Ansible Automation for IBM Power Systems -

Practical Customer Examples

John Karaba

Executive Consultant, IBM Systems Lab Services

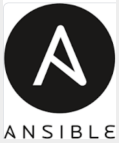
Helping Customers with their Cloud Journey

IBM Systems Lab Services — Power Cloud Team

Plan
Cloud Design Workshop

Transform the Enterprise

Innovate
OpenShift and Cloud Pak Implementation

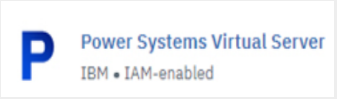


Automate
Automation for DevOps

Accelerate
PowerVC Implementation

Manage the Hybrid Multicloud

Deploy
Hybrid Cloud Implementation



Extend
Moving Power Workloads to Cloud

Manage
OpenShift and Cloud Pak Implementation

Typical areas of automation for IBM Power customers

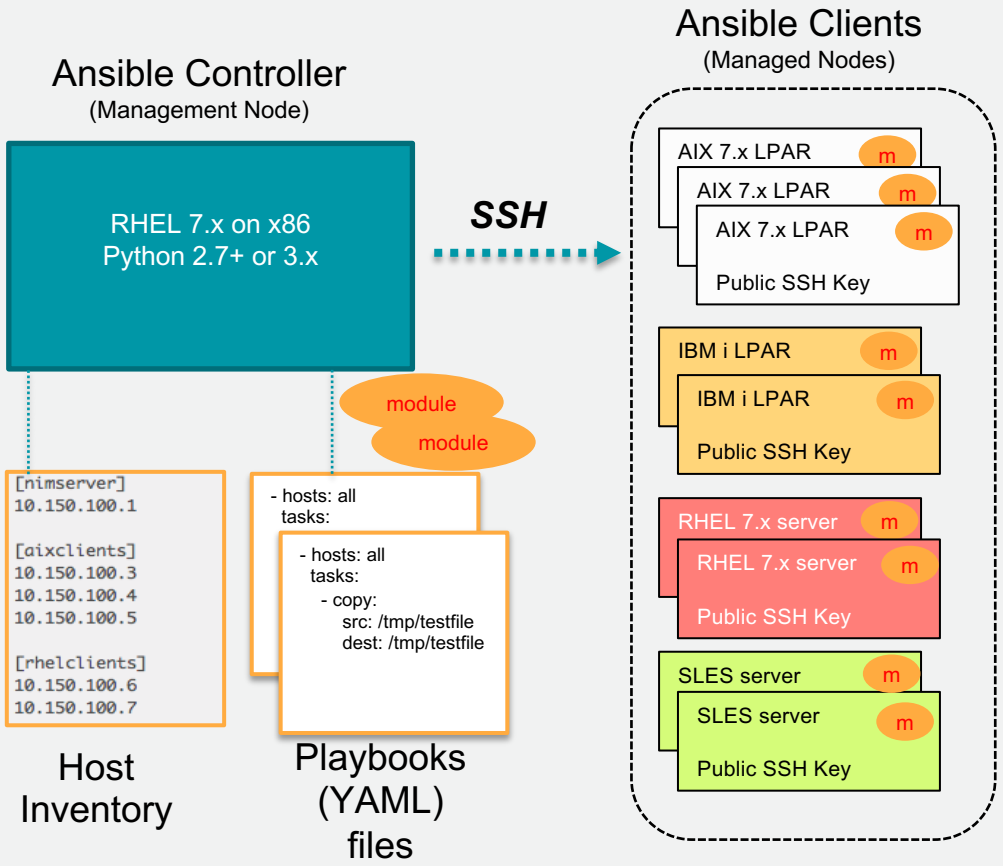
- Rapidly build new VMs (LPARs) in a consistent manner
- Apply appropriate configuration to the new LPARs (post deploy)
- Deploy and configure applications / software packages to the new LPARs
- Administer existing LPARs (Common Administration Tasks)
- Quickly provision/deprovision Test/QA environments (DevOps)
- Integration with other management tools (e.g. PowerVC Cloud Manager)

