Automated Systems Management with Ansible

Wang Yun (王云)

Business Architect

Cloud and Systems Management

Agenda

- Introduce Ansible
- Ansible and IBM i
- Examples
- Real customer use cases

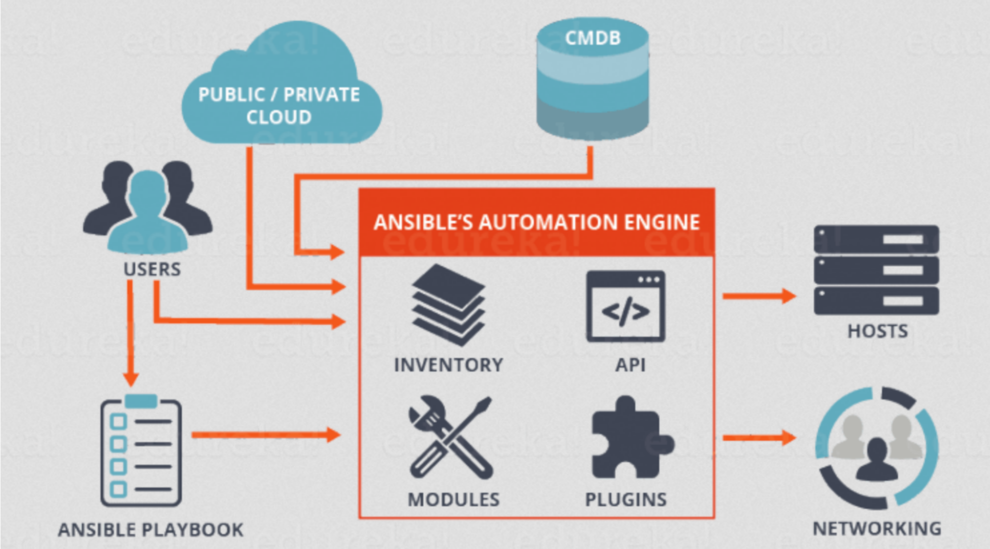
Agenda

- Introduce Ansible
- Ansible and IBM i
- Examples
- Real customer use cases

Introduction of Ansible

Ansible is a radically simple IT automation platform that makes your applications and systems easier to deploy.

– It is a free open source application with commercial options from Red Hat
– Agent-less – No need for agent installation and management
– Python/YAML based
– Highly flexible and configuration management of systems.
– Configuration roll-back in case of error



Key Ansible use cases

– Configuration management
– Application deployment
– Continuous delivery
– Provisioning
– Orchestration
– Security automation

Quick start video - https://www.ansible.com/resources/videos/quick-start-video

Ansible: Terminology

Control node – any machine with Ansible installed and is used to run playbooks

Managed node (a.k.a. endpoints) – endpoint devices (e.g., AIX, IBM i, Linux, Windows, etc.) that are managed with Ansible

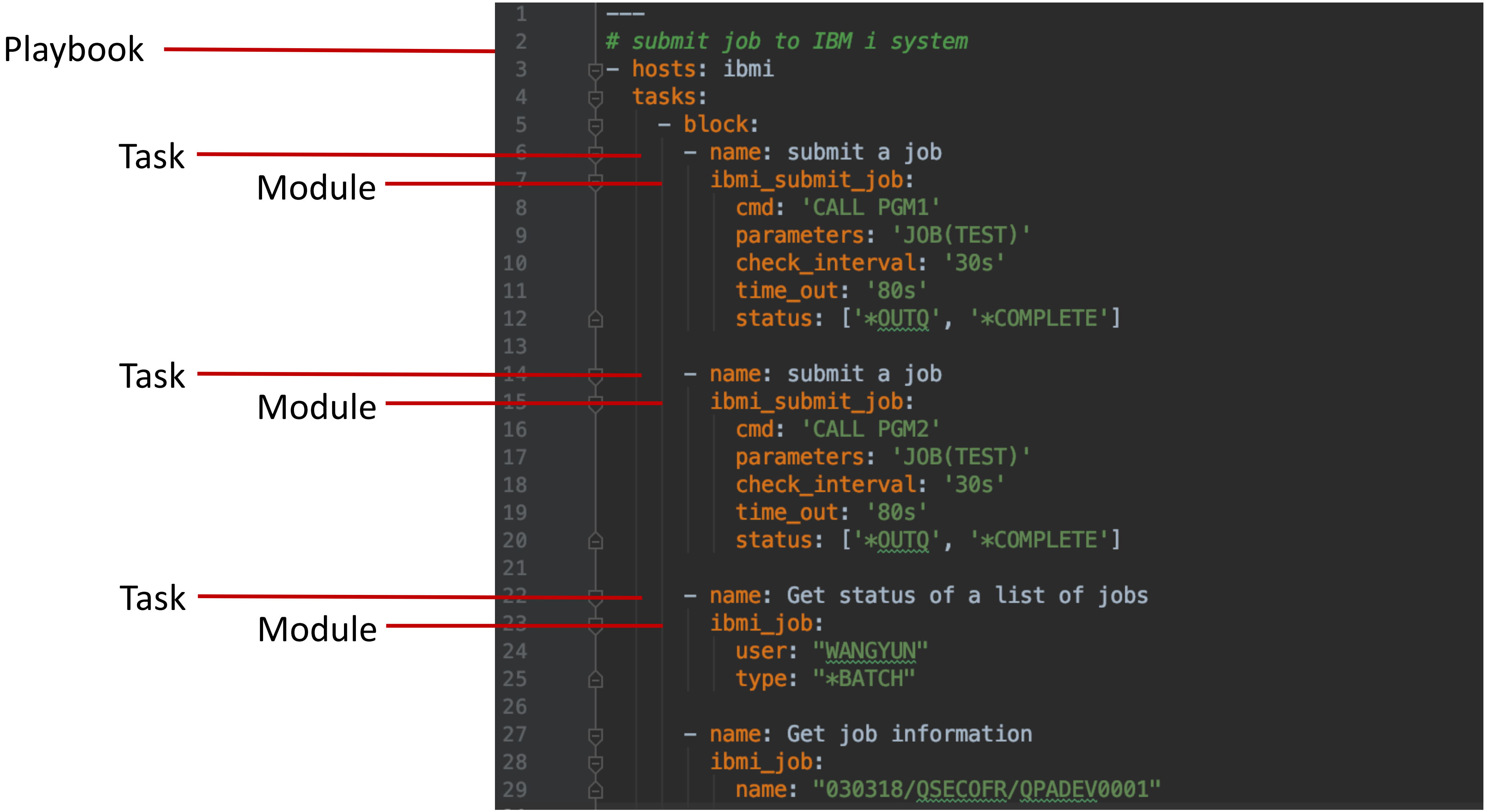
Inventory – a list of managed nodes so that Ansible understands the overall IT landscape

Modules – units of code that Ansible executes; hundreds of modules provided out-of-box; thousands of community modules also available

Tasks – units of action in Ansible (invoke a set of modules to do something useful)

Playbooks – ordered list of tasks and written in YAML

Playbook – YAML script of your tasks



Playbook

Task

Module

Task

Module

Task

Module

```yaml
---
# submit job to IBM i system
- hosts: ibmi
  tasks:
    - block:
      - name: submit a job
        ibmi_submit_job:
          cmd: 'CALL PGM1'
          parameters: 'JOB(TEST)'
          check_interval: '30s'
          time_out: '80s'
          status: ['*OUTQ', '*COMPLETE']

      - name: submit a job
        ibmi_submit_job:
          cmd: 'CALL PGM2'
          parameters: 'JOB(TEST)'
          check_interval: '30s'
          time_out: '80s'
          status: ['*OUTQ', '*COMPLETE']

      - name: Get status of a list of jobs
        ibmi_job:
          user: "WANGYUN"
          type: "*BATCH"

      - name: Get job information
        ibmi_job:
          name: "030318/QSECOFR/QPADEV0001"
```

Ansible Tower

Management interface that makes Ansible much friendlier at "enterprise scale" by way of a nice graphical interface

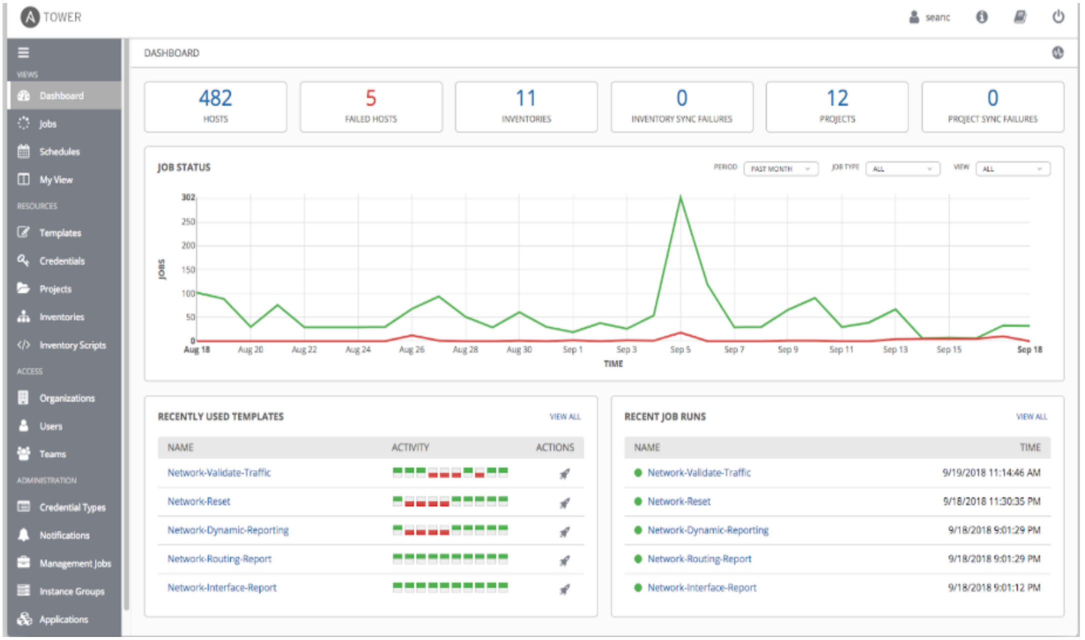
- View inventories
- Run playbooks
- Review logs and more

