Lab 12 – Creating a Full VPC Using Terraform Registry Module (Beginner-Friendly)

Creator: Sandeep Kumar Sharma

Learning Objectives

- • Understand how to use Terraform Registry modules for networking.
- • Learn how to deploy a complete AWS VPC using an official public module.
- • Learn VPC basics (CIDR, subnets, route tables) without writing huge Terraform code.
- • Deploy a VPC with public and private subnets.
- • Validate the created networking components in AWS Console.

Learning Outcome

By the end of this lab, the learner will be able to: - Use the official Terraform AWS VPC module confidently. - Create a multi-subnet VPC using a few lines of Terraform. - Understand the basic networking structure created by the module. - Prepare for advanced labs involving EC2, ALB, and RDS inside custom VPCs.

Concept Explanation (Natural Style)

Creating a VPC manually with Terraform requires: - VPC resource - Subnets - Route tables - Routes - Internet Gateway - NAT Gateway - Associations - DHCP options - Security

This easily becomes **100+ lines of code**.

But companies prefer **speed + best practices**. So instead of writing everything ourselves, we often use the **official AWS VPC module** available on the Terraform Registry:

https://registry.terraform.io/modules/terraform-aws-modules/vpc/aws/latest

This module: - Creates VPC - Creates subnets - Creates route tables - Configures IGW and NAT - Creates default security configurations

…all for you.

It's the easiest, fastest, and safest way to set up networking.

⭐Part 1: Create Project Structure

```
mkdir terraform-lab12-vpc-module
cd terraform-lab12-vpc-module
```

Create file:

```
touch main.tf
```

⭐Part 2: Write Code Using Official VPC Module

Open **main.tf** and add:

```
terraform {
  required_providers {
    aws = {
      source  = "hashicorp/aws"
      version = "~> 5.0"
    }
  }
}

provider "aws" {
  region = "ap-south-1"
}

# ---------------------------------------------
# VPC Module from Terraform Registry
# ---------------------------------------------
module "vpc" {
  source  = "terraform-aws-modules/vpc/aws"
  version = "5.1.2"   # latest stable version

  name = "lab12-vpc"
  cidr = "10.0.0.0/16"

  azs = [
    "ap-south-1a",
    "ap-south-1b",
    "ap-south-1c"
  ]
  public_subnets = [
    "10.0.1.0/24",
    "10.0.2.0/24",
    "10.0.3.0/24"
  ]

  private_subnets = [
    "10.0.4.0/24",
    "10.0.5.0/24",
    "10.0.6.0/24"
  ]

  enable_nat_gateway = true
  single_nat_gateway = true

  tags = {
    Project = "Terraform-Lab12"
  }
}

# Output VPC ID
output "vpc_id" {
  value = module.vpc.vpc_id
}

# Output Public Subnets
output "public_subnets" {
  value = module.vpc.public_subnets
}
```

⭐Explanation of Important Inputs

🦝CIDR Block

Defines VPC range:

```
10.0.0.0/16
```

Means VPC can have 65,000+ IPs.

🦝Availability Zones

We used 3 AZs to create a highly available setup.

🦝Public Subnets

Used for: - Load balancers - Public EC2 - Bastion hosts

🦝Private Subnets

Used for: - Databases - Application servers - Internal services

🦝NAT Gateway

Allows only **private subnet instances** to access the internet securely.

The module automatically handles route tables and associations.

⭐Part 3: Initialize Terraform

```
terraform init
```

Terraform will: - Download VPC module - Download AWS provider - Set up backend locally

---

⭐Part 4: Plan

```
terraform plan
```

Terraform will show: - VPC creation - Public & private subnets - NAT gateway - Route tables - IGW

⭐Part 5: Apply

```
terraform apply
```

Type **yes**.

This will take **2–3 minutes**, especially for NAT creation.

You will get:

```
vpc_id = "vpc-0abc12345xyz"
public_subnets = ["subnet-123", "subnet-456", "subnet-789"]
```

⭐Part 6: Validate in AWS Console

Go to: - **VPC → Your VPCs** → lab12-vpc - **Subnets** → You will see 6 subnets - **Route Tables** → Public + Private - **Internet Gateway** → Attached - **NAT Gateway** → Created

Everything was created automatically.

⭐Part 7: Destroy (Optional)

```
terraform destroy
```

Type **yes**.

This removes: - VPC - Subnets - NAT - IGW - Route tables

Summary

In this lab, you learned: - How to use the official Terraform AWS VPC module. - How to create a full networking architecture with minimal code. - How Terraform Registry modules simplify complex resources. - How to deploy VPC, public/private subnets, NAT, and routes.

In the next labs, we will connect EC2 inside this VPC using subnet and security groups.

End of Lab 12