

# PROPOSAL FOR ANSIBLE INSTALLATION AND AUTOMATED DEPLOYMENT

## 1. Introduction

In today's dynamic IT environments, managing and deploying software installations manually can be time-consuming, error-prone, and difficult to scale. As infrastructure grows, the need for consistent, automated, and scalable software deployment becomes critical. This document proposes using Ansible to automate the installation and configuration of software, ensuring faster, repeatable, and reliable deployment processes across various platforms.

## 2. What is Ansible?

**Ansible** is an open-source IT automation tool used to automate software provisioning, configuration management, and application deployment. It allows IT administrators and developers to automate repetitive tasks, such as server setups, software installations, and application deployments, across multiple systems.

### Key Features of Ansible:

- **Agentless:** Unlike other automation tools, Ansible does not require agents to be installed on remote systems. It communicates with nodes using SSH, making it lightweight and easy to set up.
- **Simple YAML Playbooks:** Ansible uses human-readable YAML files (called playbooks) to define the automation tasks, which makes it easy to understand and maintain.
- **Idempotency:** Ansible ensures that tasks are executed in such a way that no matter how many times a task runs, the system state remains consistent and predictable.
- **Cross-Platform:** Ansible works on a wide range of platforms, from Linux servers to cloud infrastructure, making it versatile for a wide range of use cases.

## 3. Why Do We Need Automated Installation?

Manual software installation processes have several challenges, particularly in larger environments:

1. **Time-Consuming:** Manually installing and configuring software across multiple systems can take hours or days, especially when repeated over multiple environments.

2. **Inconsistent Results:** Manual steps are prone to human error, which can lead to inconsistencies in configuration across servers, affecting the stability and performance of the infrastructure.
3. **Scalability:** As infrastructure scales, managing installations manually becomes increasingly complex. Automation allows you to scale your installations easily, ensuring the same configuration is applied consistently across systems.
4. **Configuration Drift:** Over time, differences in manual configurations can lead to a phenomenon known as "configuration drift," where different servers exhibit different behavior due to small configuration changes.

## **Benefits of Automated Installation:**

- **Speed:** Automation tools like Ansible reduce the time required to deploy software by automating repetitive installation tasks.
- **Reliability:** Automated processes ensure consistent and repeatable deployments, reducing errors caused by manual intervention.
- **Scalability:** With Ansible, you can deploy software across thousands of servers in a matter of minutes with a single playbook.
- **Version Control:** Ansible playbooks can be version-controlled (e.g., in Git), providing traceability and rollback options if something goes wrong.
- **Reduced Human Error:** Automation eliminates manual intervention, reducing the likelihood of errors and misconfigurations.

## **4. Ansible for Automated Installation**

Ansible is the ideal tool for automating software installation because of its agentless architecture and ease of use. Here's why it stands out:

- **Platform Independence:** Ansible works across various platforms, including Linux, Windows, and cloud environments, making it an excellent fit for diverse infrastructures.
- **Simplified Playbooks:** By using Ansible playbooks, administrators can define software installation steps in a straightforward, easy-to-read format.
- **Security:** Since Ansible uses SSH for communication, it adds minimal security risks. Additionally, sensitive information such as passwords can be encrypted using **Ansible Vault**.
- **Efficiency:** Ansible ensures tasks are only performed when necessary. If software is already installed and configured, Ansible skips those tasks, ensuring minimal downtime.

## 5. Steps for Installing Ansible and Using it for Automated Software Installation

### 5.1 Installing Ansible

To install Ansible on your local machine or server:

#### 1. Update Your System:

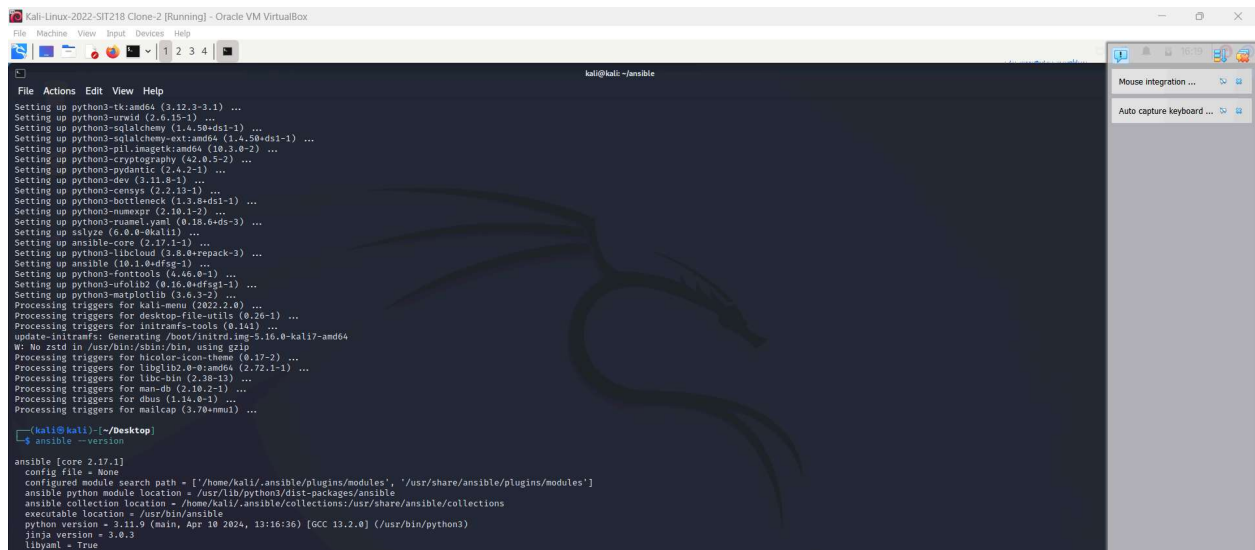
command:-sudo apt update

**2.Install Ansible:** Ansible is available in most Linux distributions' repositories. Install it using:

command:-sudo apt install ansible

**3.Verify the Installation:** Once installed, verify that Ansible is installed by running:

command:-ansible --version



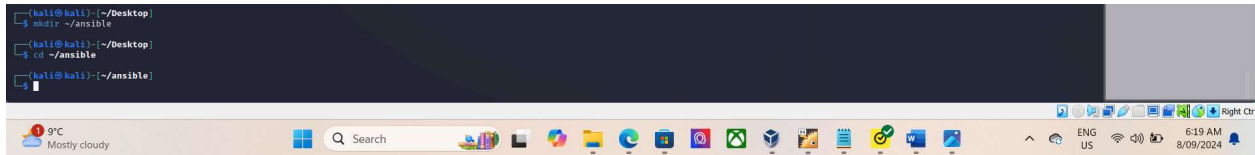
The screenshot shows a Kali Linux terminal window with a Kali Linux logo in the background. The terminal output shows the installation of various Python packages and Ansible. The command `ansible --version` is executed, displaying the following information:

```
ansible [core 2.17.1]
  config file = None
  configured module search path = ['/home/kali/.ansible/plugins/modules', '/usr/share/ansible/plugins/modules']
  ansible python module location = /usr/lib/python3/dist-packages/ansible
  ansible collection location = /home/kali/.ansible/collections:/usr/share/ansible/collections
  executable location = /usr/bin/ansible
  python version = 3.11.9 (main, Apr 10 2024, 13:16:36) [GCC 13.2.0] (/usr/bin/python3)
  jinja version = 3.1.3
  libyaml = True
```

**4. Create a Project Directory:** Create a dedicated directory to store the **inventory file** and the **Ansible playbook**

commands:-`mkdir ~/ansible_project`

`cd ~/ansible_project`



```
(kali@kali) ~/Desktop
$ mkdir ~/ansible
(kali@kali) ~/Desktop
$ cd ~/ansible
(kali@kali) ~/ansible
$
```

