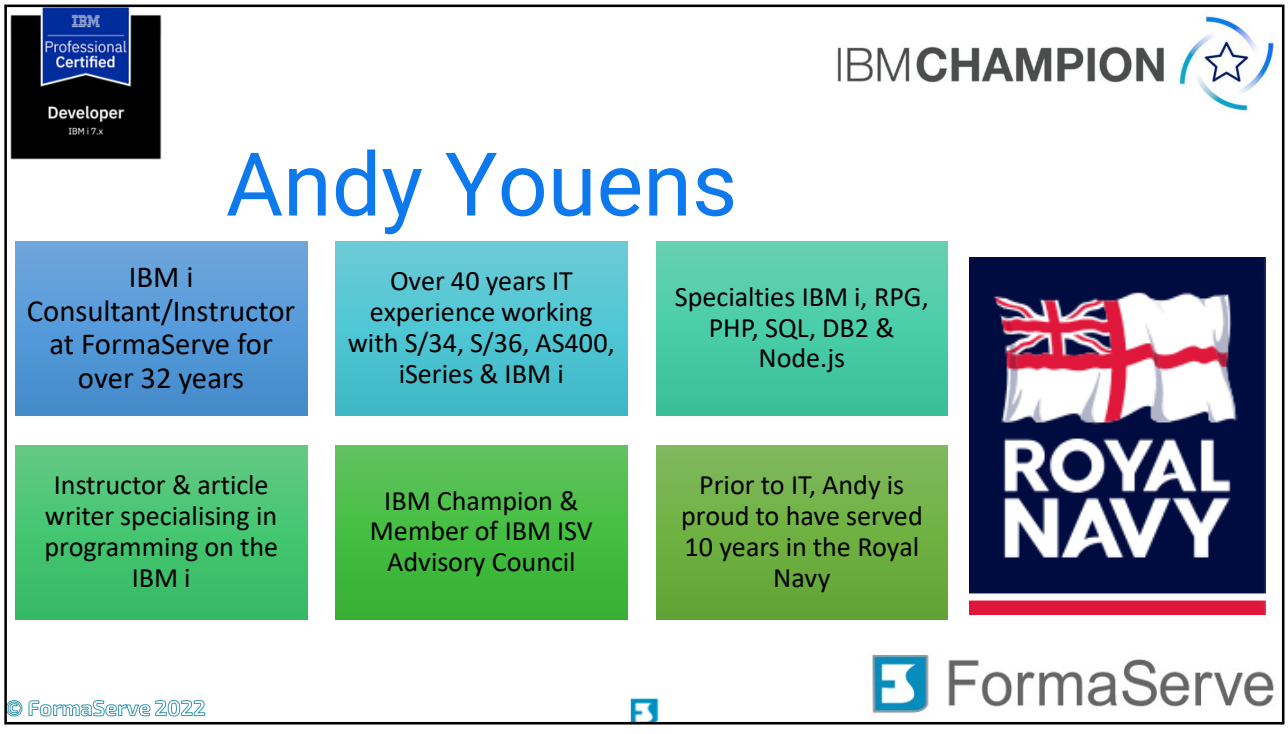

IBM i
FormaServe
IBM CHAMPION
Ansible & IBM i Workshop
Andy Youens

1


IBM Professional Certified Developer IBM i 7.x
IBM CHAMPION
Andy Youens
IBM i Consultant/Instructor at FormaServe for over 32 years
Over 40 years IT experience working with S/34, S/36, AS400, iSeries & IBM i
Specialties IBM i, RPG, PHP, SQL, DB2 & Node.js
Instructor & article writer specialising in programming on the IBM i
IBM Champion & Member of IBM ISV Advisory Council
Prior to IT, Andy is proud to have served 10 years in the Royal Navy
ROYAL NAVY
© FormaServe 2022
FormaServe

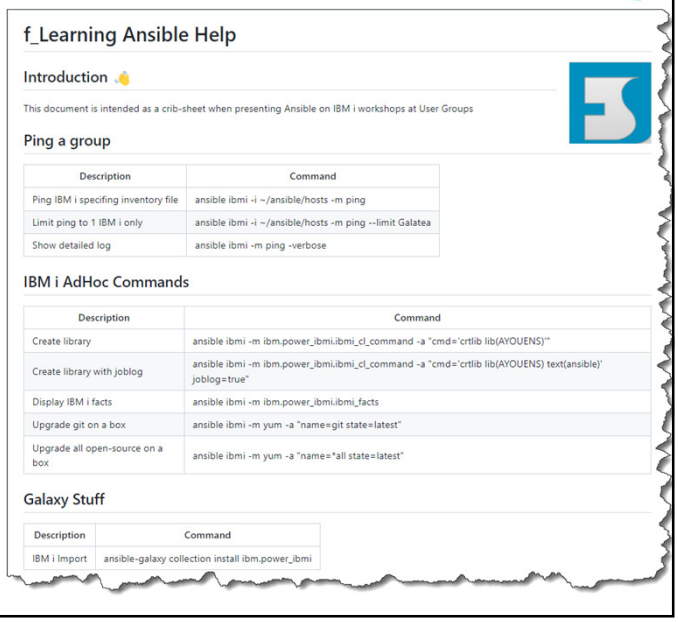
2

Ansible Workshop Agenda

- Part 1 – Andy!
  - Introduction
  - Installation
  - Configuration
  - Ad-Hoc Commands
  - YAML
  - Playbooks
  - Encryption
  - Top Tips
  - Demo
    - WSL will be the control node
    - WSL to monitor IBM i & Windows Laptop
- Part 2 – Everyone Else!
  - Monitor IBM i's
  - One IBM i will be the control node
  - Run Ad-hoc commands
  - Create/run playbooks
  - Work individually or a group
  - You decide!

GitHub Examples

- All examples from this workshop can be found in our GitHub repo
- https://github.com/FormaServe/f_Learning/tree/master/Ansible
- Presentation slides can be found in https://github.com/FormaServe/f_Learning/tree/master/pdf



f_Learning Ansible Help

Introduction

This document is intended as a crib-sheet when presenting Ansible on IBM i workshops at User Groups

Ping a group

Description	Command
Ping IBM i specifing inventory file	ansible ibmi -i ~/ansible/hosts -m ping
Limit ping to 1 IBM i only	ansible ibmi -i ~/ansible/hosts -m ping --limit Galatea
Show detailed log	ansible ibmi -m ping -verbose

IBM i AdHoc Commands

Description	Command
Create library	ansible ibmi -m ibm.power_ibmi.ibmi_cl_command -a "cmd='crtlib lib(AYOUENS)'"
Create library with joblog	ansible ibmi -m ibm.power_ibmi.ibmi_cl_command -a "cmd='crtlib lib(AYOUENS) text(ansible)' joblog=true"
Display IBM i facts	ansible ibmi -m ibm.power_ibmi.ibmi_facts
Upgrade git on a box	ansible ibmi -m yum -a "name=git state=latest"
Upgrade all open-source on a box	ansible ibmi -m yum -a "name=*all state=latest"

Galaxy Stuff

Description	Command
IBM i Import	ansible-galaxy collection install ibm.power_ibmi

4

Ansible Introduction

5

What is Ansible?

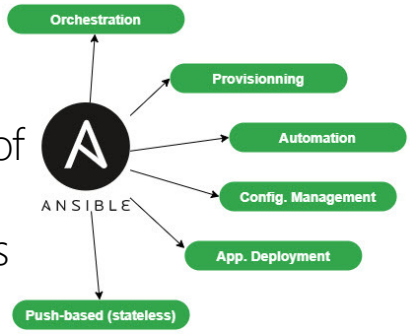
Red Hat Ansible Automation Platform

- First version 2012 from Ansible Works
- Acquired by Red Hat 2015
- A simple IT automation platform
- An open-source application from Red Hat*
- Written in Python & YAML based
- Highly flexible & configuration management of systems
- Configuration roll-back in case of error
- Report errors at https://github.com/IBM/ansible-for-i/

6

Ansible Advantages

- **Free -** It is open-source
- **Simple -** No problems in setting up & using
- **Powerful -** You can configure thousands of machines at a time
- **Flexible -** Supports multiple environments
- **Agentless -** Nothing to install on your hosts
- **Efficient -** It is light on your server



 www.FormaServe.co.uk

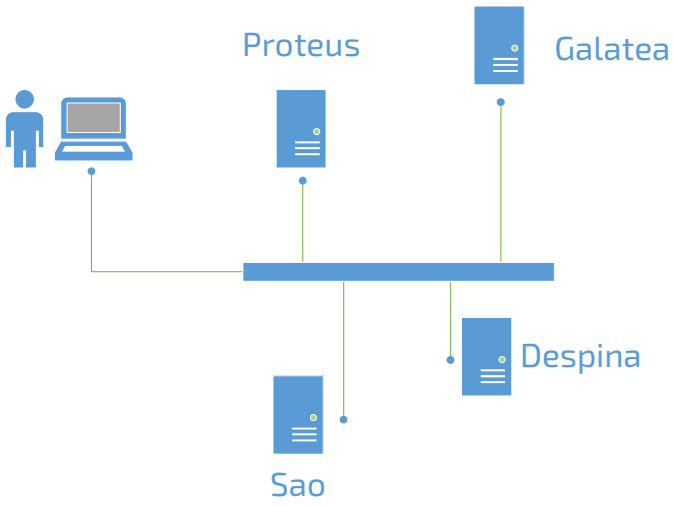
7

What can Ansible do for IBM i?

- Automate traditional IBM i administration tasks
- Improve application development process & efficiency to shorten software delivery cycle
- Manage multiple IBM i systems with interactive command lines
- Automate IBM i tasks with playbooks
- Allows operators with no IBM i skills to do basic management
- Ties the knot between open-systems & propriety systems

8

What can Ansible do for me?



