Automation with Ansible

openstack®

nectar

Automation

- Deploying complex cloud systems involves many moving parts
  - The capability of managing large number of systems
  - Easy to forget some of the details in configuring the systems
  - Manual process is always error-prone
  - Snapshots are monolithic, provides no record of what has changed

- Benefits of automation
  - Provides a record of what you did
  - Codifies knowledge about the system
  - Makes process repeatable
  - Makes it programmable – “Infrastructure as Code”

Classification of Scripting tools

- Cloud-focused

  Used to interact with Cloud services.

  - Apache JClouds (Java-based - supports multiple clouds)
  - Boto (Python - supports AWS and OpenStack)
  - OpenStackClient (Python - supports OpenStack)
  - CloudFormation (YAML/JSON - supports AWS, OpenStack Heat)

- Shell scripts
  - Bash
  - Perl

Classification of Scripting Tools

- Configuration management (CM) tools

  Configuration management refers to the process of ***systematically*** handling ***changes*** to a system in a way that it ***maintains integrity*** over time.

  Automation is the mechanism used to make servers reach a desirable state, previously defined by provisioning scripts using tool-specific languages and features.

  - Chef (uses Ruby for creating cookbooks)
  - Puppet (uses its own configuration language)
  - **Ansible** (use YAML to express playbooks)
  - Fabric (Python library that uses SSH for application deployment and administration tasks)
  - Terraform, SaltStack, Docker, …

What is configuration Management? Everything you need to know.

Ansible

- An automation tool for configuring and managing computers. Finer grained set up and configuration of software packages.
- Initial release: Feb. 2012
- Combines multi-node software deployment
- Ad-hoc task execution and configuration management

Configuring thousands of machines manually!?

Ansible: Features

- Easy to learn
  - Playbooks in YAML, templates in Jinja2 etc.
  - Sequential execution
- Minimal requirements
  - No need for centralized management servers/daemons
  - Single command to install (*pip install ansible*)
  - Uses SSH to connect to target machine
- Idempotent (repeatable)
  - Executing N times no different to executing once
  - Prevents side-effects from re-running scripts
- Extensible

  Write your own modules

Ansible: More Features

- Supports push or pull

  Push by default but can use cron job to make it pull

- Rolling updates

  Useful for continuous deployment / zero downtime deployment

- Inventory management

  - Dynamic inventory from external data sources
  - Execute tasks against host patterns

- *Ansible Vault* for encrypted data

  $ ansible-vault create demo.yaml

  $ ansible-vault decrypt demo.yaml

  $ ansible-vault encrypt demo.yaml

  $ ansible-vault rekey demo.yaml

Ansible: More Features

- Ad-hoc commands

  Execute a one-off command against your inventory

  $ ansible -i inventory_file -u ubuntu -m shell -a “reboot”

- Ansible Galaxy (https://galaxy.ansible.com/)
- Ansible Tower: Enterprise mission control for Ansible

  Dashboard, System Tracker, etc.

What is Ansible and what can it automate?

Ansible: YAML & Jinja2

- Ansible Playbooks are expressed in YAML.
  - YAML: **Y**AML **A**in't **M**arkup **L**anguage
  - YAML is a human friendly data serialization standard for all programming languages.
  - YAML Syntax: https://docs.ansible.com/ansible/latest/reference_appendices/YAMLSyntax.html

- Ansible uses Jinja2 templating for dynamic expression
  - Jinja2 is a modern and designer-friendly templating language for Python, modelled after Django's templates.
  - Jinja2 introduction: https://docs.ansible.com/ansible/latest/user_guide/playbooks_templating.html#templating-jinja2

Ansible: Installation Guide

- Linux (Ubuntu)

  $ sudo apt-get update && sudo apt-get install software-properties-common

  $ sudo apt-add-repository --yes --update ppa:ansible/ansible

  $ sudo apt-get install ansible

- macOS
  - Brew (https://brew.sh/)

