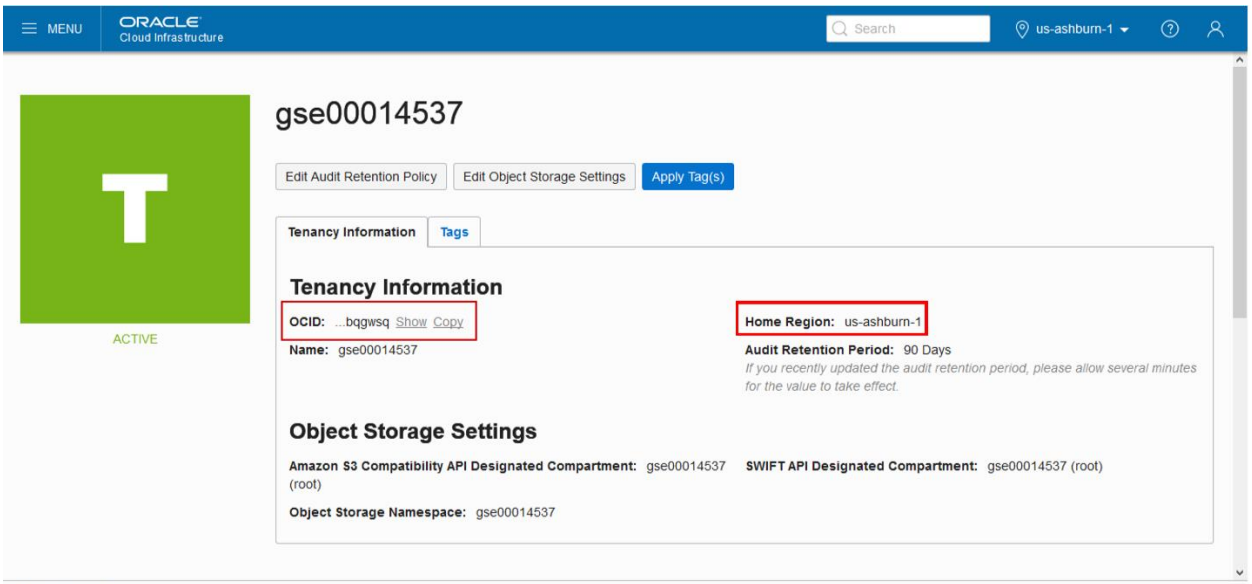
- In what directory would you like to place the install? (leave blank to use '/home/opc/lib/oracle-cli')
- In what directory would you like to place the 'oci' executable? (leave blank to use '/home/opc/bin')
- Modify profile to update your $PATH and enable shell/tab completion now? (Y/n): n

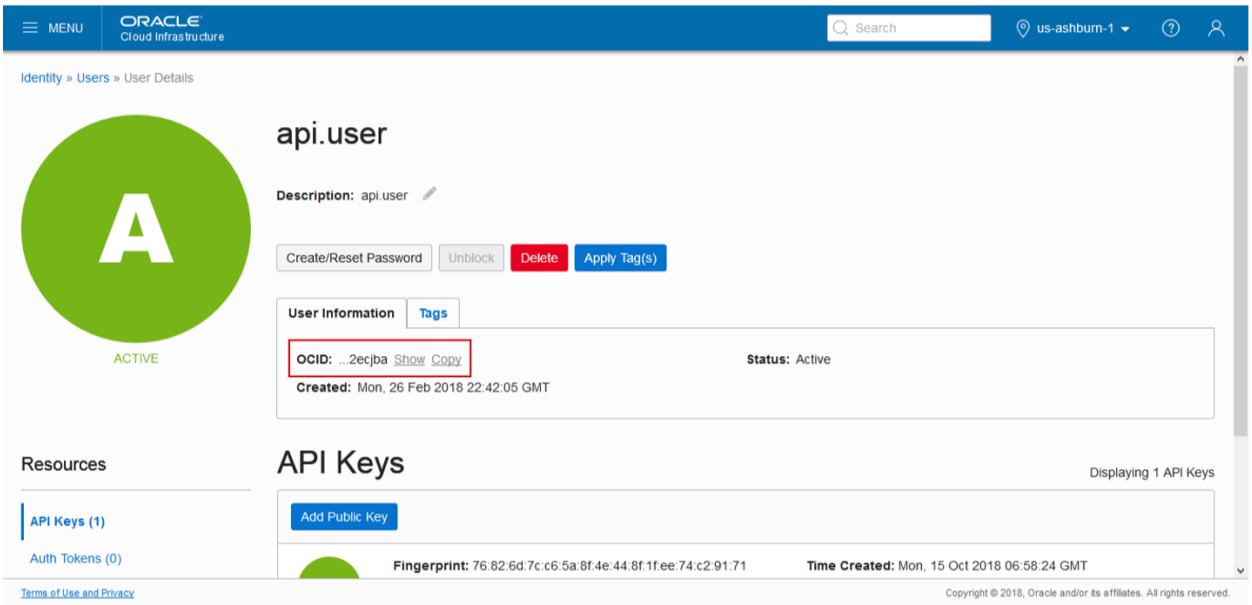
OCI-CLI Configuration

CLI OCI setup will generate the oci_api_key.pem and oci_api_key_public.pem based on the OCID information collected from the Cloud Portal page.

