Lab: Modifying and Copying Files to Hosts

Introduction:

In this Lab we will use standard Ansible modules to create, install, edit, and remove files on managed hosts and manage the permissions, ownership and SELinux contexts of those files.

Objectives:

- **Retrieve Files from managed files**
- **Verifying the results**
- **Enabling SELinux**
- **Managing all hosts**
- **Verifying Attributes**
- **SELinux for user**
- **Add linein file and blockin file module**
- **File Module to Remove the User file**

1. Retrieve Files from managed files

1.1 Let’s Create a playbook called **securebackup.yml** in the current working directory. Configure the playbook to use the fetch module to retrieve the **/var/log/secure** log file from each of managed hosts and store them on the **control node**. The playbook should create the **secure-backups** directory with sub directories named after the **hostname of each managed host.**

```
---
- name: Use the fetch module to retrive secure log files
  hosts: all
  become: yes
```

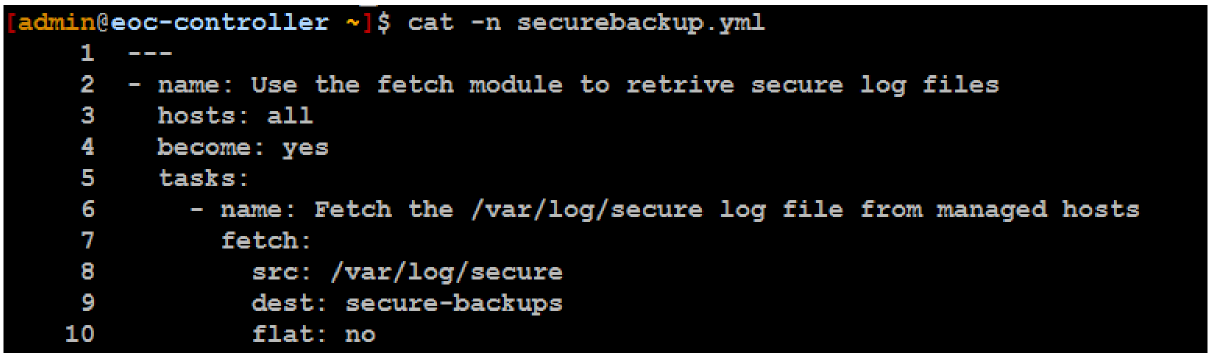
1.1 Add a task to **securebackup.yml** playbook that retrieves the **/var/log/secure** log file from the managed hosts and stores it in the secure-backups directory. The fetch modules create the **secure-backups directory** if it does not exist. Use the flat: no parameter to ensure the default behavior of appending the hostname, path and file name to destination.

```
  tasks:
    - name: Fetch the /var/log/secure log file from managed hosts
      fetch:
        src: /var/log/secure
        dest: secure-backups
        flat: no
```

1.2 Let’s view the **securebackup.yml** file.

```
# cat -n securebackup.yml
```

Output:

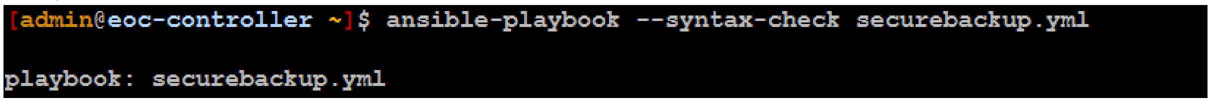


```
[admin@eoc-controller ~]$ cat -n securebackup.yml
     1  ---
     2  - name: Use the fetch module to retrive secure log files
     3    hosts: all
     4    become: yes
     5    tasks:
     6      - name: Fetch the /var/log/secure log file from managed hosts
     7        fetch:
     8          src: /var/log/secure
     9          dest: secure-backups
    10          flat: no
```

1.3 Run the ansible-playbook --syntax-check **securebackup.yml** command to verify its syntax and correct any errors.

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

Output:

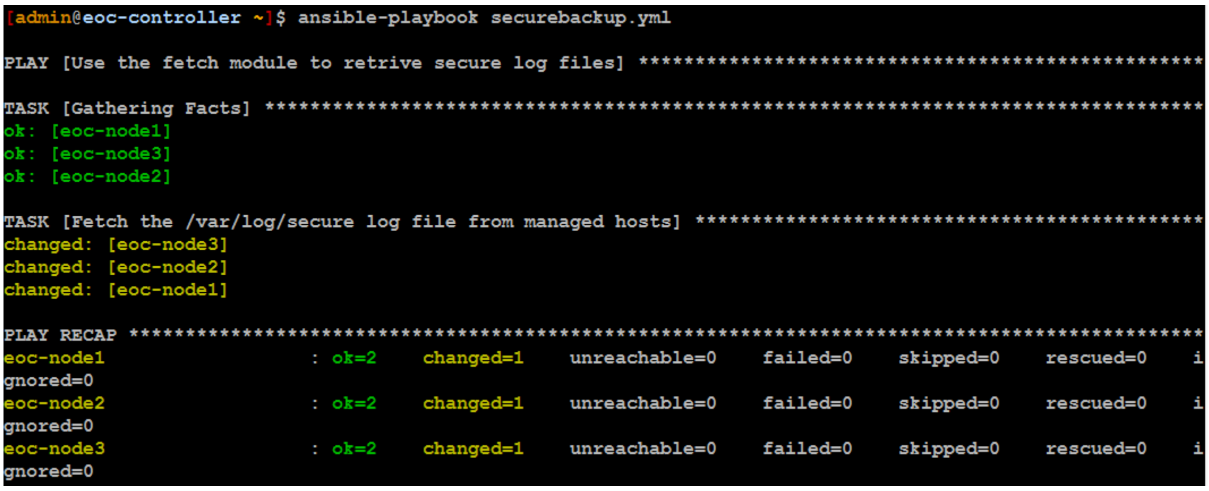


```
[admin@eoc-controller ~]$ ansible-playbook --syntax-check securebackup.yml

playbook: securebackup.yml
```

1.4 Let's run ansible-playbook **securebackup.yml** to execute the playbook.

```
# ansible-playbook securebackup.yml
```

Output:



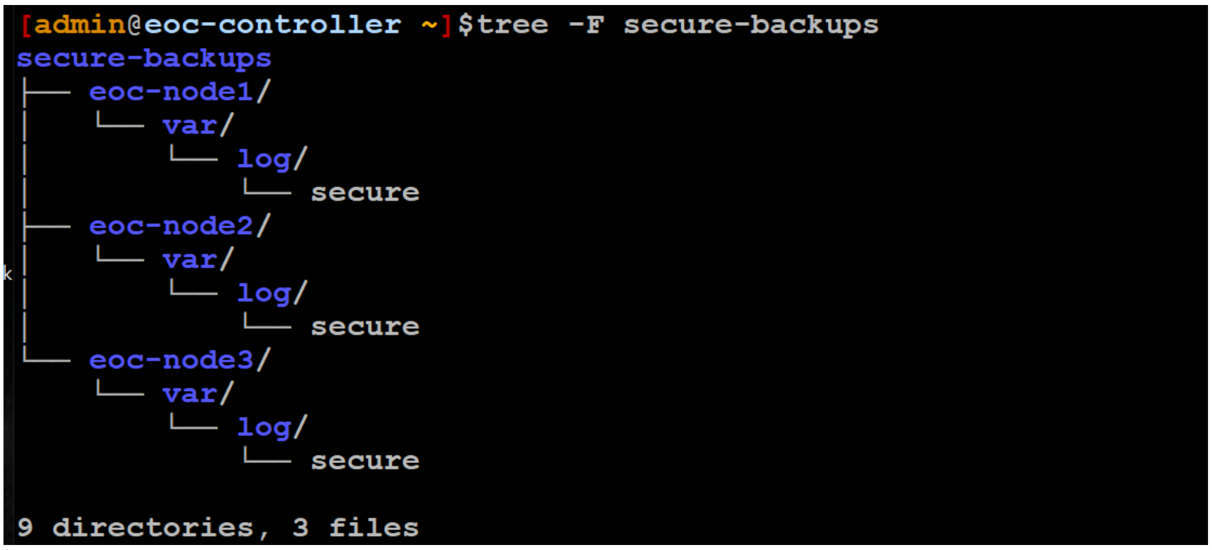
```
[admin@eoc-controller ~]$ ansible-playbook securebackup.yml

PLAY [Use the fetch module to retrive secure log files] ***************************************************

TASK [Gathering Facts] ************************************************************************************
ok: [eoc-node1]
ok: [eoc-node3]
ok: [eoc-node2]

TASK [Fetch the /var/log/secure log file from managed hosts] **********************************************
changed: [eoc-node3]
changed: [eoc-node2]
changed: [eoc-node1]

PLAY RECAP ************************************************************************************************
eoc-node1                  : ok=2    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    i
gnored=0
eoc-node2                  : ok=2    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    i
gnored=0
eoc-node3                  : ok=2    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    i
gnored=0
```

