Ansible “loop” option might look like little backward but it will very useful when you want to perform the repetitive tasks. For each section in ansible playbook, we need to provide a meaningful name. For an example, if you want to install multiple packages, you need to create the section for each package. This could increase the length of the playbook and have to create a new block of code for the same action but different packages. In such cases, “loop” function will help to repeat the action for a list of items.

**Environment**

* Ansible Server –**ansible-server**
* Remote hosts – **uaans69 , gpfslinapp1**

**Loop Function on YUM module:**

1. Login to Ansible server and view the ad-hoc inventory.  (If you do not have one, just add the remote hosts in the file)

[linadm@ansible-server automation]$ cat lin-servers.1

gpfslinapp1

uaans69

[linadm@ansible-server automation]$

2. Create a new playbook using “item” & “with\_items” to demonstrate the loop function.

[linadm@ansible-server automation]$ cat install\_packages1.yaml

---

- hosts: all

become: yes

tasks:

- name: Install Packages

yum: name={{ item }} update\_cache=yes state=latest

with\_items:

- vim

- lsof

- nano

[linadm@ansible-server automation]$

In the above playbook, “yum” module has been used to install the list of packages after updating the cache.

3. Execute the playbook using the ad-hoc inventory.

[linadm@ansible-server automation]$ ansible-playbook -i lin-servers.1 install\_packages1.yaml

PLAY [all] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

TASK [Gathering Facts] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [gpfslinapp1]

ok: [uaans69]

TASK [Install Packages] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [uaans69] => (item=[u'vim', u'lsof', u'nano'])

changed: [gpfslinapp1] => (item=[u'vim', u'lsof', u'nano'])

PLAY RECAP \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

gpfslinapp1 : ok=2 changed=1 unreachable=0 failed=0

uaans69 : ok=2 changed=0 unreachable=0 failed=0

[linadm@ansible-server automation]$

Here we could see that packages have been installed on remote hosts “gpfslinapp1” and packages exist on “uaans69”.

### ****Impact of Inefficient Code :****

1. What will be the impact of writing inefficient code?

If we would have written the playbook without “item” and “with\_items”, the playbook would look like below.  
We could see that the number of lines is more comparable to the above one.

---

- hosts: all

become: yes

tasks:

- name: Install Packages

yum: name=vim update\_cache=yes state=latest

- name: Install Packages

yum: name=lsof state=latest

- name: Install Packages

yum: name=nano state=latest

2. Let’s measure the execution time of the above-mentioned playbook.

[linadm@ansible-server automation]$ time ansible-playbook -i lin-servers.1 install\_packages1\_without\_loop.yaml

PLAY [all] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

TASK [Gathering Facts] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [gpfslinapp1]

ok: [uaans69]

TASK [Install Packages] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [uaans69]

ok: [gpfslinapp1]

TASK [Install Packages] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [uaans69]

ok: [gpfslinapp1]

TASK [Install Packages] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [uaans69]

ok: [gpfslinapp1]

PLAY RECAP \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

gpfslinapp1 : ok=4 changed=0 unreachable=0 failed=0

uaans69 : ok=4 changed=0 unreachable=0 failed=0

real 0m44.297s

user 0m5.048s

sys 0m1.269s

[linadm@ansible-server automation]$

3. Let’s measure the runtime of the playbook which has been created with loops.

[linadm@ansible-server automation]$ time ansible-playbook -i lin-servers.1 install\_packages1.yaml

PLAY [all] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

TASK [Gathering Facts] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [gpfslinapp1]

ok: [uaans69]

TASK [Install Packages] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [uaans69] => (item=[u'vim', u'lsof', u'nano'])

ok: [gpfslinapp1] => (item=[u'vim', u'lsof', u'nano'])

PLAY RECAP \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

gpfslinapp1 : ok=2 changed=0 unreachable=0 failed=0

uaans69 : ok=2 changed=0 unreachable=0 failed=0

real 0m27.082s

user 0m4.364s

sys 0m0.697s

[linadm@ansible-server automation]$

Here you could see the difference. Using the right function at the right place will always reduce the load and  
runtime.

**User Module – Loop:**

Here is another use case of “loop” in Ansible. The following playbook can be used to create the list of users on all the remote nodes.

---

- hosts: all

become: yes

- name: add generic account users on all the hosts

user:

name: "{{ item }}"

state: present

groups: "adb"

loop:

- webadmin

- dbadmin

* Usernames: “webadmin”, “dbadmin”
* Group: “adb”