

ДОДАТОК А

СЛАЙДИ ПРЕЗЕНТАЦІЇ

ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
РАДІОЕЛЕКТРОНІКИ

КВАЛІФІКАЦІЙНА РОБОТА МАГІСТРА

На тему:

Дослідження інструментів автоматизації розгортання
приватної гіперконвергентної хмари на базі OpenStack

Студент:	Коротіч Олексій Віталійович
Група:	ІМІМ-20-2
Керівник:	Костромицький Андрій Іванович

Зміст

- Види хмар
- OpenStack та його компоненти
- Інструменти автоматизації розгортання Openstack
- Порівняння інструментів автоматизації

Види хмар

- Публічні хмари
- Приватні хмари
- Гібридні хмари

Публічні хмари

Переваги

- Еластична структура
- Відносно низька ціна
- Оптимізація витрат на інфраструктуру
- Просте і зрозуміле використання

Недоліки

- Ризики пов'язані з безпекою
- Ризик втрати даних
- Ризики використання неякісного обладнання

Приватні хмари

Переваги

- Ізольованість IT-інфраструктури
- Надійне зберігання даних
- Можливість довгострокового вкладення обладнання для підтримки роботи системи
- Доступна конфігурованість

Недоліки

- Обмежений обсяг потужностей
- Необхідність тривалої підготовки до розгортання
- Висока ціна (покупка/оренда) обладнання
- Необхідність наявності IT-фахівців для адміністрування

Гібридні хмари

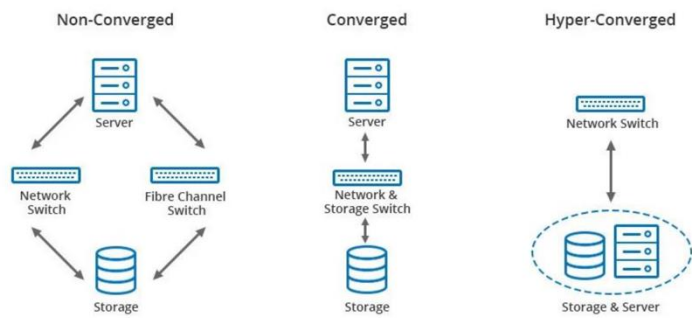
Переваги

- Створення відповідного середовища для тестування та розробки
- Зниження ймовірності збоїв системи
- Значне спрощення локальної інфраструктури
- Велике навантаження високою доступністю
- Масштабованість
- Постійний доступ до даних

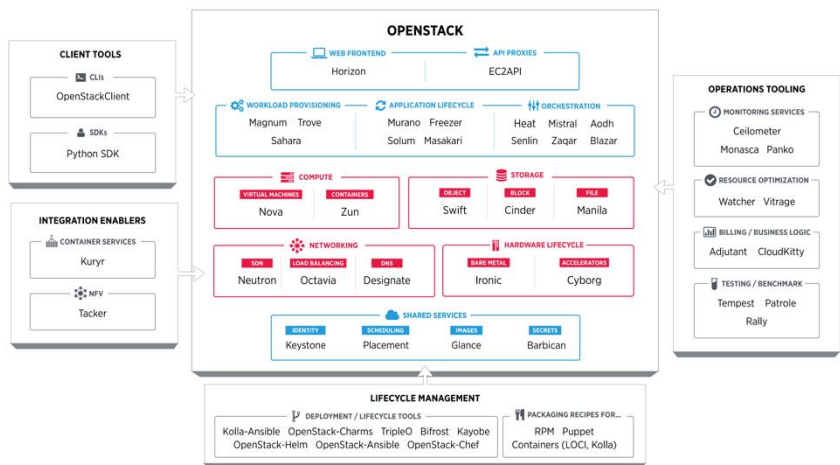
Недоліки

- Ризик втрати даних при передачі
- Ускладнене резервування
- Відстеження знаходження даних