    $ brew install ansible
  - Pip

    $ sudo pip install ansible

Ansible: Installation Guide

- Windows 10/11 (WSL)
  - Install Windows Subsystem for Linux

    https://docs.microsoft.com/en-us/windows/wsl/install-win10
  - Install Ansible

    See guide for Linux (Ubuntu)
  - Tutorial

    https://www.youtube.com/watch?v=4sMFybv74Uo

- Ansible guide:

https://docs.ansible.com/ansible/latest/installation_guide/intro_installation.html

Ansible: Structure

- Ansible scripts are called *playbooks,* written as simple YAML files
- Structured in a simple folder hierarchy

```
Playbook folder
|- variables
|    |_ vars.yaml
|- inventory
|    |_ inventory.ini
|- roles
|    |- defaults
|    |   |_main.yaml
|    |- tasks
|    |   |- task1.yaml
|    |   |_task2.yaml
|    |_ templates / files
|_ playbook.yaml
```

```
[webservers]
foo.example.com
128.250.0.1

[dbservers]
one.example.com
two.example.com
```

Ansible: Playbooks

- Executed sequentially from a YAML file

```
- hosts: webservers
  vars:
    package: ['httpd', 'python-dev']

  tasks:
    - name: Ensure the latest Apache and Python are installed
      ansible.builtin.apt:
          name: "{{ package }}"
          state: latest
    - name: Write the Apache config file
      ansible.builtin.file:
          src: /srv/httpd.conf
          dest: /etc/httpd.conf
    - name: Ensure Apache is restarted
      ansible.builtin.service:
          name: httpd
          state: restarted
```

```
[webservers]
www[01:50].example.com
192.168.0.[1:254]
```

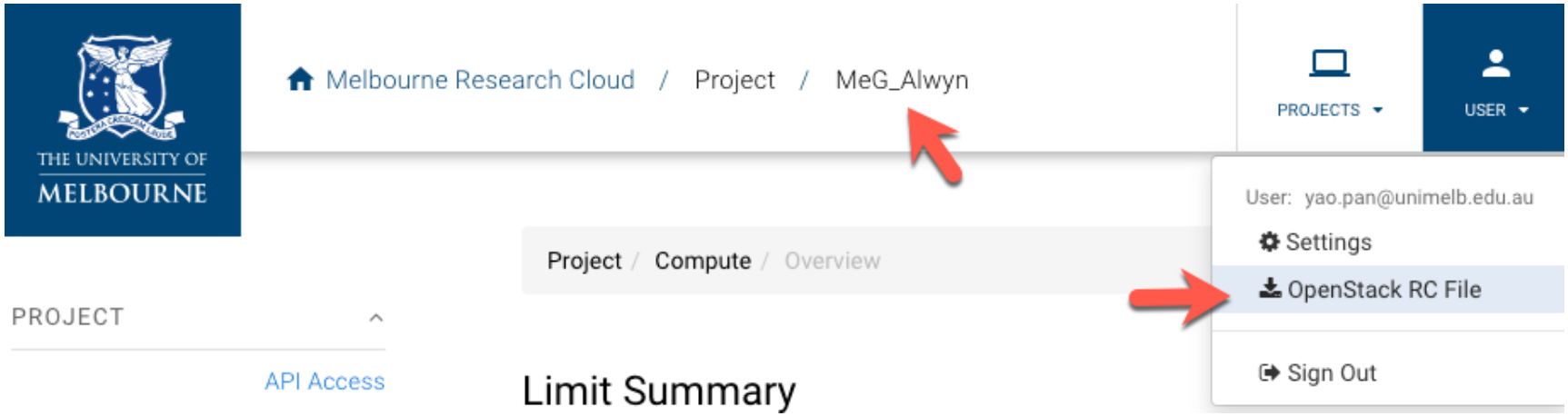
Ansible Demo: Melbourne Research Cloud

Ansible Demo 1:

- Connect to Melbourne Research Cloud
- List all images
- Create volumes
- Create security groups with security rules
- Launch an instance and attach the volume and security groups
- Create snapshots of the volumes