FormaServe IBM i Network

 www.FormaServe.co.uk

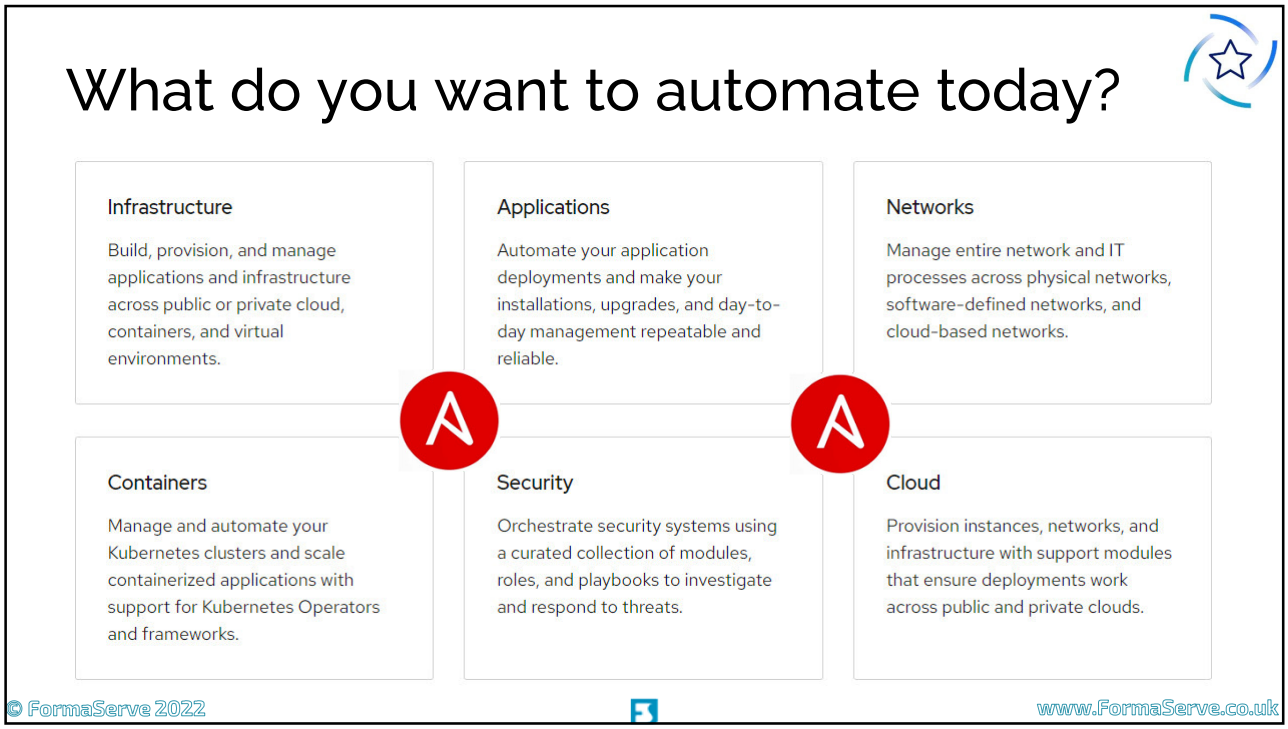
9

I Used Ansible to Setup this Workshop

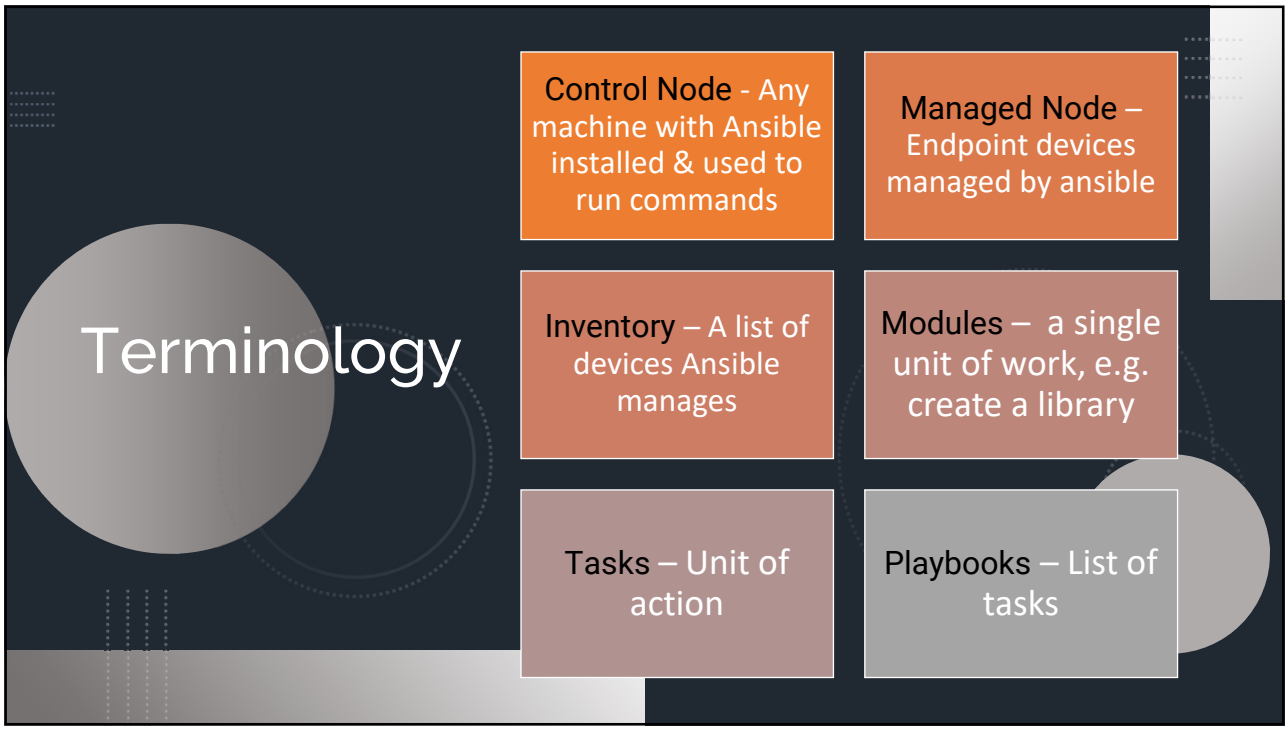
- On every partition
  - Set the default shell to be BASH
  - Installed Open Source packages need for this workshop
  - For every user
    - Create user profile with home directory
    - Create a library for user profile
    - Create home directory on IFS
    - Create sub-directory Ansible off users home directory
    - Create sub-directory .ssh off users home directory
    - In the .ssh directory create an authorized_keys file
    - Change authority on users home directory for full access, including sub-directories
    - Create user shell profile & setup OSS path & BASH cursor

 www.FormaServe.co.uk

10


What do you want to automate today?
Infrastructure
Build, provision, and manage applications and infrastructure across public or private cloud, containers, and virtual environments.
Applications
Automate your application deployments and make your installations, upgrades, and day-to-day management repeatable and reliable.
Networks
Manage entire network and IT processes across physical networks, software-defined networks, and cloud-based networks.
Containers
Manage and automate your Kubernetes clusters and scale containerized applications with support for Kubernetes Operators and frameworks.
Security
Orchestrate security systems using a curated collection of modules, roles, and playbooks to investigate and respond to threats.
Cloud
Provision instances, networks, and infrastructure with support modules that ensure deployments work across public and private clouds.
© FormaServe 2022
www.FormaServe.co.uk

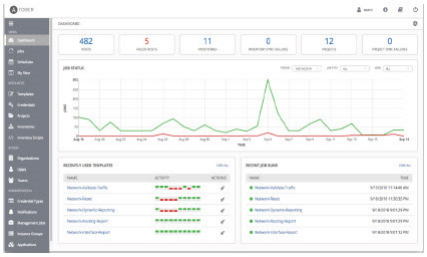
11


Terminology
Control Node - Any machine with Ansible installed & used to run commands
Managed Node – Endpoint devices managed by ansible
Inventory – A list of devices Ansible manages
Modules – a single unit of work, e.g. create a library
Tasks – Unit of action
Playbooks – List of tasks

12

Not Covered – Ansible Tower!

- Commercial offering
- Enterprise version of Ansible, helps companies & teams scale quickly & effectively
- There is a cost associated with adopting it, installing it, & getting the software up & running in your environments
- Everything else is Free!



RED HAT ANSIBLE Tower

 www.FormaServe.co.uk

13

Install Ansible

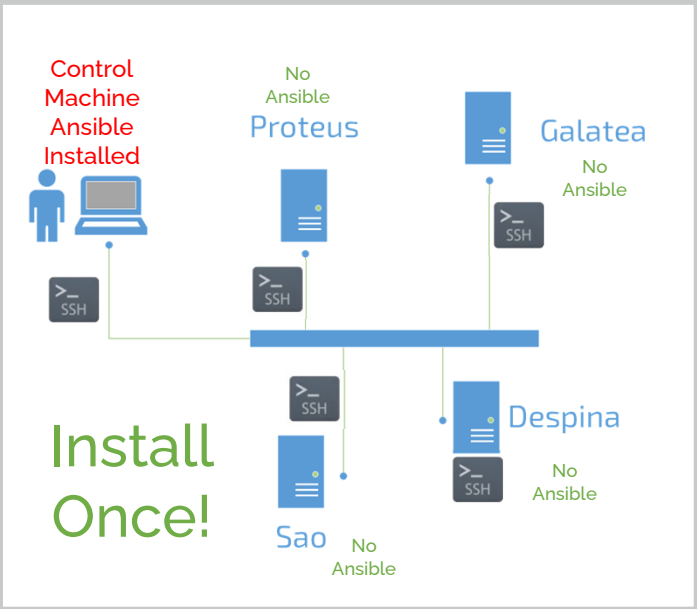
ANSIBLE

 www.FormaServe.co.uk

14

Ansible is Agentless

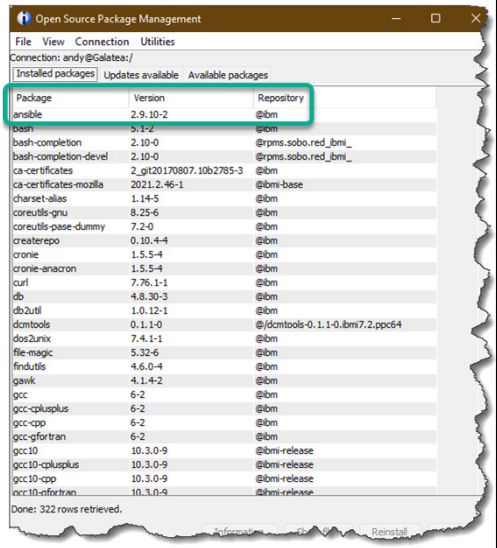
- One server with Ansible
- Ansible knows of 2 machine types
  - Control machine – Manages other machines
  - Remote machine – Handled by control machine



15

Installing Ansible

- IBM i for Control Node
  - ACS Open Source Package Management
  - YUM install ansible
- WSL
  - sudo apt install ansible
- Pre-req's on IBM i Nodes
  - IBM i V7.3 & >
  - 5733SC1 - PASE
  - 5770DG1 - HTTP
  - Python3
  - Python3-iToolkit
  - Python3-ibm_db



