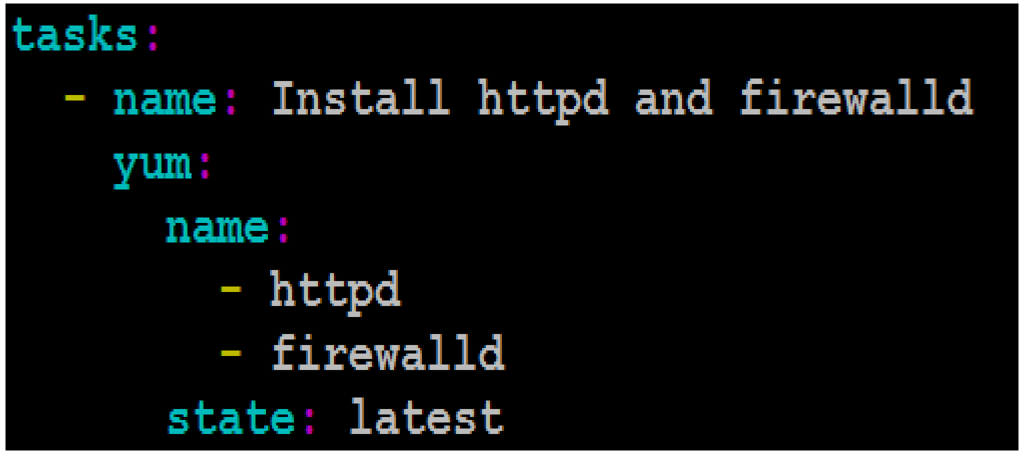
Ansible Playbooks

Eyes On Cloud

What is Task?

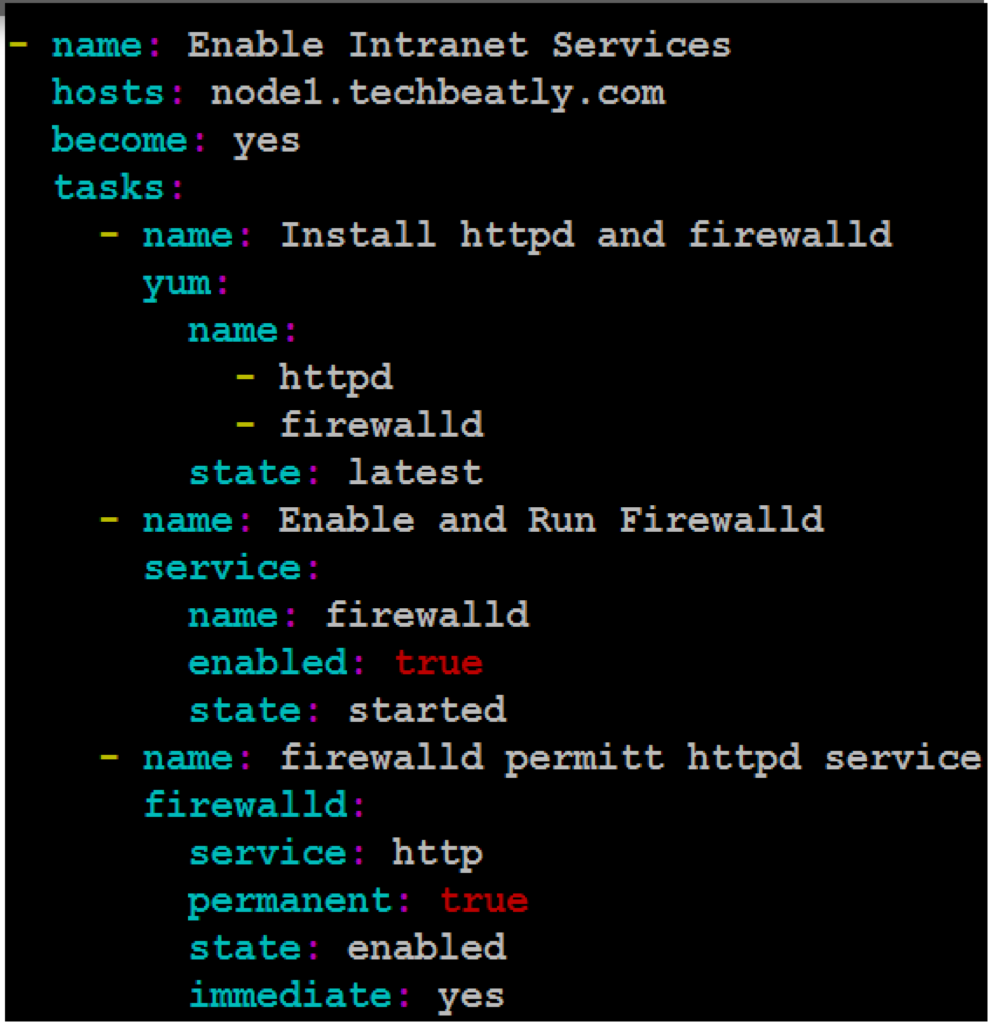
- A task is the smallest unit of action you can **automate** using an **Ansible playbook.**
- A **task** is the **application** of a module to perform a **specific unit of work.**



```
tasks:
  - name: Install httpd and firewalld
    yum:
      name:
        - httpd
        - firewalld
      state: latest
```

