# What is Vagrant?

Vagrant is a tool for building and managing virtual machine environments in a single workflow. With an easy-to-use workflow and focus on automation, Vagrant lowers development environment setup time.

Vagrant provides easy to configure, reproducible, and portable work environments built on top of industry-standard technology and controlled by a single consistent workflow to help maximize the productivity and flexibility. To achieve its target, Vagrant stands on the shoulders of giants. Machines are provisioned on top of VirtualBox, VMware, AWS, or any other provider. Then, industry-standard provisioning tools such as shell scripts, Chef, or Puppet, can automatically install and configure software on the virtual machine.

# Vagrant vs. CLI Tools

Virtualization software like VirtualBox and VMware come with command line utilities for managing the lifecycle of machines on their platform. Vagrant uses many of these utilities internally.

The difference between these CLI tools and Vagrant is that Vagrant builds on top of these utilities in a few ways while still providing a consistent workflow. Vagrant supports multiple synced folder types, multiple provisioners to setup the machine, automatic SSH setup, creating HTTP tunnels into your development environment, and more. All of these can be configured using a single simple configuration file.

Vagrant still has a few improvements over manual scripting even if you ignore all the higher-level features Vagrant provides. The command-line utilities provided by virtualization software often change each version or have subtle bugs with workarounds. Vagrant automatically detects the version, uses the correct flags, and can work around known issues.

# Getting Started

## Installation

### Install Vagrant

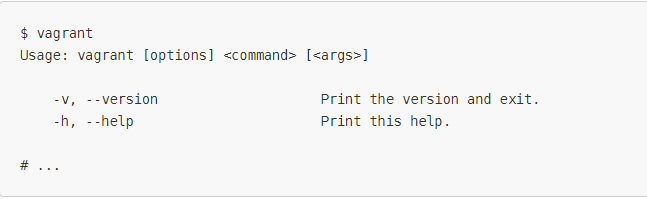
Vagrant must first be installed on the machine you want to run it on.

### Installing Vagrant

To install Vagrant, first find the appropriate package for your system and download it. Vagrant is packaged as an operating-specific package. Run the installer for your system. The installer will automatically add vagrant to your system path so that it is available in terminals.

### Verifying the Installation

After installing Vagrant, verify the installation worked by opening a new command prompt or console, and checking that vagrant is available:

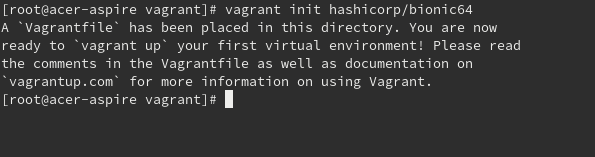


## Project Setup

The first step in configuring any Vagrant project is to create a Vagrantfile. The purpose of the Vagrantfile is twofold:

1. Mark the root directory of your project. Many of the configuration options in Vagrant are relative to this root directory.
2. Describe the kind of machine and resources you need to run your project, as well as what software to install and how you want to access it.

Vagrant has a built-in command for initializing a directory for usage with Vagrant: vagrant init.



This will place a Vagrantfile in your current directory. You can look at the Vagrantfile if you want, it is filled with comments and examples.

You can also run vagrant init in a pre-existing directory to set up Vagrant for an existing project.

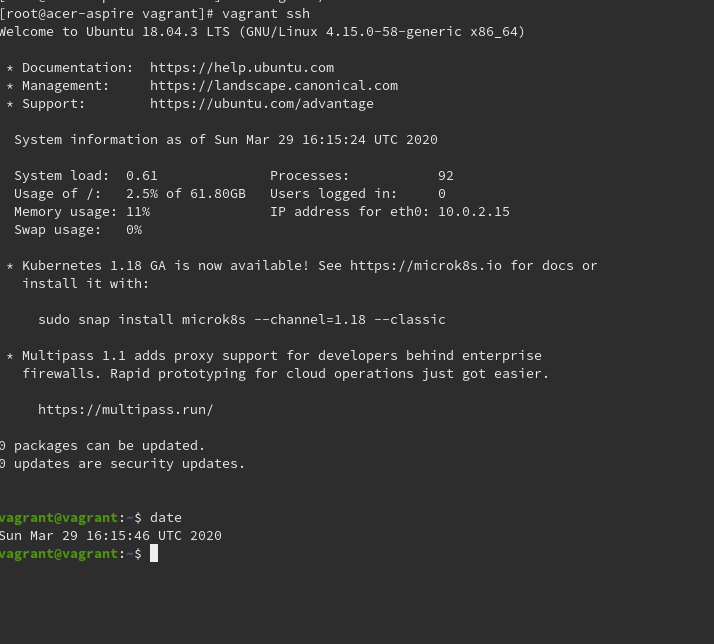
The Vagrantfile is meant to be committed to version control with your project, if you use version control. This way, every person working with that project can benefit from Vagrant without any upfront work.

## Up And SSH

To boot your first Vagrant environment. Run the following from your terminal:



In less than a minute, this command will finish, and you will have a virtual machine running Ubuntu. You will not actually see anything though, since Vagrant runs the virtual machine without a UI. To prove that it is running, you can SSH into the machine. This command will drop you into a full-fledged SSH session.



When you are done fiddling around with the machine, run vagrant halt destroy back on your host machine, and Vagrant will terminate the use of any resources by the virtual machine.