Supported on xLinux only right now; manages to all supported OS endpoints

Available for subscription purchase from Red Hat

Red Hat's commercial form of Ansible AWX

Red Hat
Ansible
Tower



482	5	11	0	12	0
HOSTS	FAILED HOSTS	INVENTORIES	INVENTORY SYNC FAILURES	PROJECTS	PROJECT SYNC FAILURES

PERIOD: PAST MONTH
JOB TYPE: ALL
VIEW: ALL

RECENTLY USED TEMPLATES

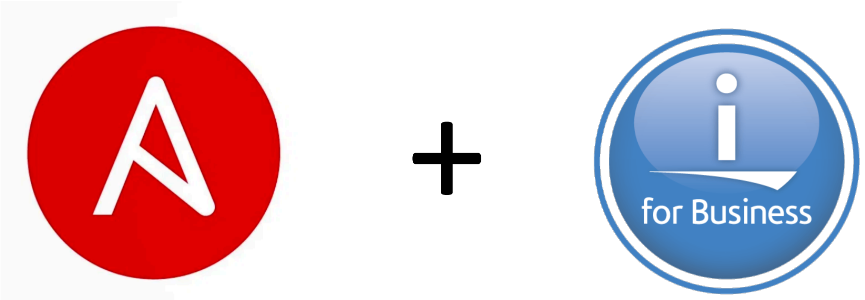
NAME	ACTIVITY	ACTIONS
Network-Validate-Traffic	🟩🟩🟩🟥🟥🟥🟩🟥🟩🟩	
Network-Reset	🟩🟥🟥🟥🟥🟩🟩🟩🟩🟩	
Network-Dynamic-Reporting	🟩🟥🟥🟥🟥🟩🟩🟩🟩🟩	
Network-Routing-Report	🟩🟩🟩🟩🟩🟩🟩🟩🟩🟩	
Network-Interface-Report	🟩🟩🟩🟩🟩🟩🟩🟩🟩🟩	

RECENT JOB RUNS

NAME	TIME
● Network-Validate-Traffic	9/19/2018 11:14:46 AM
● Network-Reset	9/18/2018 11:30:35 PM
● Network-Dynamic-Reporting	9/18/2018 9:01:29 PM
● Network-Routing-Report	9/18/2018 9:01:29 PM
● Network-Interface-Report	9/18/2018 9:01:12 PM

Agenda

- Introduce Ansible
- Ansible and IBM i
- Examples
- Real customer use cases



Use Ansible with IBM Systems

Consistency

- Integrate z/OS, AIX, IBM i and Linux into an enterprise automation strategy in a consistent way
- Enables common approach to hybrid applications and infrastructure management

Transparency

- Enable transparent visibility of z/OS, AIX, IBM i and Linux automation when orchestrated by Ansible
- Drive best practices to manage automation in source control to move towards infrastructure as code
- Contribute to breaking down cultural walls

Skills

- Ansible and Python skills are readily available in the marketplace and can be applied to z/OS, AIX, IBM i and Linux

Ansible for IBM i use cases

Common IBM i administrator tasks

- Fix installation and system maintenance
- Application and programs deployment, both native and PASE
- IBM i work management, security management, and other common IBM i tasks
- IBM i Open Source package installation and update via YUM or PIP for Python
- Adhoc or scripted CL and SQL commands for very specific IBM i admin tasks
- Create Ansible playbooks to run a workflow with multiple actions performed via available modules
- Ansible is already there for other platforms. Leverage Ansible team (with no IBM i background) to manage some of the IBM i tasks

Common IBM i development tasks

- Continuous development and testing – automate build, unit test, deploy process, etc.
- Easy re-setup / re-clone application dev/testing environment in different stage of product life cycle

IBM i cloud tasks

- Integrates with CAM and Terraform for VM provision and configuration in IBM cloud platforms
- Orchestrations to form cloud solutions

Provisioning

Configuration Management

Application Deployment

Continuous Delivery

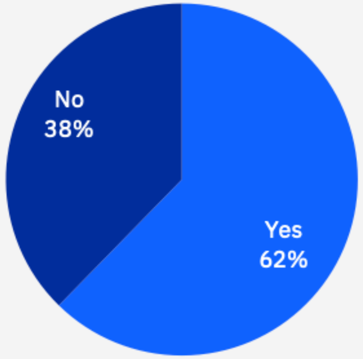
Security Automation

Orchestration

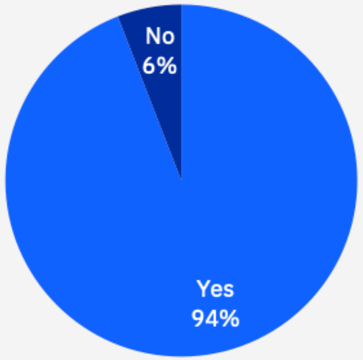
From a recent Ansible Webinar – Summary

No. of attendees: 149

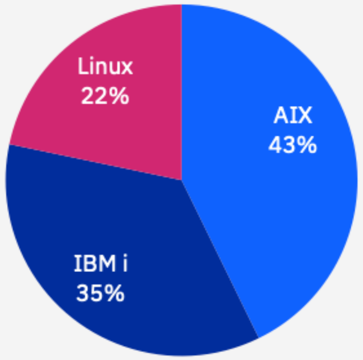
Are you familiar with Ansible?



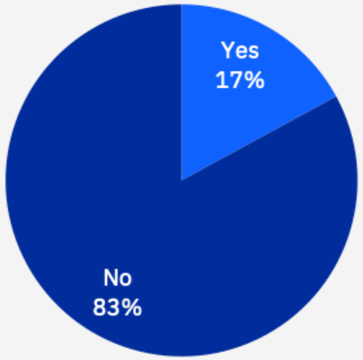
Are you using IBM Power Systems?



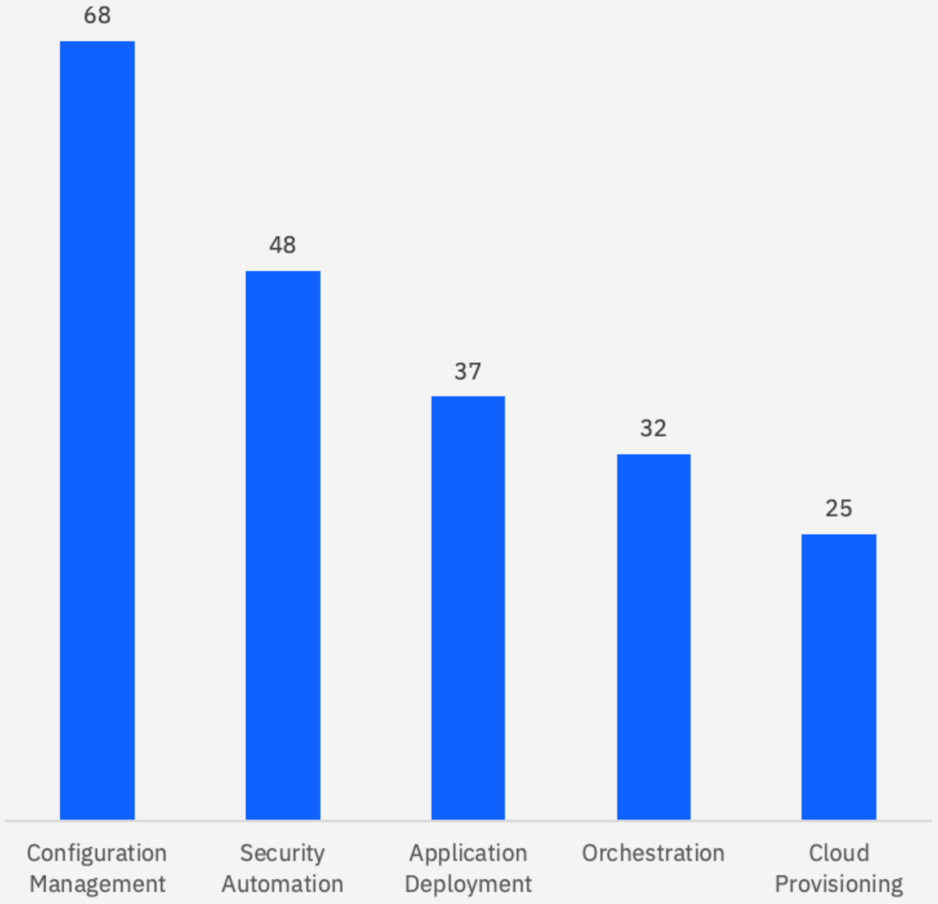
Which OS are you running on Power?



Have you used any Ansible modules to automate AIX and IBM i?



Which tasks are you planning to automate with Ansible content for AIX and IBM i?



Ansible for IBM i Roadmap

Current Available

June

- PTF and LPP Management
- Open Source Package Management
- Object Management
- PASE Support
- Work Management Runtime
- Device Management
- IASP support
- Reusable Playbook Solutions
- Handle new requirements/problems

September

- Advanced fix management
- Basic Network Configuration
- Work Management
- Security Management
- Message Handling
- Reusable Playbook Solutions
- Handle new requirements/problems

Building blocks to support IBM i

Next Step

Next release 1

- Solution and product configuration
- SQL services bundles
- Application Management
- System health bundles
- Reusable Playbook Solutions
- Handle new requirements/problems

Next release 2

- Manage in hybrid cloud
- Application management in cloud
- Enhance existing functions
- Reusable Playbook Solutions
- Handle new requirements/problems

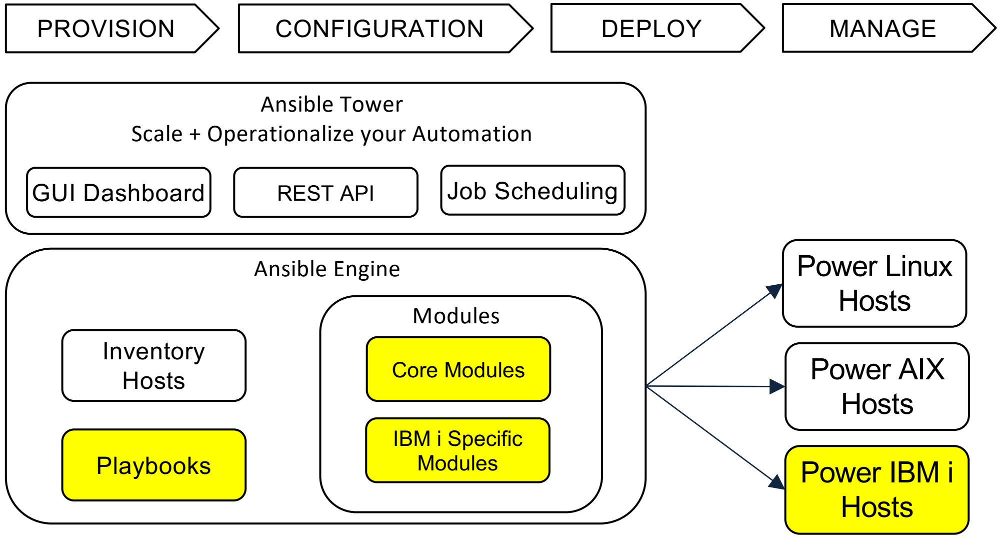
Focus more on solutions for particular use cases
Continue enhancing basic modules per requests

Note: All statements regarding IBM's future direction and intent are subject to change or withdrawal without notice and represent goals and objectives only.

What are available today?