What is Play?

- A **play** is a **sequence of tasks** to be applied, in order, to **one** or **more hosts selected** from your **inventory**.

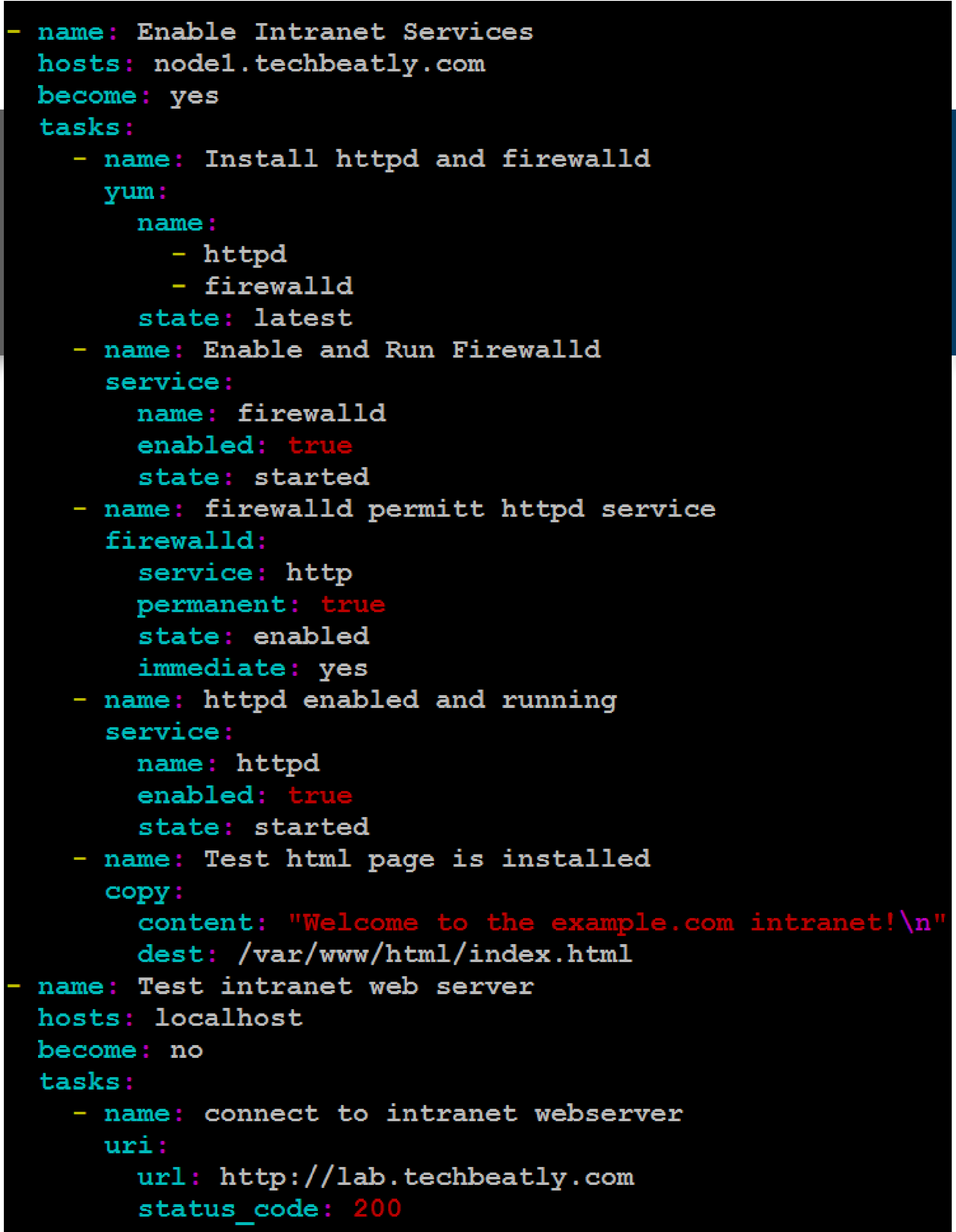


```yaml
- name: Enable Intranet Services
  hosts: node1.techbeatly.com
  become: yes
  tasks:
    - name: Install httpd and firewalld
      yum:
        name:
          - httpd
          - firewalld
        state: latest
    - name: Enable and Run Firewalld
      service:
        name: firewalld
        enabled: true
        state: started
    - name: firewalld permitt httpd service
      firewalld:
        service: http
        permanent: true
        state: enabled
        immediate: yes
```