2. Verifying the results

2.1 Let's verify the playbook results.

```
# tree -F secure-backups
```

Output:



```
[admin@eoc-controller ~]$tree -F secure-backups
secure-backups
├── eoc-node1/
│   └── var/
│       └── log/
│           └── secure
├── eoc-node2/
│   └── var/
│       └── log/
│           └── secure
└── eoc-node3/
    └── var/
        └── log/
            └── secure

9 directories, 3 files
```

3. Enabling SELinux

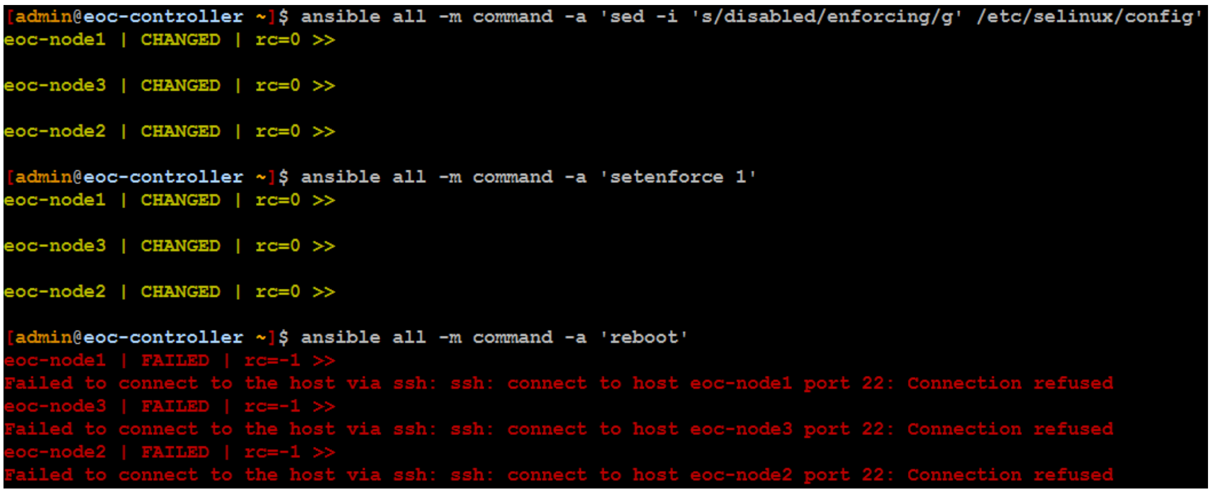
3.1 Let's enable the SELinux to perform this Lab.

```
# ansible all -m command -a 'sed -i 's/disabled/enforcing/g'
/etc/selinux/config'
```

```
# ansible all -m command -a 'setenforce 1'
```

```
# ansible all -m command -a 'reboot'
```

Output:



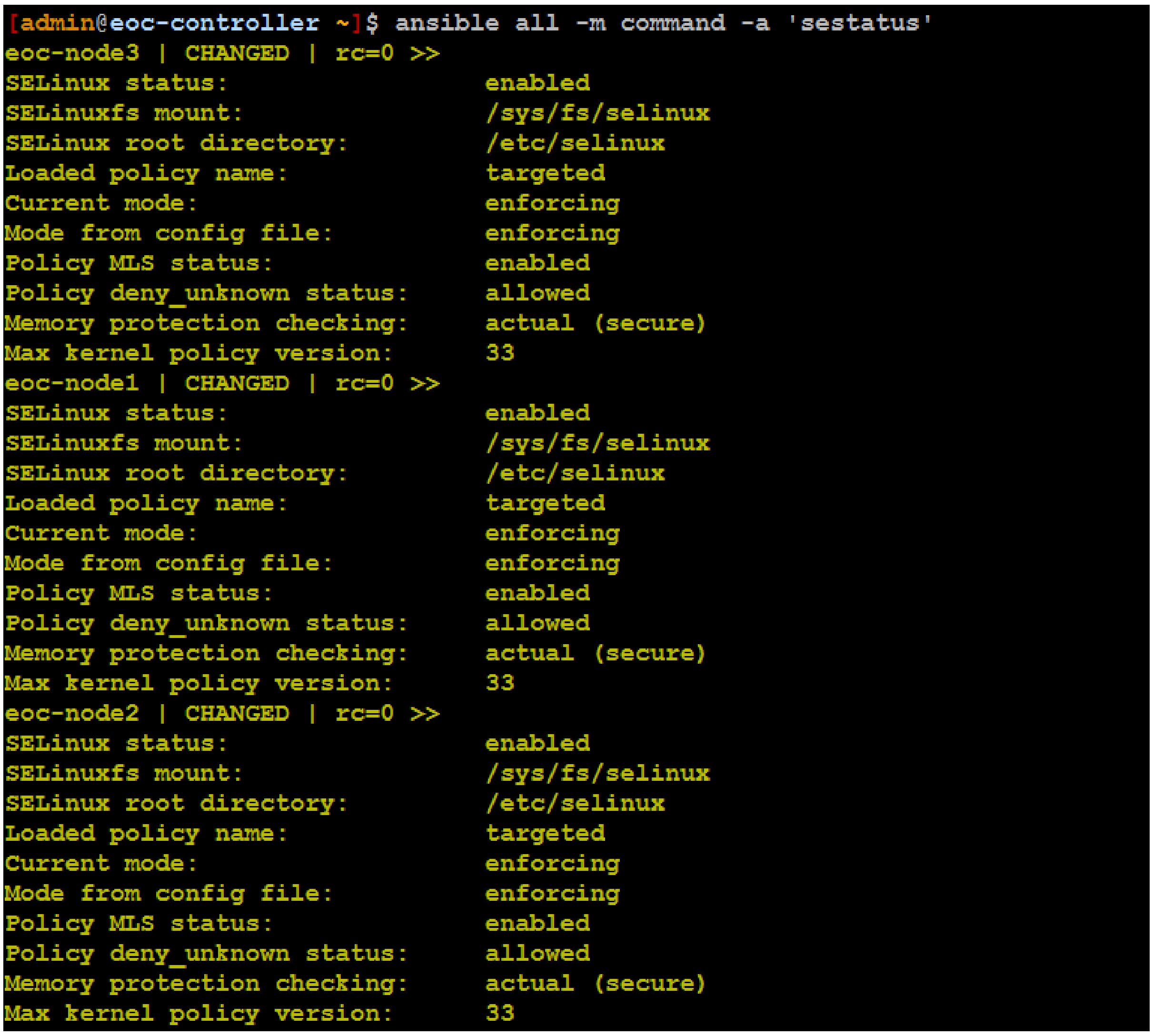
```
[admin@eoc-controller ~]$ ansible all -m command -a 'sed -i 's/disabled/enforcing/g' /etc/selinux/config'
eoc-node1 | CHANGED | rc=0 >>

eoc-node3 | CHANGED | rc=0 >>

eoc-node2 | CHANGED | rc=0 >>

[admin@eoc-controller ~]$ ansible all -m command -a 'setenforce 1'
eoc-node1 | CHANGED | rc=0 >>

eoc-node3 | CHANGED | rc=0 >>

eoc-node2 | CHANGED | rc=0 >>

[admin@eoc-controller ~]$ ansible all -m command -a 'reboot'
eoc-node1 | FAILED | rc=-1 >>
Failed to connect to the host via ssh: ssh: connect to host eoc-node1 port 22: Connection refused
eoc-node3 | FAILED | rc=-1 >>
Failed to connect to the host via ssh: ssh: connect to host eoc-node3 port 22: Connection refused
eoc-node2 | FAILED | rc=-1 >>
Failed to connect to the host via ssh: ssh: connect to host eoc-node2 port 22: Connection refused
```

Info: Wait for some minutes.