IBM i systems are managed by Ansible

- Ansible server runs on Linux (official support) or IBM i (community support)
- IBM i systems to be managed as endpoints
- Only support Python 3 at IBM i endpoint from release 1.0.2
- Modules, roles, playbooks and plugins are developed
- Available in Ansible Galaxy and Automation Hub support



The yellow parts are what Ansible for i focuses

Functions available today

- Ansible Content for IBM Power Systems - IBM i
  - Auto installation playbook to enable IBM i to be managed by Ansible
  - 50+ modules are available
  - 10+ roles (roles are playbooks that can be repeatedly used)
  - Playbook samples and test cases
- Many Ansible core modules can be executed on IBM i
  - Sanity testing has been done. Need more in the coming months.
- VM provisioning
  - Use OpenStack modules
- Resources
  - Galaxy link: https://galaxy.ansible.com/ibm/power_ibmi
  - GitHub repo: https://github.com/IBM/ansible-for-i/
  - Documentation: https://ibm.github.io/ansible-for-i/index.html
  - Automation Hub: https://cloud.redhat.com/ansible/automation-hub/ibm/power_ibmi

Ansible for IBM i modules

Object Management

ibmi_copy
ibmi_fetch
ibmi_lib_restore
ibmi_lib_save
ibmi_object_authority
ibmi_object_find
ibmi_object_restore
ibmi_object_save
ibmi_sync
ibmi_sync_files
ibmi_synchronize
ibmi_synchronize_files

Fix Management

ibmi_display_fix
ibmi_download_fix
ibmi_fix
ibmi_fix_group_check
ibmi_fix_imgclg
ibmi_fix_repo
ibmi_install_product_from_savf
ibmi_save_product_to_savf
ibmi_uninstall_product

IASP Management

ibmi_device_vary
ibmi_get_nonconfigure_disks
ibmi_iasp

Network

ibmi_ethernet_port
ibmi_nrg_link
ibmi_tcp_interface
ibmi_tcp_server_service

Command Support

ibmi_cl_command
ibmi_rtv_command
ibmi_script
ibmi_script_execute
ibmi_sql_execute
ibmi_sql_query

Work Management

ibmi_at
ibmi_display_subsystem
ibmi_end_subsystem
ibmi_host_server_service
ibmi_job
ibmi_message
ibmi_query_job_log
ibmi_reboot
ibmi_reply_message
ibmi_start_subsystem
ibmi_submit_job

Security

ibmi_sysval
ibmi_user_and_group
ibmi_user_compliance_check

Ansible core modules that support IBM i

Core modules are common modules supporting various of operating systems
They are out-of-box modules developed by Ansible core team
IBM i can leverage below core modules for automations

Ansible Core Modules

- command
- raw
- script
- shell
- pip
- yum
- pause
- wait_for_connection
- authorized_key
- ping
- setup

Ansible Core Modules

- assemble
- blockinfile
- copy
- fetch
- file
- find
- lineinfile
- stat
- synchronize
- git

- Only sanity testing has been done for these Ansible core modules for now. Will do more testing in the future.
- Core modules support IBM i under PASE

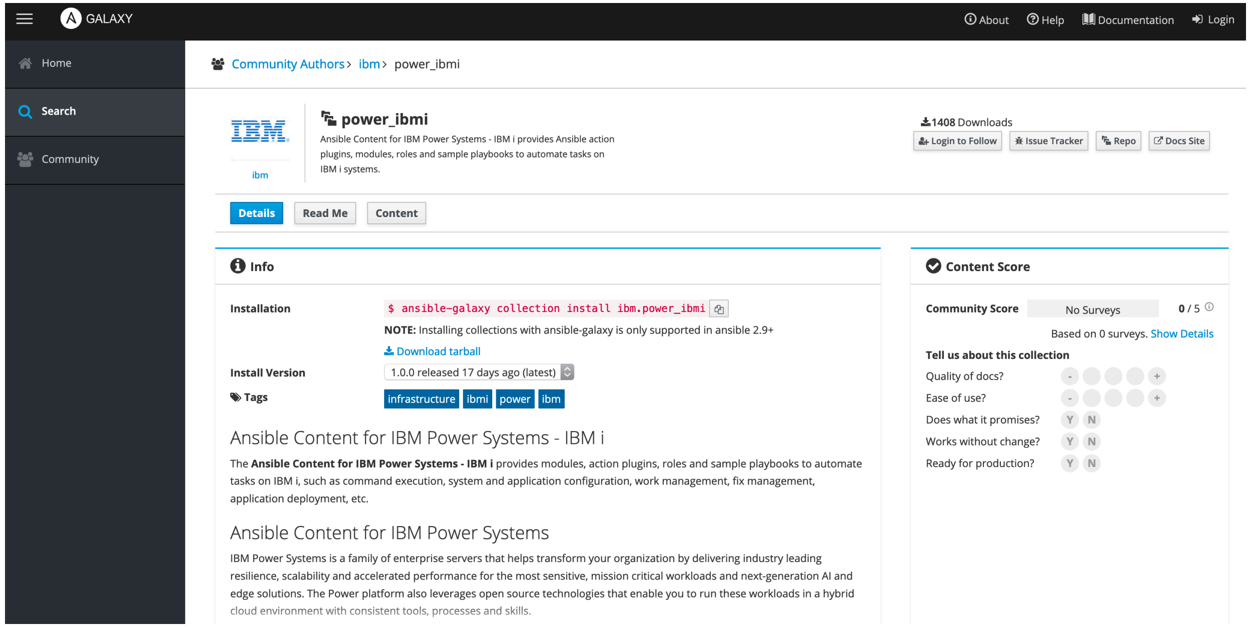
Ansible Galaxy and Automation Platform Support

Ansible Galaxy Details:

- Galaxy is the upstream community for sharing Ansible Collections
- Delivered using the collection packaging mechanism
- https://galaxy.ansible.com/ibm/power_ibmi

Certified Integration: Ansible and IBM Power Systems

- Enabling Red Hat Ansible Automation Platform across all OS environments running on Power Systems.
- Power Systems IBM i collection fits under this umbrella of content.
- https://cloud.redhat.com/ansible/automation-hub/ibm/power_ibmi



power_ibmi

Ansible Content for IBM Power Systems - IBM i provides Ansible action plugins, modules, roles and sample playbooks to automate tasks on IBM i systems.

1408 Downloads

Info

Installation `$ ansible-galaxy collection install ibm.power_ibmi`

NOTE: Installing collections with ansible-galaxy is only supported in ansible 2.9+

Download tarball

Install Version 1.0.0 released 17 days ago (latest)

Tags infrastructure ibmi power ibm

Ansible Content for IBM Power Systems - IBM i

The **Ansible Content for IBM Power Systems - IBM i** provides modules, action plugins, roles and sample playbooks to automate tasks on IBM i, such as command execution, system and application configuration, work management, fix management, application deployment, etc.

Ansible Content for IBM Power Systems

IBM Power Systems is a family of enterprise servers that helps transform your organization by delivering industry leading resilience, scalability and accelerated performance for the most sensitive, mission critical workloads and next-generation AI and edge solutions. The Power platform also leverages open source technologies that enable you to run these workloads in a hybrid cloud environment with consistent tools, processes and skills.

Content Score

Community Score No Surveys **0** / 5

Based on 0 surveys. Show Details

Tell us about this collection

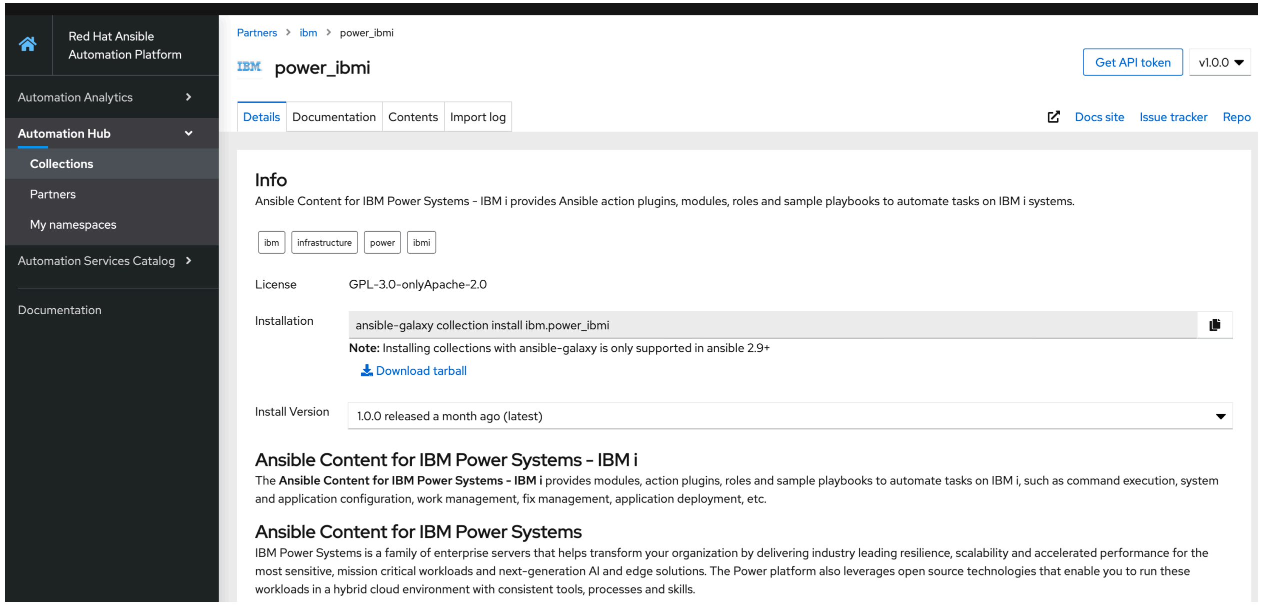
Quality of docs? - +

Ease of use? - +

Does what it promises? Y N

Works without change? Y N

Ready for production? Y N



power_ibmi

Info

Ansible Content for IBM Power Systems - IBM i provides Ansible action plugins, modules, roles and sample playbooks to automate tasks on IBM i systems.

ibm infrastructure power ibmi

License GPL-3.0-onlyApache-2.0

Installation `ansible-galaxy collection install ibm.power_ibmi`

Note: Installing collections with ansible-galaxy is only supported in ansible 2.9+

Download tarball

Install Version 1.0.0 released a month ago (latest)

