**Standard Operating Process**

**(SOP) for**

**Wipro Tech Transition\_Unix\_Linux\_ How to setup ansible inventories**

**Prepared By**

****

**Jitendra Kumar Singh, Wipro Technologies**

# 

# Contents

[1.](https://wipro365-my.sharepoint.com/l) Introduction 3

[2.](https://wipro365-my.sharepoint.com/l) Purpose 3

[3.](https://wipro365-my.sharepoint.com/l) Scope 3

[4.](https://wipro365-my.sharepoint.com/l) Responsibilities 3

[5.](https://wipro365-my.sharepoint.com/l) Summary 3

[6.](https://wipro365-my.sharepoint.com/l) Procedure 4

[7.](https://wipro365-my.sharepoint.com/l) Creating a Custom Inventory File 4

[8.](https://wipro365-my.sharepoint.com/l) Nodes listing 5

[9.](https://wipro365-my.sharepoint.com/l) Validate Ansible inventory: 5-6

[10.](https://wipro365-my.sharepoint.com/l) Show disk usage using Ansible 7

# Document Details

|  |  |
| --- | --- |
| Project Name | Wipro Tech Transition\_Unix\_Linux\_ How to setup ansible inventories |
| Account |  |
| Current Version | 1.0.0 |
| List of Contributors | Linux |
| Document Name/ID | Wipro Tech Transition\_Unix\_Linux\_ How to setup ansible inventories |

# Revision History

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Version | Date of Revision | Next Review Date | Description | Author  (Wipro) | Reviewed By (Wipro) | Approved By  (XXXXXX) |
| 1.0.0 | 01-March-2024 | 02-May -2024 | Wipro Tech Transition\_Unix\_Linux\_ How to setup ansible inventories  configuration \_v1.0.docx | Jitendra Kumar Singh | Manish Sahi |  |

**Introduction**

Ansible is a modern configuration management tool that facilitates the task of setting up and maintaining remote servers, with a minimalist design intended to get users up and running quickly. Ansible uses an inventory file to keep track of which hosts are part of your infrastructure, and how to reach them for running commands and playbooks.

**Purpose**

There are multiple ways in which you can set up your Ansible inventory file, depending on your environment and project needs. In this guide, we’ll demonstrate how to create inventory files and organize servers into groups and subgroups, how to set up host variables, and how to use patterns to control the execution of Ansible commands and playbooks per host and per group.

**Scope**

This document encompasses all the activities that a systems admin completes facilitates the task of setting up and maintaining remote servers and Ansible uses an inventory file to keep track of which hosts are part of your infrastructure.

**Responsibilities**

The Systems Administrator Performs the Activity and update this document on timely manner.

**Summary**

This describes setup and configuration of ansible inventories.

**Procedure**

## [Prerequisites](https://www.digitalocean.com/community/tutorials/how-to-set-up-ansible-inventories#prerequisites)

In order follow this guide, you’ll need:

* **One Ansible control node**: an Ubuntu 20.04 machine with Ansible installed and configured to connect to your Ansible hosts using SSH keys. Make sure the control node has a regular user with sudo permissions and a firewall enabled, as explained in our [Initial Server Setup](https://www.digitalocean.com/community/tutorials/initial-server-setup-with-ubuntu-20-04) guide. To set up Ansible, please follow our guide on [How to Install and Configure Ansible on Ubuntu 20.04](https://www.digitalocean.com/community/tutorials/how-to-install-and-configure-ansible-on-ubuntu-20-04).
* **Two or more Ansible Hosts**: two or more remote Ubuntu 20.04 servers.

## [Creating a Custom Inventory File](https://www.digitalocean.com/community/tutorials/how-to-set-up-ansible-inventories#step-1-creating-a-custom-inventory-file)

Upon installation, Ansible creates an inventory file that is typically located at /etc/ansible/hosts. This is the default location used by Ansible when a custom inventory file is not provided with the -i option, during a playbook or command execution.

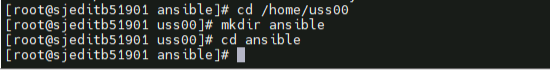
Even though you can use this file without problems, using per-project inventory files is a good practice to avoid mixing servers when executing commands and playbooks. Having per-project inventory files will also facilitate sharing your provisioning setup with collaborators, given you include the inventory file within the project’s code repository.

To get started, access your home folder and create a new directory to hold your Ansible files:

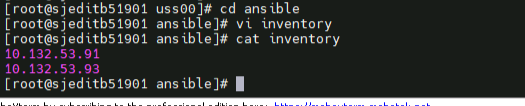
1. cd ~
2. mkdir ansible

Move to that directory and open a new inventory file using your text editor of choice. Here, we’ll use nano:

1. cd ansible
2. nano inventory

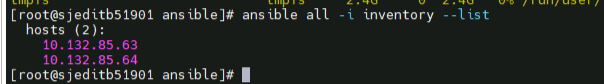


A list of your nodes, with one server per line, is enough for setting up a functional inventory file. Hostnames and IP addresses are interchangeable:



Once you have an inventory file set up, you can use the ansible-inventory command to validate and obtain information about your Ansible inventory:

1. ansible-inventory -i inventory --list



Even though we haven’t set up any groups within our inventory, the output shows 2 distinct groups that are automatically inferred by Ansible: all and ungrouped. As the name suggests, all is used to refer to all servers from your inventory file, no matter how they are organized. The ungrouped group is used to refer to servers that aren’t listed within a group.

### [Running Commands and Playbooks with Custom Inventories](https://www.digitalocean.com/community/tutorials/how-to-set-up-ansible-inventories#running-commands-and-playbooks-with-custom-inventories)

To run Ansible commands with a custom inventory file, use the -i option as follows:

1. ansible all -i inventory -m ping

This would execute the ping module on **all** hosts listed in your custom inventory file.

Similarly, this is how you execute Ansible playbooks with a custom inventory file:

1. ansible-playbook -i inventory playbook.yml

**Note**: For more information on how to connect to nodes, please refer to our [How to Use Ansible](https://www.digitalocean.com/community/cheatsheets/how-to-use-ansible-cheat-sheet-guide) guide, as it demonstrates more connection options.

So far, we’ve seen how to create a basic inventory and how to use it for running commands and playbooks. In the next step, we’ll see how to organize nodes into groups and subgroups.

### Show disk usage using Ansible Shell Module

Create a simple file with name fs\_usage.yml

- name: Check filesystem usage

ansible.builtin.shell: df -khT

register: fs\_usage

- name: Display output

ansible.builtin.debug:

var: fs\_usage.stdout\_lines

In the above example, we use the df -KhT command to show disk usage.

The command output is registered in the fs\_usage variable and displayed in the next task.

### Show disk usage using Ansible Command Module

#ansible all -I inventory -m command -a “df -khT”