16

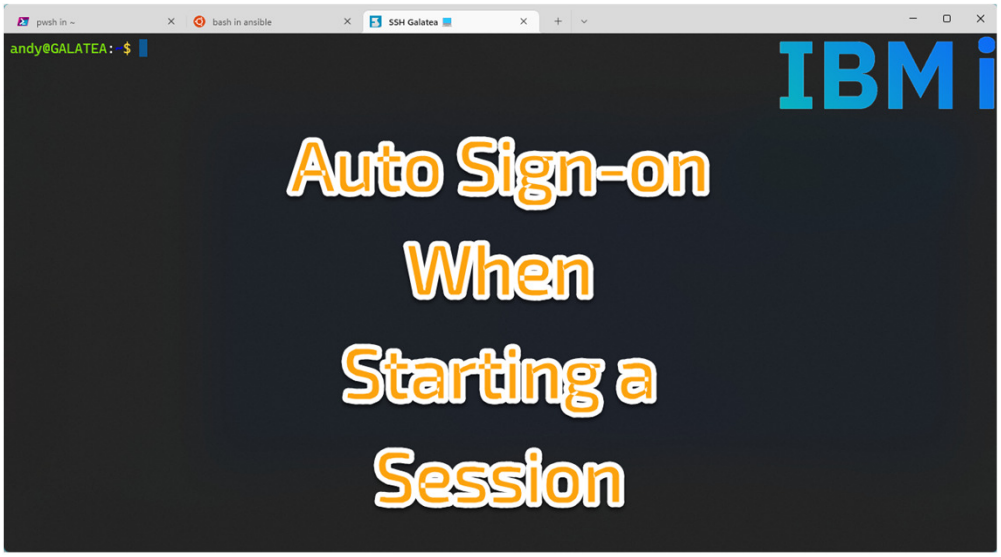
Communications Methods

- Linux/Unix
  - Uses SSH
- IBM i
  - Uses SSH
- Windows
  - Uses Powershell or SSH if configured

Communications to Hosts

- The Control node needs information to connect to nodes
  - The location of the inventory file, if not using default
  - The connection protocol to use - Default SSH
  - Whether non-standard ports are used (not 22)
  - What user it can login as
  - How Ansible should become root (sudo/QSECOFR)
  - Whether to prompt for a SSH password or key pair
- Lets make life easier for Ansible & setup SSH keys …

SSH Keys



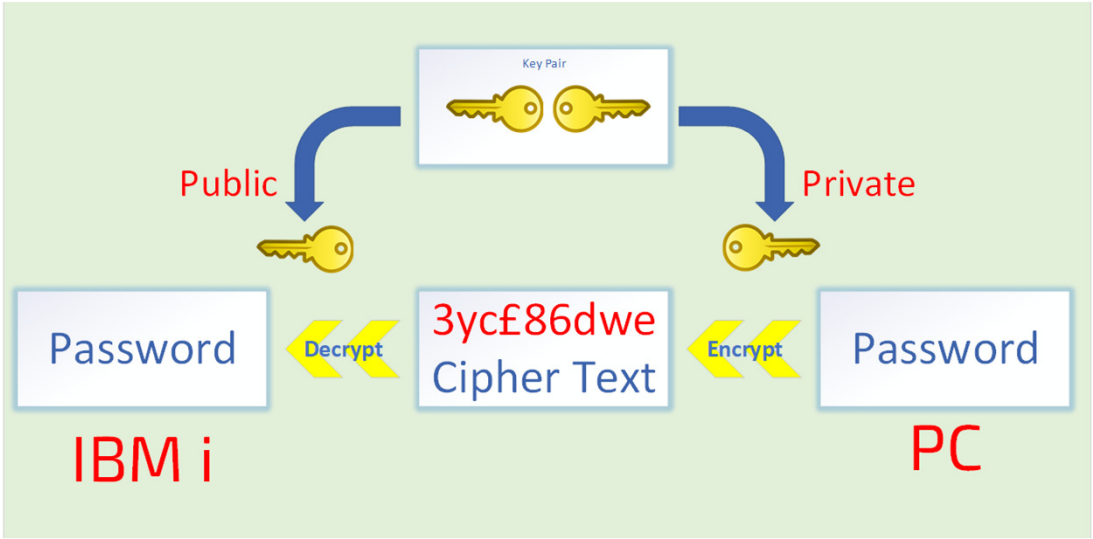
19

How Does it Work?

- We have a pair of keys, a private key & a public key
- Each key pair is unique, the two keys work together
- If you have the private key, you can prove you have it without showing what it is
- It's like proving you know a password without having to show someone the password

20

Key Authentication



21

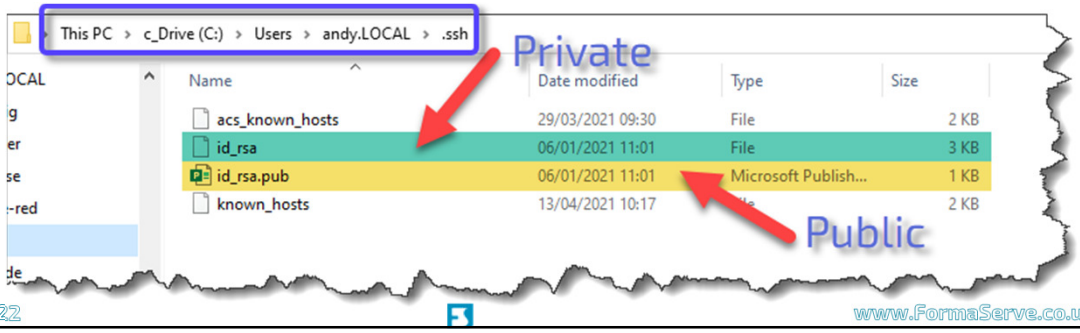
Generating Keys - Use ssh-keygen

- We can use …
  - Windows
  - Git for Windows
  - Putty

Windows 11

PuTTY

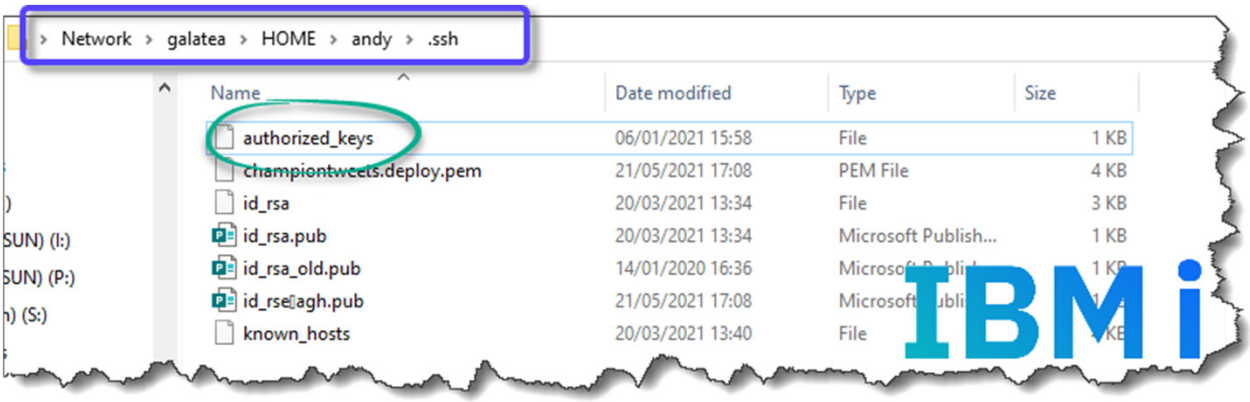
git



22

Send Public Key to IBM i

- Home .ssh Directory



23

SSH Keys Not Working?

- Home Directory
  - Are you the owner? - `ls -ld /home/Andy/`
    - chown andy /home/andy
  - Check permissions on home directory - `drwxr-sr-x`
    - chmod 755 /home/andy/
- .ssh Directory
  - Are you the owner of this directory? - `ls -ld /home/Andy/.ssh`
    - chown andy /home/andy/.ssh
  - Check authority for `-rw - drwx-S--3`
    - chgmod 700 /home/andy/.ssh/
- Authorized_keys
  - Check authority for `-rw`
    - chgmod 600 /home/andy/.ssh/authorized_keys

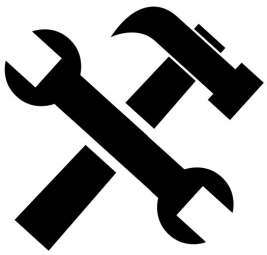
24

Configure Ansible

25

Configuring Ansible

- Two main configuration files
  - Inventory/host
  - Ansible configuration
- Configure communications



26

Inventory File

- Simplest form, a list of host names/IP addresses
- Can be named whatever you want!
- Can have any/no extension
- Common names
  - hosts/hosts.ini
  - inventory/inventory.ini
- Pick a name & stick with it!

```
10.0.0.36
10.0.0.39
10.0.0.37
10.0.0.38
81.142.39.107
10.0.2.3
10.0.0.2
norfolk
mercury
sun
jupiter
10.0.0.36
10.0.0.39
10.0.0.37
10.0.0.38
81.142.39.107
10.0.2.3
```


27

Inventory File - Groups

- Fit your hosts into groups for ease
- Three types of group
  - All - Every hosts
  - Named – Ones we configure!
  - Ungrouped - Contains every managed node that is not part of a group
- Localhost already present
- Do NOT give a group the same name as a host!

```
[adservers]
10.0.0.2

[windows]
norfolk
mercury

[ibmi]
10.0.0.36
10.0.0.39
10.0.0.37
10.0.0.38

[webservers]
81.142.39.107

[iot]
10.0.2.3
```


28

Inventory File - Groups

- A host can be a member of multiple groups
- This allows you to organise groups in different ways, for example …
  - Web servers or database servers
  - Production & testing servers
  - Development machines
  - IBM i
  - By countries/regions

```
[adservers]
10.0.0.2

[windows]
norfolk
mercury

[ibmi]
10.0.0.36
10.0.0.39
10.0.0.37
10.0.0.38

[webservers]
81.142.39.107

[iot]
10.0.2.3
```