Ansible Demo: Melbourne Research Cloud

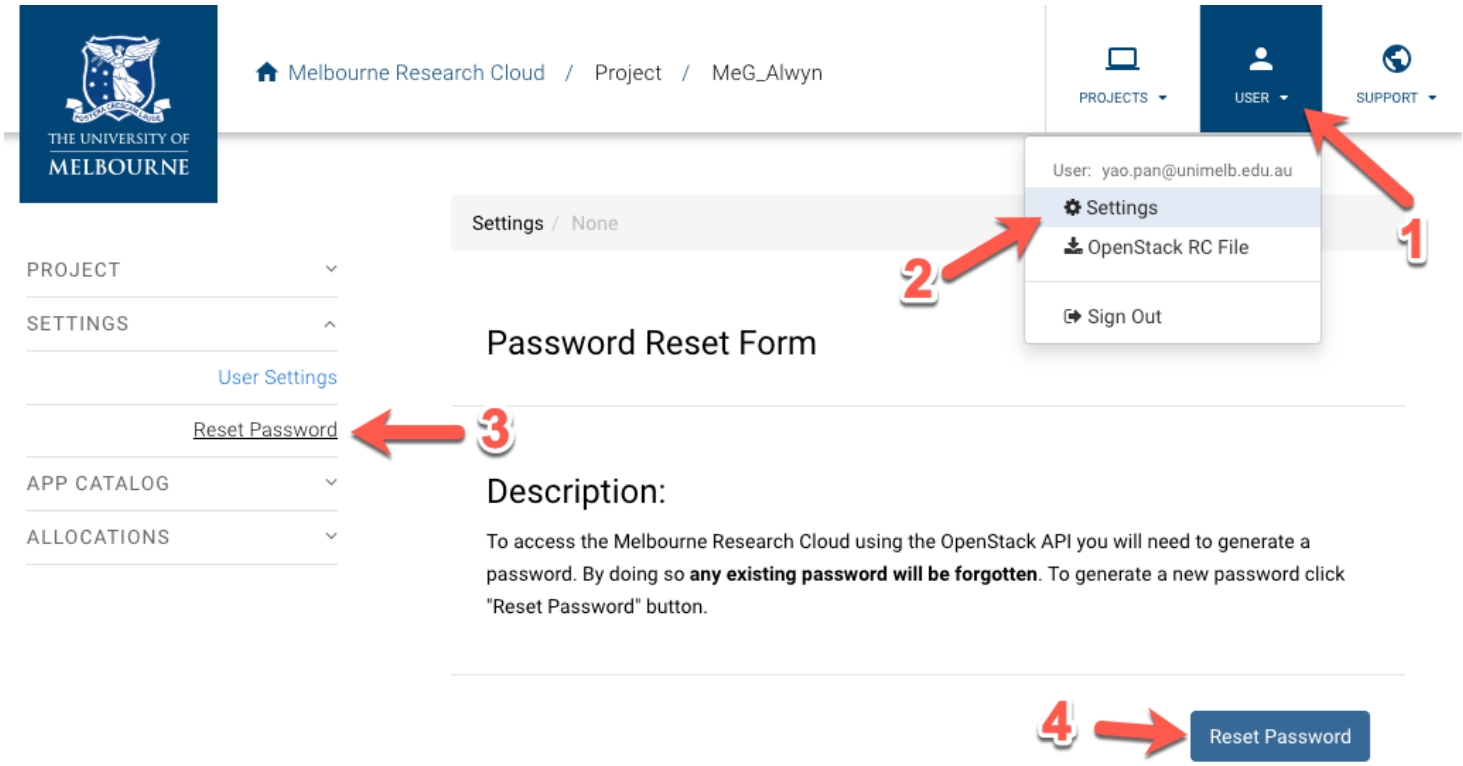
- Prerequisites:
  1. Login to https://dashboard.cloud.unimelb.edu.au
  2. Download *openrc.sh* from Dashboard
     - Make sure the correct project is selected
     - Download the OpenStack RC File



