Using Terraform to create the Terraform state bucket

Initializing the Terraform state bucket and DynamoDB lock tables

```
cd ~/environment/tfekscode/tf-setup
```

Initialize the Terraform backend environment:

```
terraform init
```

The command will create a hidden directory in your file system called ".terraform" and it will download all the resource providers that are needed, for our environment this includes the aws, external and null providers:

```
Initializing the backend...

Initializing provider plugins...
- Reusing previous version of hashicorp/null from the dependency lock file
- Reusing previous version of hashicorp/external from the dependency lock file
- Reusing previous version of hashicorp/random from the dependency lock file
- Reusing previous version of hashicorp/kubernetes from the dependency lock file
- Reusing previous version of hashicorp/helm from the dependency lock file
- Reusing previous version of hashicorp/local from the dependency lock file
- Reusing previous version of hashicorp/aws from the dependency lock file
- Using hashicorp/local v2.1.0 from the shared cache directory
- Using hashicorp/aws v4.63.0 from the shared cache directory
- Using hashicorp/null v3.1.1 from the shared cache directory
- Using hashicorp/external v2.1.1 from the shared cache directory
- Using hashicorp/random v3.5.1 from the shared cache directory
- Using hashicorp/kubernetes v2.17.0 from the shared cache directory
- Using hashicorp/helm v2.4.1 from the shared cache directory
```

Amazon EKS Terraform Workshop

1. Workshop Introduction
2. Introduction to Kubernetes
3. Start the Workshop
4. Terraform Primer (Optional)
5. Creating a private EKS Cluster with Terraform
   1. EKS Terraform Scenario
      - EKS multi-part Build with Terraform
   2. Initial Setup for Terraform state
      - **Using Terraform to create the Terraform state bucket**
      - Terraform files explanation
   3. Setting up the Network
      - Using Terraform to create VPC and other Network related resources
      - Terraform files explanation
   4. IAM Roles and Policies for EKS
      - Using Terraform to create the IAM Roles

```
Terraform has been successfully initialized!

You may now begin working with Terraform. Try running "terraform plan" to see
any changes that are required for your infrastructure. All Terraform commands
should now work.

If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.
data.aws_region.current: Reading...
data.aws_availability_zones.az: Reading...
data.aws_caller_identity.current: Reading...
data.aws_region.current: Read complete after 0s [id=eu-west-1]
data.aws_availability_zones.az: Read complete after 0s [id=eu-west-1]
data.aws_caller_identity.current: Read complete after 0s [id=440018911661]
```

Validate the Terraform code

```
1   terraform validate
```

```
Success! The configuration is valid.
```

Plan the deployment:

```
1   terraform plan -out tfplan
```

```
Terraform used the selected providers to generate the following execution plan. Resource actions are indicated w
  + create

Terraform will perform the following actions:

  # aws_dynamodb_table.terraform_locks[0] will be created
  + resource "aws_dynamodb_table" "terraform_locks" {
      + arn              = (known after apply)
      + billing_mode     = "PAY_PER_REQUEST"
      + hash_key         = "LockID"
```