29

Inventory File - Groups

- Groups can contain specific host information

What about security??

```
[ibmi]
10.0.0.36
proteus ansible_host=10.0.0.39 ansible_python_interpreter="/QOpenSys/pkgs/bin/python2"
despina ansible_ssh_user="JRiddle" ansible_ssh_pass="James"
sao
pub400 ansible_ssh_port=2222 ansible_ssh_user="AYouens" ansible_ssh_pass="NoIdea"

[ibmi:vars]
# ansible_python_interpreter="/QOpenSys/pkgs/bin/python3"
ansible_python_interpreter="/QOpenSys/pkgs/bin/python3.6"
ansible_ssh_common_args="-o StrictHostKeyChecking=no"
```


30

Inventory File - Listing

- Use to list current inventory
- ansible-inventory --list
- Great debugging tool
- Useful if getting unpredicted results

```
andy@Norfolk:~/ansible$ ansible-inventory --list
{
    "_meta": {
        "hostvars": {
            "10.0.0.36": {
                "ansible_python_interpreter": "/QOpenSys/pkgs/bin/python3.6",
                "ansible_ssh_common_args": "-o StrictHostKeyChecking=no"
            },
            "despina": {
                "ansible_python_interpreter": "/QOpenSys/pkgs/bin/python3.6",
                "ansible_ssh_common_args": "-o StrictHostKeyChecking=no",
                "ansible_ssh_pass": "James",
                "ansible_ssh_user": "JRiddle"
            },
            "proteus": {
                "ansible_host": "10.0.0.39",
                "ansible_python_interpreter": "/QOpenSys/pkgs/bin/python2",
                "ansible_ssh_common_args": "-o StrictHostKeyChecking=no"
            },
            "pub400": {
                "ansible_python_interpreter": "/QOpenSys/pkgs/bin/python3.6",
                "ansible_ssh_common_args": "-o StrictHostKeyChecking=no",
                "ansible_ssh_pass": "NoIdea",
                "ansible_ssh_port": 2222,
                "ansible_ssh_user": "AYouens"
            },
            "sao": {
                "ansible_python_interpreter": "/QOpenSys/pkgs/bin/python3.6",
                "ansible_ssh_common_args": "-o StrictHostKeyChecking=no"
            }
```


31

Ansible Configuration File

- During installation a sample config file at /etc/ansible/ansible.cfg
- It is recommended this file is copied to your default location
- A large file that contains all configuration values
- All values can be overwritten

```
# config file for ansible -- https://ansible.com/
# ===============================================

# nearly all parameters can be overridden in ansible-playbook
# or with command line flags. ansible will read ANSIBLE_CONFIG,
# ansible.cfg in the current working directory, .ansible.cfg in
# the home directory or /etc/ansible/ansible.cfg, whichever it
# finds first

# some basic default values...
[defaults]
# where is the inventory?
inventory      = ~/ansible/inventory

#library        = /usr/share/my_modules/
#module_utils   = /usr/share/my_module_utils/
#remote_tmp     = ~/.ansible/tmp
#lo             = ~/.ansible/t
```


32

Which Ansible Config File?

- Will use environment variable
  - ANSIBLE_CONFIG (export ANSIBLE_CONFIG='/ansible')
- Will search for a config file in the following order
  - ansible.cfg – Current working directory
  - ~/.ansible.cfg – dot file in users home directory
  - QOpenSys/etc/ansible/ansible.cfg – default configuration, if no other config file has been found
- ansible --version or ansible-config --version will identify which configuration file is being used

33

Default File Locations

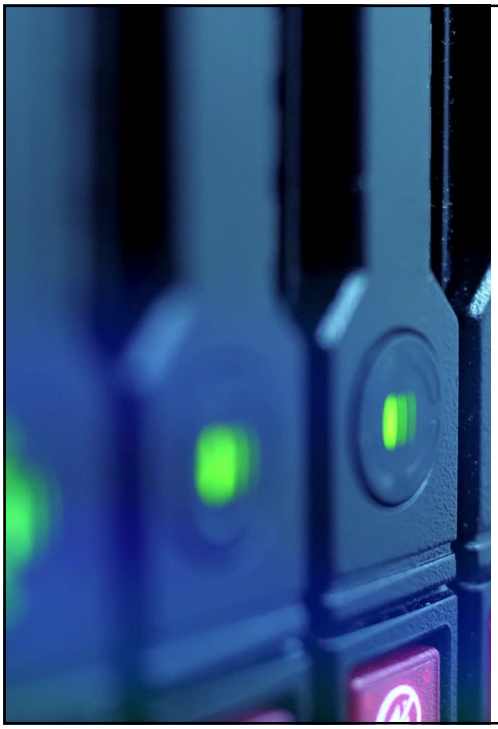
Name	Directory
Named Hosts or Inventory	(/QOpenSys)/etc/ansible/hosts
Configuration	(/QOpenSys)/etc/ansible/ansible.cfg

```
andy@Norfolk:~/ansible$ ansible --version
ansible 2.9.6
  config file = /home/andy/ansible/ansible.cfg
  configured module search path = ['/home/andy/.ansible/plugins/modules', '/usr/share/ansible/plugins/modules']
  ansible python module location = /usr/lib/python3/dist-packages/ansible
  executable location = /usr/bin/ansible
  python version = 3.8.10 (default, Mar 15 2022, 12:22:08) [GCC 9.4.0]
andy@Norfolk: /ansible$
```


34

Ansible Ad-hoc Commands

35



Ansible Modules

- Small programs that do all the actual work
- Has one specific task, e.g. creating a library
- Pushed to the target server from your Ansible controller
- They do the work, then get removed
- Modules are the backbone of Ansible
- Modules are being added to all the time
- Loads of core modules & IBM i specific

36

Core Modules

```yaml
- name: Copy file to new environment
  copy:
    src: /ansible/environment.txt
    dest: /formaserve/test/config.ini
    owner: QSECOFR
```

```yaml
- name: Install Open Source packages
  yum:
    name:
      - nginx
      - postgresql
      - postgresql-server
    state: present
```

```yaml
- name: Use git to clone Repo & archive
  git:
    repo: https://github.com/formaserve/f_learning.git
    dest: /node_apps/training
    archive: /tmp/training-examples.zip
```

```yaml
- name: System timestamp - date
  debug:
    msg:  "Current date: {{ cur_date }}"
```

```yaml
- name: Write some content in a file
  copy:
    dest: /home/formaserve/testing.txt
    content: |
      Good Morning
      Good Afternoon
```


37

Ansible Core Modules & More …

- Use ansible-doc module-name for full details
- Great for examples
- List modules
  - ansible-doc -l
- Full details
  - Ansible-doc ping
- To be selective
  - ansible-doc -l | grep ibm

```
azure_rm_aks_info                          Get Azure Kubernetes Service facts
azure_rm_aksversion_info                   Get available kubernetes versions supported by Azure ...
azure_rm_appgateway                        Manage Application Gateway instance
azure_rm_applicationsecuritygroup          Manage Azure Application Security Group
azure_rm_applicationsecuritygroup_info     Get Azure Application Security Group facts
azure_rm_appserviceplan                    Manage App Service Plan
azure_rm_appserviceplan_info               Get azure app service plan facts
azure_rm_automationaccount                 Manage Azure Automation account
azure_rm_automationaccount_info            Get Azure automation account facts
azure_rm_autoscale                         Manage Azure autoscale setting
azure_rm_autoscale_info                    Get Azure Auto Scale Setting facts
azure_rm_availabilityset                   Manage Azure Availability Set
azure_rm_availabilityset_info              Get Azure Availability Set facts
azure_rm_azurefirewall                     Manage Azure Firewall instance
azure_rm_azurefirewall_info                Get AzureFirewall info
azure_rm_batchaccount                      Manages a Batch Account on Azure
azure_rm_cdnendpoint                       Manage a Azure CDN endpoint
azure_rm_cdnendpoint_info                  Get Azure CDN endpoint facts
azure_rm_cdnprofile                        Manage a Azure CDN profile
azure_rm_cdnprofile_info                   Get Azure CDN profile facts
azure_rm_containerinstance                 Manage Azure Container
azure_rm_containerinstance_info            Get Azure Container Instance facts
azure_rm_containerregistry                 Manage an Azure Container Registry
azure_rm_containerregistry_info            Get Azure Container Registry facts
azure_rm_cosmosdbaccount                   Manage Azure Database Account instance
azure_rm_cosmosdbaccount_info              Get Azure Cosmos DB Account facts
```

ansible-doc -l | grep azure

38

Ansible Command Arguments

Argument	Description	Example
-m	Specify module to run	-m ping
-i	Specify your inventory file	-i hosts.ini
-e	Specify extra variables (Overrides)	-e "key=123"
-v	Verbose mode (Joblog)	
-vvv	Show connection details	
-u	Specify remote user	-u QSECOFR
-h	Show help	
-b	Become a user	-b ANDY
-a	Module attributes	-m yum -a 'name=git state=latest'

39

Pinging IBMi Servers

- Ping - A great starting point!
- Doesn't run TCP/IP Ping

```
andy  ansible  83.101s   ansible ibmi -m ping
galatea | SUCCESS => {
    "changed": false,
    "ping": "pong"
}
proteus | SUCCESS => {
    "changed": false,
    "ping": "pong"
}
```


40

Create Library IBMi Servers

- Log is colour coded to reflect results

```
andy@Norfolk:~/ansible$ ansible ibmi -m command -a "system 'crtlib lib(AYOUENS) text(ansible)'"
10.0.0.36 | CHANGED | rc=0 >>
CPC2102: Library AYOUENS created.
andy@Norfolk:~/ansible$ ansible ibmi -m command -a "system 'crtlib lib(AYOUENS) text(ansible)'"
10.0.0.36 | FAILED | rc=255 >>
CPF2111: Library AYOUENS already exists.non-zero return code
andy@Norfolk:~/ansible$
```


41

Colour Coding Commands

```
TASK [Create User Profile] ***********************
changed: [10.0.0.36] => (item=AJESSICAJ)
changed: [10.0.0.36] => (item=AEMILYY)
changed: [10.0.0.36] => (item=ASAMMYJ)
changed: [10.0.0.36] => (item=ACHLOEY)
changed: [10.0.0.36] => (item=AELLIEJ)

TASK [Create user library] ***********************
ok: [10.0.0.36] => (item=AJESSICAJ)
ok: [10.0.0.36] => (item=AEMILYY)
ok: [10.0.0.36] => (item=ASAMMYJ)
ok: [10.0.0.36] => (item=ACHLOEY)
ok: [10.0.0.36] => (item=AELLIEJ)
```

```
r Profile] ***********************************************
.36] (item=AJESSICAJ) => {"ansible_loop_var": "item", "changed
AJESSICAJ already exists", "rc": 256}
.36] (item=AEMILYY) => {"ansible_loop_var": "item", "changed":
ILYY already exists", "rc": 256}
failed: [10.0.0.36] (item=ASAMMYJ) => {"ansible_loop_var": "item", "changed":
ser profile ASAMMYJ already exists", "rc": 256}
failed: [10.0.0.36] (item=ACHLOEY) => {"ansible_loop_var": "item", "changed":
ser profile ACHLOEY already exists", "rc": 256}
failed: [10.0.0.36] (item=AELLIEJ) => {"ansible_loop_var": "item", "changed":
ser profile AELLIEJ already exists", "rc": 256}
```


42

YAML in 5 Minutes!