Ansible Content for IBM Power Systems - IBM i

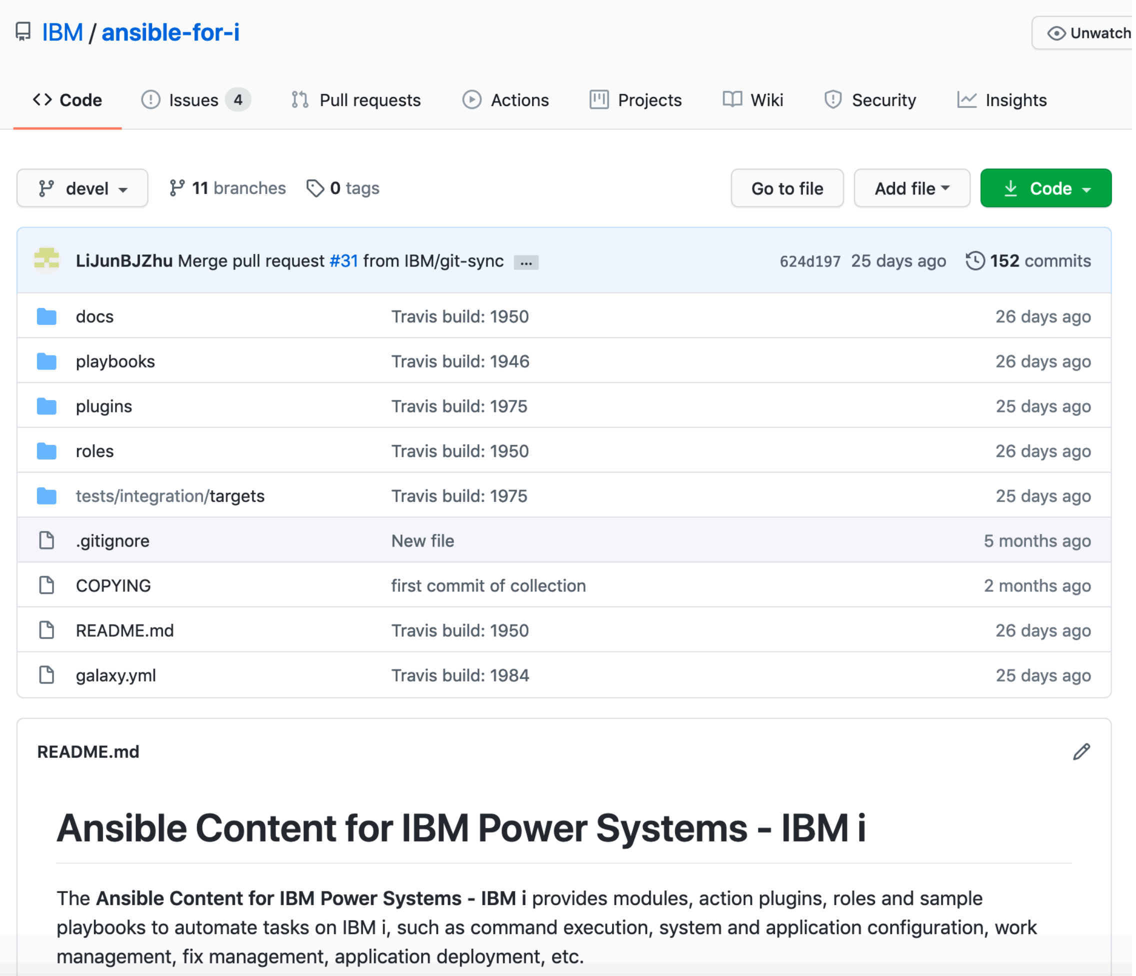
The **Ansible Content for IBM Power Systems - IBM i** provides modules, action plugins, roles and sample playbooks to automate tasks on IBM i, such as command execution, system and application configuration, work management, fix management, application deployment, etc.

Ansible Content for IBM Power Systems

IBM Power Systems is a family of enterprise servers that helps transform your organization by delivering industry leading resilience, scalability and accelerated performance for the most sensitive, mission critical workloads and next-generation AI and edge solutions. The Power platform also leverages open source technologies that enable you to run these workloads in a hybrid cloud environment with consistent tools, processes and skills.

Ansible for IBM i GitHub Repository

- Collections of IBM i modules, plugins, roles and playbook are openly stored in Github.
  - https://github.com/IBM/ansible-for-i
  - Directory of plugins contains modules and plugins.
- Frequent updates.
- Test cases used to test the modules are also in the repository – use them as examples.
  - Find the test cases in tests directory.
- Provide different branches for the source of different releases
- Partners are welcomed to contribute
  - Early testing and roles development
  - Contribute your use cases



IBM / ansible-for-i

LiJunBJZhu Merge pull request #31 from IBM/git-sync ... 624d197 25 days ago 152 commits

docs	Travis build: 1950	26 days ago
playbooks	Travis build: 1946	26 days ago
plugins	Travis build: 1975	25 days ago
roles	Travis build: 1950	26 days ago
tests/integration/targets	Travis build: 1975	25 days ago
.gitignore	New file	5 months ago
COPYING	first commit of collection	2 months ago
README.md	Travis build: 1950	26 days ago
galaxy.yml	Travis build: 1984	25 days ago

README.md

Ansible Content for IBM Power Systems - IBM i

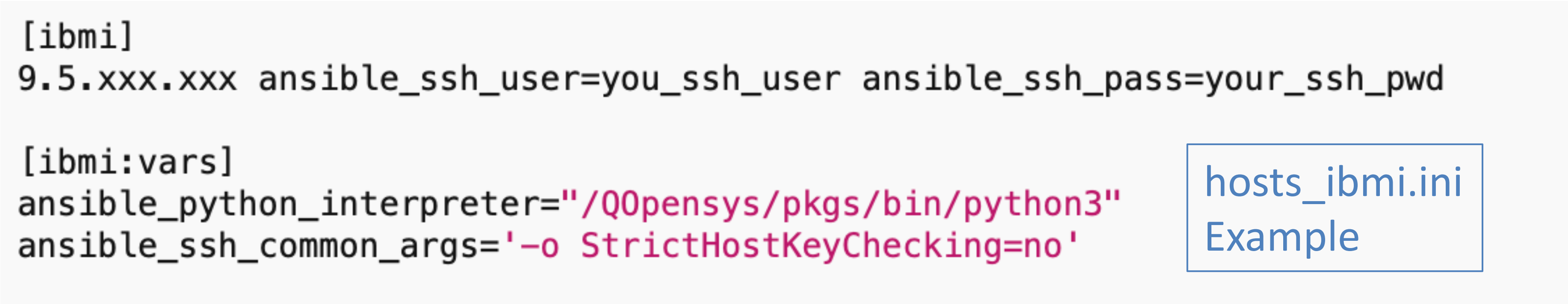
The **Ansible Content for IBM Power Systems - IBM i** provides modules, action plugins, roles and sample playbooks to automate tasks on IBM i, such as command execution, system and application configuration, work management, fix management, application deployment, etc.

Agenda

- Introduce Ansible
- Ansible and IBM i
- **Examples**
- Real customer use cases

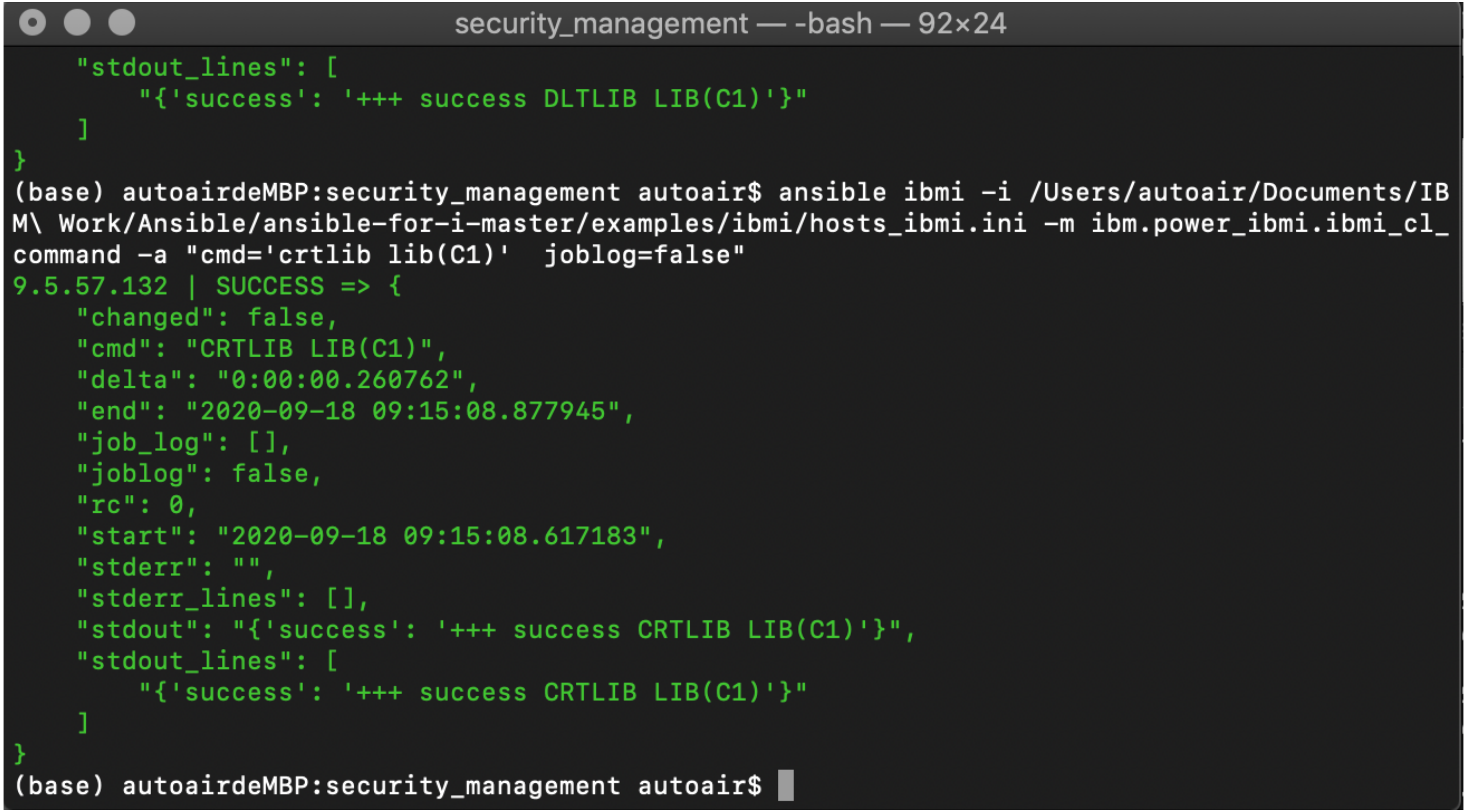
Example 1: Run an IBM i module interactively

- Run IBM i module in the command terminal.
- ansible ibmi -m ibm.power_ibmi.ibmi_cl_command -a "cmd='crtlib lib(C1)' joblog=true"
- ansible ibmi -i /yourpath/hosts_ibmi.ini -m ibm.power_ibmi.ibmi_cl_command -a "cmd='crtlib lib(C1)'"
    - '-i /yourpath/hosts_ibmi.ini' defines the endpoint information and its variables
    - 'ibmi' is the inventory group defined in /yourpath/hosts_ibmi.ini
    - '-m ibm.power_ibmi.ibmi_cl_command' defines the module used is from ibm.power_ibmi collections.



```
[ibmi]
9.5.xxx.xxx ansible_ssh_user=you_ssh_user ansible_ssh_pass=your_ssh_pwd

[ibmi:vars]
ansible_python_interpreter="/QOpensys/pkgs/bin/python3"
ansible_ssh_common_args='-o StrictHostKeyChecking=no'
```

hosts_ibmi.ini
Example



```
    "stdout_lines": [
        "{'success': '+++ success DLTLIB LIB(C1)'}"
    ]
}
(base) autoairdeMBP:security_management autoair$ ansible ibmi -i /Users/autoair/Documents/IB
M\ Work/Ansible/ansible-for-i-master/examples/ibmi/hosts_ibmi.ini -m ibm.power_ibmi.ibmi_cl_
command -a "cmd='crtlib lib(C1)'  joblog=false"
9.5.57.132 | SUCCESS => {
    "changed": false,
    "cmd": "CRTLIB LIB(C1)",
    "delta": "0:00:00.260762",
    "end": "2020-09-18 09:15:08.877945",
    "job_log": [],
    "joblog": false,
    "rc": 0,
    "start": "2020-09-18 09:15:08.617183",
    "stderr": "",
    "stderr_lines": [],
    "stdout": "{'success': '+++ success CRTLIB LIB(C1)'}",
    "stdout_lines": [
        "{'success': '+++ success CRTLIB LIB(C1)'}"
    ]
}
(base) autoairdeMBP:security_management autoair$
```