1. Collect information from OCI User & Tenancy

First you will be required to get the User OCID and Tenancy OCID as well as Tenancy Region from Oracle Cloud Infrastructure.





2. Run the setup tools to get the CLI ready to access OCI.

First step in on your Linux machine type in the command oci setup config to run the setup process.

```
login as: opc
Authenticating with public key "rsa-key-20200824"
Last login: Sun Mar  7 18:25:07 2021 from 42.190.105.135
[opc@lastoci ~]$ oci setup config
    This command provides a walkthrough of creating a valid CLI config file.

    The following links explain where to find the information required by this
    script:

    User API Signing Key, OCID and Tenancy OCID:

        https://docs.cloud.oracle.com/Content/API/Concepts/apisigningkey.htm#Oth
er

    Region:

        https://docs.cloud.oracle.com/Content/General/Concepts/regions.htm

    General config documentation:

        https://docs.cloud.oracle.com/Content/API/Concepts/sdkconfig.htm


Enter a location for your config [/home/opc/.oci/config]:
```

To note some fields needed to be entered when running the script:

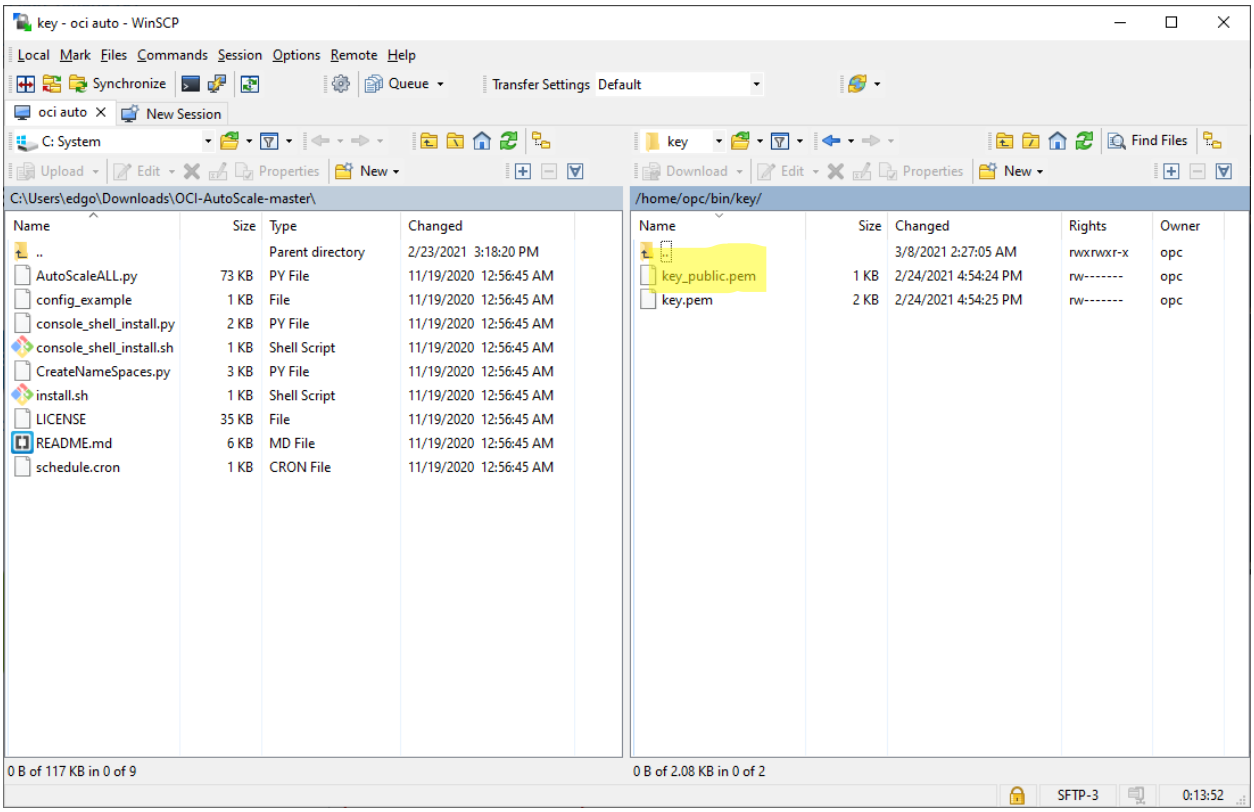
- Enter a location for your config [/home/opc/.oci/config]? (leave blank to use /home/opc/.oci/config)
- Enter a user OCID (Use your User OCID)
- Enter a tenancy OCID? (Use your tenancy OCID)
- Enter a Region (Use the region: ap-sydney-1)
- Do you want to generate a new RSA key pair (Type Y to create a new key pair that will be stored in the machine. This key will be used to add a fingerprint for your user to access OCI Instances)
- Enter a directory for your keys to be created: (Type in any name you want for the key folder)
- Enter a name for your key: (Type in any name you want your keys to be.)
- Enter a passphrase for your private key (Here you can define a password for your private key. If not required, just keep it blank and hit enter.)