Typical one-week Workshop Agenda

Agenda is customized based on customer's specific needs

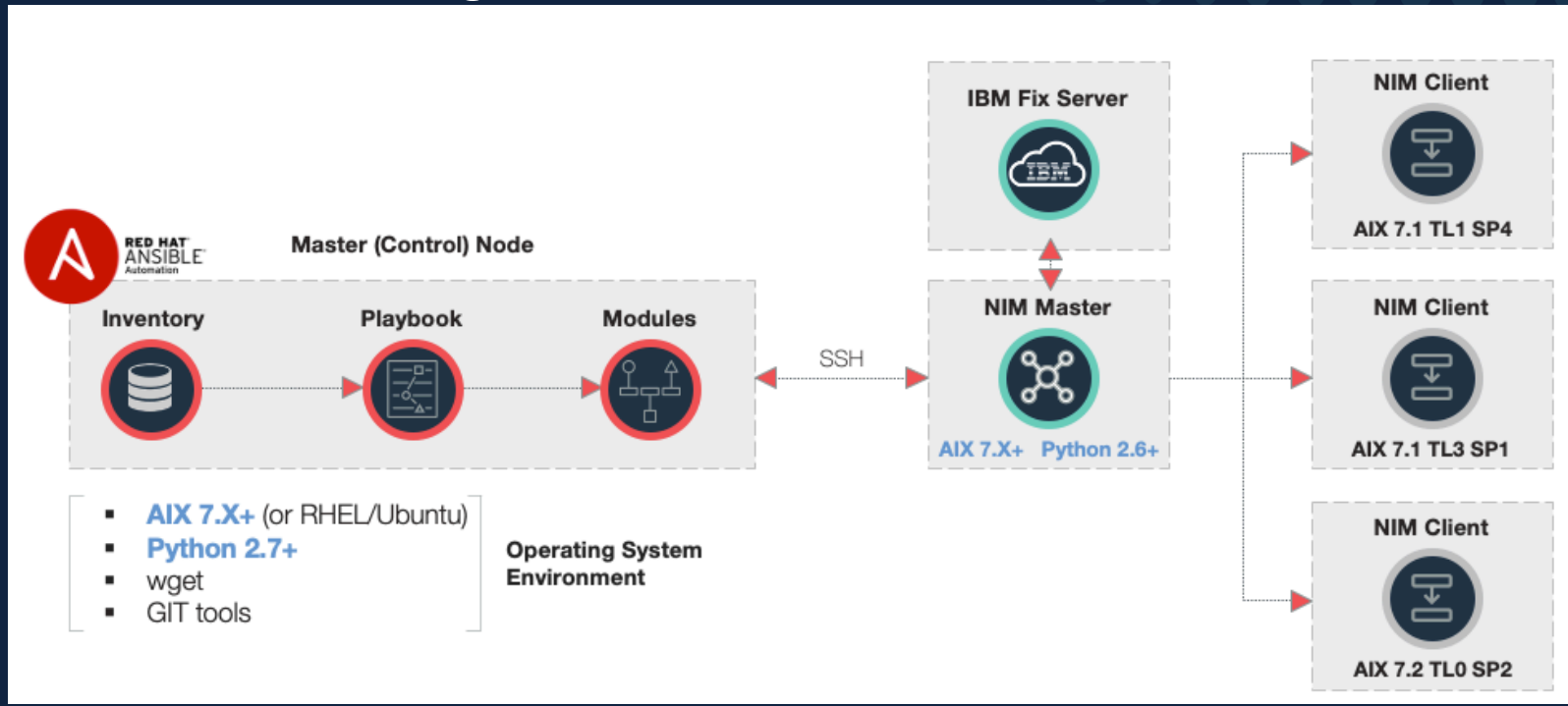
- Review current environment and Install/Configure Ansible
- Demonstrate basic Ansible modules
- Ansible - Advanced Concepts (roles, collections)
- Demonstrate Use Cases from IBM Power Systems AIX or IBM i Collection
  - 5-10 use cases
- Integration with PowerVC via REST APIs
  - 3-5 use cases

Example Setup for Ansible Workshop



AnsibleFest

Automated Patching of IBM AIX Servers



Ansible modules for AIX

Galaxy Collection Page: https://galaxy.ansible.com/ibm/power_aix

Community Modules
- Community supported
- Needs IBM review and potential contribution to ensure AIX compatibility

