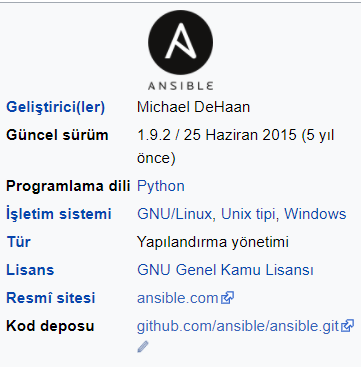
Name of task: ansible researching

Estimated duration: 3 hours

What is Ansible?



Ansible is an open-source automation tool, or platform, used for IT tasks such as configuration management, application deployment, intraservice orchestration, and provisioning. Automation is crucial these days, with IT environments that are too complex and often need to scale too quickly for system administrators and developers to keep up if they had to do everything manually. Automation simplifies complex tasks, not just making developers’ jobs more manageable but allowing them to focus attention on other tasks that add value to an organization. In other words, it frees up time and increases efficiency. And Ansible, as noted above, is rapidly rising to the top in the world of automation tools. Let’s look at some of the reasons for Ansible’s popularity.

Now that we have seen what is Ansible, let us find out the various Benefits of Ansible.

**Benefits 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.

Next, in our path to understanding what ansible is, let us find out the features and capabilities of Ansible.

**Ansible Features**

**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**

Now let’s talk a bit about the pieces that make up the Ansible environment.

**1. Modules**

Modules are like small programs that Ansible pushes out from a control machine to all the nodes or remote hosts. The modules are executed using playbooks (see below), and they control things such as services, packages, and files. Ansible executes all the modules for installing updates or whatever the required task is, and then removes them when finished. Ansible provides more than 450 modules for everyday tasks.

**2. Plugins**

As you probably already know from many other tools and platforms, plugins are extra pieces of code that augment functionality. Ansible comes with a number of its plugins, but you can write your own as well. Action, cache, and callback plugins are three examples.

**3. Inventories**

All the machines you’re using with Ansible (the control machine plus nodes) are listed in a single simple file, along with their IP addresses, databases, servers, and so on. Once you register the inventory, you can assign variables to any of the hosts using a simple text file. You can also pull inventory from sources like EC2 (Amazon Elastic Compute Cloud).

**4. Playbooks**

Ansible playbooks are like instruction manuals for tasks. They are simple files written in YAML, which stands for YAML Ain’t Markup Language, a human-readable data serialization language. Playbooks are really at the heart of what makes Ansible so popular is because they describe the tasks to be done quickly and without the need for the user to know or remember any particular syntax. Not only can they declare configurations, but they can orchestrate the steps of any manually ordered task, and can execute tasks at the same time or at different times.

Each playbook is composed of one or multiple plays, and the goal of a play is to map a group of hosts to well-defined roles, represented by tasks.

**5. APIs**

Various APIs (application programming interfaces) are available so you can extend Ansible’s connection types (meaning more than just SSH for transport), callbacks, and more.

Now that we’ve come this far to understand what Ansible is, let us next look into the Ansible tower.

metin içeren bir resim

Açıklama otomatik olarak oluşturuldu

After this command, the file layout will be formed like this. Here we explain the files a little.

default: The files where the default variables to be used in this role are stored.

files: The files that you want to upload to the remote server are hosted here.

handlers: Usually, variables are stored here that we can control the status of the services. like restart, stop, start

meta: This is the part where information about the created role is given. The information here is read by ansible and provided by the necessary dependencies.

task: We can say the most important file. The tasks to be carried out by the role can be found in different files here.

templates: If there is a configuration or something that you want to send to the server, it is kept.

vars: Here the variables that will be used by the tasks are stored.