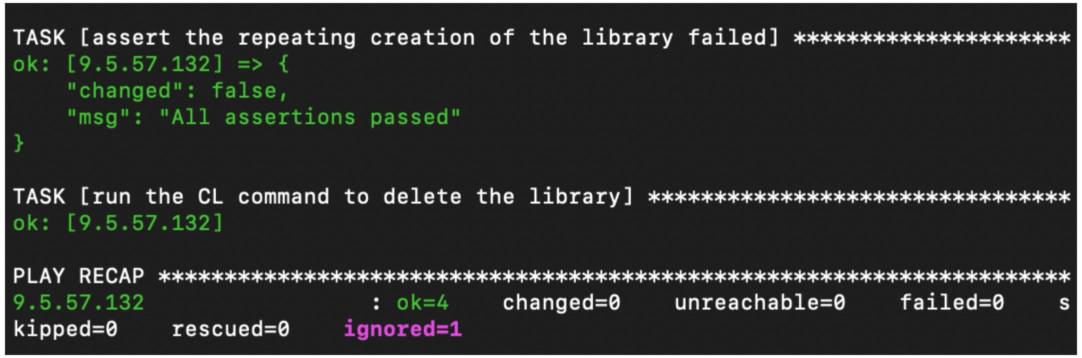
Example 2: Run Ansible Playbook with IBM i modules

ansible-playbook ibmi-cl-command-sample.yml

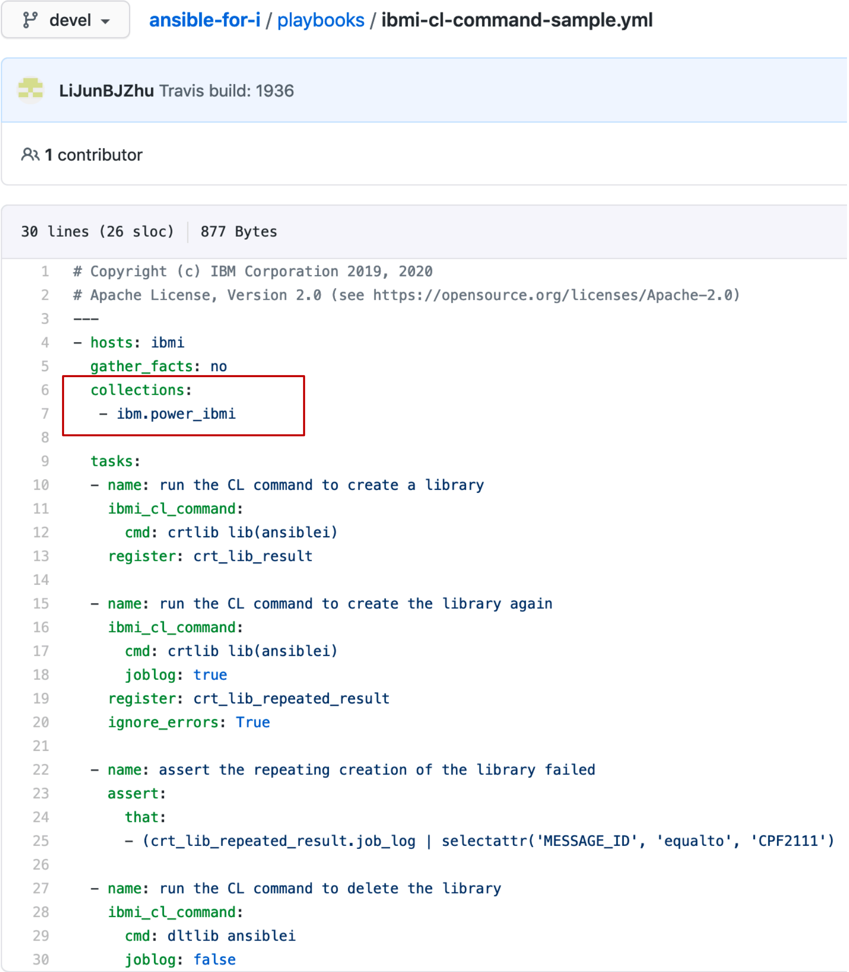
ansible-playbook -i /yourpath/hosts_ibmi.ini ibmi-cl-command-sample.yml

More Playbook Examples:

- https://github.com/IBM/ansible-for-i/tree/devel/tests/integration/targets
- https://github.com/IBM/ansible-for-i/tree/devel/playbooks



```
TASK [assert the repeating creation of the library failed] **********************
ok: [9.5.57.132] => {
    "changed": false,
    "msg": "All assertions passed"
}

TASK [run the CL command to delete the library] ********************************
ok: [9.5.57.132]

PLAY RECAP *********************************************************************
9.5.57.132                 : ok=4    changed=0    unreachable=0    failed=0    s
kipped=0    rescued=0    ignored=1
```

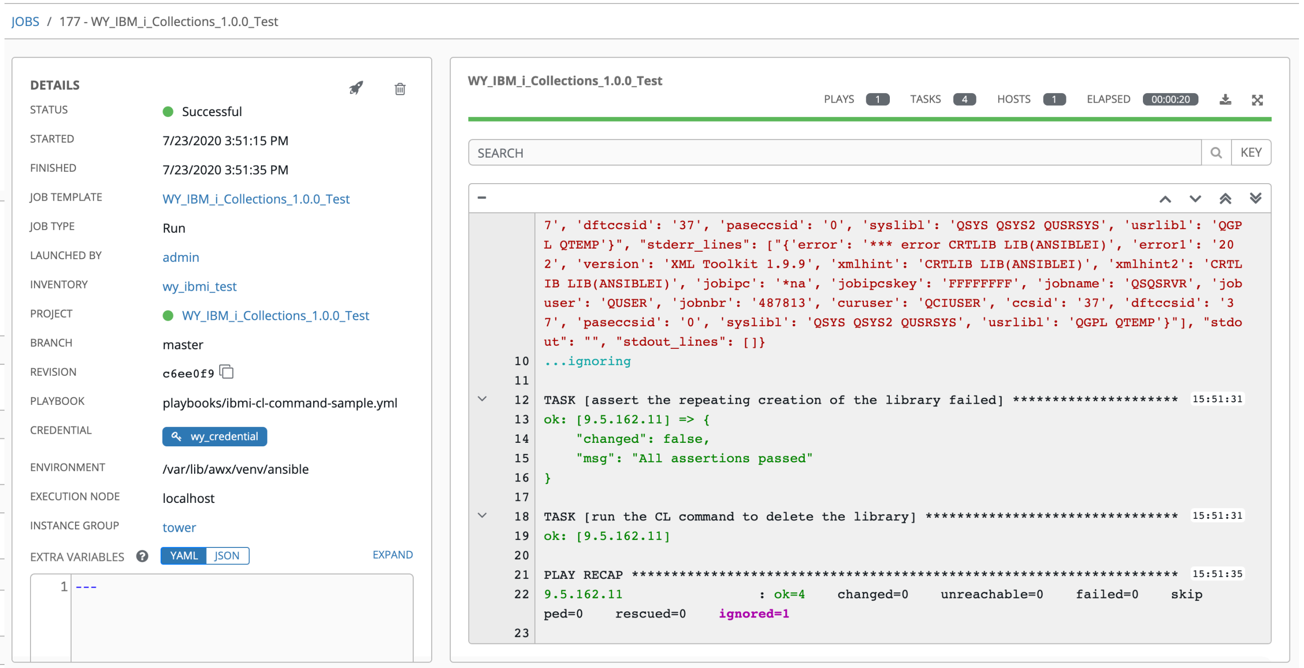


devel
ansible-for-i / playbooks / ibmi-cl-command-sample.yml
LiJunBJZhu Travis build: 1936
1 contributor
30 lines (26 sloc) | 877 Bytes

```yaml
# Copyright (c) IBM Corporation 2019, 2020
# Apache License, Version 2.0 (see https://opensource.org/licenses/Apache-2.0)
---
- hosts: ibmi
  gather_facts: no
  collections:
   - ibm.power_ibmi

  tasks:
  - name: run the CL command to create a library
    ibmi_cl_command:
      cmd: crtlib lib(ansiblei)
    register: crt_lib_result

  - name: run the CL command to create the library again
    ibmi_cl_command:
      cmd: crtlib lib(ansiblei)
      joblog: true
    register: crt_lib_repeated_result
    ignore_errors: True

  - name: assert the repeating creation of the library failed
    assert:
      that:
      - (crt_lib_repeated_result.job_log | selectattr('MESSAGE_ID', 'equalto', 'CPF2111')

  - name: run the CL command to delete the library
    ibmi_cl_command:
      cmd: dltlib ansiblei
      joblog: false
```

Example 3: Run IBM i Tasks with Ansible Tower

- Create a GitHub repository to store your playbook scripts
  - Sample repository: https://github.com/airwangyun/IBM_i_Ansible_Tower_Demo
  - The repository needs to define the collections information in the file of requirements.yml under collections directory
- Use Inventories tab to manage IBM i hosts.
- Create project to manage IBM i systems
- Use Templates tab to define jobs to run
  - Playbooks in the repository can be selected to run.



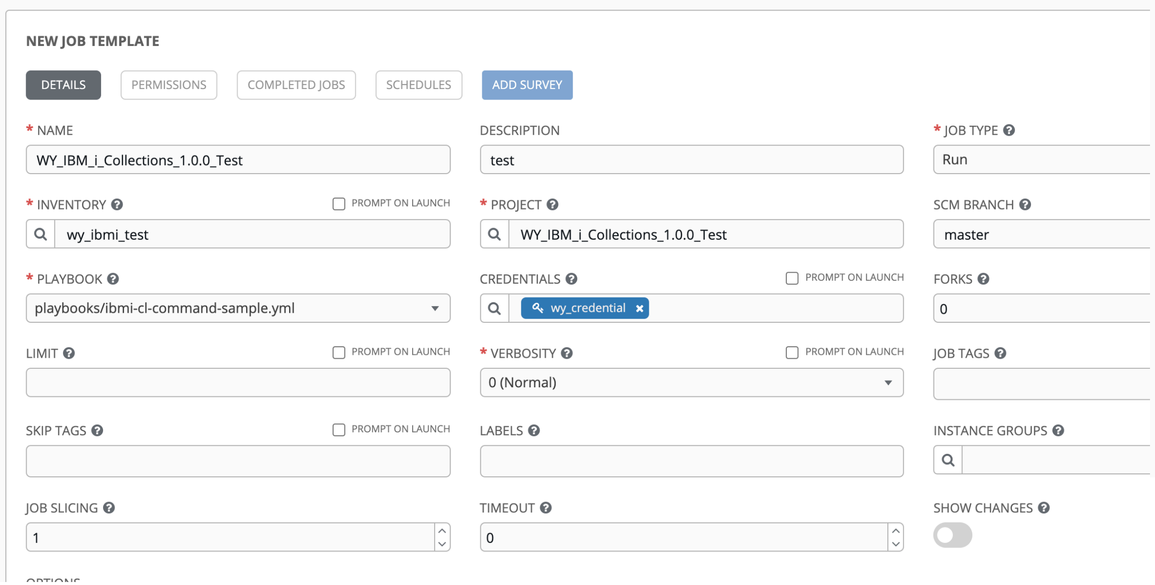
DETAILS

STATUS	● Successful
STARTED	7/23/2020 3:51:15 PM
FINISHED	7/23/2020 3:51:35 PM
JOB TEMPLATE	WY_IBM_i_Collections_1.0.0_Test
JOB TYPE	Run
LAUNCHED BY	admin
INVENTORY	wy_ibmi_test
PROJECT	● WY_IBM_i_Collections_1.0.0_Test
BRANCH	master
REVISION	c6ee0f9
PLAYBOOK	playbooks/ibmi-cl-command-sample.yml
CREDENTIAL	wy_credential
ENVIRONMENT	/var/lib/awx/venv/ansible
EXECUTION NODE	localhost
INSTANCE GROUP	tower
EXTRA VARIABLES	YAML JSON EXPAND

```
---
```

