🧪Lab 3 — Ad-Hoc Commands in Ansible (0 → Hero Level)

Author: *Sandeep Kumar Sharma*

🕑Learning Objectives

In this lab, you will learn: - What ad-hoc commands are - Why ad-hoc commands are used - How to run 15–20 essential ad-hoc commands - Equivalent package/file/service commands on **Ubuntu** and **Amazon Linux 2** - How to validate your cluster quickly with one-line automation

---

👙Learning Outcomes

After this lab you will: - Confidently execute ad-hoc commands - Manage files, packages, services, and users using ad-hoc commands - Know clear differences between Ubuntu (APT) and Amazon Linux (YUM) - Be ready for writing your first playbook in the next lab

---

🧰What Are Ad-Hoc Commands?

Ad-hoc commands are **one-line Ansible commands** executed directly from the terminal **without creating a playbook**.

These commands use Ansible **modules** to perform quick tasks such as: - Installing packages - Creating files or directories - Working with users - Checking system uptime - Restarting services

They are perfect for quick automation when you don't need a full YAML file.

🕯️ Why Do We Use Ad-Hoc Commands?

- For quick testing
- To validate connectivity between nodes
- To execute small tasks across many servers instantly
- For environment discovery
- For rapid troubleshooting

They save tremendous time and effort.

🖥️Cluster Used

We will use:

```
[dev]
Node1 (Ubuntu/Amazon Linux)
Node2 (Ubuntu/Amazon Linux)
```

All commands below run from the **Master node**, using:

```
ansible dev -m <module> -a "arguments"
```

🪙Essential 15–20 Ad-Hoc Commands

Below are the most necessary and commonly used commands. Each command includes examples for: - **Ubuntu (APT)** - **Amazon Linux 2 (YUM)**

🔗1. Check Node Connectivity (Ping Module)

```
ansible dev -m ping
```

Expected:

```
pong
```

🔗2. Check Uptime

```
ansible dev -m command -a "uptime"
```

🔗3. Create a File

```
ansible dev -m file -a "path=/tmp/demo.txt state=touch"
```

🔗4. Create Directory

```
ansible dev -m file -a "path=/opt/myapp state=directory mode=0755"
```

🔗5. Delete a File

```
ansible dev -m file -a "path=/tmp/demo.txt state=absent"
```

🔗6. Install a Package

Ubuntu

```
ansible dev -m apt -a "name=tree state=present update_cache=yes" -b
```

Amazon Linux 2

```
ansible dev -m yum -a "name=tree state=present" -b
```

🔗7. Remove a Package

Ubuntu

```
ansible dev -m apt -a "name=tree state=absent" -b
```

Amazon Linux 2

```
ansible dev -m yum -a "name=tree state=absent" -b
```

🔗8. Install Multiple Packages

Ubuntu

```
ansible dev -m apt -a "name='git,curl,wget' state=present update_cache=yes" -b
```

Amazon Linux 2

```
ansible dev -m yum -a "name='git,curl,wget' state=present" -b
```

🔗9. Check Disk Usage

```
ansible dev -m command -a "df -h"
```

🔗10. Check Logged-In Users

```
ansible dev -m command -a "who"
```

🔗11. Start a Service (Example: Apache/Nginx)

Ubuntu (apache2)

```
ansible dev -m service -a "name=apache2 state=started" -b
```

Amazon Linux 2 (httpd)

```
ansible dev -m service -a "name=httpd state=started" -b
```

🔗12. Stop a Service

Ubuntu

```
ansible dev -m service -a "name=apache2 state=stopped" -b
```

Amazon Linux 2

```
ansible dev -m service -a "name=httpd state=stopped" -b
```

🔗13. Create a User

```
ansible dev -m user -a "name=testuser state=present" -b
```

🔗14. Delete a User

```
ansible dev -m user -a "name=testuser state=absent" -b
```

🔗15. Copy a File from Master to Nodes

```
ansible dev -m copy -a "src=/tmp/localfile.txt dest=/tmp/remotefile.txt"
```

🔗16. Run Shell Commands

```
ansible dev -m shell -a "echo Hello from $(hostname) > /tmp/msg.txt"
```

🔗17. Retrieve File Content

```
ansible dev -m shell -a "cat /tmp/msg.txt"
```

🔗18. Check Memory Usage

```
ansible dev -m command -a "free -m"
```

🔗19. Reboot Servers

```
ansible dev -m reboot -b
```

🔗20. Gather Only Uptime Using One Command

```
ansible dev -a "uptime"
```

👙Lab Summary

In this lab, we learned: - What ad-hoc commands are - Why they are essential for quick automation - 20+ practical ad-hoc commands with Ubuntu & Amazon Linux differences - Managing files, services, users, and packages using a single line command

Next, we will move to: 👉**Lab 4: Gathering Facts & Exploring ansible_facts**

Get ready for deeper automation! 😢