- **installp**
- **aix_devices**
- **aix_filesystem**
- **aix_inittab**
- **aix_lvg**
- **aix_lvol**
- expect
- telnet
- openssh_cert
- openssh_keypair
- openssl_certificate
- openssl_certificate_info
- openssl_csr
- openssl_csr_info
- openssl_dhparam
- openssl_pkcs12
- openssl_privatekey
- openssl_privatekey
- openssl_publickey

Core Modules
- Supported by RedHat

- command
- raw
- script
- shell
- pip
- yum
- pause
- wait_for_connection
- at
- authorized_key
- gather_facts
- **group**
- **mount**
- ping
- reboot
- setup
- **user**
- assemble
- blockinfile
- copy
- fetch
- file
- find
- lineinfile
- stat
- synchronize

IBM Organic Modules
- In Automation Hub with IBM L2/L3 support

- emgr
- flrtvc
- geninstall
- suma
- nim
- nim_flrtvc
- nim_suma
- nim_vios_hc
- nim_vios_alt_disk
- nim_viosupgrade *
- nim_updateios *
- mktcpip

IBM versions of community and core Modules
- In Automation Hub with IBM L2/L3 support

- installp
- devices
- filesystem *
- inittab
- lvg
- lvol

Core Module Replacements
- user *
- group *
- mount

IBM Developed and Supported

Notes:

Items in **Red** are core modules that are replaced in our AIX collection.

Items in Blue are community modules that are replaced to meet desired functionality for AIX and are certified for support.

* denotes modules that are still under development

Ansible Modules for AIX (examples)

File System operations

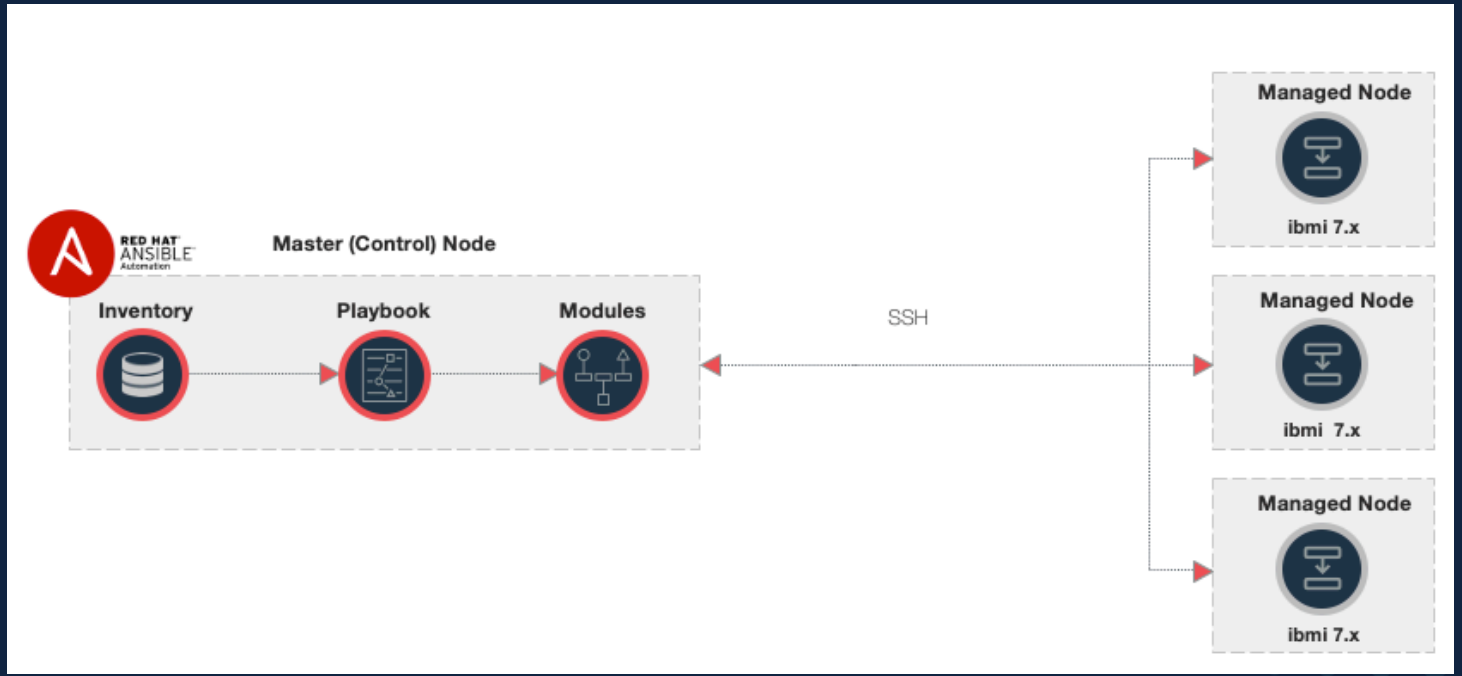
```
- name: Creation of a JFS2 filesystem
  filesystem:
    state: present
    filesystem: /mnt3
    fs_type: jfs2
    attributes: size=32768,isnapshot='no'
    mount_group: test
    vg: rootvg
- name: Increase size of a filesystem
  filesystem:
    filesystem: /mnt3
    state: present
    attributes: size=+5M
- name: Remove a NFS filesystem
  filesystem:
    filesystem: /mnt
    state: absent
    rm_mount_point: true
```

