Lab 11: Terraform State File – Understanding State, Changes, and Backup

Author: Dr. Sandeep Kumar Sharma
Level: Beginner to Intermediate
Platform: Ubuntu Linux + Microsoft Azure
Prerequisite: Lab 1 to Lab 10

Learning Objective

Participants will learn:

- What a Terraform state file is
- When the state file is created
- What information the state file contains
- How Terraform uses the state file
- How Terraform detects changes
- What happens when configuration changes
- What the backup file is
- The role of the backup file

Learning Outcome

After completing this lab, participants will be able to:

- Understand Terraform internal working
- Explain state file importance
- Track infrastructure using state
- Understand change detection mechanism
- Understand backup and recovery concept

Concept Explanation

What is Terraform State File?

Terraform state file is a file where Terraform **stores the real-world infrastructure information**.

File name:

```
terraform.tfstate
```

Terraform uses this file to: - Know what resources are created - Track resource IDs - Track relationships between resources - Map code with real cloud infrastructure

---

When is State File Created?

The state file is created:

- After first `terraform apply`
- When Terraform creates any resource

If no resources are created → no state file.

What Does the State File Contain?

The state file contains:

- Resource names
- Resource types
- Azure resource IDs
- Resource attributes
- Dependencies
- Metadata
- Provider information
- Infrastructure mapping

Terraform state = **Infrastructure database**

Why State File is Important

Terraform uses the state file to:

- Detect changes
- Plan updates
- Avoid duplication
- Track resource lifecycle
- Perform delete/update safely

Without state file → Terraform becomes blind.

Change Detection Concept

Scenario

Already created VM:

```
name = "Sandeep-machine"
```

Now change configuration to:

```
name = "Sandeep-machine-new"
```

What Terraform Does Internally

Step-by-step process:

1. Terraform reads configuration file
2. Terraform reads state file
3. Terraform compares:
4. Code configuration
5. State file data
6. Terraform detects difference
7. Terraform creates execution plan

Result

Terraform decides:

- Old VM name ≠ New VM name

So Terraform plans:

- Destroy old VM
- Create new VM

Because **VM name is immutable property** in Azure.

Practical Demonstration

Step 1: Existing VM

```
name = "Sandeep-machine"
```

Apply:

```
terraform apply
```

State file created:

```
terraform.tfstate
```

Step 2: Modify Configuration

Change:

```
name = "Sandeep-machine"
```

to

```
name = "Sandeep-machine-renamed"
```

Step 3: Plan

```
terraform plan
```

Terraform output will show:

- Resource will be destroyed
- Resource will be created

Step 4: Apply

```
terraform apply
```

Terraform will:

- Delete old VM
- Create new VM

Backup File

What is Backup File?

Terraform automatically creates a backup file when state changes.

File name:

```
terraform.tfstate.backup
```

When Backup File is Created?

Backup file is created when:

- State file is modified
- State file is updated
- New apply operation happens

What Backup File Contains?

- Previous state data
- Previous infrastructure mapping
- Old resource records

Role of Backup File

Backup file is used for:

- Recovery
- Rollback reference
- Disaster recovery
- State restoration
- Debugging

State File Flow

Terraform workflow:

```
Configuration File
        ↓
terraform.tfstate
        ↓
Real Infrastructure
```

Important Facts

- Terraform state is source of truth
- Terraform does not query Azure for full data every time
- Terraform trusts state file
- State corruption = infrastructure risk

Commands Related to State

```
terraform state list
```

```
terraform state show <resource>
```

Summary Understanding

- Code = Desired state
- State file = Current state
- Terraform = Reconciliation engine

Terraform compares:

Desired state vs Current state

and performs actions to match them.

Files Created

```
terraform.tfstate
terraform.tfstate.backup
```