Ansible Demo: Melbourne Research Cloud

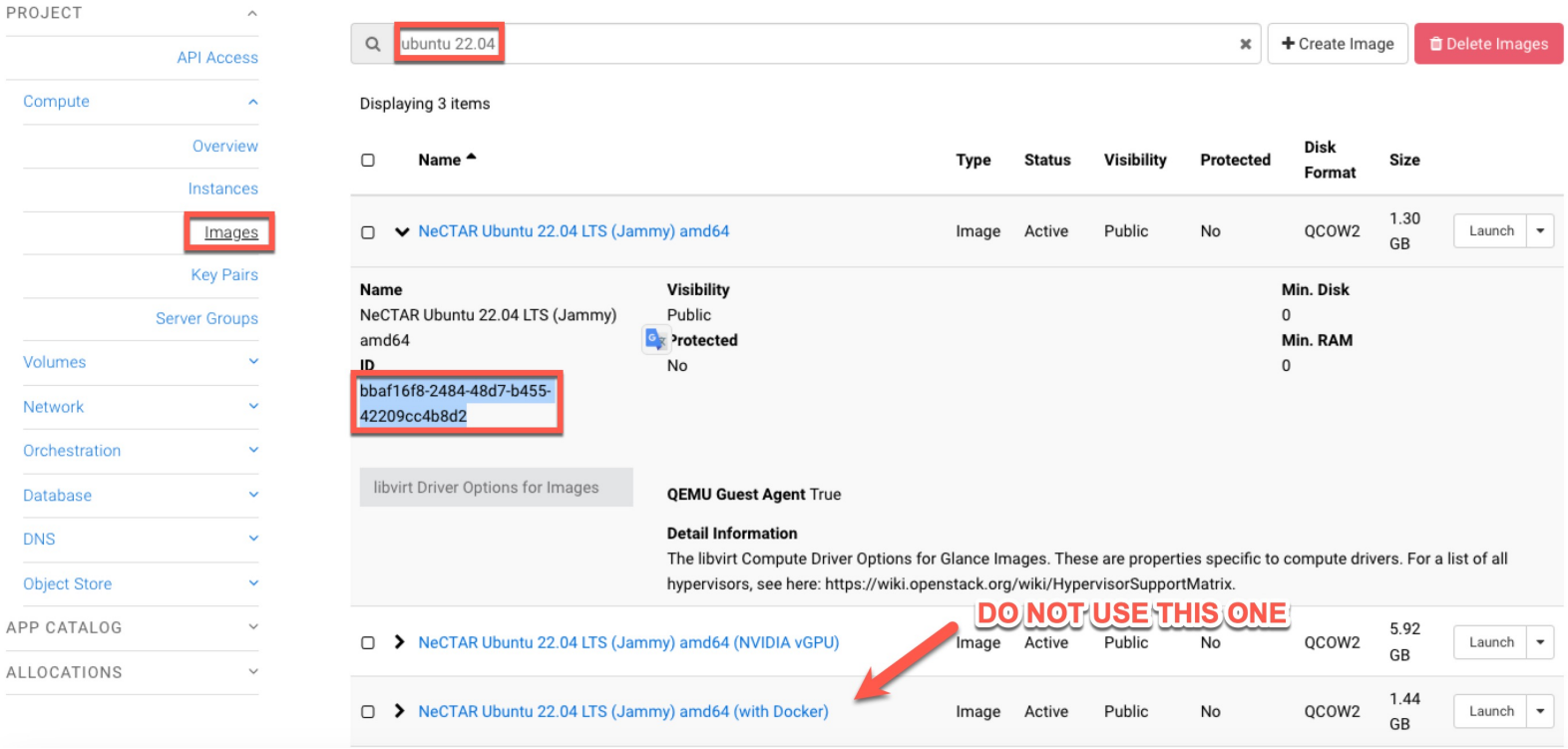
- Prerequisites:
  3. Reset API password

     Dashboard -> User -> Settings -> Reset Password



Ansible Demo: Melbourne Research Cloud

- Prerequisites:
  4. Instance Flavor: *uom.general.1c4g*
  5. Availability Zone: *melbourne-qh2-uom*
  6. Image Id: *bbaf16f8-2484-48d7-b455-42209cc4b8d2*
  7. Dashboard -> Compute -> Images