Download & Install SP and TL

```
- name: Check, download and install to latest SP of TL 7.2.4
  suma:
    action: download
    oslevel: '7200-04'
    last_sp: yes
    extend_fs: no

- name: Check, download and install to TL 7.2.3
  suma:
    action: download
    oslevel: '7200-03'
```

Automation for IBM i Servers



Ansible modules for IBM i

IBM Developed and Supported Modules

On Ansible-Galaxy: https://galaxy.ansible.com/ibm/power_ibmi
On Github: https://github.com/IBM/ansible-for-i

1. ibmi_at
2. ibmi_cl_command
3. ibmi_copy
4. ibmi_device_vary
5. ibmi_display_subsystem
6. ibmi_end_subsystem
7. ibmi_fetch
8. ibmi_fix
9. ibmi_fix_imgclg
10. ibmi_host_server_service
11. ibmi_iasp
12. ibmi_install_product_from_savf
13. ibmi_job
14. ibmi_lib_restore
15. ibmi_lib_save
16. ibmi_message
17. ibmi_object_authority
18. ibmi_object_find
19. ibmi_object_restore
20. ibmi_object_save
21. ibmi_reboot
22. ibmi_save_product_to_savf
23. ibmi_script
24. ibmi_script_execute
25. ibmi_sql_execute
26. ibmi_sql_query
27. ibmi_start_subsystem
28. ibmi_submit_job
29. ibmi_sync
30. ibmi_synchronize
31. ibmi_tcp_interface
32. ibmi_tcp_server_service
33. ibmi_uninstall_product
34. ibmi_user_and_group

Community Modules

1. assemble
2. authorized_key
3. blockinfile
4. Command
5. copy
6. fetch
7. file
8. Find
9. git
10. lineinfile
11. pause
12. ping
13. pip
14. raw
15. script
16. set_up
17. shell
18. stat
19. wait_for_connection
20. yum

Ansible Modules for IBM i (examples)

Submit a job

```
- name: Submit a batch job and run CALL QGPL/PGM1
  ibmi_submit_job:
    cmd: 'CALL QGPL/PGM1'
    parameters: 'JOB(TEST)'
    check_interval: '30s'
    time_out: '80s'
    status: ['*OUTQ', '*COMPLETE']
```