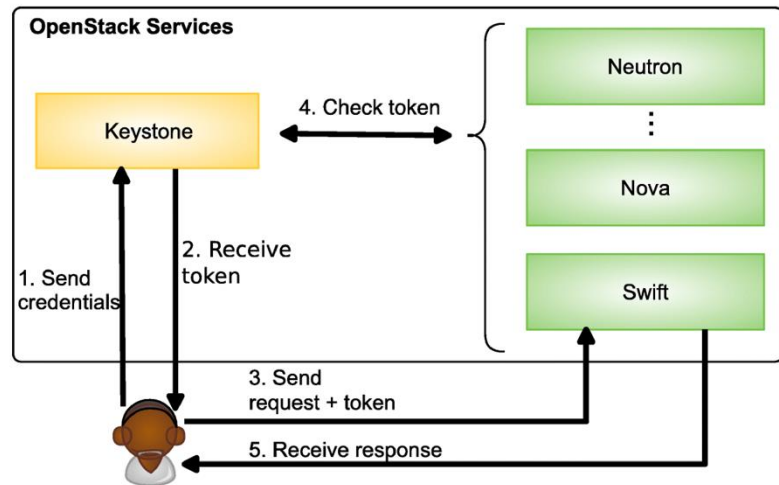
Гіперконвергентність



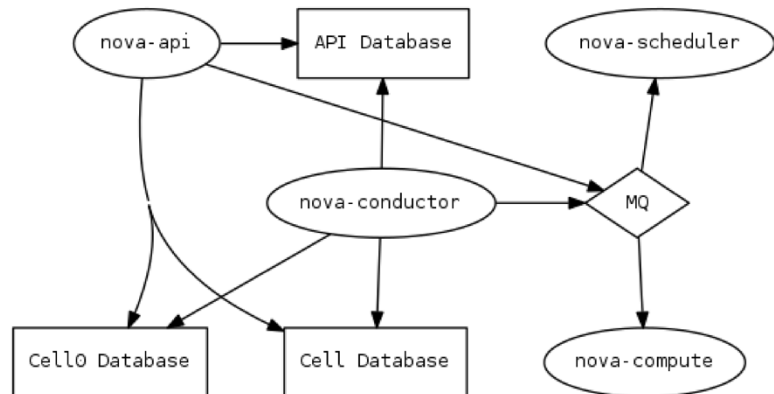
OpenStack та його КОМПОНЕНТИ



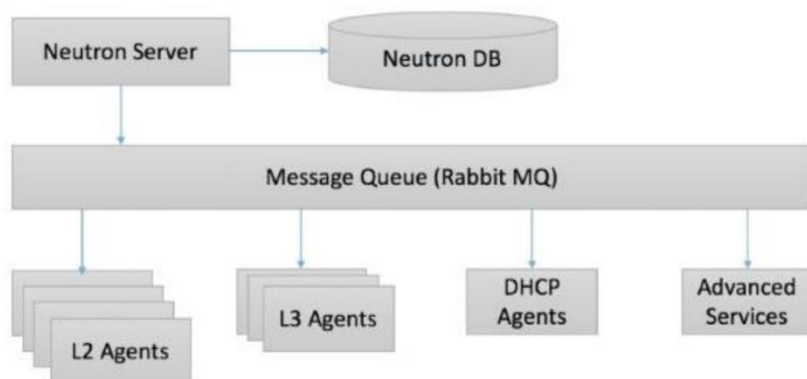
Keystone



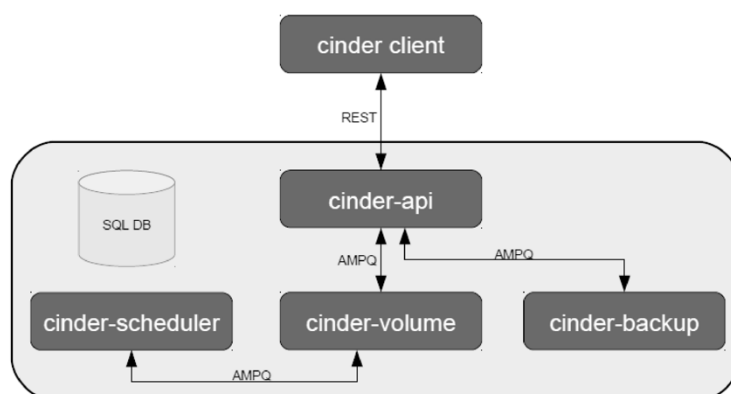
Nova



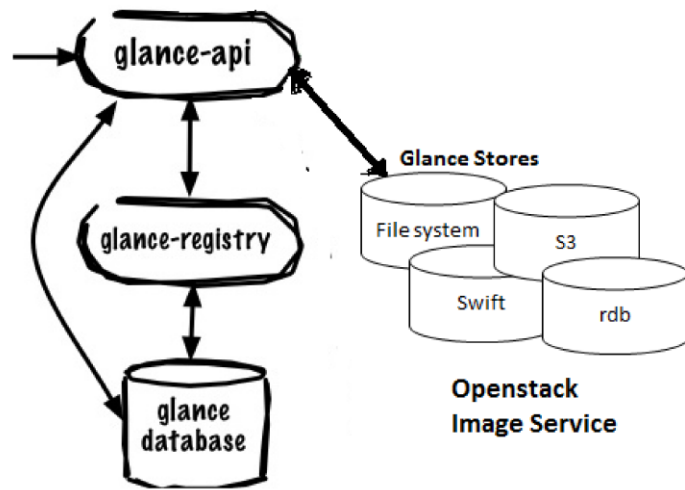
Neutron



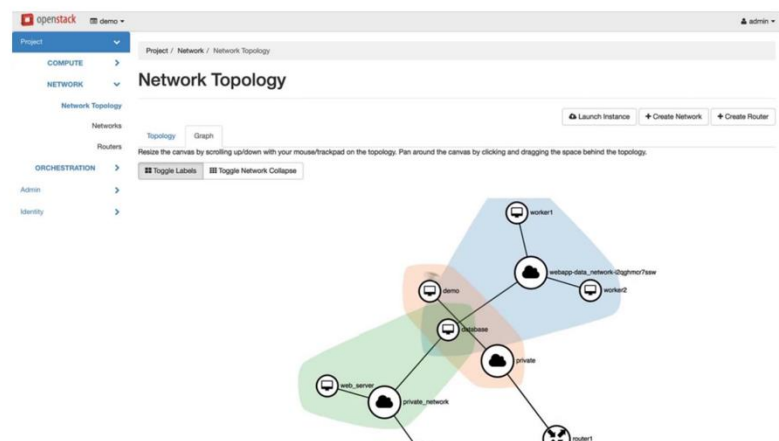
Cinder



Glance



Horizon



IaC

- Послідовність
- Зниження витрат
- Ефективність
- Швидкість
- Зниження ризиків

Інструменти
автоматизації
розгортання
Openstack



Ansible



Chef



Saltstack



Puppet

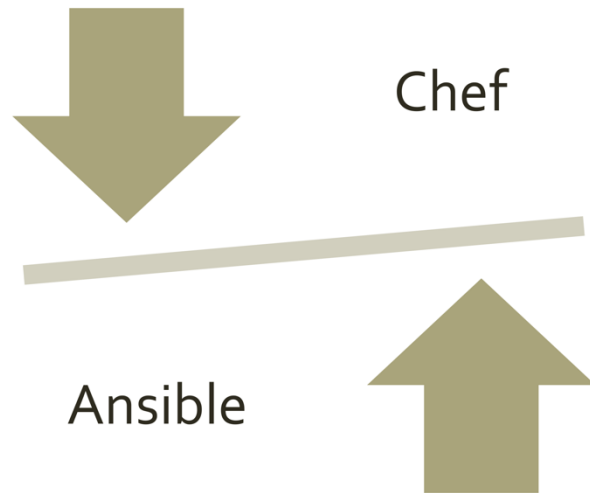


OpenStack
Autopilot

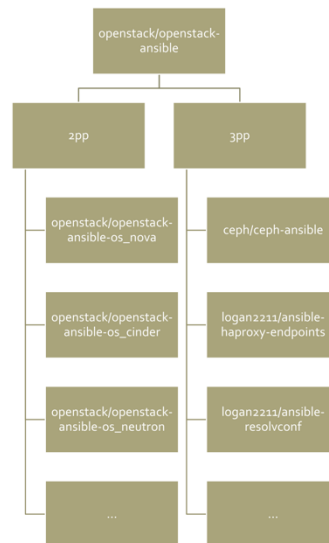


Fuel OpenStack

Порівняння
інструментів
автоматизації



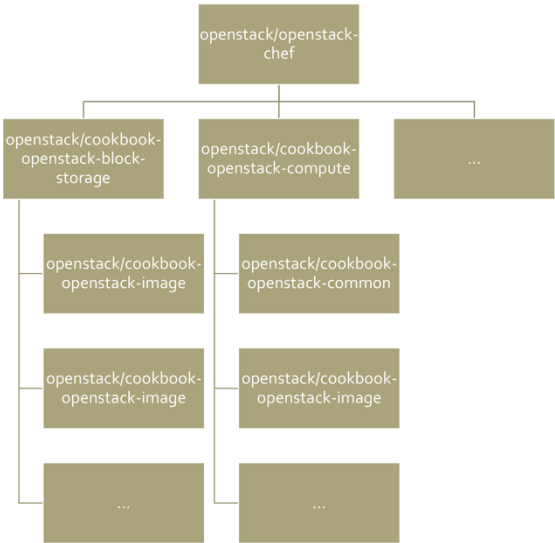
OpenStack-
Ansible



OpenStack- Ansible

Версія	Статус Ansible	Статус Openstack
Zed		У розробці
Yoga	У розробці	Підтримка
Xena	Підтримка	Підтримка
Wallaby	Підтримка	Підтримка
Victoria	Підтримка	Розширена підтримка
Ussuri	Розширена підтримка	Розширена підтримка
Train	Розширена підтримка	Розширена підтримка
Stein	Розширена підтримка	Розширена підтримка
Rocky	Розширена підтримка	Розширена підтримка
Queens	Розширена підтримка	Розширена підтримка
Pike	Розширена підтримка	Розширена підтримка
Ocata	Розширена підтримка	Не підтримується
Newton	Не підтримується	Не підтримується