3.2 Let's view the status of selinux.

```
# ansible all -m command -a 'sestatus'
```

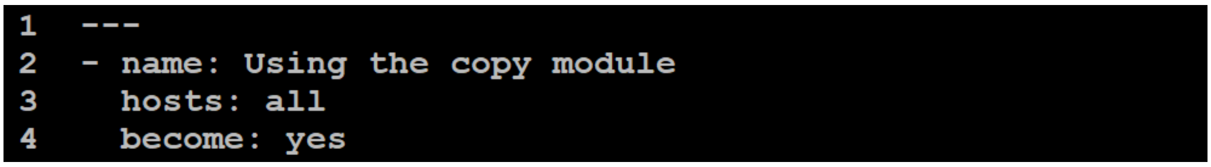
Output:



```
[admin@eoc-controller ~]$ ansible all -m command -a 'sestatus'
eoc-node3 | CHANGED | rc=0 >>
SELinux status:                 enabled
SELinuxfs mount:                /sys/fs/selinux
SELinux root directory:         /etc/selinux
Loaded policy name:             targeted
Current mode:                   enforcing
Mode from config file:          enforcing
Policy MLS status:              enabled
Policy deny_unknown status:     allowed
Memory protection checking:     actual (secure)
Max kernel policy version:      33
eoc-node1 | CHANGED | rc=0 >>
SELinux status:                 enabled
SELinuxfs mount:                /sys/fs/selinux
SELinux root directory:         /etc/selinux
Loaded policy name:             targeted
Current mode:                   enforcing
Mode from config file:          enforcing
Policy MLS status:              enabled
Policy deny_unknown status:     allowed
Memory protection checking:     actual (secure)
Max kernel policy version:      33
eoc-node2 | CHANGED | rc=0 >>
SELinux status:                 enabled
SELinuxfs mount:                /sys/fs/selinux
SELinux root directory:         /etc/selinux
Loaded policy name:             targeted
Current mode:                   enforcing
Mode from config file:          enforcing
Policy MLS status:              enabled
Policy deny_unknown status:     allowed
Memory protection checking:     actual (secure)
Max kernel policy version:      33
```

4. Managing all hosts

4.1 Create the **copyfile.yml** playbook in the current working directory. Configure the playbook to copy the **/root/ filemanage/files/users.txt** file to all managed hosts.



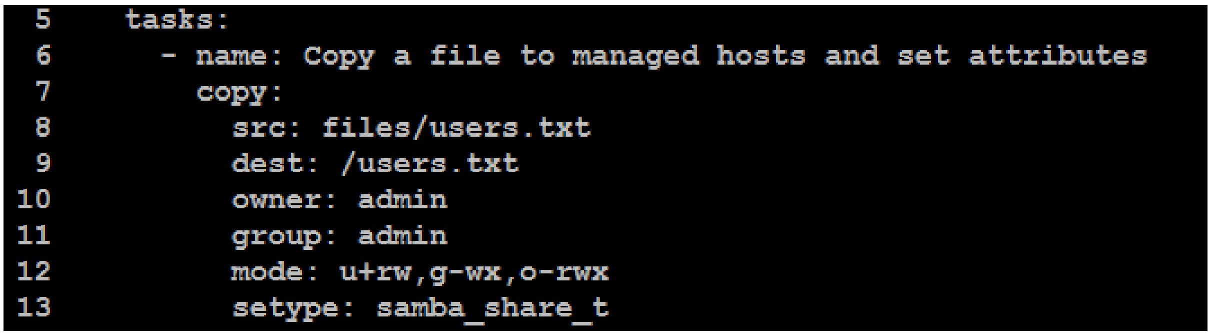
```
---
- name: Using the copy module
  hosts: all
  become: yes
```

4.2 Add a task to use the copy module to copy the /home/admin/file-manage/files/users.txt file to all managed hosts. Use the copy module to set the following parameters for the users.txt file:

PARAMETER	VALUES
src	files/users.txt
dest	/users.txt
owner	admin
group	admin

mode	u+rw,g-wx,o-rwx
setype	samba_share_t

Note: This example uses symbolic permissions notation. The letters **u**, **g**, and **o** stand for "**user**", "**group**", and "**other**". The equals sign ("**=**") means "**set the permissions exactly like this**," and the letters "**r**", "**w**", and "**x**" stand for "**read**", "**write**", and "**execute**", respectively. The commas separate the different classes of permissions, and there are no spaces between them.

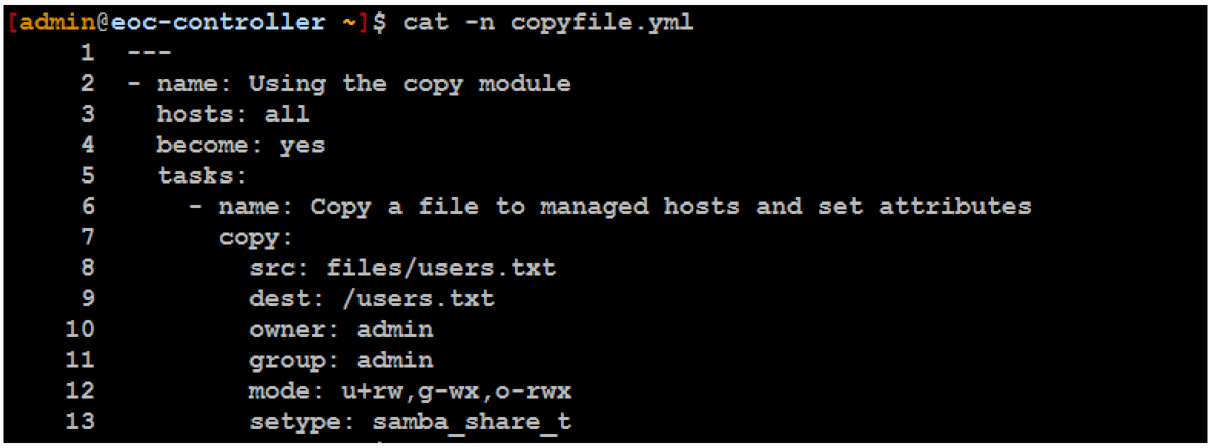


```
  tasks:
    - name: Copy a file to managed hosts and set attributes
      copy:
        src: files/users.txt
        dest: /users.txt
        owner: admin
        group: admin
        mode: u+rw,g-wx,o-rwx
        setype: samba_share_t
```

4.3 Let's view the manifest copyfile.yml.

```
# cat -n copyfile.yml
```

Output:



```
[admin@eoc-controller ~]$ cat -n copyfile.yml
     1  ---
     2  - name: Using the copy module
     3    hosts: all
     4    become: yes
     5    tasks:
     6      - name: Copy a file to managed hosts and set attributes
     7        copy:
     8          src: files/users.txt
     9          dest: /users.txt
    10          owner: admin
    11          group: admin
    12          mode: u+rw,g-wx,o-rwx
    13          setype: samba_share_t
```

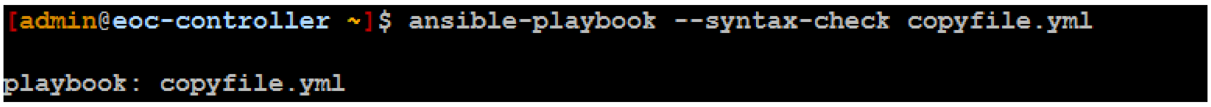
4.4 Create a **directory** by the name **files** and a **file** name **users.txt** inside the directory.

```
# mkdir -p files

# touch files/users.txt
```

4.5 Run the ansible-playbook --syntax-check **copyfile.yml** command to verify its syntax and correct any errors

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

Output:

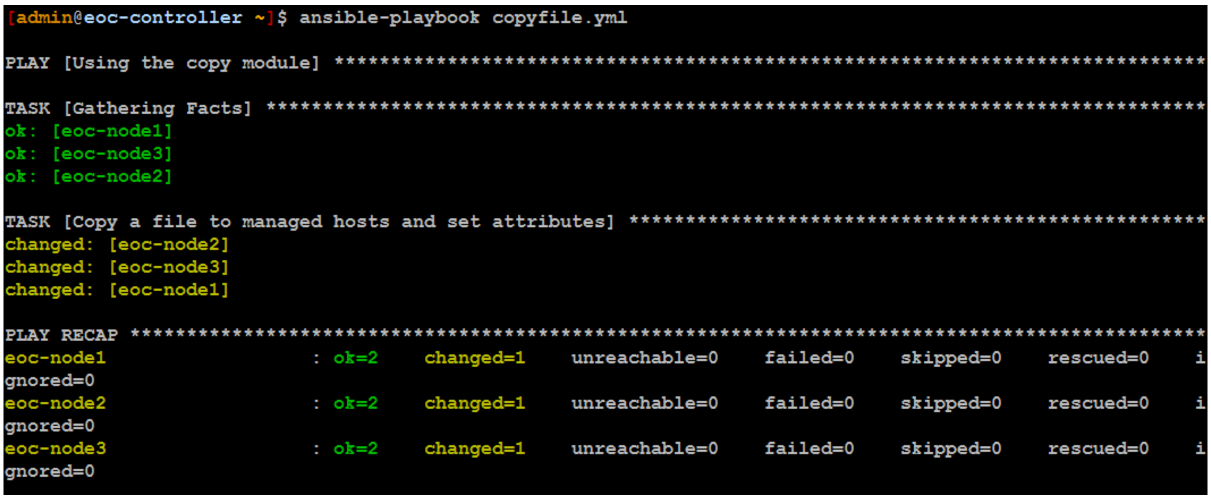


```
[admin@eoc-controller ~]$ ansible-playbook --syntax-check copyfile.yml

playbook: copyfile.yml
```