Ansible Demo: Melbourne Research Cloud

- *playbook* to interact with Melbourne Research Cloud
  - List all images
    openstack.cloud.image_info
  - Create a volume
    openstack.cloud.volume
  - Create a security group with rules
    openstack.cloud.security_group, *openstack.cloud.security_group_rule*
  - Launch an instance and attach the volume and security group
    openstack.cloud.server
  - Create a snapshot of the volume
    openstack.cloud.volume_snapshot
  - Other useful Ansible modules
    ansible.builtin.apt, *ansible.builtin.pip*, *ansible.builtin.set_fact*
    ansible.builtin.debug, *ansible.builtin.wait_for*, *ansible.builtin.add_host*
    ansible.builtin.gather_facts
    become, *register*, *loop*, *when*

Run *playbook*:

$ . ./openrc.sh; ansible-playbook [--ask-become-pass] mrc.yaml

Ansible Demo: Quiz

- Retrieve facts about the instance created
  - Create an instance and retrieve facts about that instance
- Attach existing volume(s) to an existing instance
  - Create a new volume
  - Attach the new volume to an existing instance

  Hint:
    - *Go to the Ansible Documentation and search docs for "openstack"*
    - *Find relevant Ansible module*
- Add / remove existing Security Group to existing instance
  - Create a new Security Group
  - Add Security Group Rules
  - Attach the Security Group to an existing instance

  Hint: openstack.cloud.server

Ansible Demo: WordPress

- Simple *playbook* to deploy a *WordPress* instance with *Docker**
  - Install dependencies
    ansible.builtin.apt, *ansible.builtin.pip*
  - Mount volumes and make filesystems
    community.general.filesystem, *ansible.builtin.stat*
    ansible.builtin.file, *ansible.posix.mount*
  - Install *Docker* on one of the volumes
    ansible.builtin.apt_key, *ansible.builtin.apt_repository*
  - Deploy a *WordPress* instance with *Docker*
    ansible.builtin.template, *community.docker.docker_compose*

** Docker will be introduced in the coming workshops*

References

- Hwang, Dongarra & Fox, 2011. Distributed and Cloud Computing, 1st Edition. Elsevier.
- Armbrust et al., 2010. A view of cloud computing. Communications of the ACM 53, 50-58. http://doi.acm.org/10.1145/1721654.1721672
- Revolution Not Evolution: How Cloud Computing Differs from Traditional IT and Why it Matters http://www.rackspace.com/knowledge_center/whitepaper/revolution-not-evolution-how-cloud-computing-differs-from-traditional-it-and-why-it
- The 10 Most Important Companies In Cloud Computing http://www.businessinsider.com.au/10-most-important-in-cloud-computing-2013-4?op=1#a-word-about-clouds-1
- Ansible, https://www.ansible.com/

Q & A

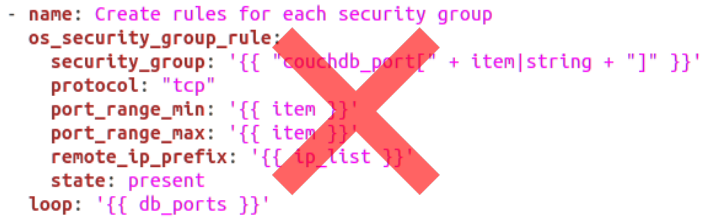
Question 1

I am trying to set up a CouchDB cluster, and in order to do that I'd have to modify my security group setting to open up the ports that CouchDB nodes need to use for communication. … I don't really feel like using a nested loop, … Just wondering if there is any other way of doing this?

Server A: open 5984, 5986, 4369, 9100 – 9200 to Server B and Server C

Server B: open 5984, 5986, 4369, 9100 – 9200 to Server A and Server C

Server C: open 5984, 5986, 4369, 9100 – 9200 to Server A and Server B



Problem: remote_ip_prefix only takes a single CIDR notation.

Solution? A nested loop with "with_nested"? Try and tell me if it works ;p

Solution! Use the security group as the source of the remote in a security rule.