What is Playbooks?

- Ansible Playbooks are **lists of tasks** that automatically execute for your specified inventory or groups of hosts.
- One or more Ansible tasks can be combined to make a play an ordered grouping of tasks mapped to specific hosts and tasks are executed in the order in which they are written
- Playbooks are Ansible’s configuration, deployment, and orchestration language.
- In a playbook, you can save the sequence of tasks in a play into a human-readable and immediately runnable form.

Example of Playbook

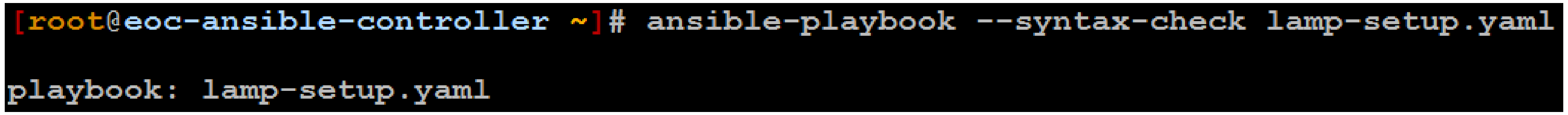


```
- name: Enable Intranet Services
  hosts: node1.techbeatly.com
  become: yes
  tasks:
    - name: Install httpd and firewalld
      yum:
        name:
          - httpd
          - firewalld
        state: latest
    - name: Enable and Run Firewalld
      service:
        name: firewalld
        enabled: true
        state: started
    - name: firewalld permitt httpd service
      firewalld:
        service: http
        permanent: true
        state: enabled
        immediate: yes
    - name: httpd enabled and running
      service:
        name: httpd
        enabled: true
        state: started
    - name: Test html page is installed
      copy:
        content: "Welcome to the example.com intranet!\n"
        dest: /var/www/html/index.html
- name: Test intranet web server
  hosts: localhost
  become: no
  tasks:
    - name: connect to intranet webserver
      uri:
        url: http://lab.techbeatly.com
        status_code: 200
```

Eyes On Cloud

Syntax Verification

- Prior to executing a **playbook**, it is good practice to perform a **verification** to ensure that the **syntax** of its contents is correct.
- The ansible-playbook command offers a **--syntax-check** option that you can use to verify the syntax of a playbook. The following example shows the successful syntax verification of a playbook.

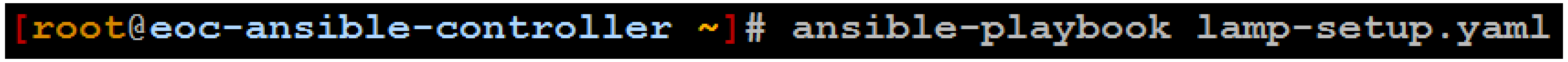


```
[root@eoc-ansible-controller ~]# ansible-playbook --syntax-check lamp-setup.yaml

playbook: lamp-setup.yaml
```

Running playbooks

- The **ansible-playbook** command is used to run **playbooks**.
- The command is executed on the control node and the name of the playbook to be run is passed as an argument:



```
[root@eoc-ansible-controller ~]# ansible-playbook lamp-setup.yaml
```