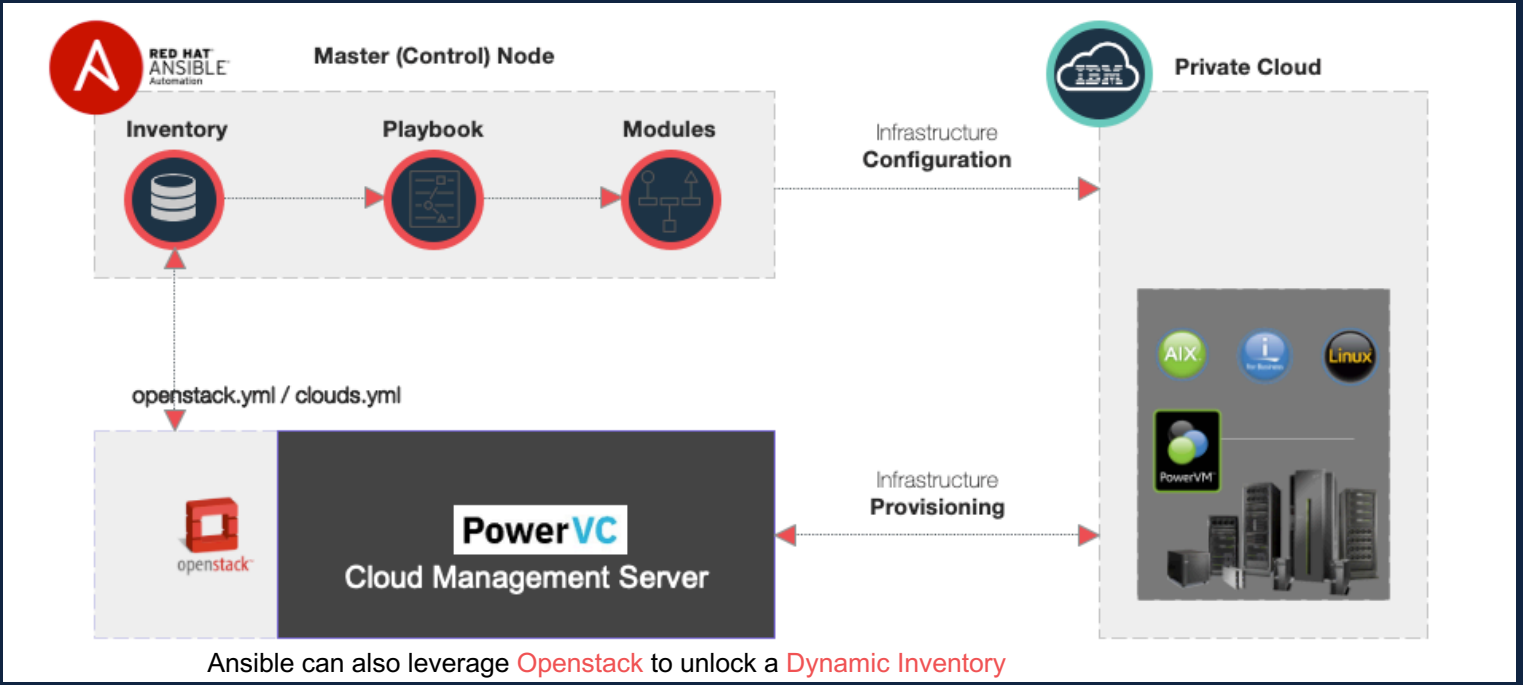
Vary on a device

```
- name: start host server service
  ibmi_device_vary:
    device_list: ['IASP1', 'IASP2']
    status: '*ON'
    joblog: True
```

Patching

```
- name: Remove a single PTF
  ibmi_fix:
    product_id: '5770DBM'
    delayed_option: "*NO"
    temp_or_perm: "*PERM"
    operation: 'remove'
    fix_list:
      - "SI72223"
- name: Install a single PTF
  ibmi_fix:
    product_id: '5770DBM'
    save_file_object: 'QSI72223'
    save_file_lib: 'QGPL'
    delayed_option: "*NO"
    temp_or_perm: "*TEMP"
    operation: 'load_and_apply'
    fix_list:
      - "SI72223"
- name: query ptf
  ibmi_fix:
    operation: 'query'
    fix_list:
      - "SI72223"
      - "SI70819"
```

Automated Server & Storage Provisioning via PowerVC & REST APIs



Ansible can also leverage Openstack to unlock a Dynamic Inventory

Client Example: Ansible for DevOps Automation Workshop
North America / Large Telecom Provider

Challenge

- Cloud and Automation are now strategic imperatives for business and IT
- Power infrastructure (AIX, SLES and RHEL on Power) needs to be easier to consume and manage.
- The team was looking for assistance from IBM
  - Run a POC and train the team in Ansible
  - **start automating existing configuration** management and deployment processes
  - integrate Ansible playbooks with existing PowerVC installations via REST API
- With the exception of one person, everyone on the team was new to Ansible

Client Example: Ansible for DevOps Automation Workshop
North America / Large Telecom Provider