Amazon EKS Terraform Workshop

1. Workshop Introduction
2. Introduction to Kubernetes
3. Start the Workshop
4. Terraform Primer (Optional)
5. Creating a private EKS Cluster with Terraform
   1. EKS Terraform Scenario
      - EKS multi-part Build with Terraform
   2. Initial Setup for Terraform state
      - **Using Terraform to create the Terraform state bucket**
      - Terraform files explanation
   3. Setting up the Network
      - Using Terraform to create VPC and other Network related resources
      - Terraform files explanation
   4. IAM Roles and Policies for EKS
      - Using Terraform to create the IAM Roles


```
    + id                = (known after apply)
    + name              = "terraform_locks_tf-setup"
    + read_capacity     = (known after apply)
    + stream_arn        = (known after apply)
    + stream_label      = (known after apply)
    + stream_view_type  = (known after apply)
    + tags_all          = (known after apply)
    + write_capacity    = (known after apply)

    + attribute {
        + name = "LockID"
        + type = "S"
      }

    + point_in_time_recovery {
        + enabled = true
      }

    + server_side_encryption {
        + enabled     = true
        + kms_key_arn = (known after apply)
      }
  }

# aws_dynamodb_table.terraform_locks[1] will be created
+ resource "aws_dynamodb_table" "terraform_locks" {
    + arn              = (known after apply)
    + billing_mode     = "PAY_PER_REQUEST"
    + hash_key         = "LockID"
    + id               = (known after apply)
    + name             = "terraform_locks_net"
    + read_capacity    = (known after apply)
    + stream_arn       = (known after apply)
    + stream_label     = (known after apply)
    + stream_view_type = (known after apply)
    + tags_all         = (known after apply)
    + write_capacity   = (known after apply)

    + attribute {
        + name = "LockID"
        + type = "S"
      }

    + point_in_time_recovery {
        + enabled = true
      }

    + server_side_encryption {
```

Amazon EKS Terraform Workshop

1. Workshop Introduction
2. Introduction to Kubernetes
3. Start the Workshop
4. Terraform Primer (Optional)
5. Creating a private EKS Cluster with Terraform
   1. EKS Terraform Scenario
      - EKS multi-part Build with Terraform
   2. Initial Setup for Terraform state
      - **Using Terraform to create the Terraform state bucket**
      - Terraform files explanation
   3. Setting up the Network
      - Using Terraform to create VPC and other Network related resources
      - Terraform files explanation
   4. IAM Roles and Policies for EKS
      - Using Terraform to create the IAM Roles


```
        + enabled     = true
        + kms_key_arn = (known after apply)
      }
  }

# aws_dynamodb_table.terraform_locks[2] will be created
+ resource "aws_dynamodb_table" "terraform_locks" {
    + arn              = (known after apply)
    + billing_mode     = "PAY_PER_REQUEST"
    + hash_key         = "LockID"
    + id               = (known after apply)
    + name             = "terraform_locks_iam"
    + read_capacity    = (known after apply)
    + stream_arn       = (known after apply)
    + stream_label     = (known after apply)
    + stream_view_type = (known after apply)
    + tags_all         = (known after apply)
    + write_capacity   = (known after apply)

    + attribute {
        + name = "LockID"
        + type = "S"
      }

    + point_in_time_recovery {
        + enabled = true
      }

    + server_side_encryption {
        + enabled     = true
        + kms_key_arn = (known after apply)
      }
  }

# aws_dynamodb_table.terraform_locks[3] will be created
+ resource "aws_dynamodb_table" "terraform_locks" {
    + arn              = (known after apply)
    + billing_mode     = "PAY_PER_REQUEST"
    + hash_key         = "LockID"
    + id               = (known after apply)
    + name             = "terraform_locks_c9net"
    + read_capacity    = (known after apply)
    + stream_arn       = (known after apply)
    + stream_label     = (known after apply)
    + stream_view_type = (known after apply)
    + tags_all         = (known after apply)
    + write_capacity   = (known after apply)
```