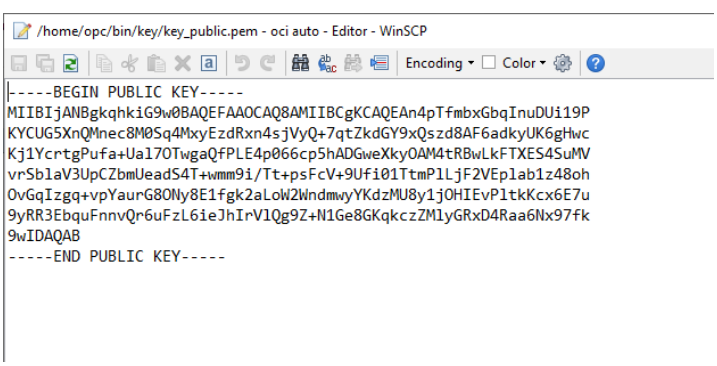
3. Add the public key to the User in OCI

Here we will be adding the created public key on our previous step to our User in OCI, allowing the script to be run through the user.

3.1 First open WinSCP to view your created public key.

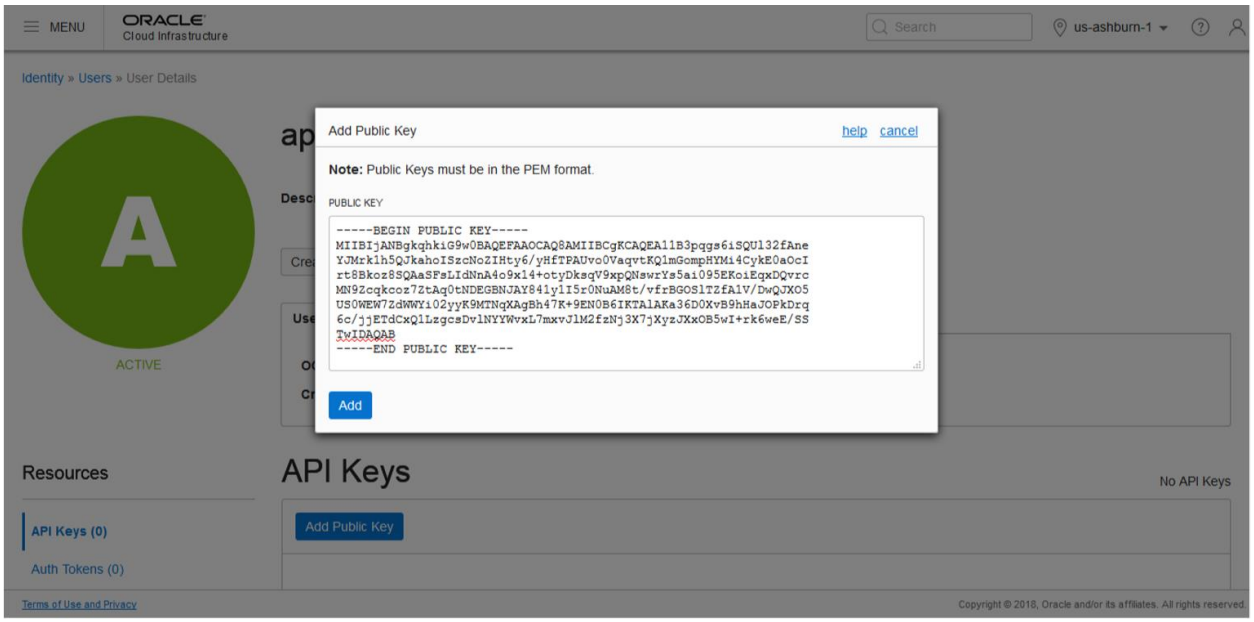
Connect to your WinSCP and open the folder in which you have stored your key. Next double click on the public.pem file to open it in an editor. Copy the text within the file.





