**What is Terraform**

Terraform is an infrastructure as code tool that lets you define both cloud and on-prem resources in human-readable configuration files that you can version, reuse, and share. You can then use a consistent workflow to provision and manage all of your infrastructure throughout its lifecycle. Terraform can manage low-level components like compute, storage, and networking resources, as well as high-level components like DNS entries and SaaS features.

**The Benefits of Using Terraform**

Traditionally, without IaC, the cloud infrastructure was managed manually. This was not the most efficient way and was prone to manual errors. Consistency was a challenge, especially when many servers and clusters were to be managed.

Configuration management tools existed, but the infrastructure’s support was very limited. Application code development and management of that code has evolved a lot with versioning tools, DevOps toolchains, development practices, and delivery methodologies. Terraform introduced the concept of [Infrastructure as Code](https://spacelift.io/blog/infrastructure-as-code), which by default leverages these advantages to managing infrastructure.

Let us take a look at a few of the benefits Terraform has to offer.

|  |  |
| --- | --- |
| **Benefits** | **Description** |
| Consistency | With infrastructure being managed via code, it becomes very easy to version and track changes. Since cloud resource provision happens logically, we can rely on its consistency to create a scaled set of infrastructure. |
| Automation | Terraform workflow manages [the lifecycle](https://spacelift.io/blog/terraform-resource-lifecycle) of cloud resources – from their creation till they are destroyed or decommissioned. This provides an opportunity to enable end-to-end automation right from the infrastructure layer. Automation workflows also assist in strengthening deployment strategies. Read more about: [Terraform automation](https://spacelift.io/blog/terraform-automation). |
| Less Risk | Using Terraform to develop infrastructure as code provides validation beforehand. I also isolates manual efforts and errors associated with it. Thus once developed, infrastructure provisioning and de-provisioning cycles execute identically. |
| Modular and DRY | Infrastructure can be developed in a modular way so that it can be reused across multiple projects. This approach also enables organizations to ingrain their security and governance practices around infrastructure resources, thus offsetting the initial efforts spent every time a new project kicks off. |

**Note:** New versions of Terraform will be placed under the BUSL license, but everything created before version 1.5.x stays open-source. [OpenTofu](https://opentofu.org/" \t "_blank) is an open-source version of Terraform that will expand on Terraform’s existing concepts and offerings. It is a viable alternative to HashiCorp’s Terraform, being forked from Terraform version 1.5.6. OpenTofu retained all the features and functionalities that had made Terraform popular among developers while also introducing improvements and enhancements. OpenTofu works with your existing Terraform state file, so you won’t have any issues when you are migrating to it.

**Terraform Features**

Some of the key features of Terraform are

|  |  |
| --- | --- |
| **Feature** | **Description** |
| Declarative | Terraform uses Hashicorp Configuration Langauge which provides a declarative syntax to develop infrastructure as code. The HCL configuration language helps declare the target state of cloud resources to be provisioned. |
| Cloud Agnostic | Terraform is a great tool to automate [multi-cloud deployments](https://spacelift.io/blog/multi-cloud-infrastructure-strategy). Its modular architecture enables working with multiple well-known cloud vendors simultaneously. |
| Ecosystem | The provider and module ecosystem of Terraform is well established. Certified modules and [providers](https://spacelift.io/blog/terraform-providers) are available on Terraform Registry to be readily used. Customers can create and publish their own modules both publicly and privately. |
| Extendible | Terraform can be extended to support lesser-known or private data centers. |
| Agentless | Terraform works with the programmatic access provided by cloud provider APIs. Thus there is no need to install agents. |

With that in mind, let us get our hands dirty by installing and setting up our Terraform environment.

**Installation and Setup**

**Terraform**

Terraform comes in two forms: open source and Terraform Cloud. [Terraform Cloud](https://spacelift.io/blog/what-is-terraform-cloud) is an online hosted platform that provides a UI for automation of provisioning tasks and management, but the code still needs to be manually developed.

You can also check out [Spacelift](https://spacelift.io/" \t "_blank), a sophisticated and compliant infrastructure delivery platform that makes Terraform management easy.