4.6 Run ansible-playbook **copyfile.yml** to execute the playbook.

```
# ansible-playbook copyfile.yml
```

Output:



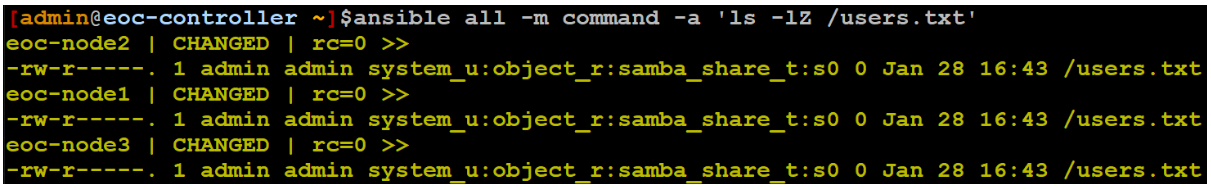
```
[admin@eoc-controller ~]$ ansible-playbook copyfile.yml

PLAY [Using the copy module] ***********************************************************************

TASK [Gathering Facts] *****************************************************************************
ok: [eoc-node1]
ok: [eoc-node3]
ok: [eoc-node2]

TASK [Copy a file to managed hosts and set attributes] *********************************************
changed: [eoc-node2]
changed: [eoc-node3]
changed: [eoc-node1]

PLAY RECAP *****************************************************************************************
eoc-node1                  : ok=2    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    i
gnored=0
eoc-node2                  : ok=2    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    i
gnored=0
eoc-node3                  : ok=2    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    i
gnored=0
```

5. Verifying Attributes

5.1 Use an ad hoc command to execute the ls –Z command as user root to verify the attributes of the users.txt file on the managed hosts.

```
# ansible all -m command -a 'ls -lZ /users.txt'
```

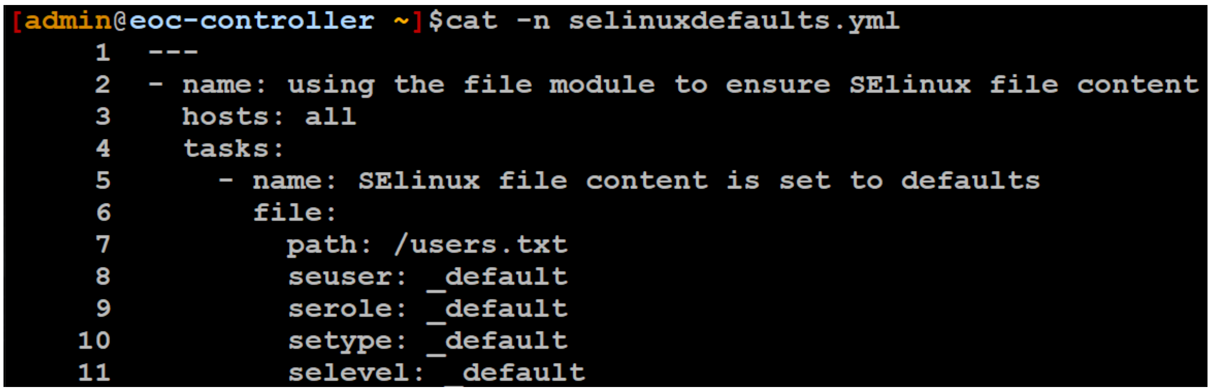
Output:



```
[admin@eoc-controller ~]$ansible all -m command -a 'ls -lZ /users.txt'
eoc-node2 | CHANGED | rc=0 >>
-rw-r-----. 1 admin admin system_u:object_r:samba_share_t:s0 0 Jan 28 16:43 /users.txt
eoc-node1 | CHANGED | rc=0 >>
-rw-r-----. 1 admin admin system_u:object_r:samba_share_t:s0 0 Jan 28 16:43 /users.txt
eoc-node3 | CHANGED | rc=0 >>
-rw-r-----. 1 admin admin system_u:object_r:samba_share_t:s0 0 Jan 28 16:43 /users.txt
```

6. SELinux for user

6.1 Create a playbook called **selinuxdefaults.yml** in the current working directory. Configure the playbook to use the file module to ensure the default SELinux context for user, role, type, and level fields.

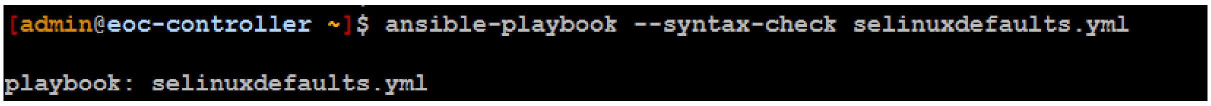


```
[admin@eoc-controller ~]$cat -n selinuxdefaults.yml
     1  ---
     2  - name: using the file module to ensure SElinux file content
     3    hosts: all
     4    tasks:
     5      - name: SElinux file content is set to defaults
     6        file:
     7          path: /users.txt
     8          seuser: _default
     9          serole: _default
    10          setype: _default
    11          selevel: _default
```

6.2 Run the ansible-playbook --syntax-check **selinuxdefaults.yml** command to verify its syntax and correct any errors.

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

Output:

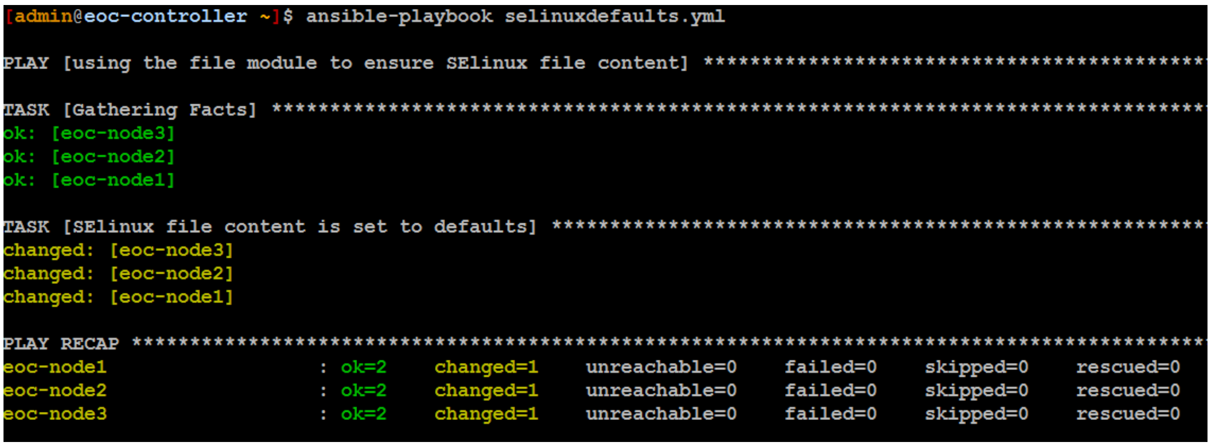


```
[admin@eoc-controller ~]$ ansible-playbook --syntax-check selinuxdefaults.yml

playbook: selinuxdefaults.yml
```

6.3 Run ansible-playbook **selinuxdefaults.yml** to execute the playbook.

```
# ansible-playbook selinuxdefaults.yml
```

Output:



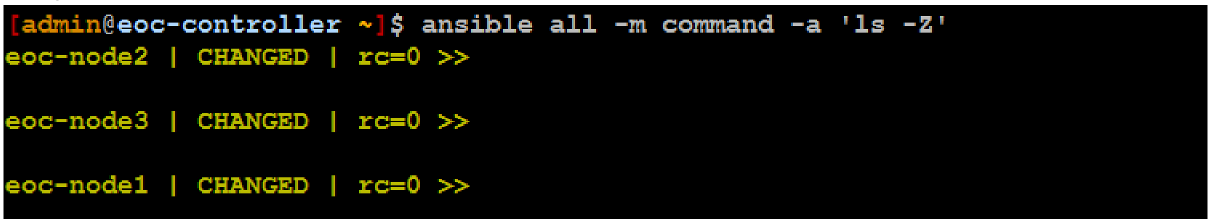
```
[admin@eoc-controller ~]$ ansible-playbook selinuxdefaults.yml

PLAY [using the file module to ensure SElinux file content] ******************************************

TASK [Gathering Facts] *******************************************************************************
ok: [eoc-node3]
ok: [eoc-node2]
ok: [eoc-node1]

TASK [SElinux file content is set to defaults] *******************************************************
changed: [eoc-node3]
changed: [eoc-node2]
changed: [eoc-node1]

PLAY RECAP *******************************************************************************************
eoc-node1                  : ok=2    changed=1    unreachable=0    failed=0    skipped=0    rescued=0
eoc-node2                  : ok=2    changed=1    unreachable=0    failed=0    skipped=0    rescued=0
eoc-node3                  : ok=2    changed=1    unreachable=0    failed=0    skipped=0    rescued=0
```

6.4 Use an ad hoc command to execute the ls –Z command as admin to verify the default file attributes on unconfined_u:object_r:user_home_t:s0.