3.2 Copy the public key and add it to your OCI user

Follow this step in OCI to add the public key. MENU -> Identity -> Users -> User -> API Keys -> Add Public Key.



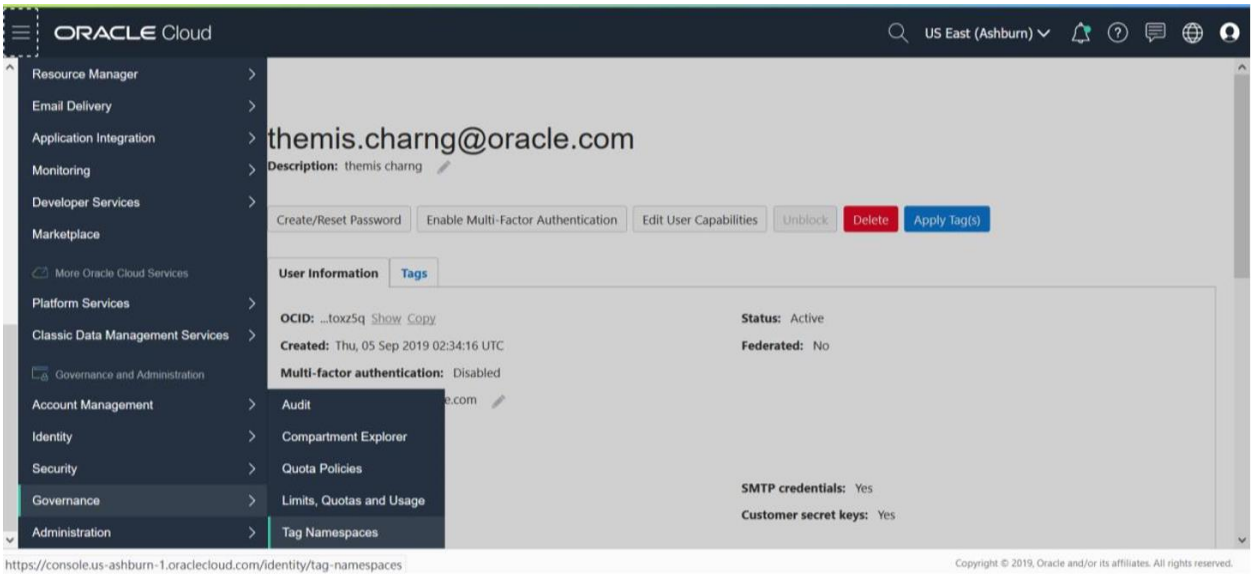
Click on Add to complete the step.

ADW and OAC configuration

To control what to power on/off, you need to create a predefined tag called Schedule. If you want to localize this, that is possible in the script. For the predefined tag, you need entries for the days of the week, weekdays, weekends and anyday. The tags names are case sensitive.

1. Create the Tag Namespace

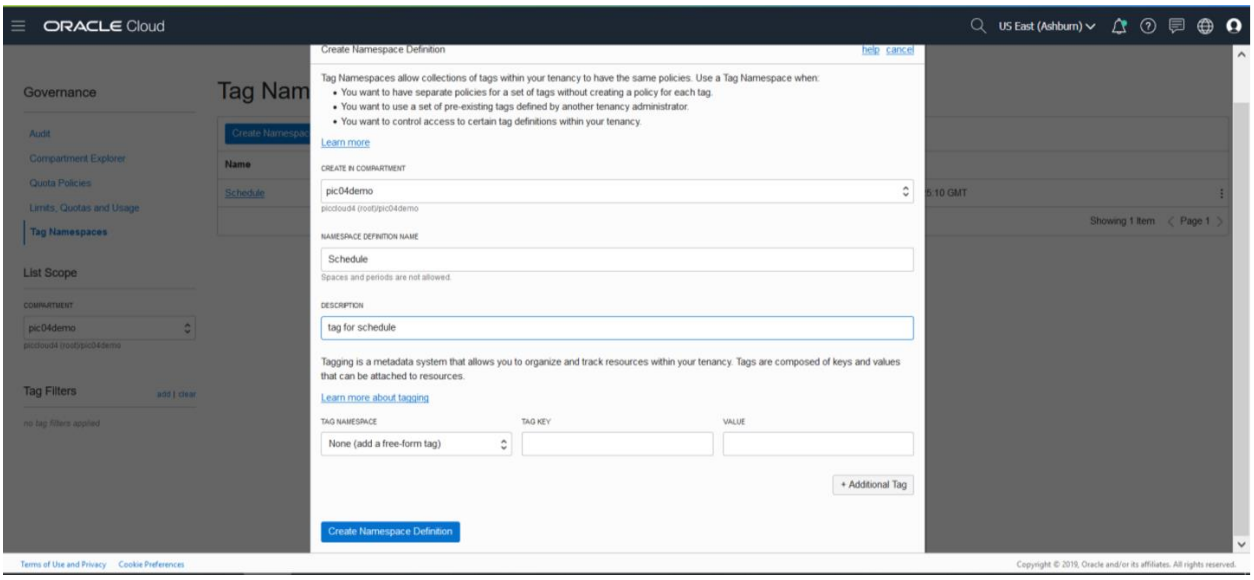
This section guides you on how to create a tag, so it can be added and used on multiple instances.