43

Basic YAML

- Easy to read
- Similar to XML & JSON
- yaml or yml file extension
- Also used for Docker, Kubernetes etc
- Commonly used for config files
- Indentation used for structure, using spaces
- Tabs are not allowed
- Use online linter to check syntax
  - https://www.yamllint.com/
  - https://yamlchecker.com/

44

Basic YAML

- Ansible uses YAML
- YAML optionally starts with ---
- Most commonly, uses simple key-value pair to represent data
- Represented in key: value pair

There should be a space between : & value

```yaml
# key-pairs syntax
app: end-of-day
port: 8080
version: 1.9.9
OS: 'IBM i 7.3'
```

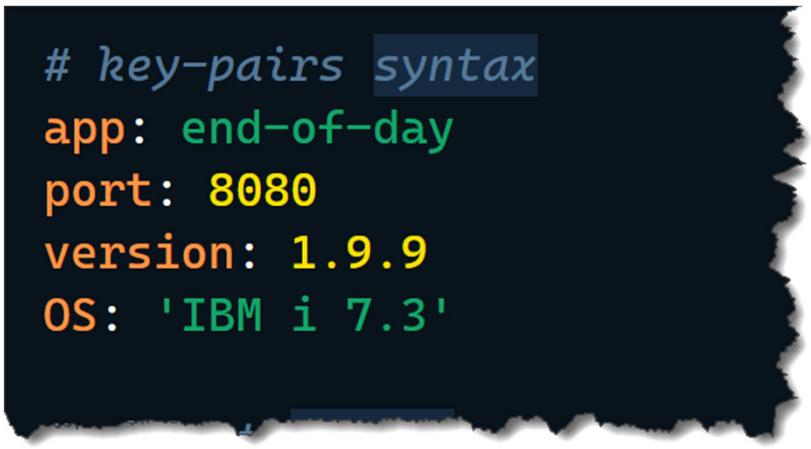

45

Basic YAML - Comments

```yaml
# Produced for iUG Workshop
# Copyright © FormaServe Systems Ltd 2022
# April 2022

---

# YAML Examples

# key-pairs syntax
app: end-of-day
port: 8080
```

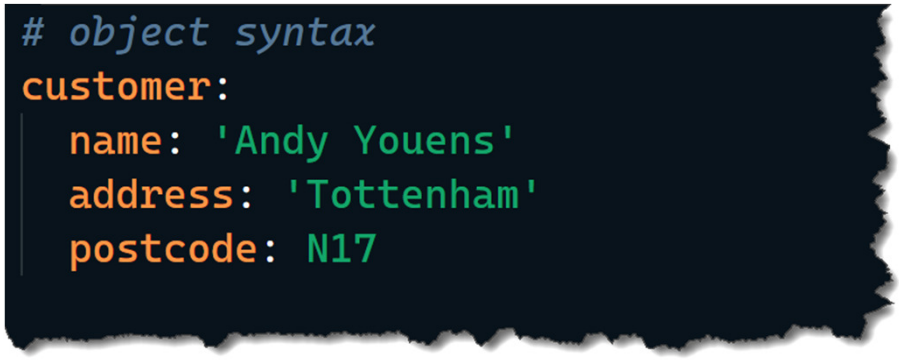

46

Basic YAML – Key Pairs



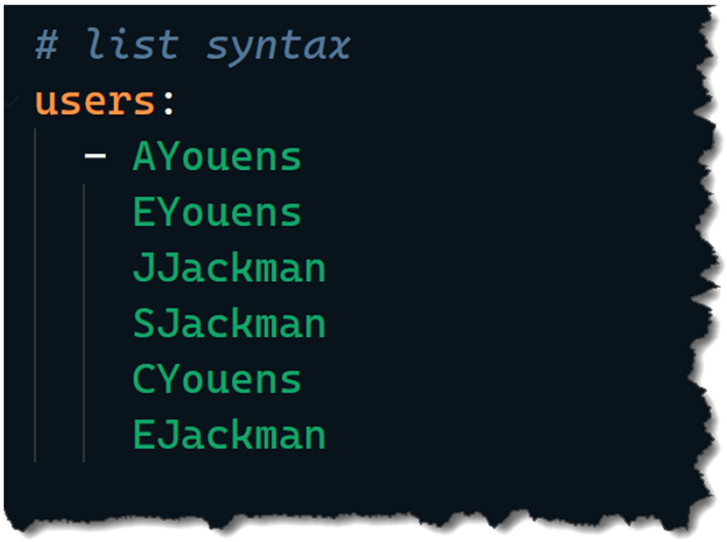
47

Basic YAML – Objects



48

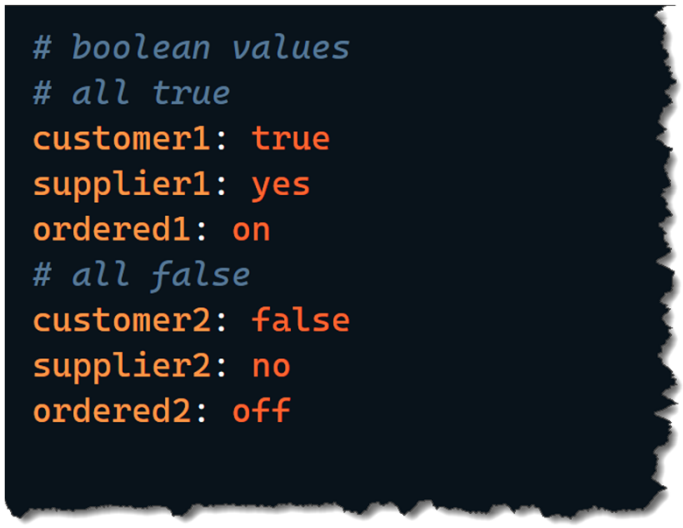
Basic YAML – Lists



```yaml
# list syntax
users:
  - AYouens
    EYouens
    JJackman
    SJackman
    CYouens
    EJackman
```


49

Basic YAML – Boolean



```yaml
# boolean values
# all true
customer1: true
supplier1: yes
ordered1: on
# all false
customer2: false
supplier2: no
ordered2: off
```


50

Multi & Single Lines in YAML

```yaml
# multi line string
multiline: |
  This is a multi line string
  that spans multiple lines
  and is indented.

# single line string
singleline: >
  This is a single line string
  that spans multiple lines
  and is indented.
```


51

Working Example of YAML

```yaml
---
apiVersion: v1
kind: pod
metadata:
  name: pod-yaml2
  labels:
    app: yaml2
spec:
  containers:
  - name: pod-yaml2
    image: busybox
    ports:
    - containerPort: 8080
      protocol: TCP
      volumeMounts:
      - mountPath: /tmp/yaml2
        name: yaml2
```

- Key-value pairs
- metadata is an object
- labels is an object
- spec is an object
- containers is a list of objects
- ports is a list
- volumeMounts is a list of objects

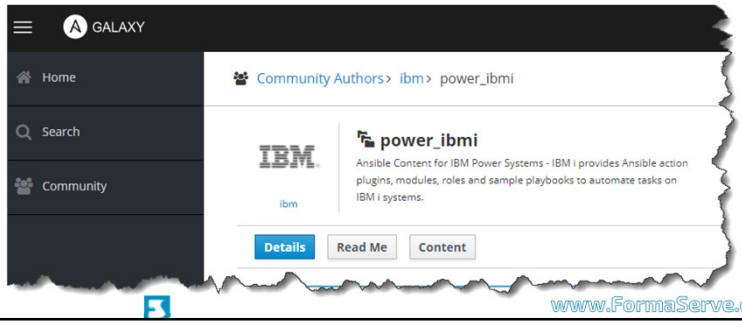
52

Ansible Galaxy

53

Ansible Galaxy

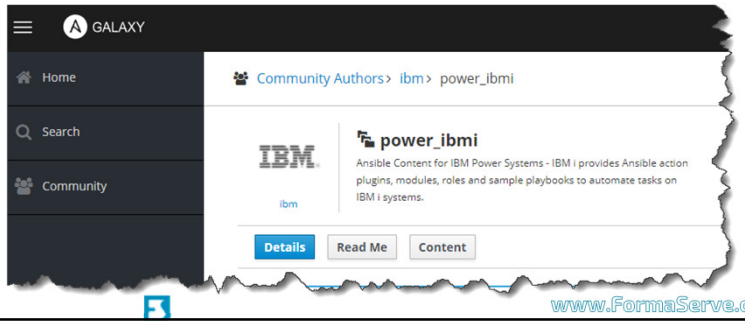
- Galaxy is a role & collection manager
- Compare it with NPM (Node Package Manager)
- Used for sharing collections
- Installed using collection package management
- https://galaxy.ansible.com/ibm/power_ibmi



54

Ansible Galaxy

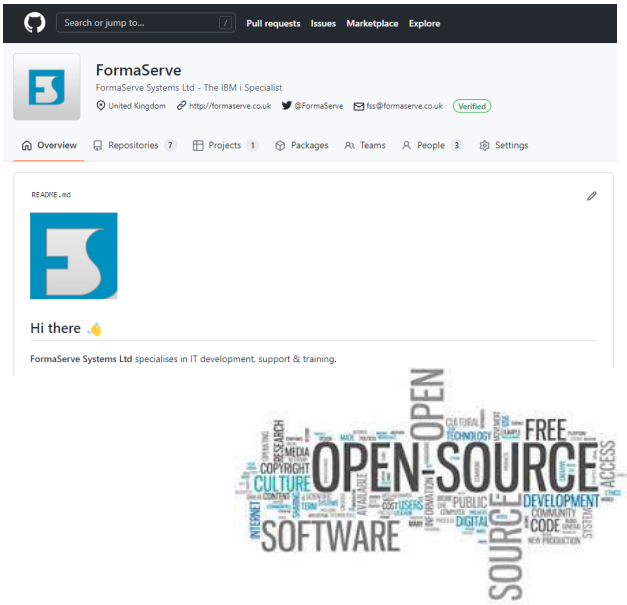
- To install Galaxy IBM i collection run
  - ansible-galaxy collection install ibm.power_ibmi
- To update use the same command as above with a -f flag to force an update
- Current version 1.7.1 as of June 2022



55

Ansible Galaxy

- If you want your Playbook to be shared with the world, why not upload it to Ansible Galaxy
- Join in with Open-Source!