Solution

- Engaged with client to perform an **Ansible for DevOps Automation workshop**
- Reviewed Ansible (Architecture/Security/Advanced Capabilities)
- Installed and configured Ansible core to be used by more than one team in current test/dev environment,
- Demonstrated the use of ansible commands, playbooks and roles to automate key AIX/Linux administration tasks.
- Demonstrated a number of use cases including:
  - Integration with NIM Server
  - Integration with custom modules and existing shell scripts
  - Building complex playbooks and leveraging roles
  - Using Ansible with PowerVC and REST APIs

Client Example: Ansible for DevOps Automation Workshop
North America / Large Telecom Provider

Benefits

- Workshop provided a strong foundation for Automation using Ansible.
- Client is now in a position to deploy Ansible core across their Test, Dev and Prod environments.
- Once fully implemented, Ansible is expected to drive,
    - Productivity gains via automation
    - Quicker time to market for IT projects
    - Faster, less-error prone turnaround on IT configuration change
    - Existing Power Infrastructure is now better positioned to support the current cloud strategy.

Using REST APIs

Get Cloud Projects via URI Module

pvc_openstack > playbooks > ! get_pvc_projects.yml > Ansible > [ ] tasks > { } debug > msg

```yaml
---

- name: Get PowerVC Projects
  hosts: localhost
  gather_facts: no

  vars:
    pvc_token: "{{ lookup('file','../vars/auth_token.
    pvc_base_url: "https://10.150.254.46:5000"

  tasks:
  - name:  Get List of tenants with IDs
    uri:
      url: "{{ pvc_base_url }}/v3/projects"
      method: GET
      headers:
        Content-Type: "application/json"
        X-Auth-Token: "{{ pvc_token }}"
      validate_certs: False
    register: results
    no_log: true

  - debug:
      msg: |
        Found "{{ results.json.projects |length }}
        ------------------------------------
        {% for items in results.json.projects %}
        {{items.id}}  -{{items.name}}   ID: {{item
        {% endfor %}
```

```
[root@jk-ansible-ctl demo]# ansible-playbook get_pvc_projects.yml

PLAY [Get PowerVC Projects] ****************************************************

TASK [Get List of tenants with IDs] ********************************************
ok: [localhost]

TASK [debug] *******************************************************************
ok: [localhost] =>
  msg: |-
    Found "7"  projects
    ------------------------------------
    5339ca6b10e549f388ba7cd95b69fe82  -Team 4   ID: Cloud Workshop Team 4
    549e3c48bda4449180847243002fb124  -Team 1   ID: Cloud Workshop Team1
    98e3dc2784a64d97ae75e0cb06fd8988  -ibm-default   ID: IBM Default Tenant
    cb74e18734b9486093e09c5dc931ebe0  -service   ID: IBM Service Tenant for service users and groups
    d928ea0a7db64995abede4f8ce94c1a4  -Team 3   ID: Cloud Workshop Team3
    e66c237401cb4bd0bb2572dde266d456  -Team 2   ID: Cloud workshop Team2
    e6eb13fb32194c2393fa47bd8c2bde4f  -powervm   ID: IBM Tenant for storing VM NVRAM data for remote restart

TASK [set_fact] ****************************************************************
ok: [localhost]

PLAY RECAP *********************************************************************
localhost                  : ok=3    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
```

