



What is Ansible?

Ansible is a software tool that provides simple but powerful automation for cross-platform computer support. It is primarily intended for IT professionals, who use it for application deployment, updates on workstations and servers, cloud provisioning, configuration management, intra-service orchestration, and nearly anything a systems administrator does on a weekly or daily basis. Ansible doesn't depend on agent software and has no additional security infrastructure, so it's easy to deploy.

Because Ansible is all about automation, it requires instructions to accomplish each job. With everything written down in simple script form, it's easy to do version control. The practical result of this is a major contribution to the "infrastructure as code" movement in IT: the idea that the maintenance of server and client infrastructure can and should be treated the same as software development, with repositories of self-documenting, proven, and executable solutions capable of running an organization regardless of staff changes.

While Ansible may be at the forefront of automation, systems administration, and DevOps, it's also useful to everyday users. Ansible allows you to configure not just one computer, but potentially a whole network of computers at once, and using it requires no programming skills. Instructions written for Ansible are human-readable. Whether you're entirely new to computers or an expert, Ansible files are easy to understand.

Advantages of Ansible :

Free: Ansible is an open-source tool.

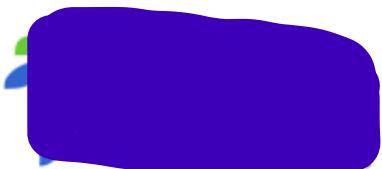
Very simple to set up and use: No special coding skills are necessary to use Ansible's playbooks (more on playbooks later).

Powerful: Ansible lets you model even highly complex IT workflows.

Flexible: You can orchestrate the entire application environment no matter where it's deployed. You can also customize it based on your needs.

Agentless: You don't need to install any other software or firewall ports on the client systems you want to automate. You also don't have to set up a separate management structure.





Efficient: Because you don't need to install any extra software, there's more room for application resources on your server.

Ansible's Features and Capabilities :

1. Configuration Management


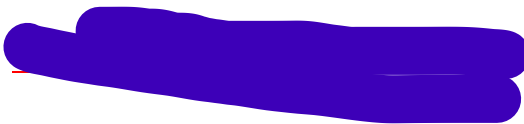
Ansible is designed to be very simple, reliable, and consistent for configuration management. If you're already in IT, you can get up and running with it very quickly. Ansible configurations are simple data descriptions of infrastructure and are both readable by humans and parsable by machines. All you need to start managing systems is a password or an SSH (Secure Socket Shell, a network protocol) key. An example of how easy Ansible makes configuration management: If you want to install an updated version of a specific type of software on all the machines in your enterprise, all you have to do is write out all the IP addresses of the nodes (also called remote hosts) and write an Ansible playbook to install it on all the nodes, then run the playbook from your control machine.

2. Application Deployment

Ansible lets you quickly and easily deploy multitier apps. You won't need to write custom code to automate your systems; you list the tasks required to be done by writing a playbook, and Ansible will figure out how to get your systems to the state you want them to be in. In other words, you won't have to configure the applications on every machine manually. When you run a playbook from your control machine, Ansible uses SSH to communicate with the remote hosts and run all the commands (tasks).

3. Orchestration

As the name suggests, orchestration involves bringing different elements into a beautifully run whole operation—similar to the way a musical conductor brings the notes produced by all the different instruments into a cohesive artistic work. For example, with application deployment, you need to manage not just the front-end and backend services but the databases, networks, storage, and so on. You also need to make sure that all the tasks are handled in the proper order. Ansible uses automated workflows, provisioning, and more to make orchestrating tasks easy. And once you've defined your infrastructure using the Ansible playbooks, you can use that same orchestration wherever you need to, thanks to the portability of Ansible playbooks.



