Terraform is an infrastructure as code tool that allows you to define and manage your infrastructure resources in a declarative manner. It uses providers to interact with various cloud providers or services. Each provider has its own set of resources and data sources that can be managed using Terraform.

Terraform provider versioning is a mechanism that allows you to specify the version of a provider you want to use in your Terraform configuration. It ensures that your infrastructure is consistently deployed using a specific version of the provider, which helps maintain stability and reproducibility.

Provider versions are defined using a version constraint in the `required_providers` block within your Terraform configuration. The version constraint can be an exact version, a range of versions, or other version constraints supported by Terraform's version constraint syntax.

Here's an example of specifying a provider version constraint in a Terraform configuration:

```hcl
terraform {
  required_providers {
    aws = ">= 3.0, < 4.0"
  }
}

provider "aws" {
  region  = "us-west-2"
  version = "~> 3.0"
}

resource "aws_instance" "example" {
  # ...
}
```

In this example, the configuration specifies that it requires the AWS provider version 3.0 or later but less than version 4.0. The `~> 3.0` constraint indicates that any version greater than or equal to 3.0 but less than 4.0 is acceptable. This allows you to automatically use the latest patch releases within the 3.x version range.

By specifying the provider version constraint in this way, you can ensure that your Terraform configuration is compatible with the expected provider version. It helps prevent unexpected changes or breaking updates when new versions of the provider are released.

To apply this configuration, you would run the `terraform init` command, which initializes the working directory and downloads the specified provider version or installs it from the Terraform registry.

Overall, provider versioning in Terraform allows you to explicitly control the versions of providers used in your infrastructure deployments, ensuring consistency and avoiding unexpected behavior due to provider updates.