Question 1

Add Rule

Rule *
Custom TCP Rule

Description

Direction
Ingress

Open Port *
Port Range

From Port *
9100

To Port *
9200

Remote *
Security Group

Security Group
Demo_Q1 (current)

Ether Type
IPv4

Description:

Rules define which traffic is allowed to instances assigned to the security group. A security group rule consists of three main parts:

Rule: You can specify the desired rule template or use custom rules, the options are Custom TCP Rule, Custom UDP Rule, or Custom ICMP Rule.

Open Port/Port Range: For TCP and UDP rules you may choose to open either a single port or a range of ports. Selecting the "Port Range" option will provide you with space to provide both the starting and ending ports for the range. For ICMP rules you instead specify an ICMP type and code in the spaces provided.

Remote: You must specify the source of the traffic to be allowed via this rule. You may do so either in the form of an IP address block (CIDR) or via a source group (Security Group). Selecting a security group as the source will allow any other instance in that security group access to any other instance via this rule.

Cancel Add

Question 2

Q: Random errors, even when run simple “hello world” playbook or the playbooks used in demo.

```
TASK [wordpress : Run docker compose] ******************************************
fatal: [172.26.38.146]: FAILED! => {"changed": false, "msg": "Cannot have both t
he docker-py and docker python modules installed together as they use the same n
amespace and cause a corrupt installation. Please uninstall both packages, and r
e-install only the docker-py or docker python module. It is recommended to insta
ll the docker module if no support for Python 2.6 is required. Please note that
simply uninstalling one of the modules can leave the other module in a broken st
ate."}
```

A: Check Ansible version and make sure the latest Ansible has been installed.

ansible --version

Question 3

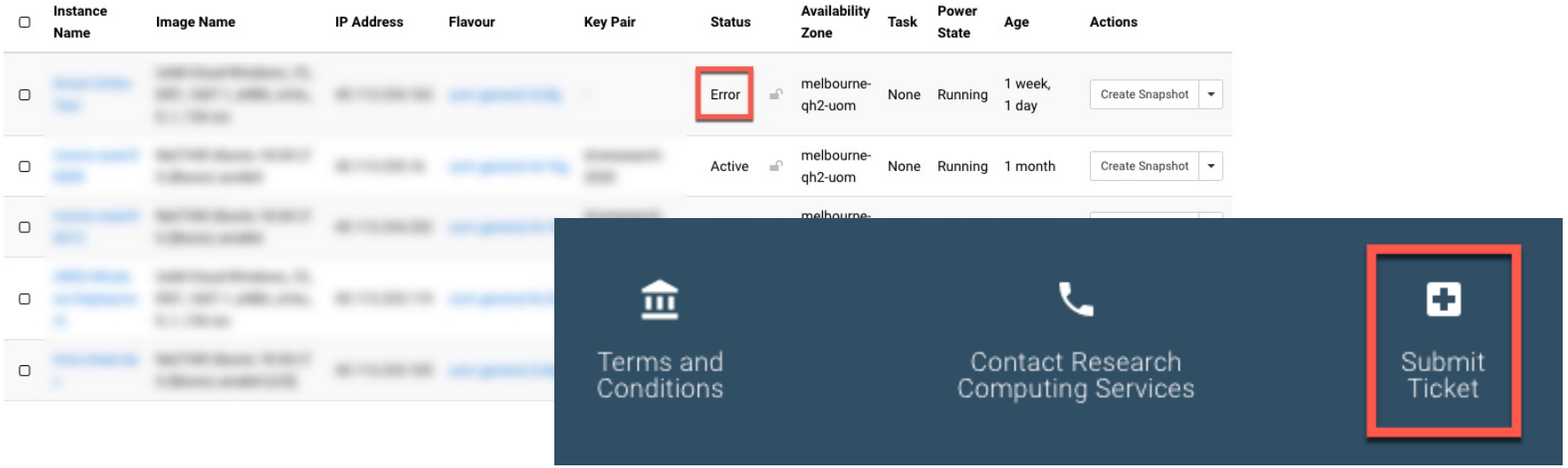
Q: Cannot SSH to the server just created on NeCTAR / MRC

```
- name: Wait for connection
  wait_for:
    host: "{{ item.openstack.public_v4 }}"
    port: 22
    timeout: 120
    search_regex: OpenSSH
  loop: '{{ os_instance.results }}'
  when: item.openstack is defined
```

A: The servers are not ready. Use wait_for module to wait until the servers are ready.

