→ Chapter: 09 TROUBLESHOOTING ANSIBLE

Troubleshooting playbooks

By default; Red Hat Ansible Engine ≠ configure log output to any log file.

Have to; 'log-path' in ansible.cfg in 'defaults' section to create logs & store in file.
Aso $-ANSIBLE_LOG_PATH env can be used.

logs are stored from 'ansible' commands & 'ansible-playbook' comands.

Debug module.

```
- name: Display free memory.
  debug:
    msg: "Free memory is {{ansible_facts['memfr...}}

- name: vars.
  debug:
    var: result
```

Debug module provides insight into what is happening in the play.

```
$ ansible-playbook playbook.yml --syntax-check
```
→ Checks yml syntax for the playbook.

```
$ ansible-playbook play.yml --step
```
→ Interactively prompts for confirmation that you want each task to run.

$ ansible-playbook playbook.yml --start-at-task="name of the task"

→ Start execution of a playbook from a specific task.

Play header ⇒ name of the play to be executed.
TASK ⇒ Name of ~~a~~ task followed by name of managed host & status ⇒ ok
fatal
Changed
Skipping.

play recap ⇒ Number of tasks executed for each managed hosts.

'-v' ⇒ increases verbosity of the output.

-v ⇒ Output data
-vv ⇒ (output + input) data
-vvv ⇒ —"— + connections to managed hosts.
-vvvv ⇒ —"— + scripts that are executed on manged hosts & user that is executing.

→ Recommended Practice for writting playbook ←
→ Name the play and tasks.
→ Include comments.
→ Organize vertically
→ Consistent horizontal indentation
→ Keep playbook simple.

Troubleshooting ansible manged hosts

$ ansible-playbook --check playbook.yml
→ Dry run of playbook without making any changes.

parameter 'check_mode: yes' ⇒ Always dry run
in task (no changes made)

parameter 'check_mode: no' ⇒ Always executes
in task normally

To check weather playbook is running in check mode;
variable ⇒ ansible_check_mode ⇒ true

always_run: yes ⇒ parameter ⇒ always runs even
in --check option.

$ ansible-playbook --diff play.yml
→ Reports changes made to the template file on
managed node.

uri module
checks that RESTful API is returning
the required content.

```
tasks:
  - uri :
      url: xyz
      return_content: yes
      register: apiresponse
  - fail :
      msg: 'version was not provided'
    when: "'version' not in apiresponse.content"
```

ø script module
executes a script on managed hosts.
fails if returncode for that script is zero.

ø stat module
gather info. of file ⇏ register it ⇏ use the output

```
tasks :
  - stat :
      path : xyz
      register : lock
  - fail :
    when : lock.stat.exists
```

if the file does not exists ⇏ variable will report false for *.stat.exists.

ø Assert module
Alternative to fail ⇏ with addition of 'that' conditional parameter.
use success_msg & fail_msg option ???

Using ping module to test weather you can connect to managed host.