Lab: Handling Task Failure

Introduction:

Ansible evaluates the return code of each task to determine whether the task succeeded or failed. Normally, when a task fails Ansible immediately aborts the rest of the play on that host, skipping all subsequent tasks. However, sometimes you might want to have play execution continue even if a task fails. For example, you might expect that a particular task could fail, and you might want to recover by running some other tasks conditionally. There are a number of Ansible features that can be used to manage task errors.

Objectives:

- **How Can You Handle Error in Ansible?**
- **Specifying Task Failure Conditions**
- **Managing Changed Status**
- **Using Ansible Blocks**
- **Using Ansible Blocks with Rescue and Always Statement**

1. How Can You Handle Error in Ansible?

1.1 Let's create the **failure.yml** playbook which contains a play with two tasks.

A. The first task with a **deliberate error** to cause failure

B. The second task to **install** some package

1.2 Let's define tasks that use the yum module and the two variables web_package and db_package The tasks will install the required packages

```
1  - name: Task Failure Exercise
2    hosts: eoc-node1
3    vars:
4      web_package: http
5      db_package: mariadb-server
6      db_service: mariadb
```

1.3 Let's install the Web Package.

```
 7    tasks:
 8      - name: Install {{ web_package }} package
 9        dnf:
10          name: "{{ web_package }}"
11          state: present
```

1.4 Let's install the db package.

```
12        - name: Install {{ db_package }} package
13          dnf:
14            name: "{{ db_package }}"
15            state: present
```

Info: **Typo in the package name.**

1.5 Let's view the manifest **failure.yml.**

```
# cat -n failure.yml
```

Output:

```
[root@eoc-controller ~]# cat -n failure.yml
     1  - name: Task Failure Exercise
     2    hosts: eoc-node1
     3    vars:
     4      web_package: http
     5      db_package: mariadb-server
     6      db_service: mariadb
     7    tasks:
     8      - name: Install {{ web_package }} package
     9        dnf:
    10          name: "{{ web_package }}"
    11          state: present
    12      - name: Install {{ db_package }} package
    13        dnf:
    14          name: "{{ db_package }}"
    15          state: present
```

1.6 Let's verify the syntax of **failure.yml** by executing below command

```
# ansible-playbook --syntax-check failure.yml
```

Output:

```
[root@eoc-controller ~]# ansible-playbook --syntax-check failure.yml

playbook: failure.yml
```

1.7 Let's run the playbook by executing below command.

```
# ansible-playbook failure.yml
```

Output:

```
[root@eoc-controller ~]# ansible-playbook failure.yml

PLAY [Task Failure Exercise] ***************************************************************************

TASK [Gathering Facts] *********************************************************************************
ok: [eoc-node1]

TASK [Install http package] ****************************************************************************
fatal: [eoc-node1]: FAILED! => {"changed": false, "failures": ["No package http available."], "msg": "Fai
led to install some of the specified packages", "rc": 1, "results": []}

PLAY RECAP *********************************************************************************************
eoc-node1                  : ok=1    changed=0    unreachable=0    failed=1    skipped=0    rescued=0
ignored=0
```

Note: The task failed because there is no such package called **http**. Because the first task **failed**, the second task was not run.

2. Specifying Task Failure Conditions

2.1 Let's edit & update the first task to ignore any errors by adding the **ignore_errors** keyword.

```
1  ---
2  - name: Task Failure lab
3    hosts: eoc-node1
4    vars:
5      web_package: http
6      db_package: mariadb-server
7      db_service: mariadb
```

2.2 Let's install the Web Package with ignore_errors keyword

```
 8    tasks:
 9      - name: Install {{ web_package }} package
10        dnf:
11          name: "{{ web_package }}"
12          state: present
13        ignore_errors: yes
```

2.3 Let's install the db package.

```
14      - name: Install {{ db_package }} package
15        dnf:
16          name: "{{ db_package }}"
17          state: present
```

2.4 Let's view the yml file.

```
# cat -n failure.yml
```

Output:

```
[admin@eoc-controller ~]$cat -n failure.yml
     1  ---
     2  - name: Task Failure lab
     3    hosts: eoc-node1
     4    vars:
     5      web_package: http
     6      db_package: mariadb-server
     7      db_service: mariadb
     8
     9    tasks:
    10      - name: Install {{ web_package }} package
    11        dnf:
    12          name: "{{ web_package }}"
    13          state: present
    14        ignore_errors: yes
    15      - name: Install {{ db_package }} package
    16        dnf:
    17          name: "{{ db_package }}"
    18          state: present
```

2.5 Let's verify the **failure.yml** manifest by running below command

```
# ansible-playbook --syntax-check failure.yml
```

Output:

```
[root@eoc-controller ~]# ansible-playbook --syntax-check failure.yml

playbook: failure.yml
```