56

Ansible Playbooks

57

Writing Playbooks

Ad–hoc Command

ansible	ibmi	-m yum	-a “name=Nodejs18 state=latest”

Consider ad-hoc commands
as CL commands
&
Playbook as a CL Program

Playbook

```
---
- name: playbook name
  hosts: ibmi
  tasks:
    - name: installing node
      yum:
        name: nodejs18
        state: latest
```


58

Writing Playbooks

- A playbook can have multiple play tasks designed to execute against a set of hosts
- Every playbook must contain
  - A list of hosts to run/configure against
  - A list of tasks to be performed on those hosts
- The rest are optional

59

Writing Playbooks

- The main way to automate tasks in Ansible
- A YAML based text file listing commands to run in a defined order
- Playbooks can change long, detailed admin tasks into a very easily routine with predicted & successful outcomes
- Has .yml extension
- Indentation with a 2 space character - Not TAB!
  - Data elements at the same level must have the same indentation
  - Items that are children of another item must be indented more than their parents

60

A Simple Playbook

1. Play name - Good doc's!
2. Which hosts to run on
3. Where the collection is stored
4. Name of the task
5. Which module to run
6. The parameters of the module (Create a library)
7. Display joblog if fails

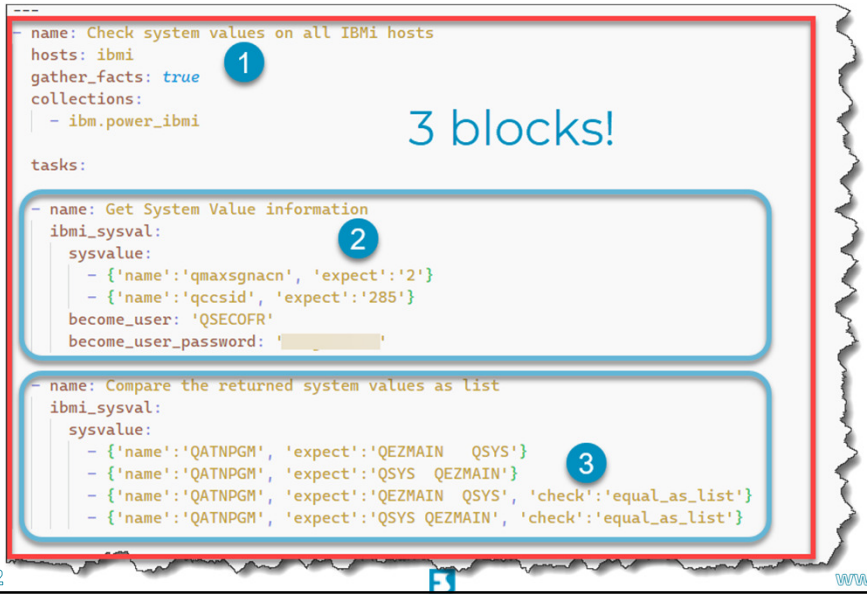
61

Playbook Example

1. Use Stat module to check if git installed
   Put result into git_status
2. If git_status returns not exists install git
3. If git_status returns found, update git

62

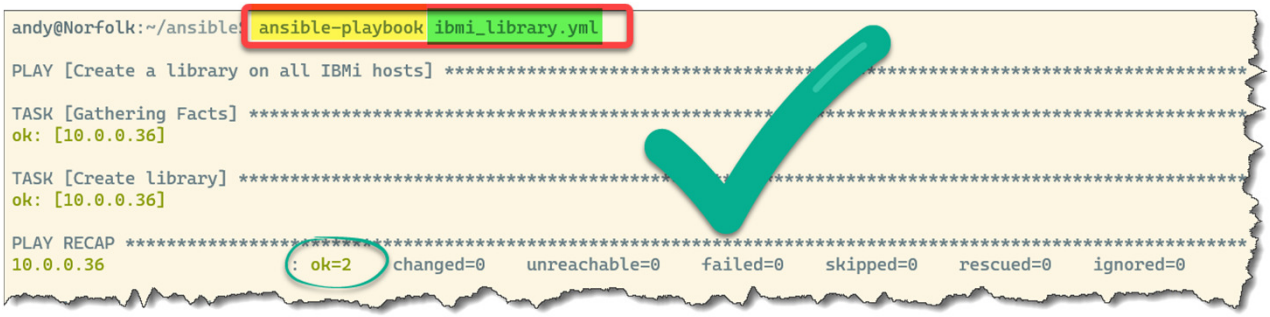
Playbook Example



63

Running Playbook

- Use the ansible-playbook command
- Pass the name of the playbook as an argument/parameter



64

Variables in Playbooks

- Store values for reuse throughout your playbook
- Simplifies the creation & maintenance of a playbook
- Reduces the number of errors
- Provide a convenient way to manage dynamic values
- Examples
  - TCP servers to stop/start
  - Licensed programs to install
  - User profile details

 www.FormaServe.co.uk

65

Variables in Playbooks

- Variable names
  - Must start with a letter
  - Can only contain
    - Letters
    - Numbers
    - Underscore _
- Variable in playbooks are very similar to using variables in any programming language, CL, RPG, PHP etc.

 www.FormaServe.co.uk

66

Variables in Playbooks

- We can use variables in three forms
  1. A single line variable declaration like we do in any programming language
     - Use vars to define inline variables within the playbook
  2. A separate file with variables & values like a properties file
     - A key pair (name: 'Andy Youens')
     - Use vars_files to import files with variables
  3. Use include_vars module for including variables from a set of files
     - Used at runtime for a task

67

Defining Variables

- In a vars block at the beginning of a playbook

```
---
- name: Create a user on all IBMi hosts
  hosts: ibmi
  vars:
    user_profile: JRIDDLE
    user_state: present
  gather_facts: true
  collections:
    - ibm.power_ibmi
```


68

Defining Variables

- In a vars_files section

```
- name: Create a user on all IBMi hosts
  hosts: ibmi
  vars_files:
    vars/user_profile.yml
  gather_facts: true
  collections:
```

user_profile.yml

```
1  user_profile: JRIDDLE
2  user_state: present
3
4
```

vars_file.yml

69

Defining Variables

- In a include_vars
- Is only activated within this step of the playbook
- Does not have to exist to run playbook, but expects it to be there when reaching that stage
- Could have been created in a previous step

```
- name: Check all ok!
  stat:
    path: /QOpensys/pkgs/bin/git
  register: git_stat
  include_vars: vars/task2.yml
```


70

Defining Variables

- Example of all 3 methods we have discussed
- Scoping
  - Who Wins?

```yaml
- hosts: ibmi
  name: Create new file on the IFS
  vars:                                  # 1
    company: FormaServe
  vars_files:                            # 2
    - vars/user_profile.yml
  collections:
    - ibm.power_ibmi

  tasks:
  - name: Create new file in test
    copy:
      dest: "vars/git.yml"
      content: |
        line git init
        line git add .
        line git commit -m Initial

  - name: Check all ok!
    stat:
      path: /QOpensys/pkgs/bin/git
    register: git_stat
    include_vars: vars/git.yml           # 3
```


71

Referencing Variables

- After declaring variables, you can use them in tasks
- Reference a variable by placing its name in double braces
  - {{ variable_name }}
- Ansible will substitute the variable, with its value at runtime

```yaml
---
- name: Create a user library on all IBMi hosts
  hosts: ibmi
  vars:
    user_profile: JRIDDLE
  gather_facts: true
  collections:
    - ibm.power_ibmi

  tasks:
  - name: Create library for {{ user_profile }}
    ibmi_cl_command:
      cmd: crtlib lib({{ user_profile }})
      joblog: true
    register: crt_lib_result
```

```
andy@Norfolk:~/ansible ansible-playbook ibmi_vars.yml

PLAY [Create a user library on all IBMi hosts] *********************************

TASK [Gathering Facts] *********************************************************
ok: [10.0.0.36]

TASK [Create library for JRIDDLE] **********************************************
ok: [10.0.0.36]

PLAY RECAP *********************************************************************
10.0.0.36                  : ok=2    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
```


72

Loops in Playbook - Example

```
tasks:
  name: Create new user
  ibmi_user_and_group:
    operation: 'create'
    user: 'AEmilyY'
    become_user: 'QSECOFR'
    become_user_password: '[redacted]'
    text: 'Created using Ansible Playbook'
    user_class: '*SECOFR'

  name: Create anuvver new user
  ibmi_user_and_group:
    operation: 'create'
    user: 'AJessicaJ'
    become_user: 'QSECOFR'
    become_user_password: '[redacted]'
    text: 'Created using Ansible Playbook'
    user_class: '*SECOFR'
```

Is there a better way?

73

Loops in Playbook - Example

```
- name: Create a new user on all IBMi hosts
  hosts: ibmi
  vars:
    user_profile:
      - AJRiddle
      - AEmilyY
      - AJessicaJ
  collections:
    - ibm.power_ibmi

  tasks:
  - name: Create new users
    ibmi_user_and_group:
      operation: 'create'
      user: "{{ item }}"
      password: ABC123
      user_class: '*SECOFR'
      become_user: 'QSECOFR'
      become_user_password: '[redacted]'
      text: 'Created using Ansible Playbook'
    loop: "{{ user_profile }}"
```

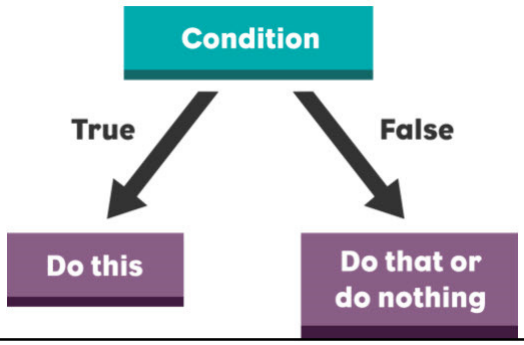
1. A list of users in variable user_profile
2. Use a loop to step thro'
3. Use each item to create a new user

```
PLAY [Create a new user on all IBMi hosts] *************************

TASK [Gathering Facts] *********************************************
ok: [10.0.0.36]

TASK [Create new users] ********************************************
changed: [10.0.0.36] => (item=AJRiddle)
changed: [10.0.0.36] => (item=AEmilyY)
changed: [10.0.0.36] => (item=AJessicaJ)

PLAY RECAP *********************************************************
10.0.0.36                  : ok=2    changed=1    unreachable=0    f
0
```


74

Run Task Conditionally

- We an use conditionals to run, or not run, tasks when conditions dictate at run-time
- Both variables & facts can be examined by conditions
- A When statement is used to run a task conditionally



75

Run Task Conditionally - Operators

Operation	Symbol
Equal	==
Less than	<
Greater than	>
Less than or equal	<=
Greater than or equal	>=
Not equal	!
Variable exists	is defined
Variable does not exist	Is not defined

76

Run Task Conditionally

- Register
  - Will capture the output of a command or task
  - They only hold values while playbook is running
  - The output is saved into a temporary variable that can be used
    - For debugging purposes
    - To check the status of something
  - Great for checking the status of a previous command
  - Beware - each command will return a different status!