WY_IBM_i_Collections_1.0.0_Test

```
7', 'dftccsid': '37', 'paseccsid': '0', 'syslibl': 'QSYS QSYS2 QUSRSYS', 'usrlibl': 'QGP
L QTEMP'}", "stderr_lines": ["{'error': '*** error CRTLIB LIB(ANSIBLEI)', 'error1': '20
2', 'version': 'XML Toolkit 1.9.9', 'xmlhint': 'CRTLIB LIB(ANSIBLEI)', 'xmlhint2': 'CRTL
IB LIB(ANSIBLEI)', 'jobipc': '*na', 'jobipcskey': 'FFFFFFFF', 'jobname': 'QSQSRVR', 'job
user': 'QUSER', 'jobnbr': '487813', 'curuser': 'QCIUSER', 'ccsid': '37', 'dftccsid': '3
7', 'paseccsid': '0', 'syslibl': 'QSYS QSYS2 QUSRSYS', 'usrlibl': 'QGPL QTEMP'}"], "stdo
ut": "", "stdout_lines": []}
...ignoring

TASK [assert the repeating creation of the library failed] ********************* 15:51:31
ok: [9.5.162.11] => {
    "changed": false,
    "msg": "All assertions passed"
}

TASK [run the CL command to delete the library] ******************************** 15:51:31
ok: [9.5.162.11]

PLAY RECAP ********************************************************************* 15:51:35
9.5.162.11                 : ok=4    changed=0    unreachable=0    failed=0    skip
ped=0    rescued=0    ignored=1

```


NEW JOB TEMPLATE

* NAME: WY_IBM_i_Collections_1.0.0_Test	DESCRIPTION: test	* JOB TYPE: Run
* INVENTORY (□ PROMPT ON LAUNCH): wy_ibmi_test	* PROJECT: WY_IBM_i_Collections_1.0.0_Test	SCM BRANCH: master
* PLAYBOOK: playbooks/ibmi-cl-command-sample.yml	CREDENTIALS (□ PROMPT ON LAUNCH): wy_credential	FORKS: 0
LIMIT (□ PROMPT ON LAUNCH):	* VERBOSITY (□ PROMPT ON LAUNCH): 0 (Normal)	JOB TAGS:
SKIP TAGS (□ PROMPT ON LAUNCH):	LABELS:	INSTANCE GROUPS:
JOB SLICING: 1	TIMEOUT: 0	SHOW CHANGES (□ PROMPT ON LAUNCH):

OPTIONS

Agenda

- Introduce Ansible
- Ansible and IBM i
- Examples
- Real customer use cases

Use Case 1: Fix management

- Requirements
  - Know how current are the PTF groups on my IBM i systems. Notify me when I’m behind.
  - Timely download PTF groups from IBM.
  - Send and install PTFs and PTF groups to single or multiple systems.
  - How many PTFs in a group are NOT installed? (PTFs have been removed for different reasons)
  - What if I want to further manage all the downloaded SAVFs and Images?

Building Blocks for Fix Management

Modules and roles are provided for fix management

- Both modules and roles can be directly used for your simple fix management tasks
- Go to https://github.com/IBM/ansible-for-i/tree/devel/tests/integration/targets for module examples
- Go to README.md under https://github.com/IBM/ansible-for-i/tree/devel/roles for role example

Fix Management Modules

ibmi_display_fix
ibmi_download_fix
ibmi_fix
ibmi_fix_group_check
ibmi_fix_imgclg
ibmi_fix_repo
ibmi_install_product_from_savf
ibmi_save_product_to_savf
ibmi_uninstall_product

Fix Management Roles

apply_all_loaded_ptfs
check_group
check_product
check_ptf
download_individual_ptfs
load_apply_ptfs
load_ptf
sync_apply_individual_ptfs

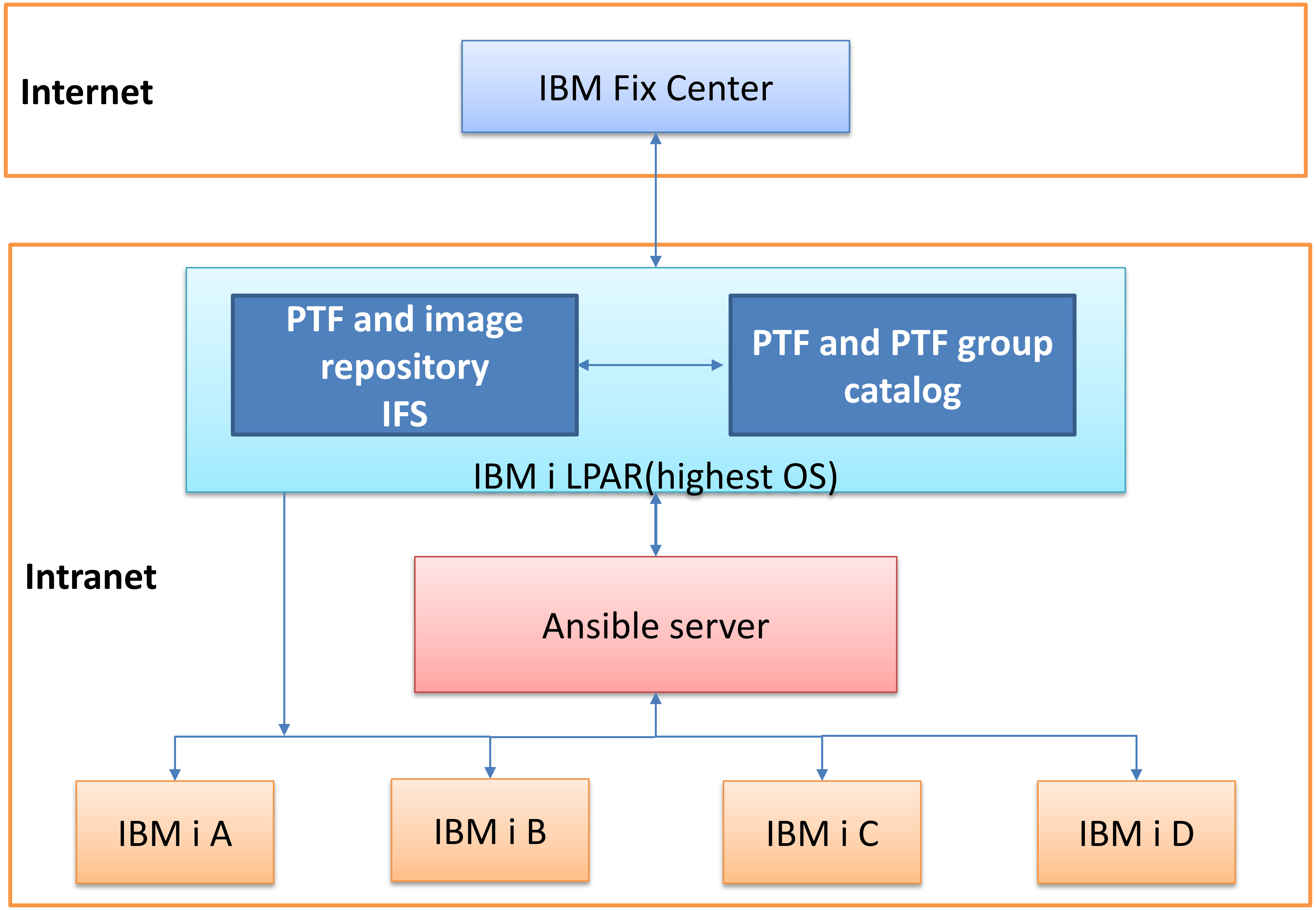
Use Case 1: Fix management

- Requirements
  - Know how current are the PTF groups on my IBM i systems. Notify me when I'm behind.
    - ibmi_fix_group_check: Retrieve the latest PTF group information from PSP server
  - Timely download PTF groups from IBM.
    - ibmi_download_fix
  - Send and install PTFs and PTF groups to single or multiple systems.
    - ibmi_fix: Install, remove or query an individual fix or a set of fixes on to IBM i system
    - ibmi_fix_imgclg: Install fixes such as PTF, PTF Group to the target IBM i system by image catalog
    - ibmi_synchronize_files: Sync files to target IBM i systems
  - How many PTFs in a group are NOT installed? (PTFs have been removed for different reasons)
  - What if I want to further manage all the downloaded SAVFs and Images?

Use Case 1: Fix management

- Requirements
  - Know how current are the PTF groups on my IBM i systems. Notify me when I’m behind.
    - ibmi_fix_group_check: Retrieve the latest PTF group information from PSP server
  - Timely download PTF groups from IBM.
    - ibmi_download_fix
  - Send and install PTFs and PTF groups to single or multiple systems.
    - ibmi_fix: Install, remove or query an individual fix or a set of fixes on to IBM i system
    - ibmi_fix_imgclg: Install fixes such as PTF, PTF Group to the target IBM i system by image catalog
    - ibmi_synchronize_files: Sync files to target IBM i systems
  - How many PTFs in a group are NOT installed? (PTFs have been removed for different reasons)
  - What if I want to further manage all the downloaded SAVFs and Images?