2.6 Let's run the playbook **failure.yml** by executing below command

```
# ansible-playbook failure.yml
```

Output:

```
[root@eoc-controller ~]# ansible-playbook failure.yml

PLAY [Task Failure lab] ********************************************************************************

TASK [Gathering Facts] *********************************************************************************
ok: [eoc-node1]

TASK [Install http package] ****************************************************************************
fatal: [eoc-node1]: FAILED! => {"changed": false, "failures": ["No package http available."], "msg": "Fai
led to install some of the specified packages", "rc": 1, "results": []}
...ignoring

TASK [Install mariadb-server package] ******************************************************************
changed: [eoc-node1]

PLAY RECAP *********************************************************************************************
eoc-node1                  : ok=3    changed=1    unreachable=0    failed=0    skipped=0    rescued=0
ignored=1
```

Note: Despite the fact that the first task failed, ansible executed the second one

2.7 In this step you will set up a **block keyword** so you can experiment with how they work.

- Update the playbook by nesting the final task in a block clause and remove the line that sets ignore_errors: yes
- Next the task that installs the mariadb-server package in a **rescue** clause. The task will execute if the task is listed in the block clause fails.
- Finally add an always clause to start the database server upon installation using the service module.

```
1  ---
2  - name: Task Failure lab
3    hosts: eoc-node1
4    vars:
5      web_package: http
6      db_package: mariadb-server
7      db_service: mariadb
```

```
 8    tasks:
 9      - name: Attempt to set up a webserver
10        block:
11          - name: Install {{ web_package }} package
12            yum:
13              name: "{{ web_package }}"
14              state: present
```

control how Ansible responds to task errors using blocks with **rescue** and **always** sections.

Note: Rescue blocks specify tasks to run when an earlier task in a block fails. This approach is similar to exception handling in many programming languages. Ansible only runs rescue blocks after a task returns a 'failed' state. Bad task definitions and unreachable hosts will not trigger the rescue block

```
15        rescue:
16          - name: Install {{ de_package }} package
17            yum:
18              name: "{{ db_package }}"
19              state: present
```

Note: The tasks in the **block** execute normally. If any tasks in the block return **failed**, the **rescue** section executes tasks to recover from the error. The **always** section runs regardless of the results of the **block** and rescue sections.

```
20        always:
21          - name: strat {{ db_service }}
22            service:
23              name: "{{ db_service }}"
24              state: started
```

2.8 Let's view the manifest file.

```
# cat -n failure.yml
```

Output:

```
[root@eoc-controller ~]# cat -n failure.yml
     1  ---
     2  - name: Task Failure lab
     3    hosts: eoc-node1
     4    vars:
     5      web_package: http
     6      db_package: mariadb-server
     7      db_service: mariadb
     8    tasks:
     9      - name: Attempt to set up a webserver
    10        block:
    11          - name: Install {{ web_package }} package
    12            yum:
    13              name: "{{ web_package }}"
    14              state: present
    15        rescue:
    16          - name: Install {{ de_package }} package
    17            yum:
    18              name: "{{ db_package }}"
    19              state: present
    20        always:
    21          - name: strat {{ db_service }}
    22            service:
    23              name: "{{ db_service }}"
    24              state: started
```

2.9 Let's verify the syntax of **failure.yml** by executing below command.

```
# ansible-playbook --syntax-check failure.yml
```

Output:

```
[root@eoc-controller ~]# ansible-playbook --syntax-check failure.yml

playbook: failure.yml
```

2.10 Let's run the playbook by executing below command.

```
# ansible-playbook failure.yml
```

Output:

```
[root@eoc-controller ~]# ansible-playbook failure.yml

PLAY [Task Failure lab] ********************************************************************************

TASK [Gathering Facts] *********************************************************************************
ok: [eoc-node1]

TASK [Install http package] ****************************************************************************
fatal: [eoc-node1]: FAILED! => {"changed": false, "failures": ["No package http available."], "msg": "Fai
led to install some of the specified packages", "rc": 1, "results": []}

TASK [Install {{ de_package }} package] ****************************************************************
ok: [eoc-node1]

TASK [strat mariadb] ***********************************************************************************
changed: [eoc-node1]

PLAY RECAP *********************************************************************************************
eoc-node1                  : ok=3    changed=1    unreachable=0    failed=0    skipped=0    rescued=1
ignored=0
```

2.11 Let's edit the manifest **failure.yml** with correct package name as httpd.

```
 1  ---
 2  - name: Task Failure lab
 3    hosts: eoc-node1
 4    vars:
 5      web_package: httpd
 6      db_package: mariadb-server
 7      db_service: mariadb
 8    tasks:
 9      - name: Attempt to set up a webserver
10        block:
11          - name: Install {{ web_package }} package
12            yum:
13              name: "{{ web_package }}"
14              state: present
15        rescue:
16          - name: Install {{ de_package }} package
17            yum:
18              name: "{{ db_package }}"
19              state: present
20        always:
21          - name: strat {{ db_service }}
22            service:
23              name: "{{ db_service }}"
24              state: started
```

2.12 Let's verify the syntax of **failure.yml** by executing below command.

```
# ansible-playbook --syntax-check failure.yml
```