Appendix

8

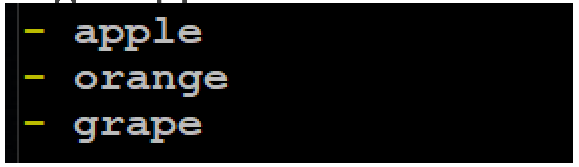
Eyes On Cloud

Formatting an Ansible Playbook(1-5)

- A playbook is a **text file** written in **YAML format**, and is normally saved with the **extension yaml** or **yml**.
- The playbook uses **indentation** with space characters to indicate the structure of its data.
- **YAML** does not place **strict requirements** on how many spaces are used for the indentation, but there are **two basic rules**.
  - **Same Indentation**.
  - **Items that are children of another item must be indented more than their parents**.

Formatting an Ansible Playbook (2-5)

- A playbook begins with a line consisting of three dashes (---) as a start of document marker.
- It may end with three dots (...) as an end of document marker
- In between those markers, the playbook is defined as a list of plays.
- An item in a YAML list starts with a single dash followed by a space. For example, a YAML list might appear as follows:

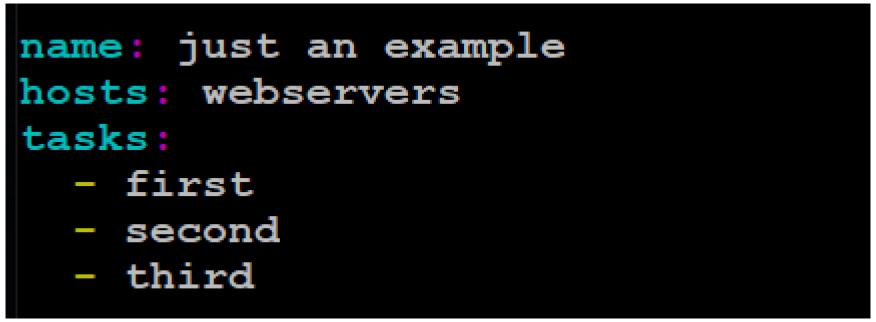


```
- apple
- orange
- grape
```

- In the preceding playbook example, the line after --- begins with a dash and starts the first (and only) play in the list of plays.
- The play itself is a collection of **key-value** pairs.
- Keys in the same play should have the same indentation.

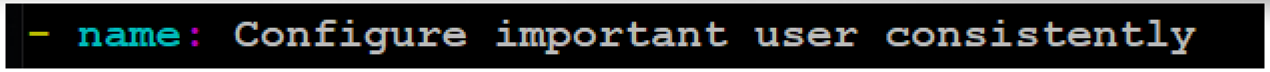
Formatting an Ansible Playbook (3-5)

- The following example shows a **YAML** snippet with three keys.
- The first two keys have simple values.
- The third has a list of three items as a value.



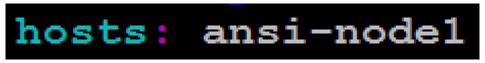
```
name: just an example
hosts: webservers
tasks:
  - first
  - second
  - third
```

Formatting an Ansible Playbook (4-5)



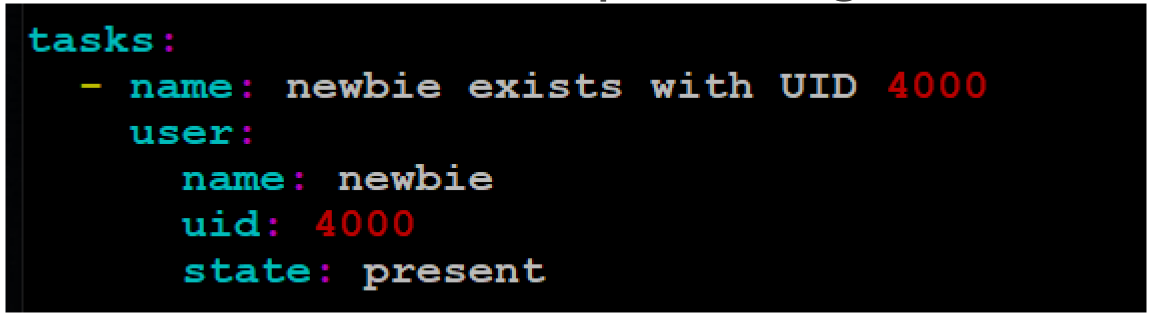
```
- name: Configure important user consistently
```