Fix management – Advanced



Download Fix – Download the fixes from IBM fix center to local fix repo

Fix Repo Mgmt

- Manage PTF and Group PTF information
- Record the PTF and Group information into PTF/Group catalog

Compare Fix

- Compare the fix level between PSP and fix repo
- Compare the fix level between local repo and IBM i endpoint system

Send fix

- Send over the fixes from repo to endpoint systems

Install the fixes

- Install PTF and Group to IBM i endpoint systems

Fix management – Advanced

- Provide infrastructure to manage PTF repository
  - Automatically check and download PTF groups
    - SNDPTFORD needs to be enabled
  - Repository is managed to store SAVFs and images downloaded from IBM fix center
  - Catalog(SQLite database tables) to manage PTF and Group information
    - What has been downloaded
    - Detail PTF list in a specific group
    - Support individual PTF and PTF group
  - Support manual put and update PTF into repository
  - Compare and send fixes from repository to target IBM i systems
  - Compare PTF difference between endpoint IBM i systems and repository
- Use case solution sample for you to download and reuse
  - https://github.com/IBM/ansible-for-i/tree/devel/usecases/fix_management

Use Case 2: Security management

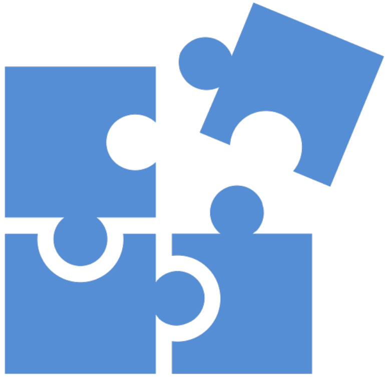
- Requirements
  - Security compliance checking
    - System value checking
    - User profile checking
    - Object authority checking
    - Network security checking
    - Other security related checking
  - Change security settings for the incompliance
  - Tasks run under user profile with least authorities
    - Only run the task under powerful user when needed

Use Case 2: Security management

- Requirements
  - Security compliance checking
    - System value checking - ibmi_sysval
    - User profile checking - ibmi_user_compliance_check
    - Object authority checking - ibmi_object_authority
    - Other security related checking – use ibmi_cl_command, ibmi_sql_command and ibmi_ ibmi_rtv_command
  - Tasks run under user profile with least authorities
    - Only run the task under powerful user when needed
      - 'become' function is provided for most of the IBM i modules. ('become' option is provided as module arguments, not as become plugin)
      - Switch to a more powerful user when needed
- Security compliance checking use case samples
  - https://github.com/IBM/ansible-for-i/tree/devel/usecases/security_management
  - You could directly use the samples and do changes for your cases

Use Case 3: Application deployment

- Requirements
  - Move application SAVF to target IBM i system
  - Restore SAVF at target IBM i system
  - Run scripts to deploy applications.
    - Some are long running ones
  - Configure system settings for applications.
    - Eg. Create job schedule entry
  - Back out when error occurs
    - Remove libraries, remove IFS objects
    - Remove job schedule entry



Use Case 3: Application deployment

- Requirements
  - Move application SAVF to target IBM i system
    - ibmi_sync, ibmi_sync_files, ibmi_synchronize, ibmi_synchronize_files
  - Restore SAVF at target IBM i system
    - ibmi_lib_restore, ibmi_object_restore
  - Run scripts to deploy applications.
    - Some are long running ones
      - ibmi_submit_job, ibmi_job, ibmi_query_job_log
  - Configure system settings for applications.
    - Eg. Create job schedule entry
      - ibmi_at
  - Back out when error occurs
    - Remove libraries, remove IFS objects – ibmi_cl_command, command
    - Remove job schedule entry - ibmi_cl_command

Use Case 4: Back up data by Ansible operator

- Requirements
  - Ansible operator does not have IBM i skills
  - Get error messages for failures without allowing non-IBM i operator to logon to the green screen
  - Automatically solve common errors
  - Use a list of save/restore commands to back up data from one system to another one
    - Objects and IFS files

© 2018 IBM Corporation

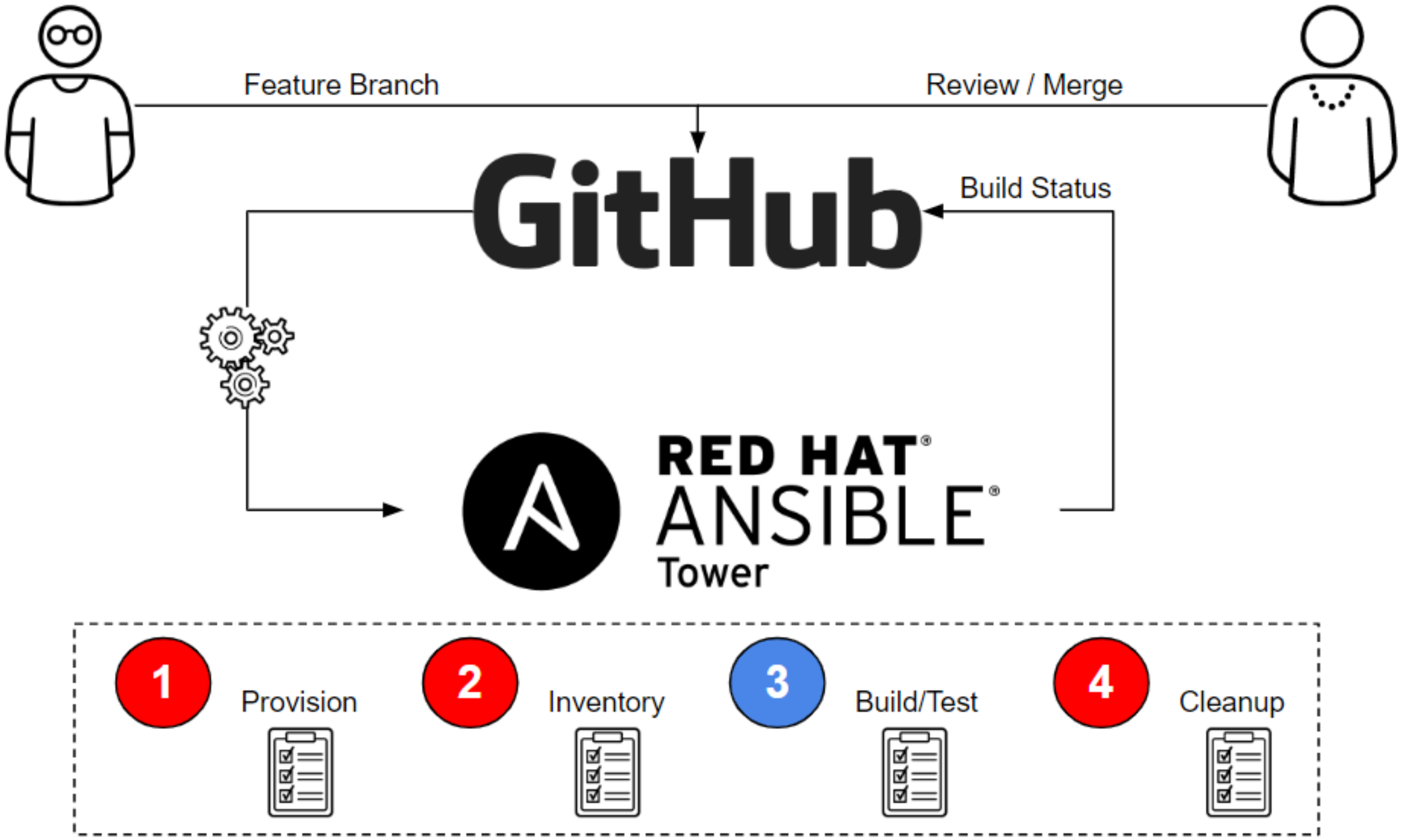
Use Case 5: Continous testing

- Requirements
  - Provision IBM i virtual machine for testing
  - Install extra dependencies to the virtual machine
  - Build latest code on newly created virtual machine
  - Kick off automated test cases
  - Clean up everything after testing has been done
    - Eg. Delete the virtual machine

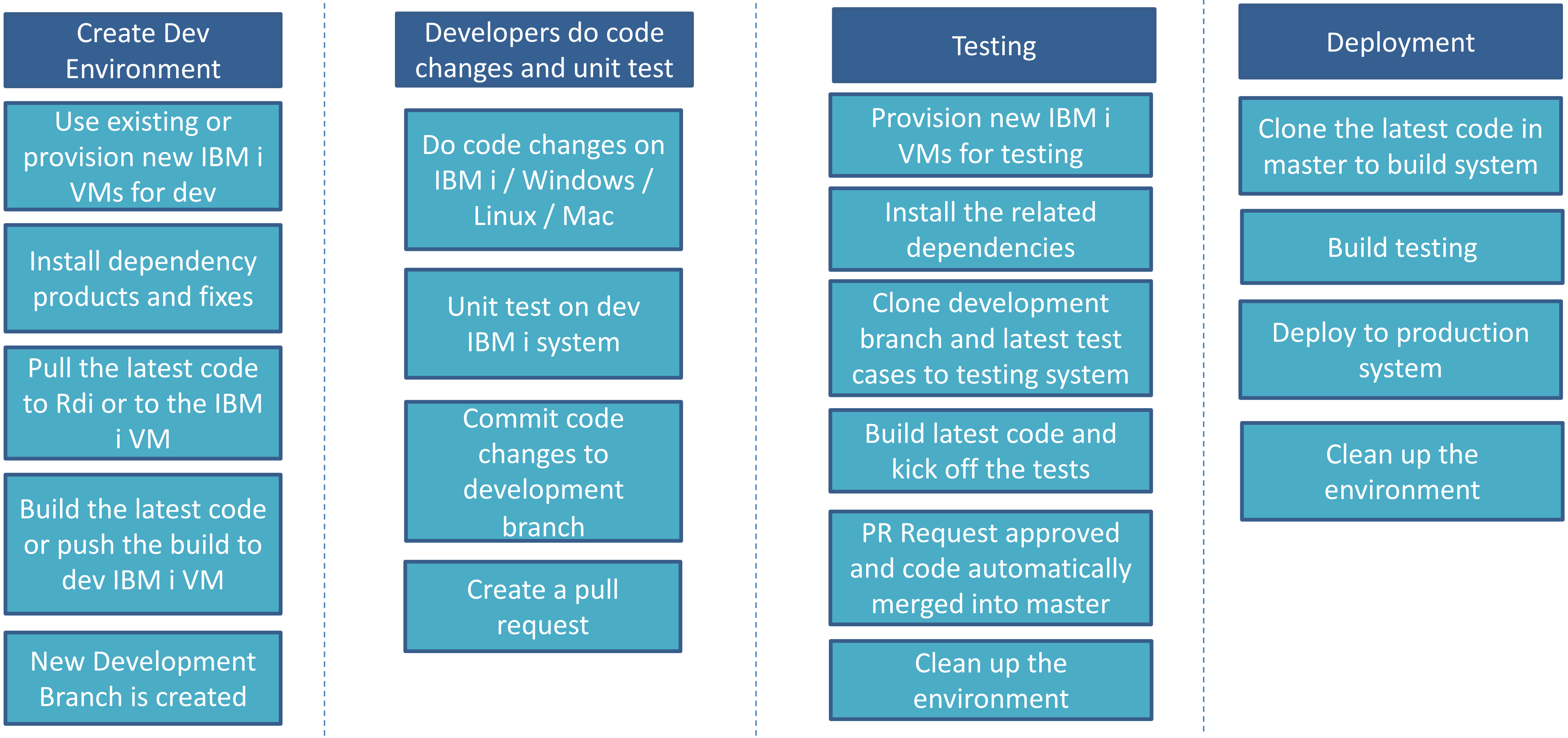
Use Case 5: Continous testing

- Requirements
  - Provision IBM i virtual machine for testing
    - add_host
  - Install extra dependencies to the virtual machine
    - ibmi_fix, ibmi_install_product_from_savf.py, ibmi_sync_files.py
  - Build latest code on newly created virtual machine
    - ibmi_script.py, git, git_config
  - Kick off automated test cases
    - ibmi_cl_command, ibmi_submit_job, ibmi_job, ibmi_query_job_log
  - Clean up everything after testing has been done
    - Eg. Delete the virtual machine
    - os_server (OpenStack module)