Note: -Z ---> print any security context of each file

```
# ansible all -m command -a 'ls -Z'
```

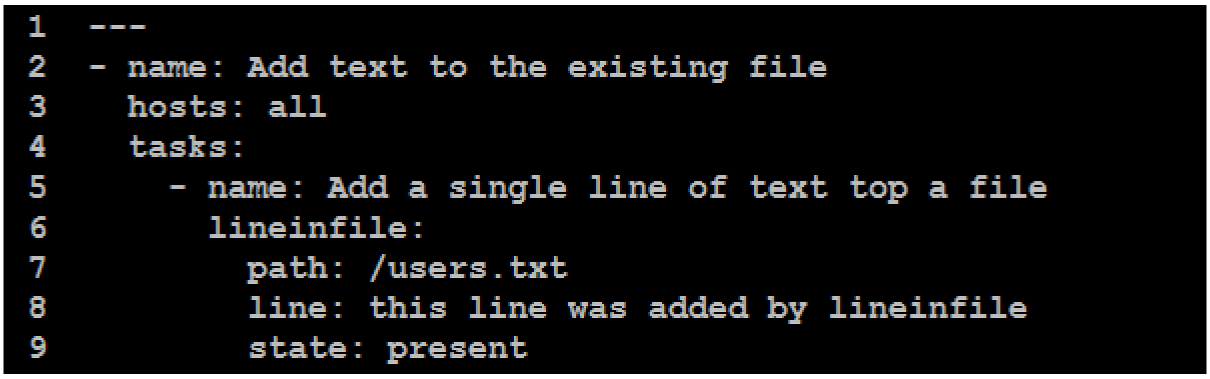
Output:



```
[admin@eoc-controller ~]$ ansible all -m command -a 'ls -Z'
eoc-node2 | CHANGED | rc=0 >>

eoc-node3 | CHANGED | rc=0 >>

eoc-node1 | CHANGED | rc=0 >>
```

7. Add linein file and blockin file module

7.1 Create a playbook called **addline.yml** in the current working directory. Configure the playbook to use the **lineinfile** module to append the line. This line wad added by the lineinfile module to the **/users.txt** file on all managed hosts.

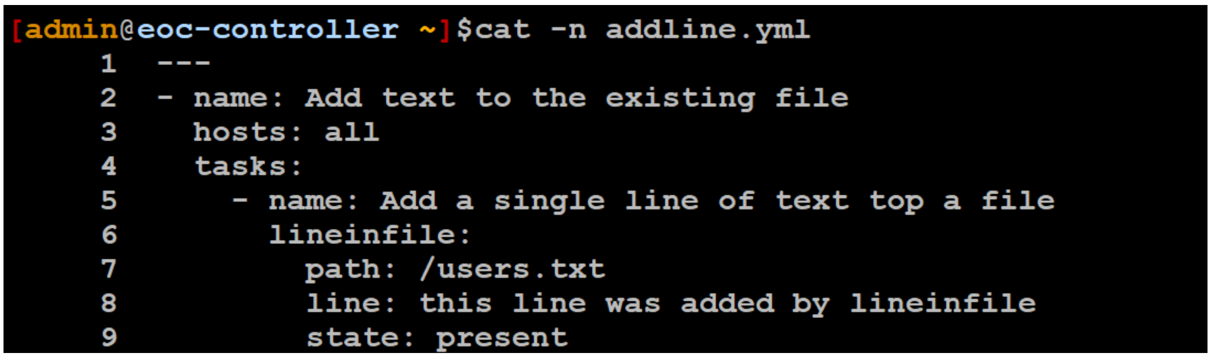


```
---
- name: Add text to the existing file
  hosts: all
  tasks:
    - name: Add a single line of text top a file
      lineinfile:
        path: /users.txt
        line: this line was added by lineinfile
        state: present
```

7.2 Let's view the **addline.yml** manifest file.

```
# cat -n addline.yml
```

Output:

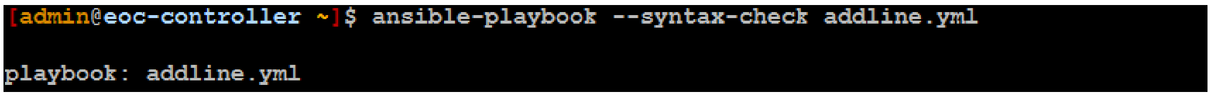


```
[admin@eoc-controller ~]$cat -n addline.yml
     1  ---
     2  - name: Add text to the existing file
     3    hosts: all
     4    tasks:
     5      - name: Add a single line of text top a file
     6        lineinfile:
     7          path: /users.txt
     8          line: this line was added by lineinfile
     9          state: present
```

7.3 Run the ansible-playbook --syntax-check **addline.yml** command to verify its syntax and correct any errors.

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

Output:

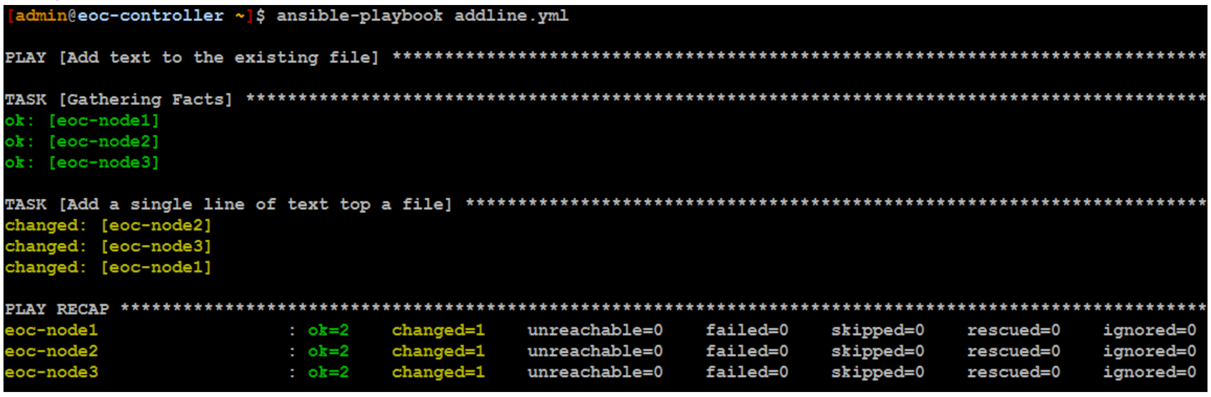


```
[admin@eoc-controller ~]$ ansible-playbook --syntax-check addline.yml

playbook: addline.yml
```

7.4 Run ansible-playbook **addline.yml** to execute the playbook.

```
# ansible-playbook addline.yml
```

Output:

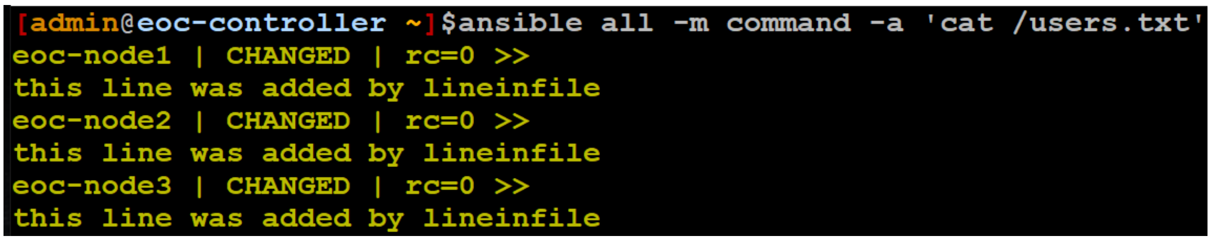


```
[admin@eoc-controller ~]$ ansible-playbook addline.yml

PLAY [Add text to the existing file] ************************************************************************

TASK [Gathering Facts] **************************************************************************************
ok: [eoc-node1]
ok: [eoc-node2]
ok: [eoc-node3]

TASK [Add a single line of text top a file] *****************************************************************
changed: [eoc-node2]
changed: [eoc-node3]
changed: [eoc-node1]

PLAY RECAP **************************************************************************************************
eoc-node1                  : ok=2    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
eoc-node2                  : ok=2    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
eoc-node3                  : ok=2    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
```

7.5 Use the command module with the cat option as the devops user, to verify the content of the **users.txt** file on the managed hosts.