Using REST APIs

Get Cloud Projects via openstack module

```
---

- name: "Gather list of projects in {{ cloud }}"
  os_project_info:
    cloud: "{{ cloud }}"
    auth:
      username: "{{ user_name }}"
      password: "{{ user_pass }}"
  register: os_proj

- name: show list of projects
  debug:
    msg: |
       List of projects
        {{'%-20.28s %-40s %-15.13s %-10.8s %-35.35s' | format('pr
        {{ '-' * 123 }}
        {% for item in os_proj.openstack_projects %}
       {{'%-20.18s %-40s %-15.13s %-10.8s %-35.35s' | format(item
        {% endfor %}
```

```
(openstack) [root@jk-ansible-ctl pvc]# ansible-playbook roles/pvc/tests/test.yml -e "pvc_action=list-projects"

PLAY [Roles testing playbook] ***********************************************************************************************

TASK [Gathering Facts] ******************************************************************************************************
ok: [localhost]

TASK [pvc : include_tasks] **************************************************************************************************
included: /myplaybooks/pvc/roles/pvc/tasks/list-projects.yml for localhost

TASK [pvc : Gather list of projects in pvc144cloud] *************************************************************************
ok: [localhost]

TASK [pvc : show list of projects] ******************************************************************************************
ok: [localhost] => {}

MSG:

List of projects
 project name         project id                               cloud          enabled    project description
 ---------------------------------------------------------------------------------------------------------------------------
 Team 4               5339ca6b10e549f388ba7cd95b69fe82         pvc144cloud    True       Cloud Workshop Team 4
 Team 1               549e3c48bda4449180847243002fb124         pvc144cloud    True       Cloud Workshop Team1
 ibm-default          98e3dc2784a64d97ae75e0cb06fd8988         pvc144cloud    True       IBM Default Tenant
 service              cb74e18734b9486093e09c5dc931ebe0         pvc144cloud    True       IBM Service Tenant for service user
 Team 3               d928ea0a7db64995abede4f8ce94c1a4         pvc144cloud    True       Cloud Workshop Team3
 Team 2               e66c237401cb4bd0bb2572dde266d456         pvc144cloud    True       Cloud Workshop Team2
 powervm              e6eb13fb32194c2393fa47bd8c2bde4f         pvc144cloud    True       IBM Tenant for storing VM NVRAM dat



PLAY RECAP ******************************************************************************************************************
localhost                  : ok=4    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
```

Using custom modules with REST APIs to deploy VM in PowerVC

vm_deploy.yml

```yaml
---
# ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
# First Play in the playbook
# check input files/parameters ...then and deploy VMs ONLY if all checks passed
# ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
- name: "Deploy New VM using REST APIs"
  hosts: localhost
  # ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
  # Note: PVC_connection dictionary, could be used
  # as argument with custom modules (not yet implemented)
  # for more human readable output change these paramters in /etc/ansible/ansible.cfg:
  #     stdout_callback = yaml
  #     #bin_ansible_callbacks = True
  # ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
...
  # ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
  # Deploy VM(S)
  # ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
  - name: "Deploy VM"
    DeployPVCVM:
      action: 'deploy'
      description: 'deploy new VMs'
      inputparmdict: "{{ pvc_connection_parms_dict}}"
      VMfile: "{{ vm_requests_file }}"
      keypair: "{{ keypair }}"
      newVMDict: "{{ parse_results.deploy_output[0] }}"
    register: deploy_results
    when: deploy_flag == true
```

vm_deploy.py Module …sample code extract

```python
def main():


    DEBUG_DATA = []
    DEPLOY_OUTPUT = []
    DEPLOY_PARAMS = {}
    DEPLOY_NODE = {}
    DEPLOY_CHANGED = False

    global logfile
    logfile = '/tmp/ansible_deploy_debug.log'
    #logfile = 'test.log'

    MODULE = AnsibleModule(
        argument_spec=dict(
            description=dict(required=False, type='str'),
            inputparmfile=dict(required=False, type='str'),
            VMfile=dict(required=False, type='str'),
            newVMDict=dict(required=False, type='str'),
            keypair=dict(required=False, type='str'),
            async=dict(choices=['true', 'false'], default='false', type='str'),
            action=dict(choices=['parse','check', 'deploy', 'remove'], type='str'),
        ),
        required_if=[
            ['action', 'parse', ['VMfile']],
            ['action', 'check', ['inputparmfile', 'newVMDict']],
            ['action', 'deploy', ['inputparmfile', 'newVMDict']],
            ['action', 'remove', ['inputparmfile', 'VMfile']]
        ],
        supports_check_mode=False
    )


    # Open log file
    logging.basicConfig(filename=logfile,
                        format='[%(asctime)s] %(levelname)s: [%(funcName)s:%(thread)d] %(message)s',
                        level=logging.DEBUG)


    logging.info('*** START DeployPVCVM action ****************************************')
```

Using Ansible openstack modules to deploy VMs in PowerVC

Invocation:

ansible-playbook roles/pvc/tests/test.yml -e "pvc_action=deploy-vm vm_req=vm_list.yml"