- The second key in the play is a hosts **attribute**, which specifies the hosts against which the play's tasks are run.
- Like the argument for the ansible command, the hosts attribute takes a host pattern as a value, such as the names of managed hosts or groups in the inventory.



```
hosts: ansi-node1
```

- Finally, the last key in the play is the tasks attribute, whose value specifies a list of tasks to run for this play. This example has a single task, which runs the user module with specific arguments



```
tasks:
  - name: newbie exists with UID 4000
    user:
      name: newbie
      uid: 4000
      state: present
```

Formatting an Ansible Playbook (5-5)

The tasks attribute is the part of the play that actually lists, in order, the tasks to be run on the managed hosts.

Each task in the list is itself a collection of **key-value** pairs.

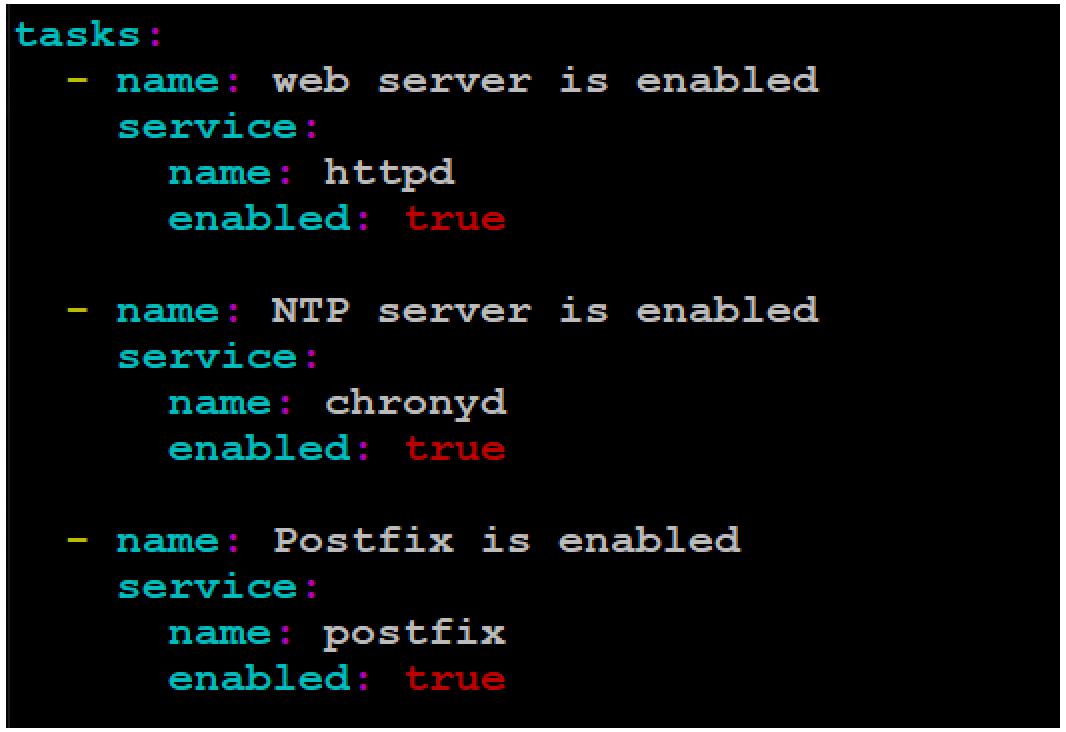
In this example, the only task in the play has two keys:

- Name is an optional label documenting the purpose of the task.
- User is the module to run for this task.
- Its arguments are passed as a collection of **key-value** pairs, which are children of the module

Eyes On Cloud

The following is another example of a tasks attribute with multiple tasks:

- Using the service module to ensure that several network services are enabled to start at boot:



```
tasks:
  - name: web server is enabled
    service:
      name: httpd
      enabled: true

  - name: NTP server is enabled
    service:
      name: chronyd
      enabled: true

  - name: Postfix is enabled
    service:
      name: postfix
      enabled: true
```