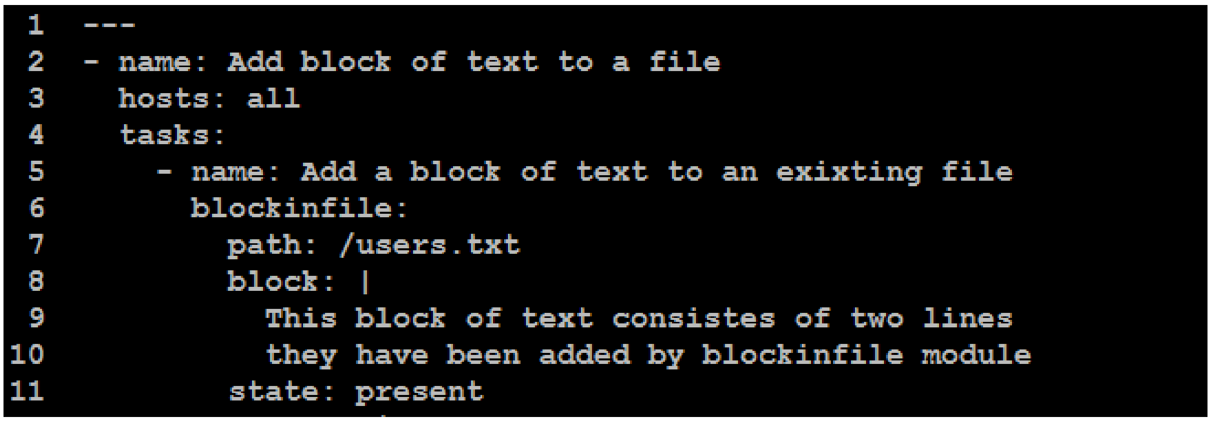
```
# ansible all -m command -a 'cat /users.txt'
```

Output:



```
[admin@eoc-controller ~]$ansible all -m command -a 'cat /users.txt'
eoc-node1 | CHANGED | rc=0 >>
this line was added by lineinfile
eoc-node2 | CHANGED | rc=0 >>
this line was added by lineinfile
eoc-node3 | CHANGED | rc=0 >>
this line was added by lineinfile
```

7.6 Create a playbook called **addblock.yml** in the current working directory. Configure the playbook to use the blockinfile module to append the following block to text to the /users.txt file on all managed hosts.

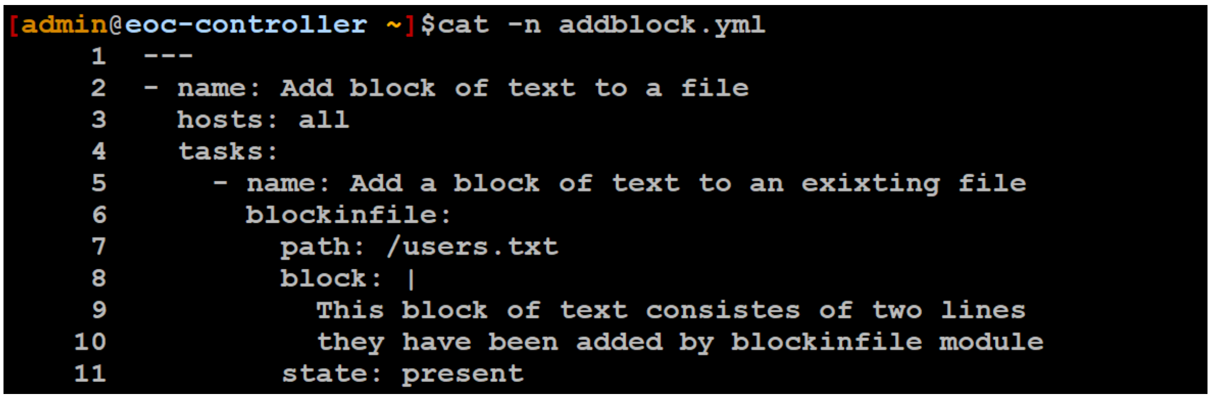


```
---
- name: Add block of text to a file
  hosts: all
  tasks:
    - name: Add a block of text to an exixting file
      blockinfile:
        path: /users.txt
        block: |
          This block of text consistes of two lines
          they have been added by blockinfile module
        state: present
```

7.7 Let's view the yaml manifest file.

```
# cat -n addblock.yml
```

Output:

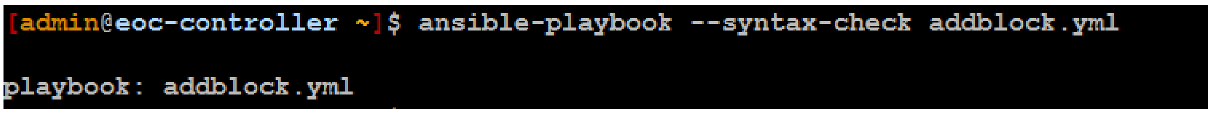


```
[admin@eoc-controller ~]$cat -n addblock.yml
     1  ---
     2  - name: Add block of text to a file
     3    hosts: all
     4    tasks:
     5      - name: Add a block of text to an exixting file
     6        blockinfile:
     7          path: /users.txt
     8          block: |
     9            This block of text consistes of two lines
    10            they have been added by blockinfile module
    11          state: present
```

7.8 Let's run the ansible-playbook --syntax-check **addblock.yml** command to verify its syntax and correct any errors.

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

Output:

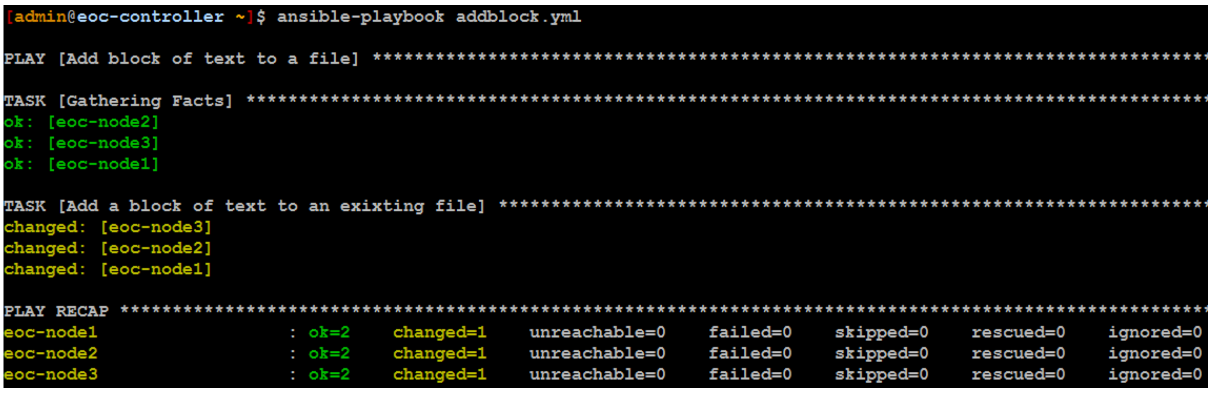


```
[admin@eoc-controller ~]$ ansible-playbook --syntax-check addblock.yml

playbook: addblock.yml
```

7.9 Let's run ansible-playbook **addblock.yml** to execute the playbook.

```
# ansible-playbook addblock.yml
```

Output:

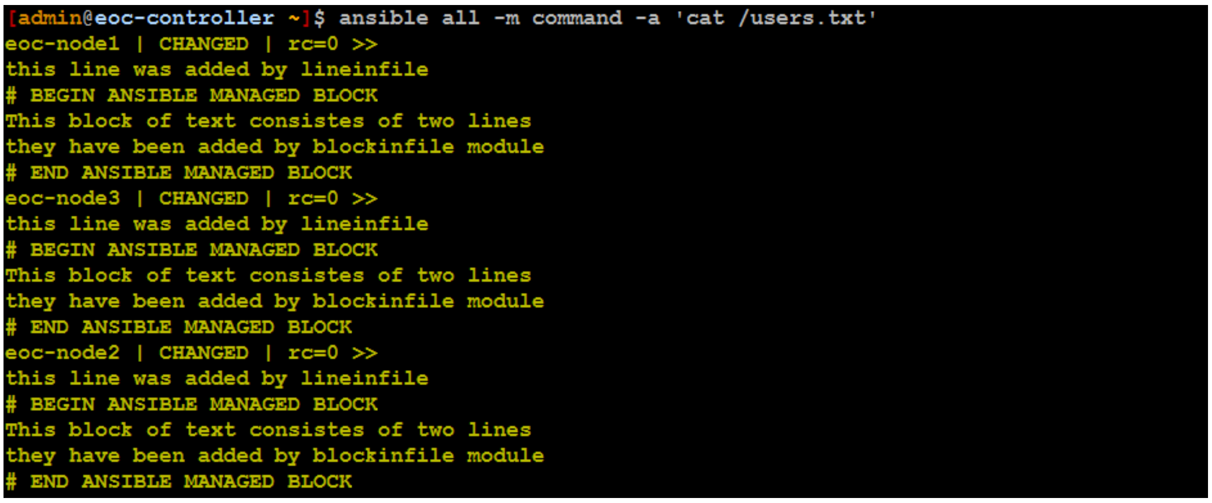


```
[admin@eoc-controller ~]$ ansible-playbook addblock.yml

PLAY [Add block of text to a file] *****************************************************************

TASK [Gathering Facts] *****************************************************************************
ok: [eoc-node2]
ok: [eoc-node3]
ok: [eoc-node1]

TASK [Add a block of text to an exixting file] ******************************************************
changed: [eoc-node3]
changed: [eoc-node2]
changed: [eoc-node1]

PLAY RECAP *****************************************************************************************
eoc-node1                  : ok=2    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
eoc-node2                  : ok=2    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
eoc-node3                  : ok=2    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
```

7.10 Use the command module with the cat command to verify the correct content of the **/users.txt** file to the managed hosts.

```
# ansible all -m command -a 'cat /users.txt'
```

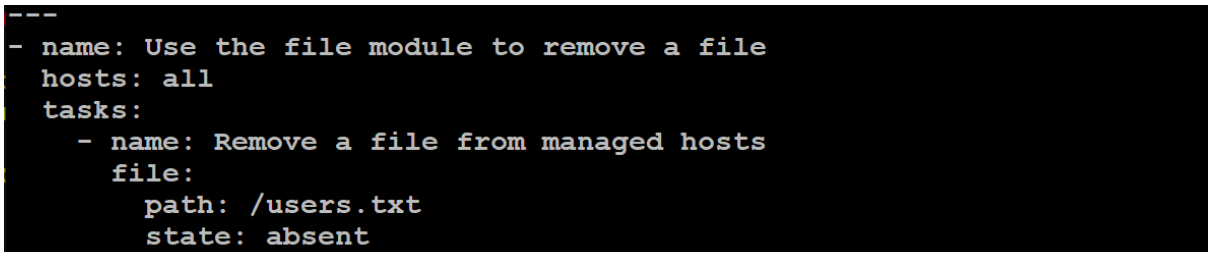
Output:



```
[admin@eoc-controller ~]$ ansible all -m command -a 'cat /users.txt'
eoc-node1 | CHANGED | rc=0 >>
this line was added by lineinfile
# BEGIN ANSIBLE MANAGED BLOCK
This block of text consistes of two lines
they have been added by blockinfile module
# END ANSIBLE MANAGED BLOCK
eoc-node3 | CHANGED | rc=0 >>
this line was added by lineinfile
# BEGIN ANSIBLE MANAGED BLOCK
This block of text consistes of two lines
they have been added by blockinfile module
# END ANSIBLE MANAGED BLOCK
eoc-node2 | CHANGED | rc=0 >>
this line was added by lineinfile
# BEGIN ANSIBLE MANAGED BLOCK
This block of text consistes of two lines
they have been added by blockinfile module
# END ANSIBLE MANAGED BLOCK
```

8. File Module to Remove the User file.

8.1 Create a playbook called **removefile.yml** in the current working directory. Configure the playbook to use the file module to remove the /users.txt file from all managed hosts.

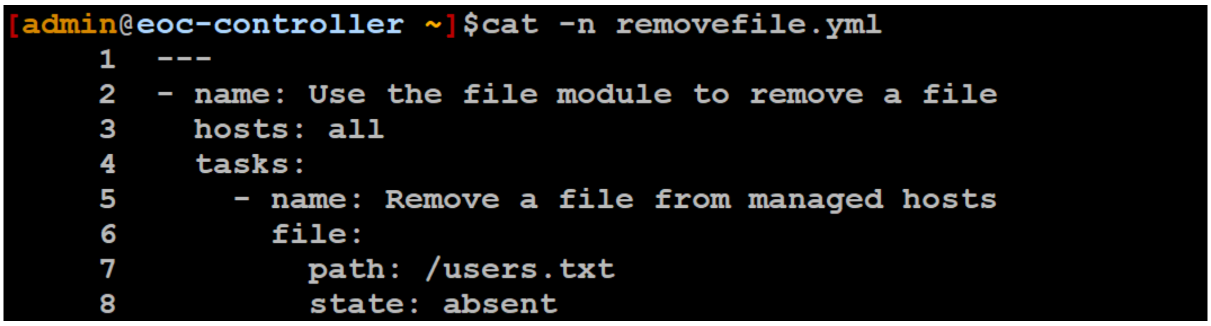


```
---
- name: Use the file module to remove a file
  hosts: all
  tasks:
    - name: Remove a file from managed hosts
      file:
        path: /users.txt
        state: absent
```

8.2 Let's view the yaml manifest file.

```
# cat -n removefile.yml
```

Output:

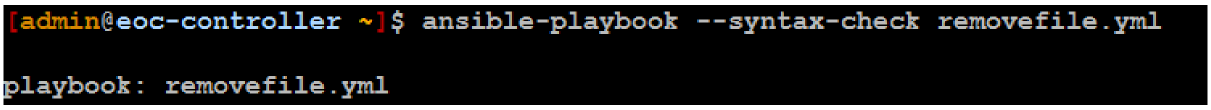


```
[admin@eoc-controller ~]$cat -n removefile.yml
     1  ---
     2  - name: Use the file module to remove a file
     3    hosts: all
     4    tasks:
     5      - name: Remove a file from managed hosts
     6        file:
     7          path: /users.txt
     8          state: absent
```

8.3 Let's run the ansible-playbook --syntax-check **removefile.yml** command to verify its syntax and correct any errors.

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

Output:

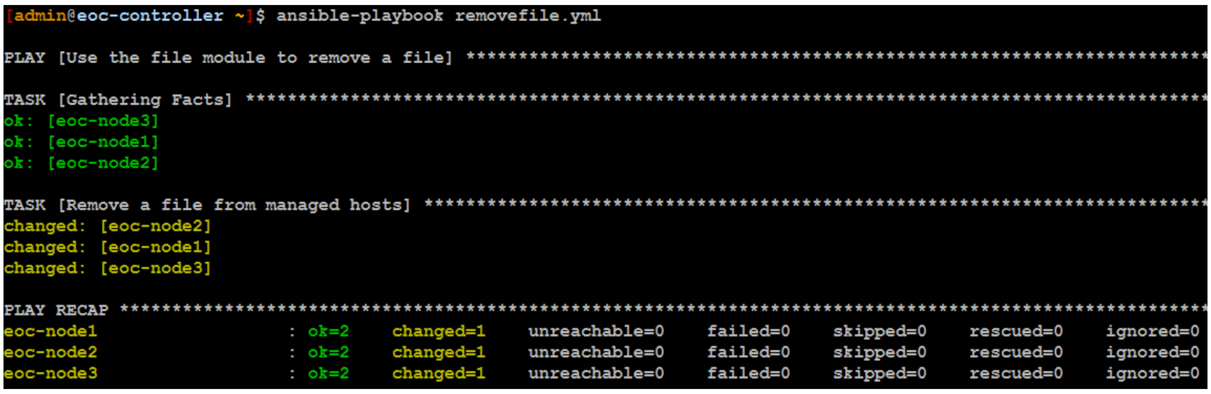


```
[admin@eoc-controller ~]$ ansible-playbook --syntax-check removefile.yml

playbook: removefile.yml
```

8.4 Let's run ansible-playbook **removefile.yml** to execute the playbook.

```
# ansible-playbook removefile.yml
```

Ouptu:

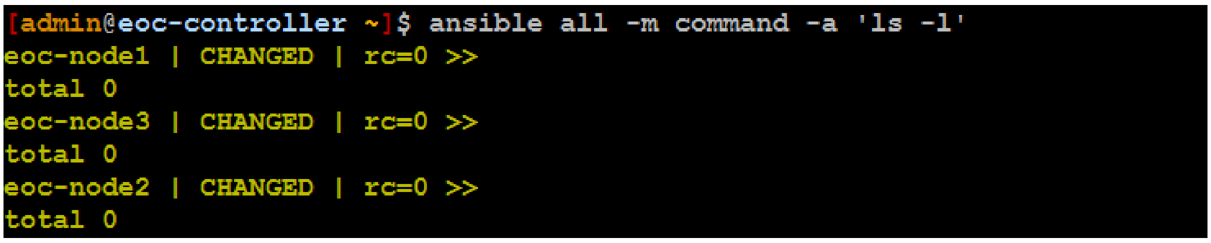


```
[admin@eoc-controller ~]$ ansible-playbook removefile.yml

PLAY [Use the file module to remove a file] ************************************************************

TASK [Gathering Facts] *********************************************************************************
ok: [eoc-node3]
ok: [eoc-node1]
ok: [eoc-node2]

TASK [Remove a file from managed hosts] ****************************************************************
changed: [eoc-node2]
changed: [eoc-node1]
changed: [eoc-node3]

PLAY RECAP *********************************************************************************************
eoc-node1                  : ok=2    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
eoc-node2                  : ok=2    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
eoc-node3                  : ok=2    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
```

8.5 Use an ad hoc command to execute **ls -**l command to confirm that the **users.txt** file no longer exists on the managed host.

```
# ansible all -m command -a 'ls -l'
```

Output:



```
[admin@eoc-controller ~]$ ansible all -m command -a 'ls -l'
eoc-node1 | CHANGED | rc=0 >>
total 0
eoc-node3 | CHANGED | rc=0 >>
total 0
eoc-node2 | CHANGED | rc=0 >>
total 0
```

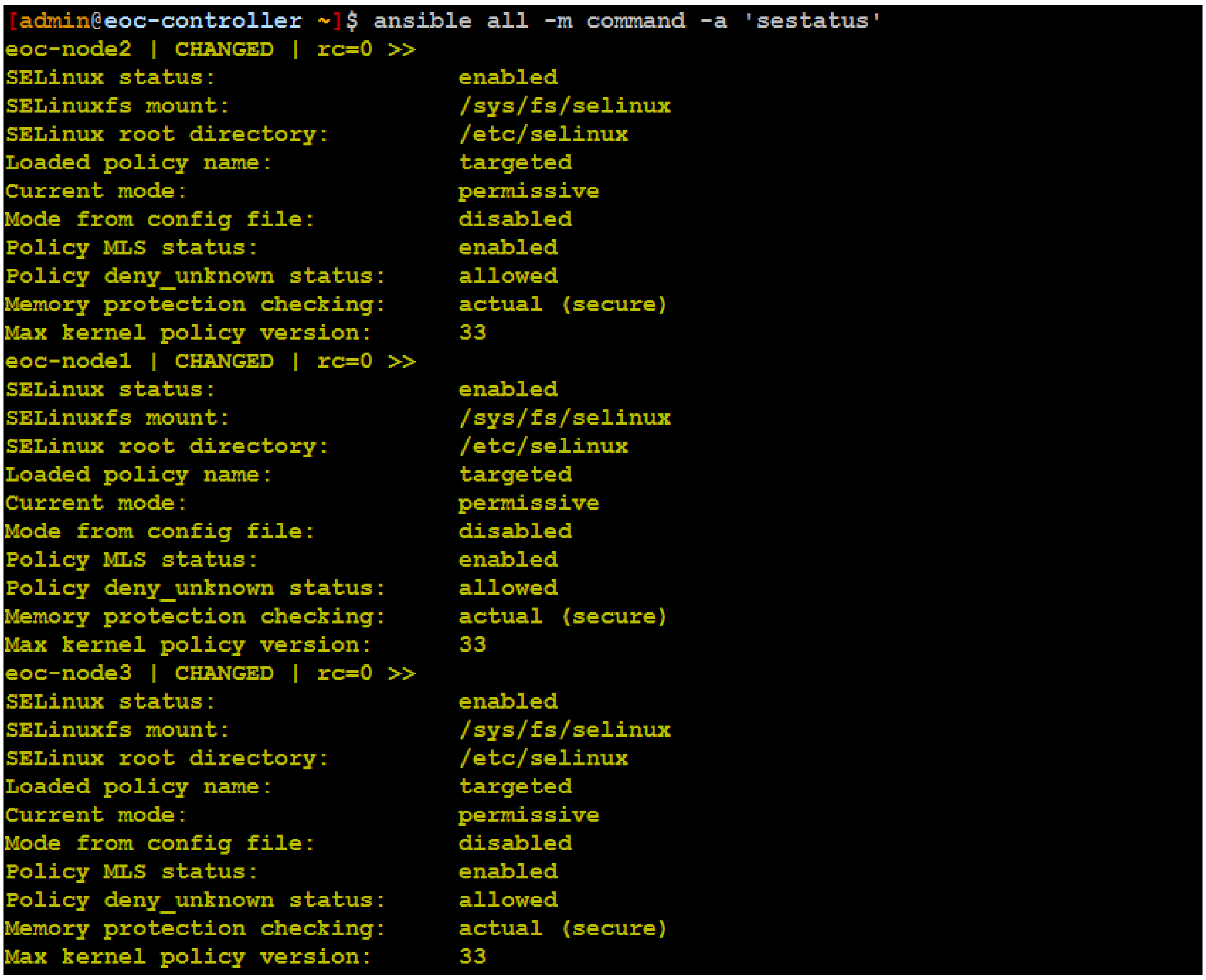
8.6 Let’s disable the **SELinux** for smooth execution of future labs.

```
# ansible all -m command -a 'sed -i 's/enforcing/disabled/g'
/etc/selinux/config'
```

```
# ansible all -m command -a 'setenforce 0'
```

```
# ansible all -m command -a 'sestatus'
```

Output:

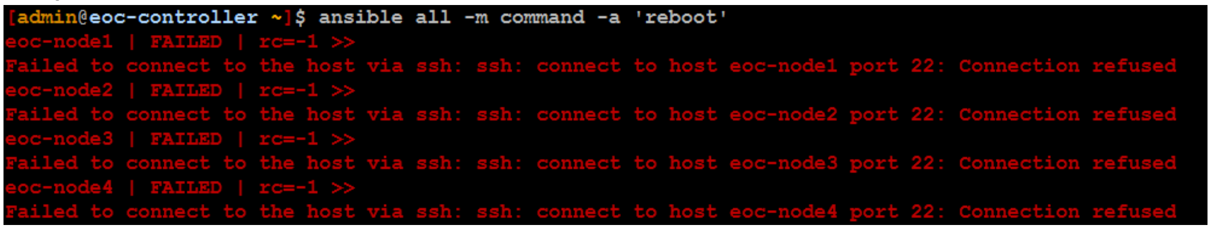


```
[admin@eoc-controller ~]$ ansible all -m command -a 'sestatus'
eoc-node2 | CHANGED | rc=0 >>
SELinux status:                 enabled
SELinuxfs mount:                /sys/fs/selinux
SELinux root directory:         /etc/selinux
Loaded policy name:             targeted
Current mode:                   permissive
Mode from config file:          disabled
Policy MLS status:              enabled
Policy deny_unknown status:     allowed
Memory protection checking:     actual (secure)
Max kernel policy version:      33
eoc-node1 | CHANGED | rc=0 >>
SELinux status:                 enabled
SELinuxfs mount:                /sys/fs/selinux
SELinux root directory:         /etc/selinux
Loaded policy name:             targeted
Current mode:                   permissive
Mode from config file:          disabled
Policy MLS status:              enabled
Policy deny_unknown status:     allowed
Memory protection checking:     actual (secure)
Max kernel policy version:      33
eoc-node3 | CHANGED | rc=0 >>
SELinux status:                 enabled
SELinuxfs mount:                /sys/fs/selinux
SELinux root directory:         /etc/selinux
Loaded policy name:             targeted
Current mode:                   permissive
Mode from config file:          disabled
Policy MLS status:              enabled
Policy deny_unknown status:     allowed
Memory protection checking:     actual (secure)
Max kernel policy version:      33
```

8.7 Reboot the machine to apply the changes.

```
# ansible all -m command -a 'reboot'
```

Output:



```
[admin@eoc-controller ~]$ ansible all -m command -a 'reboot'
eoc-node1 | FAILED | rc=-1 >>
Failed to connect to the host via ssh: ssh: connect to host eoc-node1 port 22: Connection refused
eoc-node2 | FAILED | rc=-1 >>
Failed to connect to the host via ssh: ssh: connect to host eoc-node2 port 22: Connection refused
eoc-node3 | FAILED | rc=-1 >>
Failed to connect to the host via ssh: ssh: connect to host eoc-node3 port 22: Connection refused
eoc-node4 | FAILED | rc=-1 >>
Failed to connect to the host via ssh: ssh: connect to host eoc-node4 port 22: Connection refused
```