Create Namespace Definition -> Choose Compartment -> Fill in Namespace Definition Name -> Fill in Description

▪ Namespace Definition Name: “Schedule”

▪ Description: “(File in description you want to specify)”

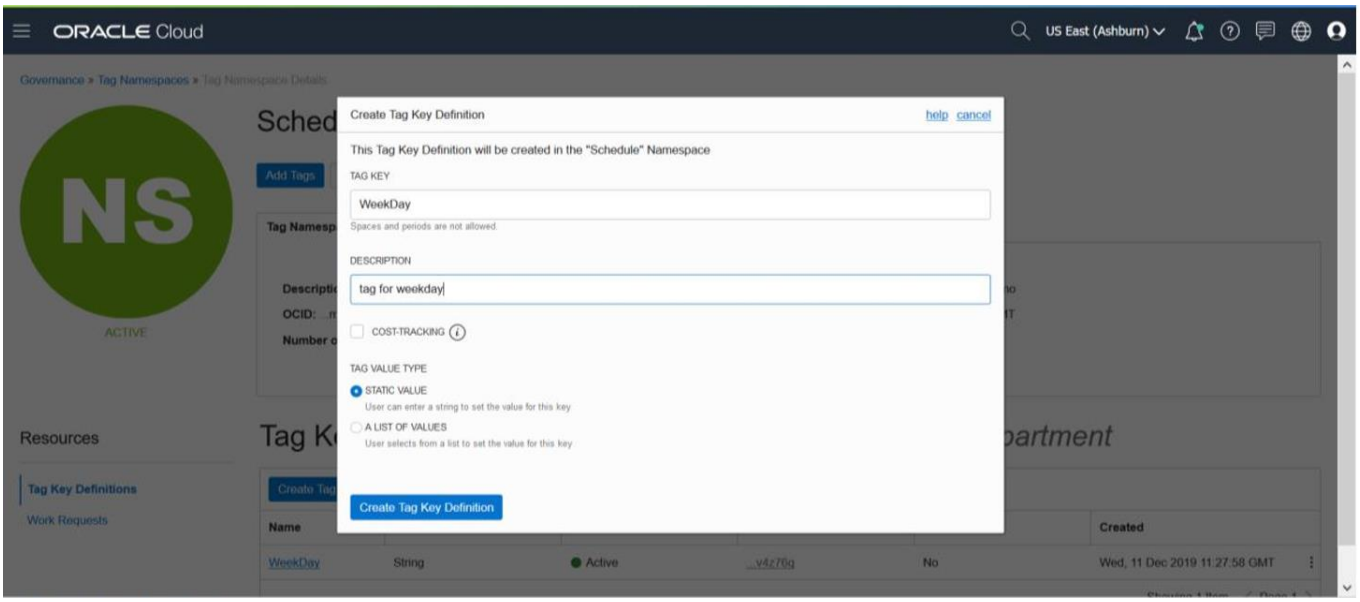


Create Tag Key Definition -> Fill in TAG KEY -> DESCRIPTION -> Choose TAG VALUE TYPE

▪ TAG KEY could be set as “WeekDay”, “Weekend”, “AnyDay” or a specific day e.g. Monday, Tuesday,…. A Weekend/Weekday tag overrules an AnyDay tag. A specific day of the week tag (ie. Monday) overrules all other tags in the scaling python script.

▪ DESCRIPTION should be filled in with words you want to specify.

▪ TAG VALUE TYPE should be chosen as STATIC VALUE for further configuration.