Output:

```
[root@eoc-controller ~]# ansible-playbook --syntax-check failure.yml

playbook: failure.yml
```

2.13 Let's Run the playbook.

```
# ansible-playbook failure.yml
```

Output:

```
[root@eoc-controller ~]# ansible-playbook failure.yml

PLAY [Task Failure lab] ********************************************************************************

TASK [Gathering Facts] *********************************************************************************
ok: [eoc-node1]

TASK [Install httpd package] ***************************************************************************
ok: [eoc-node1]

TASK [strat mariadb] ***********************************************************************************
ok: [eoc-node1]

PLAY RECAP *********************************************************************************************
eoc-node1                  : ok=3    changed=0    unreachable=0    failed=0    skipped=0    rescued=0
ignored=0
```

2.14 This step explores how to control the condition that causes a task to be reported as "changed" to the managed host.

Edit the playbook **failure.yml** to add two tasks to the start of the play, preceding the block. The first task uses the command module to run the date command and register the result in the command_result variable. The second task uses the debug module to print the standard output of the first task's command.

```
---
- name: Task Failure lab
  hosts: eoc-node1
  vars:
    web_package: httpd
    db_package: mariadb-server
    db_service: mariadb
  tasks:
    - name: check the local time
      command: date
      register: command_result
    - name: print local time
      debug:
        var: command_result.stdout
    - name: Attempt to set up a webserver
      block:
        - name: Install {{ web_package }} package
          yum:
            name: "{{ web_package }}"
            state: present
      rescue:
        - name: Install {{ de_package }} package
          yum:
            name: "{{ db_package }}"
            state: present
      always:
        - name: strat {{ db_service }}
          service:
            name: "{{ db_service }}"
            state: started
```

2.15 Let's verify the syntax **failure.yml** by executing below command

```
# ansible-playbook --syntax-check failure.yml
```

Output:

```
[root@eoc-controller ~]# ansible-playbook --syntax-check failure.yml

playbook: failure.yml
```

2.16 Let's run the playbook by executing below command.

```
# ansible-playbook failure.yml
```

Output:

```
[root@eoc-controller ~]# ansible-playbook failure.yml

PLAY [Task Failure lab] ********************************************************************************

TASK [Gathering Facts] *********************************************************************************
ok: [eoc-node1]

TASK [check the local time] ****************************************************************************
changed: [eoc-node1]

TASK [print local time] ********************************************************************************
ok: [eoc-node1] => {
    "command_result.stdout": "Thu Nov  9 11:02:18 IST 2023"
}

TASK [Install httpd package] ***************************************************************************
ok: [eoc-node1]

TASK [strat mariadb] ***********************************************************************************
ok: [eoc-node1]

PLAY RECAP *********************************************************************************************
eoc-node1                  : ok=5    changed=1    unreachable=0    failed=0    skipped=0    rescued=0
ignored=0
```

That **command** task should not report changed every time it runs because it is not changing the managed host. Because you know that the task will never change a managed host, add the line **changed_when: false** to the task to suppresses the change.

```
[root@eoc-controller ~]# cat -n failure.yml
     1  ---
     2  - name: Task Failure lab
     3    hosts: eoc-node1
     4    vars:
     5      web_package: httpd
     6      db_package: mariadb-server
     7      db_service: mariadb
     8    tasks:
     9      - name: check the local time
    10        command: date
    11        register: command_result
    12        changed_when: false
    13      - name: print local time
    14        debug:
    15          var: command_result.stdout
    16      - name: Attempt to set up a webserver
    17        block:
    18          - name: Install {{ web_package }} package
    19            yum:
    20              name: "{{ web_package }}"
    21              state: present
    22        rescue:
    23          - name: Install {{ de_package }} package
    24            yum:
    25              name: "{{ db_package }}"
    26              state: present
    27        always:
    28          - name: strat {{ db_service }}
    29            service:
    30              name: "{{ db_service }}"
    31              state: started
```

2.17 Let's verify the syntax of **failure.yml by** executing below command.

```
# ansible-playbook --syntax-check failure.yml
```

Output:

```
[root@eoc-controller ~]# ansible-playbook --syntax-check failure.yml

playbook: failure.yml
```

2.18 Let's run the playbook by executing below command.

```
# ansible-playbook failure.yml
```

Output:

```
[root@eoc-controller ~]# ansible-playbook failure.yml

PLAY [Task Failure lab] ********************************************************************************

TASK [Gathering Facts] *********************************************************************************
ok: [eoc-node1]

TASK [check the local time] ****************************************************************************
ok: [eoc-node1]

TASK [print local time] ********************************************************************************
ok: [eoc-node1] => {
    "command_result.stdout": "Thu Nov  9 11:06:11 IST 2023"
}

TASK [Install httpd package] ***************************************************************************
ok: [eoc-node1]

TASK [strat mariadb] ***********************************************************************************
ok: [eoc-node1]

PLAY RECAP *********************************************************************************************
eoc-node1                  : ok=5    changed=0    unreachable=0    failed=0    skipped=0    rescued=0
ignored=0
```