Question 4

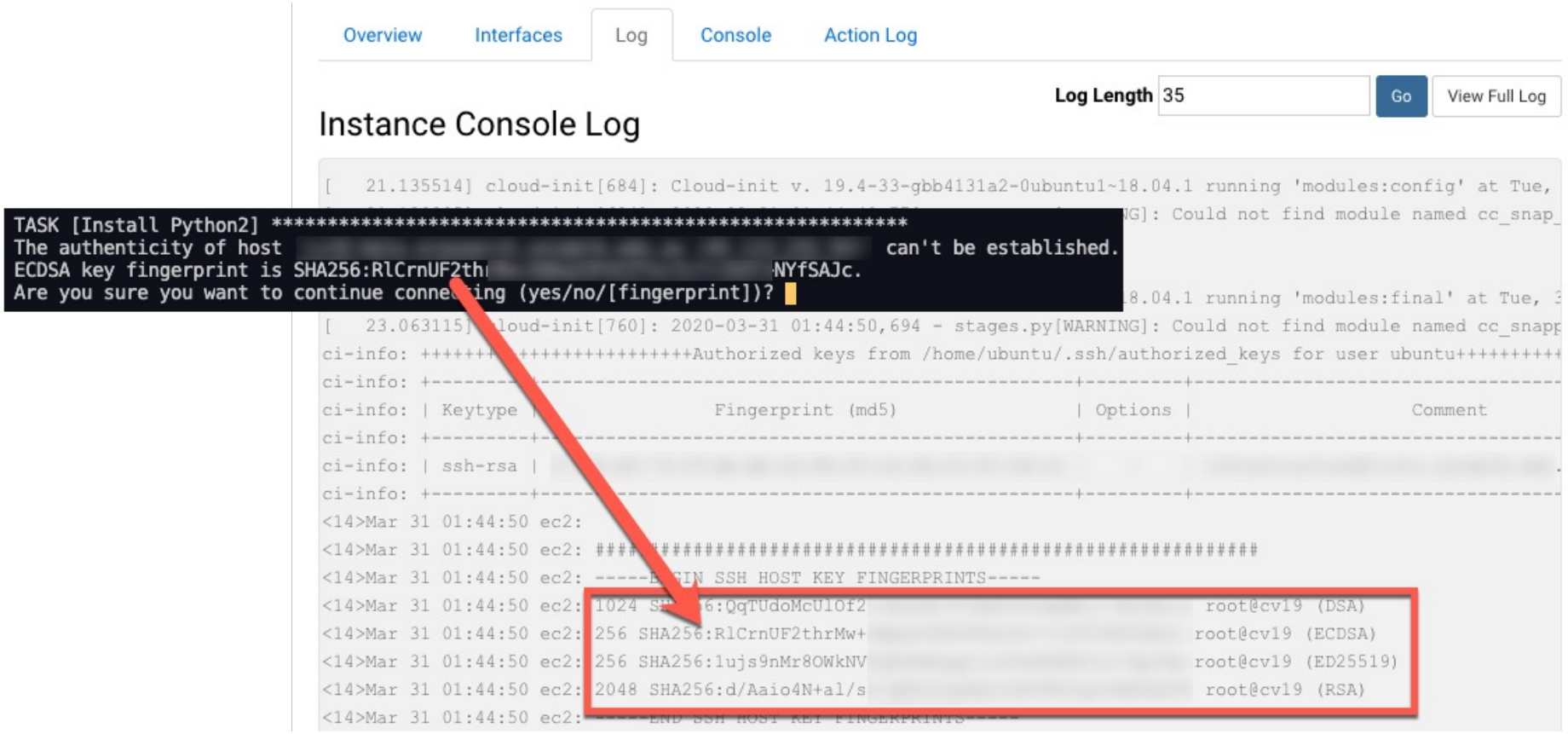
Q: I created an instance on MRC and the status showing “Error”



A: Welcome to the FREE cloud. It happens, try to delete the instance if you can. Otherwise Submit a Ticket and the MRC staff will help you out.