Mitaka	Не підтримується	Не підтримується

OpenStack- Chef



OpenStack- Chef

Версія	Статус Chef	Статус Openstack
Zed		У розробці
Yoga		Підтримка
Xena		Підтримка
Wallaby		Підтримка
Victoria		Розширена підтримка
Ussuri	У розробці	Розширена підтримка
Train	Підтримка	Розширена підтримка
Stein	Підтримка	Розширена підтримка
Rocky	Підтримка	Розширена підтримка
Queens	Підтримка	Розширена підтримка
Pike	Розширена підтримка	Розширена підтримка
Ocata	Розширена підтримка	Не підтримується
Newton	Не підтримується	Не підтримується
Mitaka	Не підтримується	Не підтримується

Порівняння

	OpenStack-Ansible	OpenStack-Chef
Стан	Активна розробка	Підтримка
Час дослідження	Однаково	Однаково
Час розгортання	Більше	Менше
Охоплення	Більше	Менше
Стабільність	Однаково	Однаково
Орієнтованість	Адміністратори	Розробники



Дякую за увагу



ДОДАТОК Б

КОД ПРОЕКТУ

```
source 'https://supermarket.chef.io'

solver :ruby, :required

%w(
  bare-metal
  block-storage
  common
  compute
  dashboard
  dns
  identity
  image
  integration-test
  network
  ops-database
  ops-messaging
  orchestration
  telemetry
).each do |cookbook|
  if Dir.exist?("../cookbook-openstack-#{cookbook}")
    cookbook "openstack-#{cookbook}", path: "../cookbook-openstack-#{cookbook}"
  else
    cookbook "openstack-#{cookbook}", git: "https://opendev.org/openstack/cookbook-openstack-#{cookbook}"