Register Statement - Example

1. I'm using STAT to see if directory exists
2. Store the return from STAT into git_stat
3. If STAT return doesn't exists, then omit this task
4. If STAT return was exists, then omit this task

```
tasks:
- name: Verify git has been installed
  stat:
    path: /QOpensys/pkgs/bin/git
  register: git_stat

- name: Install git if it is not there
  command: /QOpensys/pkgs/bin/yum install git -y
  when: not git_stat.stat.exists

- name: Update git if installed
  command: /QOpensys/pkgs/bin/yum update git -y
  when: git_stat.stat.exists
```

Error Handling

- Ansible **stops** executing tasks on a host when a task fails on that host
- Use **ignore_errors** to continue, ignoring the problem

```
---
- name: Install basic packages
  hosts: ibmi
  tasks:
    - name: install open-source
      yum:
        name: andy
        state: present
      ignore_errors: true

    - name: install next
      yum:
        name: jane
        state: present
```

```
TASK [Gathering Facts] *********************************************************
ok: [galatea]

TASK [install open-source] *****************************************************
fatal: [galatea]: FAILED! => {"ansible_facts": {"pkg_mgr": "unknown"}, "changed
...ignoring

TASK [install next] ************************************************************
fatal: [galatea]: FAILED! => {"ansible_facts": {"pkg_mgr": "unknown"}, "changed

PLAY RECAP *********************************************************************
galatea                    : ok=2    changed=0    unreachable=0    failed=1
```


79

Error Handling - Use -vvv Parameter

```
tasks:
- name: Create library
  ibmi_cl_command:
    cmd: crtlib lib(JYOUENS)
    joblog: true
```

```
"job_log": [
    {
        "FROM_INSTRUCTION": "0D10",
        "FROM_LIBRARY": "QSYS",
        "FROM_MODULE": null,
        "FROM_PROCEDURE": null,
        "FROM_PROGRAM": "QLICRLIB",
        "FROM_USER": "ANDY",
        "MESSAGE_FILE": "QCPFMSG",
        "MESSAGE_ID": "CPF2111",
        "MESSAGE_LIBRARY": "QSYS",
        "MESSAGE_SECOND_LEVEL_TEXT": "&N Recovery  . . . :   Before creating
IB command). &P -- Use DSPLIB ASPDEV(*ALLAVL) to search for the library. If the
e library may be only partially created. To remove the partially created librar
lists in a primary or secondary ASP, vary off then vary on the ASP device.",
        "MESSAGE_SUBTYPE": "EXCEPTION HANDLED",
        "MESSAGE_TEXT": "Library JYOUENS already exists.",
        "MESSAGE_TIMESTAMP": "2022-05-03T11:36:30.116500",
        "MESSAGE_TYPE": "ESCAPE",
        "ORDINAL_POSITION": 7,
        "SEVERITY": 40,
        "TO_INSTRUCTION": "5841",
```

-vvv parameter

80

Sensitive Data

81

Sensitive Data

- No problem, Ansible Vault to the rescue
- Ansibles security module
- Used for
  - Passwords
  - API keys
  - Other secrets …
- Is used to manage these security details
- A crucial tool when it comes to data security & encrypting confidential information

82

Sensitive Data

- ansible-vault commands
  - Create new files
    - ansible-vault create filename
  - View
    - ansible-vault view filename
  - Edit
    - ansible-vault edit filename
  - Encrypt existing file
    - Ansible-vault encrypt filename
  - Decrypt a file
    - Ansible-vault decrypt filename

83

Sensitive Data - Examples

- Change the password of an encrypted file
  - ansible-vault rekey filename
- Prompts for current password
- Then prompts for new password

```
andy  ansible  11.573s  ansible-vault rekey inventenc.ini
Vault password:
New Vault password:
Confirm New Vault password:
Rekey successful
```

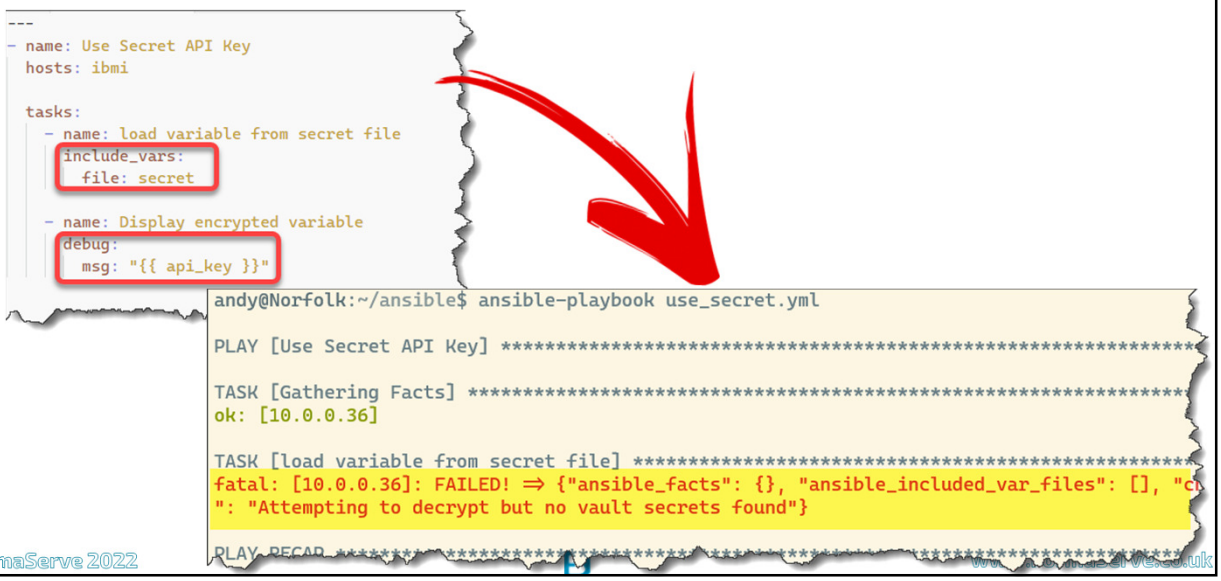

84

Sensitive Data - Encrypt Example

1. List file
2. Encrypt
3. Password
4. Success
5. List file

85

Sensitive Data - Examples



86

Sensitive Data - Examples

```
---
- name: Use Secret API Key
  hosts: ibmi

  tasks:
    - name: load variable from secret file
      include_vars:
        file: secret

    - name: Display encrypted variable
      debug:
        msg: "{{ api_key }}"
```

```
andy@Norfolk:~/ansible$ ansible-playbook --ask-vault-pass use_secret.yml
Vault password:

PLAY [Use Secret API Key] ******************************************************

TASK [Gathering Facts] *********************************************************
ok: [10.0.0.36]

TASK [load variable from secret file] ******************************************
ok: [10.0.0.36]

TASK [Display encrypted variable] **********************************************
ok: [10.0.0.36] => {
    "msg": "abc123"
}

PLAY RECAP *********************************************************************
10.0.0.36                  : ok=3    changed=0    unreachable=0    failed=0    ski
```


87

Sensitive Data - Examples

- Don't show sensitive data in log
- Can make debugging difficult!

```
tasks:
  - name: load variable from secret file
    include_vars:
      file: secret

  - name: Display encrypted variable
    debug:
      msg: "{{ api_key }}"
    no_log: true
```

```
TASK [load variable from secret file] ******************
ok: [10.0.0.36]

TASK [Display encrypted variable] **********************
ok: [10.0.0.36]

PLAY RECAP *********************************************
10.0.0.36                  : ok=3    changed=0    unreach
0
```


88

Sensitive Data - Decrypt File

- Use ansible-vault decrypt command
- If correct password input, file will be in the clear

```
andy@Norfolk:~/ansible$ ansible-vault decrypt secret
Vault password:
Decryption successful
andy@Norfolk:~/ansible$ cat secret
api_key: abc123
```


89

Sensitive Data - Configuration

- Does your config file contain sensitive data?
- If so, use ansible-vault to encrypt it!