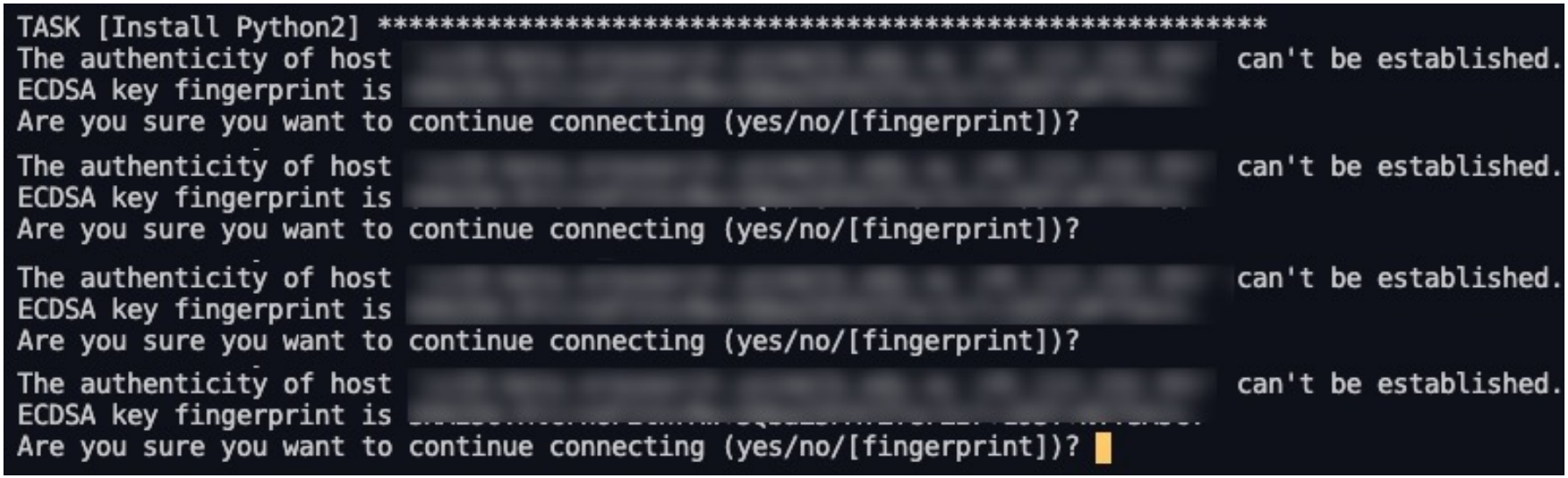
Question 5

Q: How to check the host key fingerprint?



Question 6

Q: Ansible keeps prompting host key confirmation, sometimes type “yes” only works for the first host.



Question 6

A: SSH with “-o StrictHostKeyChecking=no” option

[COMP90024]

[COMP90024:vars]
ansible_python_interpreter=/usr/bin/python3
ansible_user=ubuntu
ansible_ssh_private_key_file=<some private key>
ansible_ssh_common_args='-o StrictHostKeyChecking=no'

Question 7

Q: How to safely store the Ansible variables for credentials in a Git repo?

A: Don't push it or use Ansible Vault.

To encrypt a file: *ansible-vault encrypt file.yaml*

To decrypt a file: *ansible-vault decrypt file.yaml*

To view an encrypted file: *ansible-vault view file.yaml*

To run an encrypted playbook file:

ansible-playbook --vault-id @prompt file.yaml

Question 8

Q: What can we do if we want multiple instances to access data stored in a volume? Can we do this by Ansible?

A: Yes and No.

Yes: OpenStack supports multi-attach since Cinder version 3.50.

> ⚠ Warning
>
> It is the responsibility of the user to ensure that a multiattach or clustered file system is used on the volumes. Otherwise there may be a high probability of data corruption.

No: NeCTAR / MRC does not support multi-attach