  end
end

if Dir.exist?('../cookbook-openstackclient')
  cookbook 'openstackclient', path: '../cookbook-openstackclient'
else
  cookbook 'openstackclient', git: 'https://opendev.org/openstack/cookbook-openstackclient'
end

cookbook 'openstack_test', path: 'test/cookbooks/openstack_test'
```

```
{
  "name": "allinone",
  "description": "This will deploy all of the services for Openstack Compute to function on a single box.",
  "run_list": [
    "role[common]",
    "role[ops_database]",
    "role[ops_messaging]",
    "role[identity]",
    "role[image]",
```

```

"role[network]",
"role[compute]",
"role[block_storage]",
"role[bare_metal]",
"role[orchestration]",
"role[telemetry]",
"role[dns]",
"role[dashboard]"
]
}

```

```

{
  "name": "image",
  "description": "Deploy image services",
  "run_list": [
    "role[identity]",
    "role[image]",
    "role[network]",
    "recipe[openstack-compute::nova-setup]",
    "recipe[openstack-compute::identity_registration]",
    "recipe[openstack-compute::conductor]",
    "recipe[openstack-compute::api-os-compute]",
    "recipe[openstack-compute::api-metadata]",
    "recipe[openstack-compute::placement_api]",
    "recipe[openstack-compute::scheduler]",
    "recipe[openstack-compute::vncproxy]",
    "recipe[openstack-compute::compute]"
  ]
}