More complex: CI/CD with Ansible + GitHub



More complex: CI/CD with Ansible + GitHub



Create Dev Environment

- Use existing or provision new IBM i VMs for dev
- Install dependency products and fixes
- Pull the latest code to Rdi or to the IBM i VM
- Build the latest code or push the build to dev IBM i VM
- New Development Branch is created

Ansible Playbook Run

Developers do code changes and unit test

- Do code changes on IBM i / Windows / Linux / Mac
- Unit test on dev IBM i system
- Commit code changes to development branch
- Create a pull request

Developer Manual Operations

Testing

- Provision new IBM i VMs for testing
- Install the related dependencies
- Clone development branch and latest test cases to testing system
- Build latest code and kick off the tests
- PR Request approved and code automatically merged into master
- Clean up the environment

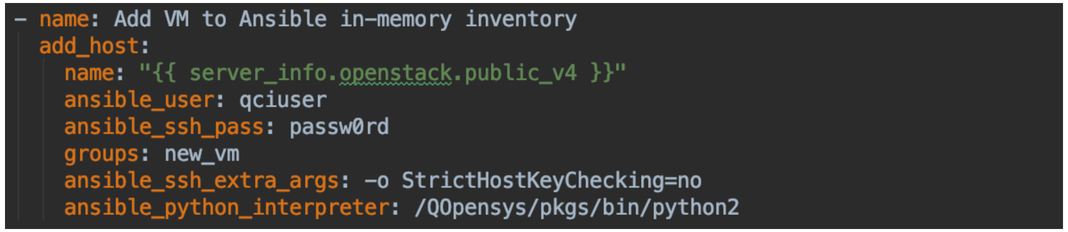
Ansible Playbook Run

Deployment

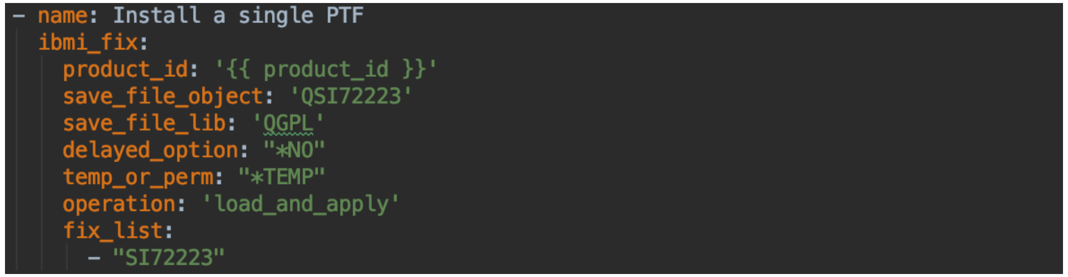
- Clone the latest code in master to build system
- Build testing
- Deploy to production system
- Clean up the environment

Ansible Playbook Run

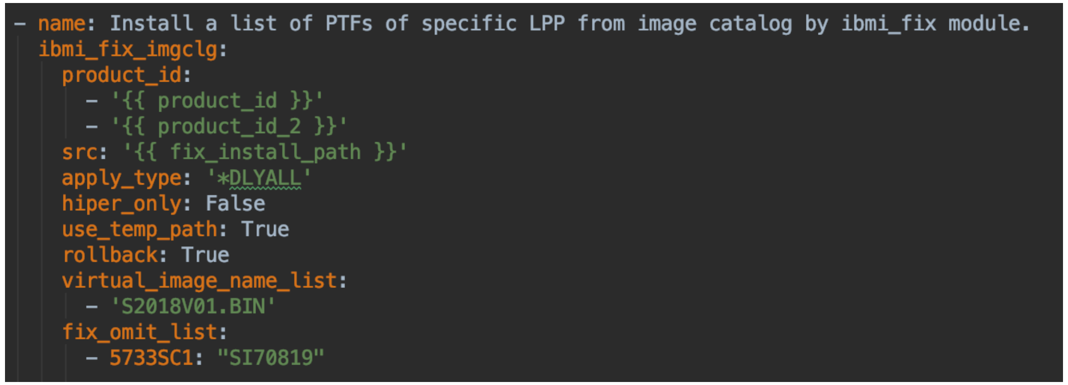
Some sample playbook fragments for creating dev environment



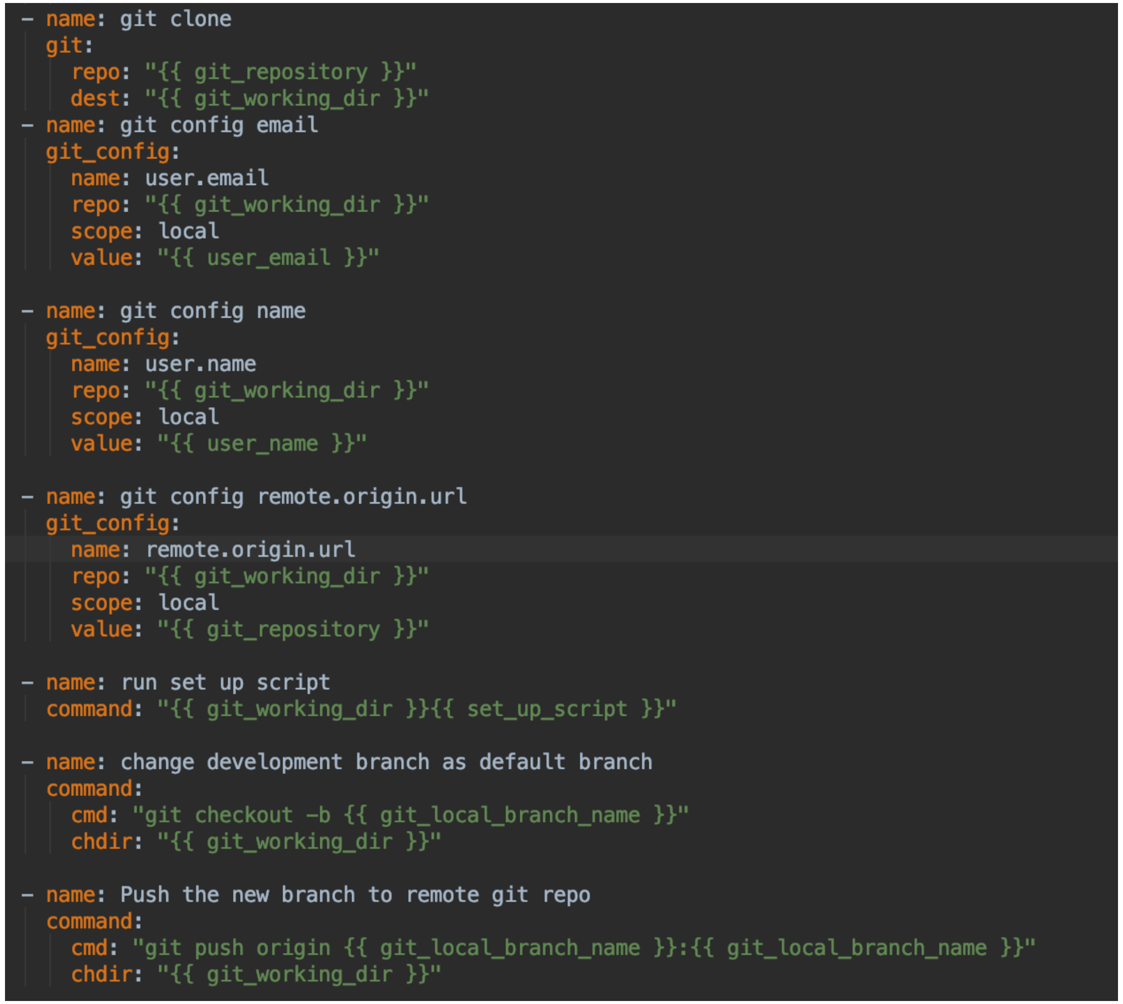
```yaml
- name: Add VM to Ansible in-memory inventory
  add_host:
    name: "{{ server_info.openstack.public_v4 }}"
    ansible_user: qciuser
    ansible_ssh_pass: passw0rd
    groups: new_vm
    ansible_ssh_extra_args: -o StrictHostKeyChecking=no
    ansible_python_interpreter: /QOpensys/pkgs/bin/python2
```



```yaml
- name: Install a single PTF
  ibmi_fix:
    product_id: '{{ product_id }}'
    save_file_object: 'QSI72223'
    save_file_lib: 'QGPL'
    delayed_option: "*NO"
    temp_or_perm: "*TEMP"
    operation: 'load_and_apply'
    fix_list:
      - "SI72223"
```



```yaml
- name: Install a list of PTFs of specific LPP from image catalog by ibmi_fix module.
  ibmi_fix_imgclg:
    product_id:
      - '{{ product_id }}'
      - '{{ product_id_2 }}'
    src: '{{ fix_install_path }}'
    apply_type: '*DLYALL'
    hiper_only: False
    use_temp_path: True
    rollback: True
    virtual_image_name_list:
      - 'S2018V01.BIN'
    fix_omit_list:
      - 5733SC1: "SI70819"
```



```yaml
- name: git clone
  git:
    repo: "{{ git_repository }}"
    dest: "{{ git_working_dir }}"
- name: git config email
  git_config:
    name: user.email
    repo: "{{ git_working_dir }}"
    scope: local
    value: "{{ user_email }}"

- name: git config name
  git_config:
    name: user.name
    repo: "{{ git_working_dir }}"
    scope: local
    value: "{{ user_name }}"

- name: git config remote.origin.url
  git_config:
    name: remote.origin.url
    repo: "{{ git_working_dir }}"
    scope: local
    value: "{{ git_repository }}"

- name: run set up script
  command: "{{ git_working_dir }}{{ set_up_script }}"

- name: change development branch as default branch
  command:
    cmd: "git checkout -b {{ git_local_branch_name }}"
    chdir: "{{ git_working_dir }}"

- name: Push the new branch to remote git repo
  command:
    cmd: "git push origin {{ git_local_branch_name }}:{{ git_local_branch_name }}"
    chdir: "{{ git_working_dir }}"
```

Summary

Ansible handles different automation tasks

- Configuration management
- Application deployment
- Continuous delivery
- Provisioning
- Orchestration
- Security automation

IBM i is supported as endpoint of Ansible

- Modules, plug-ins roles and sample playbooks are available
- Both interactive and automated tasks can be achieved for IBM i
- Key use cases are available for you to reference

Resources

- Ansible for IBM i Galaxy link: https://galaxy.ansible.com/ibm/power_ibmi
- Ansible for IBM i GitHub repo: https://github.com/IBM/ansible-for-i/
- Ansible for IBM i Documentation: https://ibm.github.io/ansible-for-i/index.html
- IBM i collections in Automation Hub: https://cloud.redhat.com/ansible/automation-hub/ibm/power_ibmi
- Ansible for IBM i Articles:
  - https://ibm.github.io/cloud-i-blog/archivers/2020_0602_automate_your_ibm_i_tasks_with_ansible
  - https://developer.ibm.com/tutorials/ansible-automation-for-power/
- Blog:
  - https://ibm.github.io/cloud-i-blog/
- Ansible documents:
  - https://docs.ansible.com/

Thanks very much!