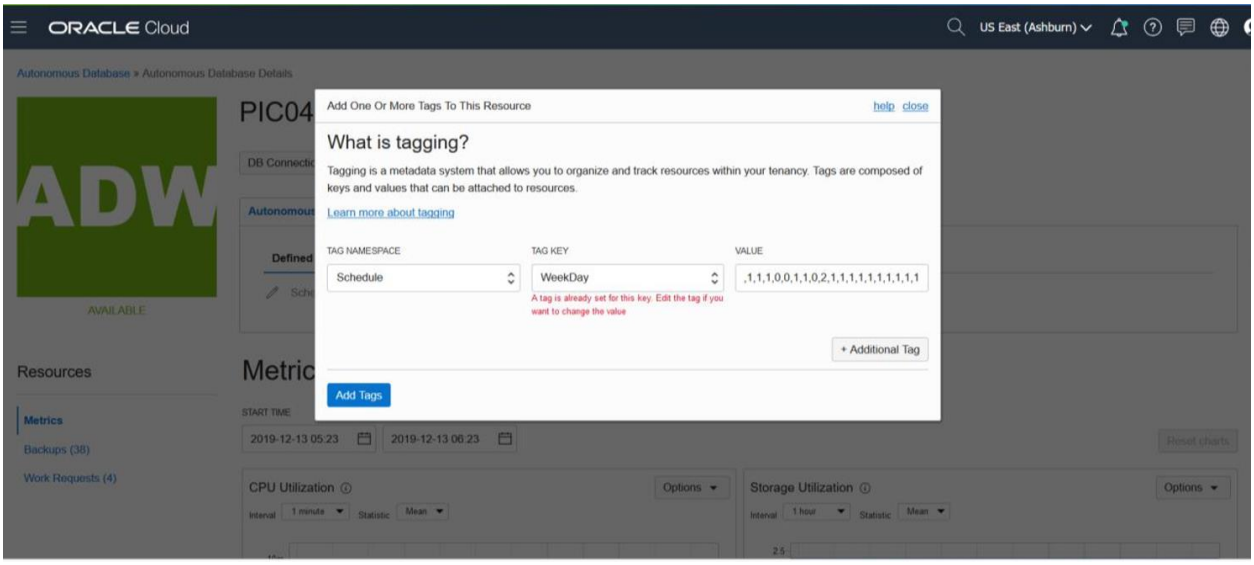


2. Add tag to ADW instances (Same method as for OAC)

A single resource can contain multiple tags. A Weekend/Weekday tag overrules an AnyDay tag. A specific day of the week tag (ie. Monday) overrules all other tags.

The value of the tag needs to contain 24 numbers (else it is ignored), seperated by commas. If the value is 0 it will power off the resource (if that is supported for that resource). Any number higher than 0 will re-scale the resource to that number. If the resource is powered off, it first will power-on the resource and then scale to the correct size.

Keep in mind the 24 numbers follows the 24 hour format, where the first number is 0000, second is 0100, and ends with 2300.



Prepare and Run the Script on Linux

Run python script to power on/ off and scale up/ down instances.

1. Setup environment for python script

In your Linux instance, run this command curl https://bootstrap.pypa.io/get-pip.py -o get-pip.py to download pip to within your Linux machine before we can run our scripts.

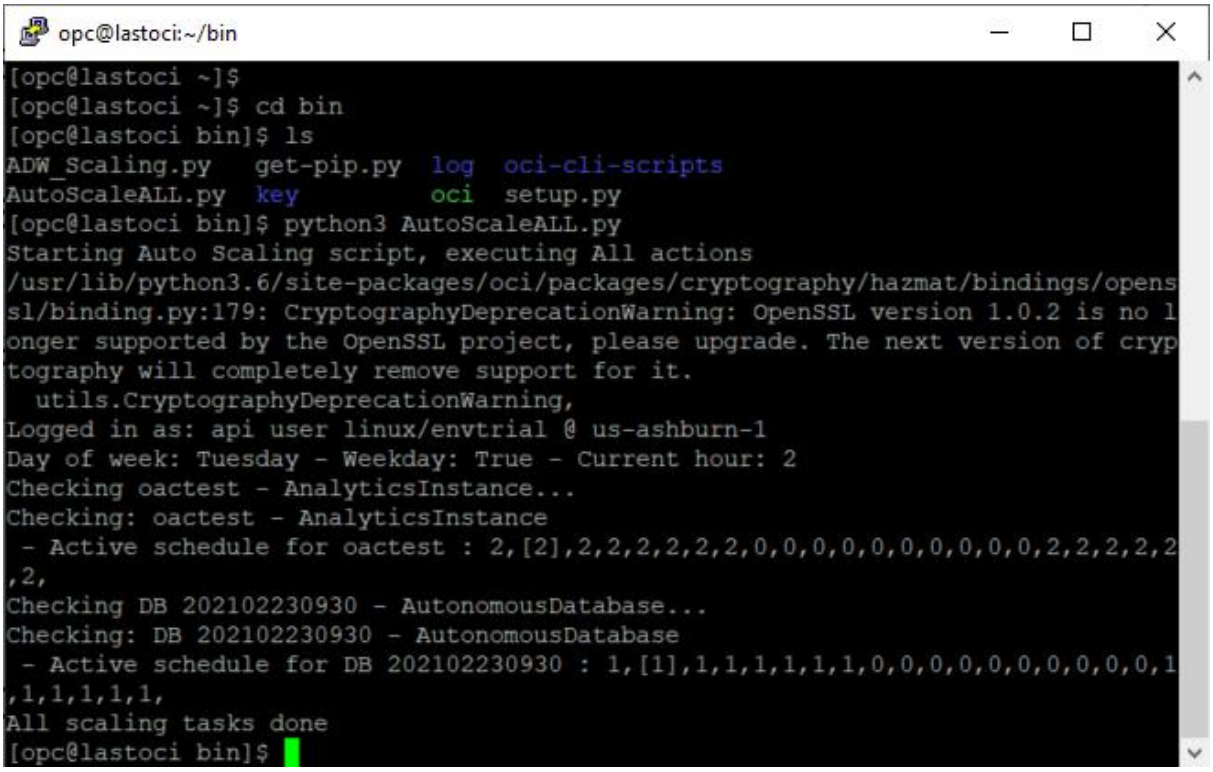
Next, we need to setup the pip. To do that, we just need to run the command python3 get-pip.py to start the setup.