```

```

{
  "name": "allinone",
  "description": "Environment used in testing the upstream cookbooks and reference Chef repository with vagrant. To be used with the vagrantfile-allinone vagrantfile. Defines the necessary attributes for a working all-in-one openstack deployment, using neutron for the networking component, and the openvswitch neutron plugin",
  "default_attributes": {
    "apache": {
      "listen": [

    ]
    }
  },
  "override_attributes": {
    "openstack": {
      "is_release": true,
      "apt": {
        "update_apt_cache": "true"
      },
      "telemetry": {
        "conf": {

```

```

    "DEFAULT": {
        "meter_dispatchers": "database"
    }
},
"dashboard": {
    "server_hostname": "localhost"
},
"memcached_servers": [
    "127.0.0.1:11211"
],
"mq": {
    "user": "admin"
},
"network": {
    "conf": {
        "DEFAULT": {
            "service_plugins": "router"
        }
    }
},
"image": {
    "image_upload": true
},
"compute": {
    "conf": {
        "libvirt": {
            "cpu_type": "none",
            "virt_type": "qemu"
        }
    }
}
}
}
}
}

```

```

## Shell Opts -----
set -e -u -x

## Variables -----
# Extra options to pass to the AIO bootstrap process
export BOOTSTRAP_OPTS=${BOOTSTRAP_OPTS:-}

# Store the clone repo root location
export OSA_CLONE_DIR="${OSA_CLONE_DIR:-$(readlink -f $(dirname $0)/..)}"

# This script should be executed from the root directory of the cloned repo
cd "$(dirname "${0}")/.."

## Main -----

# When running in the gate, enable data device detection in the bootstrap

```

```

# role. This is needed on RAX nodepool instances because the root data disk
# is too small for an OSA AIO, and a second data disk is attached which must
# be formatted and used.
if [[ -d '/etc/nodepool' ]]; then
    export BOOTSTRAP_HOST_DETECT_DATA_DISK=true
fi

# Ensure that some of the wrapper options are overridden
# to prevent interference with the AIO bootstrap.
export ANSIBLE_INVENTORY="${OSA_CLONE_DIR}/tests/test-inventory.ini"
export ANSIBLE_VARS_PLUGINS="/dev/null"
export HOST_VARS_PATH="/dev/null"
export GROUP_VARS_PATH="/dev/null"

# Run AIO bootstrap playbook
pushd tests
if [ -z "${BOOTSTRAP_OPTS}" ]; then
    /usr/local/bin/ansible-playbook bootstrap-aio.yml
else
    export BOOTSTRAP_OPTS_ITEMS="
    for BOOTSTRAP_OPT in ${BOOTSTRAP_OPTS}; do
        BOOTSTRAP_OPTS_ITEMS=${BOOTSTRAP_OPTS_ITEMS}-e "${BOOTSTRAP_OPT}" "
    done
    /usr/local/bin/ansible-playbook bootstrap-aio.yml \
        ${BOOTSTRAP_OPTS_ITEMS}
fi
popd

# Now unset the env var overrides so that the defaults work again
unset ANSIBLE_INVENTORY
unset ANSIBLE_VARS_PLUGINS
unset HOST_VARS_PATH
unset GROUP_VARS_PATH

```

```

## Vars -----
export HTTP_PROXY=${HTTP_PROXY:-""}
export HTTPS_PROXY=${HTTPS_PROXY:-""}
# The Ansible version used for testing
export ANSIBLE_PACKAGE=${ANSIBLE_PACKAGE:-"ansible-core==2.12.5"}
export ANSIBLE_ROLE_FILE=${ANSIBLE_ROLE_FILE:-"ansible-role-requirements.yml"}
export ANSIBLE_COLLECTION_FILE=${ANSIBLE_COLLECTION_FILE:-"ansible-collection-requirements.yml"}
export USER_ROLE_FILE=${USER_ROLE_FILE:-"user-role-requirements.yml"}
export USER_COLLECTION_FILE=${USER_COLLECTION_FILE:-"user-collection-requirements.yml"}
export SSH_DIR=${SSH_DIR:-"/root/.ssh"}
export DEBIAN_FRONTEND=${DEBIAN_FRONTEND:-"noninteractive"}
# check whether to install the ARA callback plugin
export SETUP_ARA=${SETUP_ARA:-"false"}

# Use pip opts to add options to the pip install command.
# This can be used to tell it which index to use, etc.
export PIP_OPTS=${PIP_OPTS:-""}