Folder Structure

```
Playbook1.yml
Playbook2.yml
....
   roles/
      pvc/
        defaults/
            main.yml
        files/
            vm_list.yml
            ..
        tasks/
            main.yml
            create-vm.yml
            list-vms.yml
            delete-vms.yml
        templates
            template1.j2
            ...
        ... /
        tests/
            test.yml
```

roles/pvc/tests/test.yml (1)

```
---
- name: Roles testing playbook
  hosts: localhost
  gather_facts: yes

  roles:
    - pvc
```

roles/pvc/defaults/main.yml (2)

```
---
pvc_base_url: https://10.150.31.50:5000
cloud: pvc144cloud
project_name: ibm-default
user_name: root
user_pass: lpar1team

pvc_token: ''

pvc_connection:
  project_name: "{{ project_name }}"
  username: "{{ user_name }}"
  password: "{{ user_pass }}"
```

roles/pvc/tasks/main.yml (3)

```
---

- include_tasks: "{{ pvc_action }}.yml"
  when: pvc_action is defined
~
```

roles/pvc/files/vm_list.yml (4)

```
new_vm_deploy_requests:
- vm_name: os_ibmi_vm1
  image_id: TESTiACT2withSSH
  flavor_id: basic
  ssh_key_name: ansible-ssh-key
  nics_id: 'VLAN 1'
  vm_group_name: Production
- vm_name: os_aix-vm1
  image_id: AIX71
  flavor_id: basic
  ssh_key_name: ansible-ssh-key
  nics_id: 'VLAN 1'
  vm_group_name: Production
~
```

roles/pvc/tasks/deploy-vm.yml (5)

```
---

- name: load new-vms-request.yml file
  include_vars:
    file: "files/{{ vm_req }}.yml"

- name: Create vms
  include_tasks: create-vm.yml
  vars:
     vm_name: "{{ item.vm_name }}"
     image_id: "{{ item.image_id }}"
     flavor_id: "{{ item.flavor_id }}"
     ssh_key_name: "{{ item.ssh_key_name }}"
     nics_id: "{{ item.nics_id }}"
     vm_group_name: "{{ item.vm_group_name }}"
     availability_zone: "{{ item.availability_zone }}"
     volumes: "{{ item.volumes }}"
  loop: "{{ new_vm_deploy_requests }}"
  when: new_vm_deploy_requests is defined
```

Thank you

Red Hat is the world's leading provider of enterprise open source software solutions. Award-winning support, training, and consulting services make Red Hat a trusted adviser to the Fortune 500.

youtube.com/user/RedHatVideos

linkedin.com/company/Red-Hat

facebook.com/ansibleautomation

twitter.com/ansible

Start / Stop VM using URI module

```
- name:  "Start VM: {{ selected_vm }}"
  uri:
    url: "{{ pvc_base_url }}/powervc/openstack/compute/v2.1/{{ pvc_tenant_ID }}/servers/{{ selected_vm_id }}/action"
    method: POST
    headers:
      Content-Type: "application/json"
      X-Auth-Token: "{{ pvc_token }}"
    body: {"os-start":null}
    body_format: json
    validate_certs: False
   # force_basic_auth: yes
    status_code: 202
  register: vm_stort_results
  ignore_errors: true
  #no_log: true
```

Contact: Vess Natchev (vess@us.ibm.com)

How to Engage IBM Systems Lab Services(SLS) on Cloud

My business challenge is:	Product Offering	SLS Service Offerings
Plan for journey to cloud	Multiple	Cloud Design Workshop*
Innovate with cloud-native apps	Red Hat OpenShift	OpenShift and Cloud Pak Implementation on Power*
Automate system management	Red Hat Ansible and Ansible Tower	Ansible Automation for DevOps*
Cut costs with "pay as you go" and bursting to the cloud	Power Virtual Server in IBM Cloud Virtual Private Cloud on Power in IBM Cloud	Moving Power Workloads to IBM Cloud Hybrid Cloud for Power Implementation
Manage and deploy apps in VMs and containers on multiple clouds	IBM Cloud Paks	OpenShift and Cloud Pak Implementation on Power*
Improve time to value for core-business apps	PowerVC VMware vRealize	PowerVC Implementation* vRealize Integration*