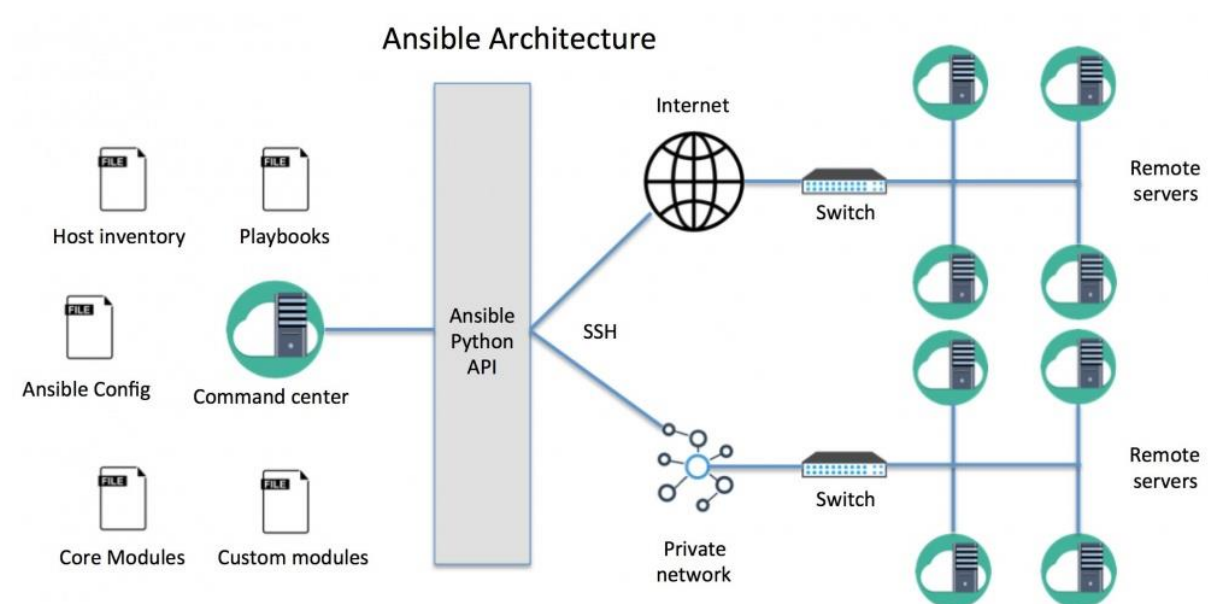
4. Security and Compliance

As with application deployment, sitewide security policies (such as firewall rules or locking down users) can be implemented along with other automated processes. If you configure the security details on the control machine and run the associated playbook, all the remote hosts will automatically be updated with those details. That means you won't need to monitor each machine for security compliance continually manually. And for extra security, an admin's user ID and password aren't retrievable in plain text on Ansible.

5. Cloud Provisioning

The first step in automating your applications' life cycle is automating the provisioning of your infrastructure. With Ansible, you can provision cloud platforms, virtualized hosts, network devices, and bare-metal servers.

Ansible Architecture :



NOTE :

- Default Ansible Configuration file : **/etc/ansible/ansible.cfg**
- Default Ansible Inventory file : **/etc/ansible/hosts**



How Ansible Works?

In Ansible, there are two categories of computers: the control node and managed nodes. The control node is a computer that runs Ansible. There must be at least one control node. A managed node is any device being managed by the control node.

Ansible works by connecting to nodes (clients, servers, or whatever you're configuring) on a network, and then sending a small program called an Ansible module to that node. Ansible executes these modules over SSH and removes them when finished. The only requirement for this interaction is that your Ansible control node has login access to the managed nodes. SSH keys are the most common way to provide access.

Ansible management node is the controlling node, which controls the entire execution of the Playbook. It's the node from which you are running the installation, and the inventory file provides the list of the host where the modules need to be run. The management node makes ssh connection, and then it executes the modules on the host machines and installs the product. It removes the modules once they are installed. So that's how ansible works

Installing Ansible

Control node requirements :

Currently Ansible can be run from any machine with Python 2 (version 2.7) or Python 3 (versions 3.5 and higher) installed. This includes Red Hat, Debian, CentOS, macOS, any of the BSDs, and so on. Windows is not supported for the control node.

Installing Ansible on RHEL, CentOS :



RPMs for currently supported versions of RHEL and CentOS are also available from EPEL repository. First you need to configure EPEL repository in RHEL or CentOS using the below steps :

RHEL/CentOS 7:

```
#yum install https://dl.fedoraproject.org/pub/epel/epel-release-latest-7.noarch.rpm
```

RHEL/CentOS 8:

```
#yum install https://dl.fedoraproject.org/pub/epel/epel-release-latest-8.noarch.rpm
```





Once you are done with setting up EPEL repository, Verify Python is pre installed or not.

```
#python --version
```

If Python is Installed, Now install ansible using YUM.

```
#yum install ansible
```

Verify ansible is installed :

```
#ansible --version
```