```



```

export OSA_WRAPPER_BIN="${OSA_WRAPPER_BIN:-scripts/openstack-ansible.sh}"

# This script should be executed from the root directory of the cloned repo
cd "$(dirname "${0}")/.."

## Functions -----
info_block "Checking for required libraries." 2> /dev/null ||
    source scripts/scripts-library.sh

## Main -----
info_block "Bootstrapping System with Ansible"

# Store the clone repo root location
export OSA_CLONE_DIR="$(pwd)"

# Set the variable to the role file to be the absolute path
ANSIBLE_ROLE_FILE="$(readlink -f "${ANSIBLE_ROLE_FILE}")"
OSA_INVENTORY_PATH="$(readlink -f inventory)"
OSA_ANSIBLE_PYTHON_INTERPRETER="auto"

# Create the ssh dir if needed
ssh_key_create

# Determine the distribution which the host is running on
determine_distro

# Install the base packages
case ${DISTRO_ID} in
    rocky|centos|rhel)
        dnf -y install \
            git curl autoconf gcc gcc-c++ nc \
            python38 python38-devel libselinux-python3 \
            systemd-devel pkgconf \
            openssl-devel libffi-devel \
            rsync wget
        if [[ ${DISTRO_ID} == "rocky" ]]; then
            OSA_ANSIBLE_PYTHON_INTERPRETER="/usr/bin/python3"
        fi
        PYTHON_EXEC_PATH="$(which python3.8)"
        ;;
    ubuntu|debian)
        apt-get update
        DEBIAN_FRONTEND=noninteractive apt-get -y install \
            git-core curl gcc netcat \
            python3 python3-dev \
            libssl-dev libffi-dev \
            libsystemd-dev pkg-config \
            python3-apt python3-venv \
            python3-minimal wget
        ;;
esac

```

```

# Load nodepool PIP mirror settings
load_nodepool_pip_opts

# Ensure we use the HTTPS/HTTP proxy with pip if it is specified
if [ -n "$HTTPS_PROXY" ]; then
    PIP_OPTS+="--proxy $HTTPS_PROXY"

elif [ -n "$HTTP_PROXY" ]; then
    PIP_OPTS+="--proxy $HTTP_PROXY"
fi

PYTHON_EXEC_PATH="${PYTHON_EXEC_PATH:-$(which python3)}"

# Obtain the SHA of the upper-constraints to use for the ansible runtime venv
TOX_CONSTRAINTS_SHA=$(awk '/requirements_git_install_branch:/ {print $2}'
playbooks/defaults/repo_packages/openstack_services.yml)

# if we are in CI, grab the u-c file from the locally cached repo, otherwise download
TOX_CONSTRAINTS_PATH="/opt/ansible-runtime-constraints-${TOX_CONSTRAINTS_SHA}.txt"
if [[ -z "${ZUUL_SRC_PATH+defined}" ]] || ! -d "${ZUUL_SRC_PATH:-}" ]; then
    wget ${TOX_CONSTRAINTS_FILE:-
"https://opendev.org/openstack/requirements/raw/${TOX_CONSTRAINTS_SHA}/upper-constraints.txt"} -O
${TOX_CONSTRAINTS_PATH}
else
    git --git-dir=${ZUUL_SRC_PATH}/opendev.org/openstack/requirements/.git show ${TOX_CONSTRAINTS_SHA}:upper-
constraints.txt > ${TOX_CONSTRAINTS_PATH}
fi

export TOX_CONSTRAINTS_FILE="file://${TOX_CONSTRAINTS_PATH}"

if [[ -z "${SKIP_OSA_RUNTIME_VENV_BUILD+defined}" ]]; then
    build_ansible_runtime_venv
fi

# Install and export the ARA callback plugin
if [ "${SETUP_ARA}" == "true" ]; then
    setup_ara
fi

# Get current code version (this runs at the root of OSA clone)
export CURRENT_OSA_VERSION=$(cd ${OSA_CLONE_DIR}; /opt/ansible-runtime/bin/python setup.py --version)

# Ensure that Ansible binaries run from the venv
pushd /opt/ansible-runtime/bin
for ansible_bin in $(ls -1 ansible*); do
    if [ "${ansible_bin}" == "ansible" ] || [ "${ansible_bin}" == "ansible-playbook" ]; then
        # For the 'ansible' and 'ansible-playbook' commands we want to use our wrapper
        ln -sf /usr/local/bin/openstack-ansible /usr/local/bin/${ansible_bin}

    else
        # For any other commands, we want to link directly to the binary