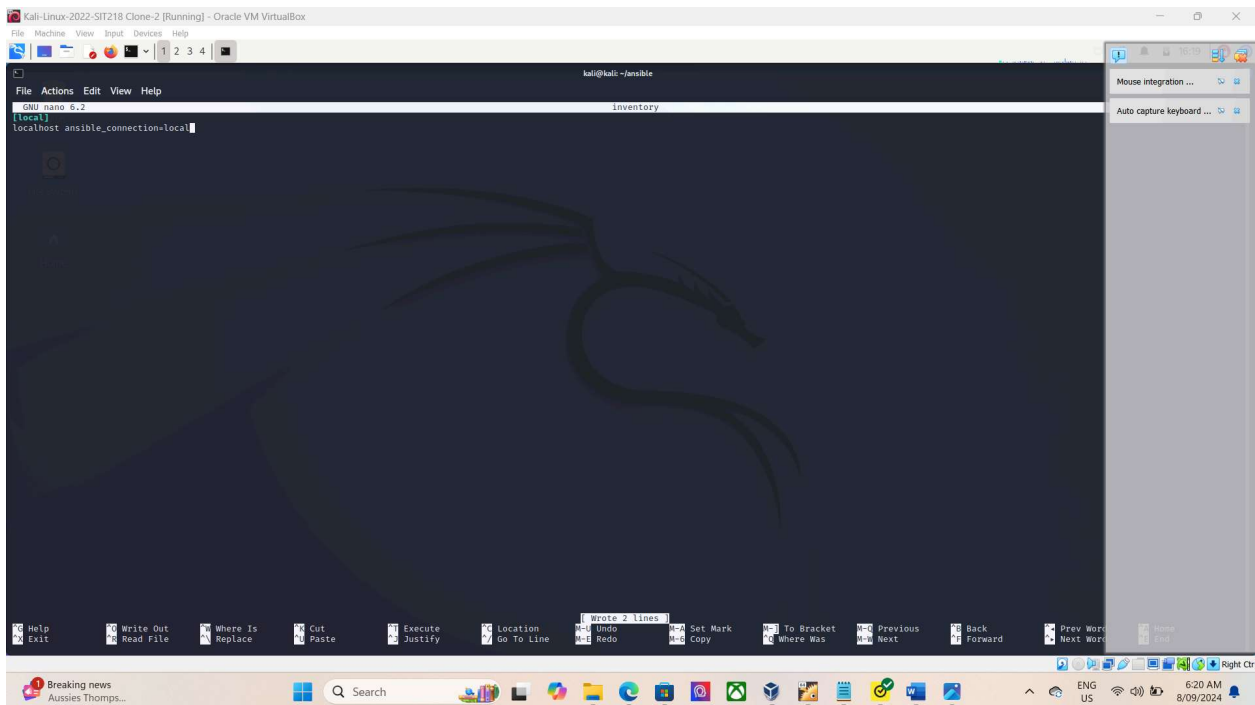
**5..Set Up an Inventory File:** Create an inventory file that defines the hosts where Ansible will execute tasks.

commands:-`nano inventory`

This file can be as simple as:

`[servers]`

`localhost ansible_connection=local`



```
kali@kali - ansible
File Actions Edit View Help
GNU nano 6.2
[local]
localhost ansible_connection=local
```

## 5.2 Automating Software Installation with Ansible

To automate software installation using Ansible, follow these steps:

1. **Create a Playbook:** An Ansible playbook is a YAML file that defines the tasks you want to automate. For example, a playbook to install Elasticsearch, Logstash, Kibana, and Filebeat can look like this:

```
commands:-nano install_elk.yml
```

Adding content in yml file

```
---
```

```
- hosts: local
```

```
  become: yes
```

```
  tasks:
```

```
    - name: Install Java
```

```
      apt:
```

```
        name: openjdk-11-jdk
```

```
        state: present
```

```
    - name: Add Elasticsearch GPG key
```

```
      apt_key:
```

```
        url: https://artifacts.elastic.co/GPG-KEY-elasticsearch
```

```
        state: present
```

```
    - name: Add Elasticsearch repository
```

apt\_repository:

repo: 'deb https://artifacts.elastic.co/packages/7.x/apt stable main'

state: present

filename: 'elastic-7.x'

- name: Install Elasticsearch, Logstash, and Kibana

apt:

name: "{{ item }}"

state: present

loop:

- elasticsearch

- logstash

- kibana

- name: Start and enable Elasticsearch

systemd:

name: elasticsearch

enabled: true

state: started

- name: Start and enable Logstash

systemd:

name: logstash

enabled: true

state: started

- name: Start and enable Kibana

systemd:

name: kibana

enabled: true

state: started

- name: Install Filebeat

apt:

name: filebeat

state: present

- name: Configure Filebeat to connect to Logstash

copy:

dest: /etc/filebeat/filebeat.yml

content: |

output.logstash:

hosts: ["localhost:5044"]

filebeat.inputs:

- type: log

enabled: true

paths:

- /var/log/syslog

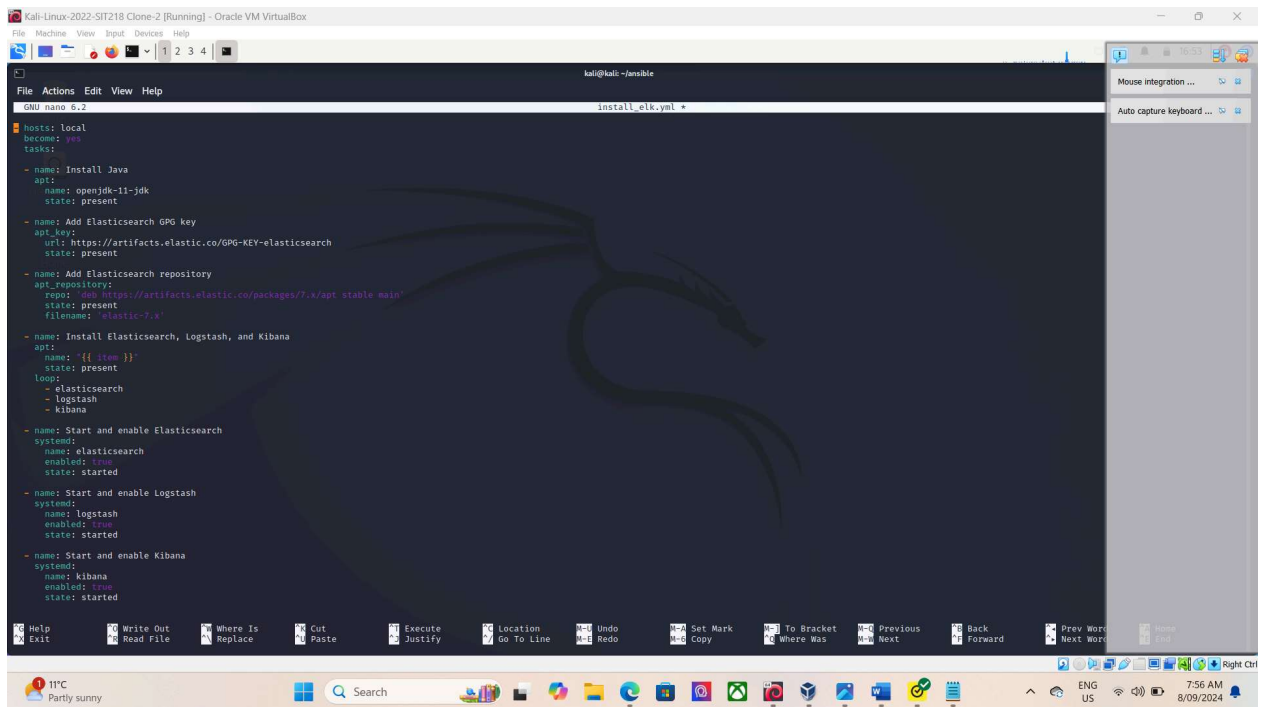
- name: Start and enable Filebeat

systemd:

name: filebeat

enabled: true

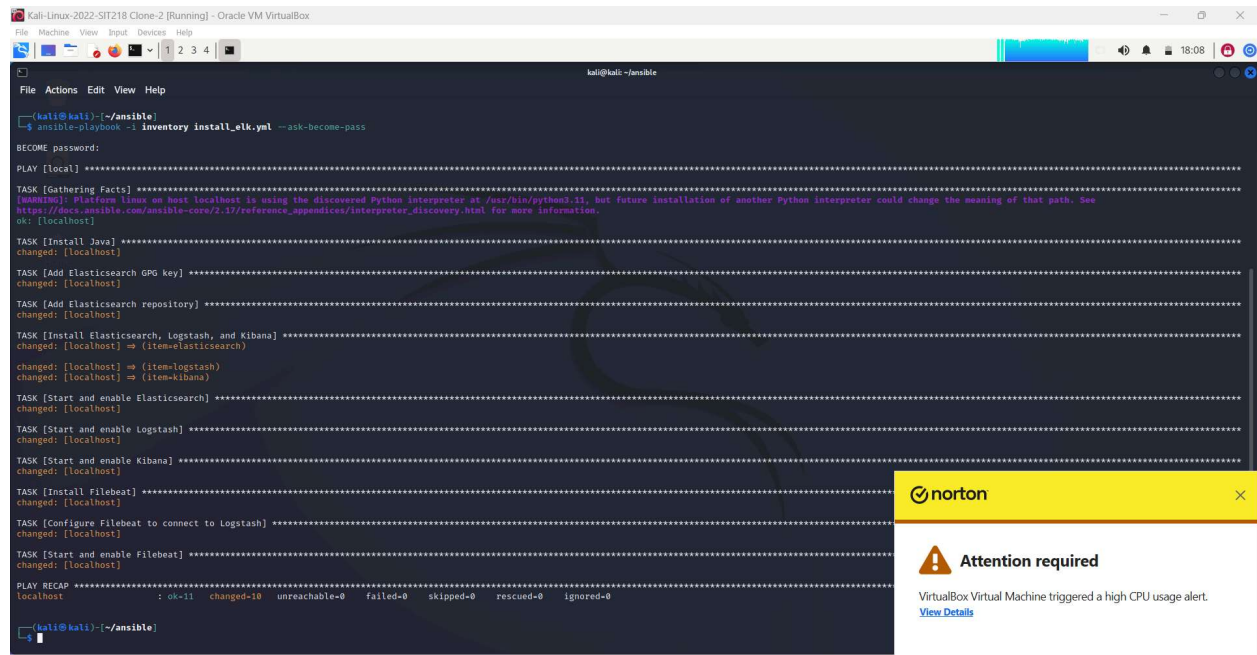
state: started



**2.Run the Ansible Playbook:-**Now that the playbook is created and the inventory file is configured, you can execute the playbook to automatically install and configure the ELK Stack.

command:-`ansible-playbook -i inventory install_elk.yml --ask-become-pass`





```
Kali-Linux-2022-SIT218 Clone-2 [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
kali@kali: ~/ansible
File Actions Edit View Help
[kali@kali]~$ ansible-playbook -i inventory install_elk.yml --ask-become-pass
BECOME password:
PLAY [local]
TASK [Gathering Facts]
[WARNING]: Platform linux on host localhost is using the discovered Python interpreter at /usr/bin/python3.11, but future installation of another Python interpreter could change the meaning of that path. See
https://docs.ansible.com/ansible-core/2.17/reference_appendices/interpreter_discovery.html for more information.
ok: [localhost]
TASK [Install Java]
changed: [localhost]
TASK [Add Elasticsearch GPG key]
changed: [localhost]
TASK [Add Elasticsearch repository]
changed: [localhost]
TASK [Install Elasticsearch, Logstash, and Kibana]
changed: [localhost] => (item=elasticsearch)
changed: [localhost] => (item=logstash)
changed: [localhost] => (item=kibana)
TASK [Start and enable Elasticsearch]
changed: [localhost]
TASK [Start and enable Logstash]
changed: [localhost]
TASK [Start and enable Kibana]
changed: [localhost]
TASK [Install Filebeat]
changed: [localhost]
TASK [Configure Filebeat to connect to Logstash]
changed: [localhost]
TASK [Start and enable Filebeat]
changed: [localhost]
PLAY RECAP
localhost : ok=11 changed=10 unreachable=0 failed=0 skipped=0 rescued=0 ignored=0
[kali@kali]~$
```

## 6. Conclusion

Using Ansible to automate software installations is essential for modern IT environments, where consistency, reliability, and scalability are critical. Ansible's ease of use, agentless architecture, and idempotency make it an ideal choice for automating installations on multiple platforms without manual intervention. Automating the ELK Stack deployment with Ansible ensures that software is installed consistently and quickly, reducing errors and saving time.

By adopting Ansible, we can streamline the software installation process, enhance operational efficiency, and ensure reliable, scalable deployments across various environments.