90

Sensitive Data - Configuration

1. Encrypt inventory.ini
2. Enter password & encrypt
3. List password

```
andy  ansible  5.861s  ansible-vault encrypt inventory.ini --output inventenc.ini   1
New Vault password:
Confirm New Vault password:   2
Encryption successful
andy  ansible  9.427s  cat inventenc.ini   3
$ANSIBLE_VAULT;1.1;AES256
33353663306131666432383833396464303564336164336661633161333832633937393264643564
32393765323263666166363430613564373234303630323360a383034643464316439663666306634
65613866326435353936386363343164313431666139396530313137346339643835633938616139
38613834643038613370a37346466356537653334343635613431356330373336356561306266333 7
33383639366332663931336336616136383634313863303836376666376535343062326363623837
```


91

Sensitive Data - Configuration

1. Run ping with encrypted inventory file
2. Will not run as no password input!
3. Use –ask-vault-pass
4. Command runs successfully

```
andy  ansible  10ms  ansible ibmi -m ping -i inventenc.ini   1
[WARNING]:  * Failed to parse /home/andy/ansible/inventenc.ini with ini plugin: Attempting to decrypt but no vault secrets found
[WARNING]: Unable to parse /home/andy/ansible/inventenc.ini as an inventory source
[WARNING]: No inventory was parsed, only implicit localhost is available   2
[WARNING]: provided hosts list is empty, only localhost is available. Note that the implicit localhost does not match 'all'
[WARNING]: Could not match supplied host pattern, ignoring: ibmi
andy  ansible  1.081s  ansible ibmi -m ping -i inventenc.ini --ask-vault-pass   3
Vault password:
galatea | SUCCESS => {   4
    "changed": false,
    "ping": "pong"
```


92

My Ansible Top Tips

93

Ansible Top Tips

- Add --step to step thro' & omit tasks

```
andy@Norfolk:~/ansible$ ansible-playbook ibmi_user1.yml --step

PLAY [Create a new user on all IBMi hosts] ********************************
Perform task: TASK: Gathering Facts (N)o/(y)es/(c)ontinue: y

Perform task: TASK: Gathering Facts (N)o/(y)es/(c)ontinue: ****************

TASK [Gathering Facts] ****************************************************
ok: [10.0.0.36]
Perform task: TASK: Create new user (N)o/(y)es/(c)ontinue: n

Perform task: TASK: Create new user (N)o/(y)es/(c)ontinue: ****************
Perform task: TASK: Create anuvver new user (N)o/(y)es/(c)ontinue:
```


94

Ansible Top Tips

- Use ansible_facts to retrieve ansible host details

```
- name: Uses shell module to retrieve domain info

  tasks:

  - name: What is our domain?
    debug:
      msg: "Domain name: {{ ansible_facts['domain'] }}"
```

```
TASK [What is our domain?] ********************
ok: [localhost] => {
    "msg": "Domain name: local.formaserve.com"
}
```


95

Ansible Top Tips

- Validate a Playbook
  - Add the --syntax-check options
  - Ansible-playbook --syntax-check ibmi_library.yml

```
andy@Norfolk:~/ansible$ ansible-playbook --syntax-check ibmi_library.yml
ERROR! couldn't resolve module/action 'ibmi_cl_commands'. This often indicates a misspelling, missing collection, or inc
orrect module path.

The error appears to be in '/home/andy/ansible/ibmi_library.yml': line 10, column 7, but may
be elsewhere in the file depending on the exact syntax problem.

The offending line appears to be:

  tasks:
    - name: Create library
      ^ here
andy@Norfolk:~/ansible$
```


96

Ansible Top Tips

- A YAML Variable Gotcha!
  - What would the app_path variable be?

```
- hosts: db_servers
  vars:
    app_path: {{ ansible_config_file }}/22
```

  - We need to use quotes!

```
- hosts: db_servers
  vars:
      app_path: "{{ base_path }}/22"
```


97

Ansible Top Tips

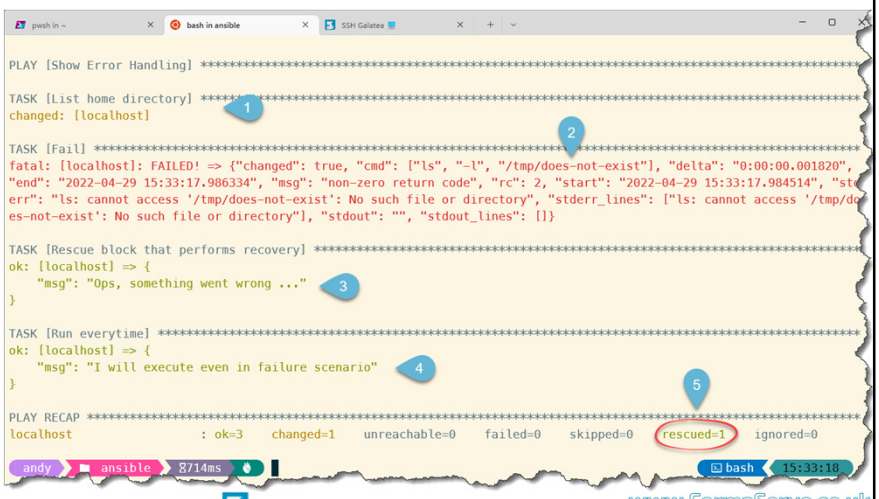
- Limit Playbook Execution
  - You can limit the hosts you target on a particular run with the --limit flag
  - Ansible-playbook eod.yml --limit galatea

98

Ansible Top Tips - Error Handling

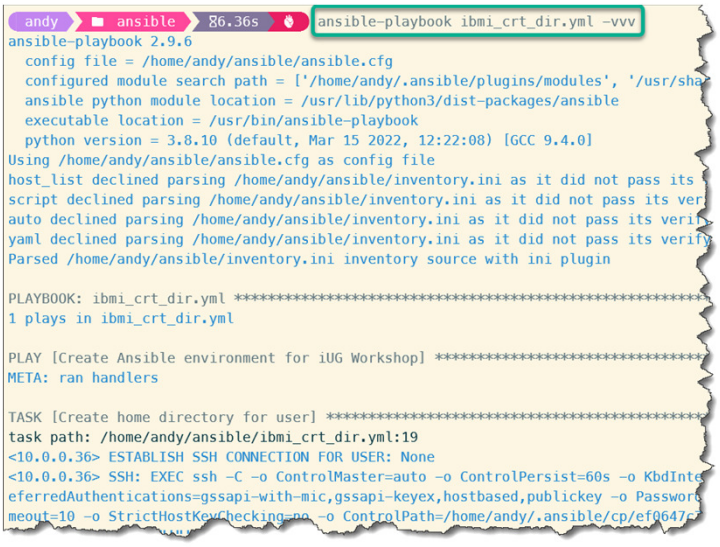
- Block, rescue & always

1. Task runs okay
2. Task fails
3. Rescue block
4. Always - runs every time



99

Ansible Top Tips Use -vvv for Diagnostic



100

Ansible Top Tips - Use tags

```
tasks:
- name: Query Disk Status
  ibmi_sql_query:
    sql: "select unit_number, resource_name, protection_status from Q
    joblog: true
  register: disk_results
  tags:
    - query

- name: Show disk output
  debug:
    msg: "Failed disk count: {{ disk_results.row_count }} Results: {{
  tags:
    - debug

- name: Send email to support
  mail:
    host:
    port:
    username:
    password:
    to: Andy Youens <andy@formaserve.co.uk>
    from: FormaServe Admin<fss@formaserve.co.uk>
    subject: "Disk Check on {{ ansible_hostname }}"
    subtype: html
    body: '<h3>Ansible disk checker.</h3>
      <p>There are {{ disk_results.row_count }} disk(s) in error.
      <br><br>
      <p>Results are:</p>
      <p>{{ disk_results.row }}</p>'
  tags:
    - email
```

to skip debug run
ansible-playbook ibmi_check_disks.yml --skip-tags debug

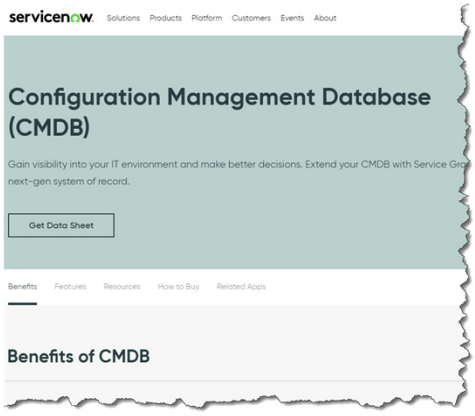
to skip email run
ansible-playbook ibmi_check_disks.yml --skip-tags email

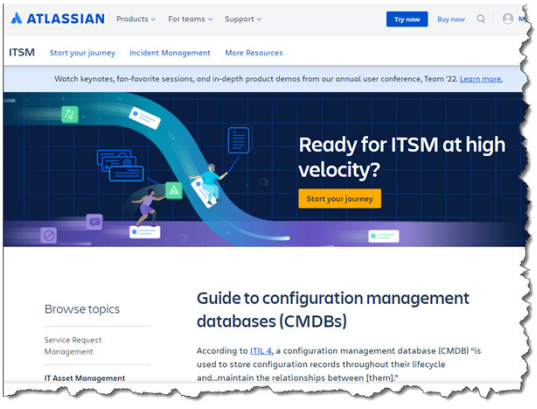
to only run SQL
ansible-playbook ibmi_check_disks.yml --tags query

101

Ansible Top Tips - CMDBs

- Find limitations on using static inventory files too limited?
- Take a look at Configuration Management Databases





102

Ansible Resources

103

Resources

- Workshop examples https://github.com/FormaServe/f_Learning
- Ansible for IBM i Galaxy 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/automationhub/ibm/power_ibmi
- Ansible for IBM i Articles
  - https://ibm.github.io/cloud-iblog/archivers/2020_0602_automate_your_ibm_i_tasks_with_ansible
  - https://developer.ibm.com/tutorials/ansible-automation-for-power/
  - https://powerwire.eu/an-ansible-ibm-i-disk-checker
- Blog https://ibm.github.io/cloud-i-blog/
- Ansible documents https://docs.ansible.com/
- Ansible IBM i modules https://ibm.github.io/ansible-for-i/modules.html

104

CI/CD Example

Create Dev Environment	Developers do code changes and unit test	Testing	Deployment
Use existing or provision new IBM i VMs for dev	Do code changes on IBM i / Windows / Linux / Mac	Provision new IBM i VMs for testing	Clone the latest code in master to build system
Install dependency products and fixes	Unit test on dev IBM i system	Install the related dependencies	Build testing
Pull the latest code to Rdi or to the IBM i VM	Commit code changes to development branch	Clone development branch and latest test cases to testing system	Deploy to production system
Build the latest code or push the build to dev IBM i VM	Create a pull request	Build latest code and kick off the tests	Clean up the environment
New Development Branch is created		PR Request approved and code automatically merged into master	
		Clean up the environment	
Ansible Playbook Run	Developer Manual Operations	Ansible Playbook Run	Ansible Playbook Run

105

Anything Else?

FormaServe

+44 (0)330 010958

Andy@formaserve.co.uk

www.formaserve.co.uk

@FormaServe & @AndyYouens

Thank You!

106

FormaServe

Still Training!

Even in these strange times

We are still giving virtual or on-site training

Its your choice ☺

Get in touch!

Email Andy@formaserve.co.uk

https://www.formaserve.co.uk/training.php

https://learning.formaserve.co.uk

107

Legal Bumph!

FormaServe Systems

- © Copyright FormaServe Systems 1990 - 2022
- All rights reserved
- All trademarks, trade names, service marks & logos referenced herein belong to their respective companies
- No unauthorised use, copying or distribution permitted

- This presentation is for informational purposes only
- FormaServe assumes no responsibility for the accuracy or completeness of the information within

108