```

```

ln -sf /opt/ansible-runtime/bin/${ansible_bin} /usr/local/bin/${ansible_bin}

fi
done
popd

# Write the OSA Ansible rc file
sed "s|OSA_INVENTORY_PATH|${OSA_INVENTORY_PATH}|g" scripts/openstack-ansible.rc >
/usr/local/bin/openstack-ansible.rc
sed -i "s|OSA_ANSIBLE_PYTHON_INTERPRETER|${OSA_ANSIBLE_PYTHON_INTERPRETER}|g"
/usr/local/bin/openstack-ansible.rc
sed -i "s|OSA_ANSIBLE_FORKS|${ANSIBLE_FORKS}|g" /usr/local/bin/openstack-ansible.rc

# Create openstack ansible wrapper tool
cp -v ${OSA_WRAPPER_BIN} /usr/local/bin/openstack-ansible
# Mark the current OSA git repo clone directory, so we don't need to compute it every time.
sed -i "s|OSA_CLONE_DIR|${OSA_CLONE_DIR}|g" /usr/local/bin/openstack-ansible
# Mark the current OSA version in the wrapper, so we don't need to compute it every time.
sed -i "s|CURRENT_OSA_VERSION|${CURRENT_OSA_VERSION}|g" /usr/local/bin/openstack-ansible

# Ensure wrapper tool is executable
chmod +x /usr/local/bin/openstack-ansible

echo "openstack-ansible wrapper created."

# If the Ansible plugins are in the old location remove them.
[[ -d "/etc/ansible/plugins" ]] && rm -rf "/etc/ansible/plugins"

# Update dependent roles
if [ -f "${ANSIBLE_ROLE_FILE}" ] && [[ -z "${SKIP_OSA_ROLE_CLONE+defined}" ]]; then
    # NOTE(cloudnull): When bootstrapping we don't want ansible to interact
    #                 with our plugins by default. This change will force
    #                 ansible to ignore our plugins during this process.
    export ANSIBLE_LIBRARY="${OSA_CLONE_DIR}/playbooks/library"
    export ANSIBLE_LOOKUP_PLUGINS="/dev/null"
    export ANSIBLE_FILTER_PLUGINS="/dev/null"
    export ANSIBLE_ACTION_PLUGINS="/dev/null"
    export ANSIBLE_CALLBACK_PLUGINS="/dev/null"
    export ANSIBLE_CALLBACKS_ENABLED="/dev/null"
    export ANSIBLE_TEST_PLUGINS="/dev/null"
    export ANSIBLE_VARS_PLUGINS="/dev/null"
    export ANSIBLE_STRATEGY_PLUGINS="/dev/null"
    export ANSIBLE_CONFIG="none-ansible.cfg"
    export ANSIBLE_COLLECTIONS_PATH="/etc/ansible"

    pushd scripts
    /opt/ansible-runtime/bin/ansible-playbook get-ansible-collection-requirements.yml \
        -e collection_file="${ANSIBLE_COLLECTION_FILE}" -e
user_collection_file="${USER_COLLECTION_FILE}"

    /opt/ansible-runtime/bin/ansible-playbook get-ansible-role-requirements.yml \
        -e role_file="${ANSIBLE_ROLE_FILE}" -e user_role_file="${USER_ROLE_FILE}"
    popd

```

```
unset ANSIBLE_LIBRARY
unset ANSIBLE_LOOKUP_PLUGINS
unset ANSIBLE_FILTER_PLUGINS
unset ANSIBLE_ACTION_PLUGINS
unset ANSIBLE_CALLBACK_PLUGINS
unset ANSIBLE_CALLBACKS_ENABLED
unset ANSIBLE_TEST_PLUGINS
unset ANSIBLE_VARS_PLUGINS
unset ANSIBLE_STRATEGY_PLUGINS
unset ANSIBLE_CONFIG
unset ANSIBLE_COLLECTIONS_PATH
fi

echo "System is bootstrapped and ready for use."
```

Позначення					Найменування					Дод. відомості		
					Текстові документи							
1.					Пояснювальна записка					70 с.		
					Графічні документи							
					Слайди презентації					23 с.		