Last step on the setup is to run the script setup.py in which you have added into your Linux machine's bin folder, through WinSCP. To run the setup script, just run python3 setup.py on your machine, in the location in which the file is stored.

2. Run the Auto Power On/Off Script

For running the script, ensure that the automation script is already added into the folder you want within Linux through WinSCP. Next, all you need to do is to run the command python3 AutoScaleALL.py. Once it is completed running, the instances ADW and OAC will then be turned on/off base on your tags on the instances.

If the tag is 0, the instance will be turned off. If it is 1 it will be turned on.

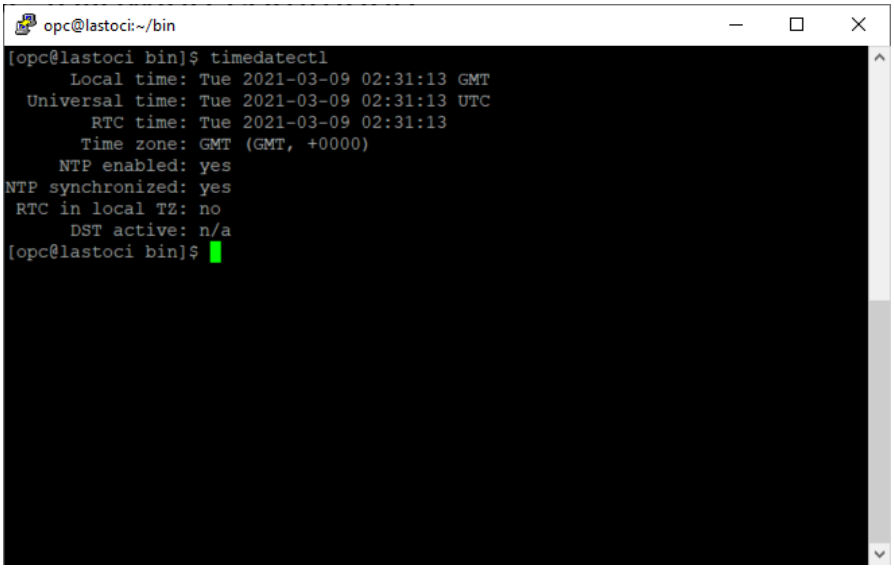


Create the Scheduler to run script when needed

In Linux, schedule scaling or power on/ off with windows task scheduler in every hour by using crontab

1. First, let's change the time zone of the Linux Machine.

Execute the command timedatectl in your Linux Machine to find out what timezone the machine is in.



Next to change the timezone, all you need to do is run the command sudo timedatectl set-timezone Australia/Sydney to change the timezone to Sydney.

```
opc@lastoci:~/bin
  Universal time: Tue 2021-03-09 02:31:13 UTC
        RTC time: Tue 2021-03-09 02:31:13
       Time zone: GMT (GMT, +0000)
     NTP enabled: yes
NTP synchronized: yes
 RTC in local TZ: no
      DST active: n/a
[opc@lastoci bin]$ sudo timedatectl set-timezone Australia/Sydney
[opc@lastoci bin]$ timedatectl
      Local time: Tue 2021-03-09 13:32:52 AEDT
  Universal time: Tue 2021-03-09 02:32:52 UTC
        RTC time: Tue 2021-03-09 02:32:52
       Time zone: Australia/Sydney (AEDT, +1100)
     NTP enabled: yes
NTP synchronized: yes
 RTC in local TZ: no
      DST active: yes
 Last DST change: DST began at
                  Sun 2020-10-04 01:59:59 AEST
                  Sun 2020-10-04 03:00:00 AEDT
 Next DST change: DST ends (the clock jumps one hour backwards) at
                  Sun 2021-04-04 02:59:59 AEDT
                  Sun 2021-04-04 02:00:00 AEST
[opc@lastoci bin]$
```

2. Configure Crontab to set the Scheduler

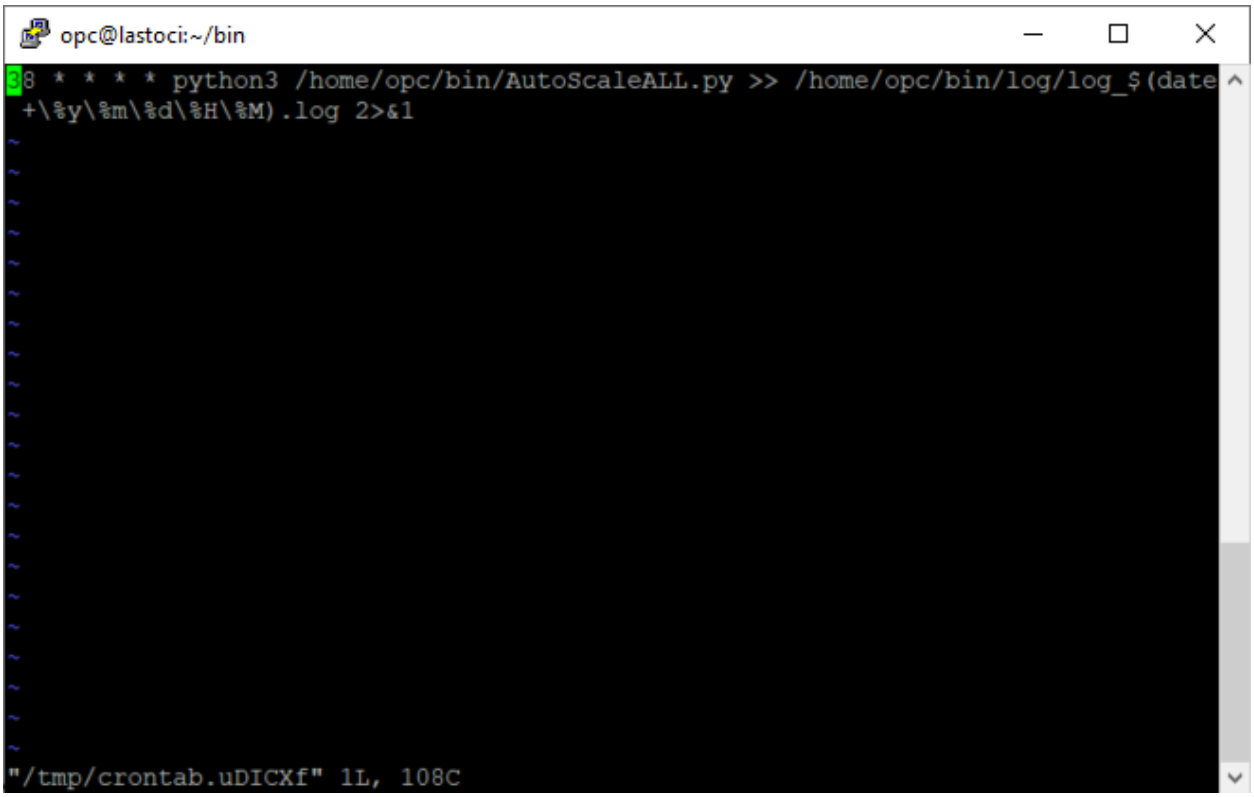
Run the below command to set up the Crontab: crontab -e

This will open an editor to allow you to create the schedule that you want. When you are in the editor, hit i on your keyboard to start editing.

Next, we will use the following script for the scheduling:

05 * * * * python3 /home/opc/bin/AutoScaleALL.py >> /home/opc/bin/log/log_$(date +\%y\%m\%d\%H\%M).log 2>&1

The above script will tell Linux to run the script every 5th minute of each our, for example it will run every 2.05pm, 3.05pm, 4.05pm and so on.



Once you have finished the script, enter ESC on your keyboard to stop editing the script. Once ESC is entered, type in :wq to save the script. You can use the command crontab -l to view the schedule that you have created.

Activity Complete

And that is all you need to do to completely implement the scripts on Linux as well as create a scheduler to run the script base on your requirement. Now you can easily scale as well as turn instances On/Off on your own accord. From the Admin's point, you will not need to again access the scripts or scheduling to change the requirement. The tags on OCI instance is the only thing needed to be changed.