Amazon EKS Terraform Workshop

1. Workshop Introduction
2. Introduction to Kubernetes
3. Start the Workshop
4. Terraform Primer (Optional)
5. Creating a private EKS Cluster with Terraform
   1. EKS Terraform Scenario
      - EKS multi-part Build with Terraform
   2. Initial Setup for Terraform state
      - **Using Terraform to create the Terraform state bucket**
      - Terraform files explanation
   3. Setting up the Network
      - Using Terraform to create VPC and other Network related resources
      - Terraform files explanation
   4. IAM Roles and Policies for EKS
      - Using Terraform to create the IAM Roles

```
    + attribute {
        + name = "LockID"
        + type = "S"
      }

    + point_in_time_recovery {
        + enabled = true
      }

    + server_side_encryption {
        + enabled     = true
        + kms_key_arn = (known after apply)
      }
  }

# aws_dynamodb_table.terraform_locks[4] will be created
+ resource "aws_dynamodb_table" "terraform_locks" {
    + arn              = (known after apply)
    + billing_mode     = "PAY_PER_REQUEST"
    + hash_key         = "LockID"
    + id               = (known after apply)
    + name             = "terraform_locks_cluster"
    + read_capacity    = (known after apply)
    + stream_arn       = (known after apply)
    + stream_label     = (known after apply)
    + stream_view_type = (known after apply)
    + tags_all         = (known after apply)
    + write_capacity   = (known after apply)

    + attribute {
        + name = "LockID"
        + type = "S"
      }

    + point_in_time_recovery {
        + enabled = true
      }

    + server_side_encryption {
        + enabled     = true
        + kms_key_arn = (known after apply)
      }
  }

# aws_dynamodb_table.terraform_locks[5] will be created
+ resource "aws_dynamodb_table" "terraform_locks" {
    + arn              = (known after apply)
    + billing_mode     = "PAY_PER_REQUEST"
```

Amazon EKS Terraform Workshop

1. Workshop Introduction
2. Introduction to Kubernetes
3. Start the Workshop
4. Terraform Primer (Optional)
5. Creating a private EKS Cluster with Terraform
   1. EKS Terraform Scenario
      - EKS multi-part Build with Terraform
   2. Initial Setup for Terraform state
      - **Using Terraform to create the Terraform state bucket**
      - Terraform files explanation
   3. Setting up the Network
      - Using Terraform to create VPC and other Network related resources
      - Terraform files explanation
   4. IAM Roles and Policies for EKS
      - Using Terraform to create the IAM Roles


```
      + hash_key         = "LockID"
      + id               = (known after apply)
      + name             = "terraform_locks_nodeg"
      + read_capacity    = (known after apply)
      + stream_arn       = (known after apply)
      + stream_label     = (known after apply)
      + stream_view_type = (known after apply)
      + tags_all         = (known after apply)
      + write_capacity   = (known after apply)

      + attribute {
          + name = "LockID"
          + type = "S"
        }

      + point_in_time_recovery {
          + enabled = true
        }

      + server_side_encryption {
          + enabled     = true
          + kms_key_arn = (known after apply)
        }
    }

# aws_dynamodb_table.terraform_locks[6] will be created
+ resource "aws_dynamodb_table" "terraform_locks" {
      + arn              = (known after apply)
      + billing_mode     = "PAY_PER_REQUEST"
      + hash_key         = "LockID"
      + id               = (known after apply)
      + name             = "terraform_locks_cicd"
      + read_capacity    = (known after apply)
      + stream_arn       = (known after apply)
      + stream_label     = (known after apply)
      + stream_view_type = (known after apply)
      + tags_all         = (known after apply)
      + write_capacity   = (known after apply)

      + attribute {
          + name = "LockID"
          + type = "S"
        }

      + point_in_time_recovery {
          + enabled = true
        }
```

Amazon EKS Terraform Workshop

1. Workshop Introduction
2. Introduction to Kubernetes
3. Start the Workshop
4. Terraform Primer (Optional)
5. Creating a private EKS Cluster with Terraform
   1. EKS Terraform Scenario
      - EKS multi-part Build with Terraform
   2. Initial Setup for Terraform state
      - **Using Terraform to create the Terraform state bucket**
      - Terraform files explanation
   3. Setting up the Network
      - Using Terraform to create VPC and other Network related resources
      - Terraform files explanation
   4. IAM Roles and Policies for EKS
      - Using Terraform to create the IAM Roles


```
      + server_side_encryption {
          + enabled     = true
          + kms_key_arn = (known after apply)
        }
    }

  # aws_dynamodb_table.terraform_locks[7] will be created
  + resource "aws_dynamodb_table" "terraform_locks" {
      + arn              = (known after apply)
      + billing_mode     = "PAY_PER_REQUEST"
      + hash_key         = "LockID"
      + id               = (known after apply)
      + name             = "terraform_locks_sampleapp"
      + read_capacity    = (known after apply)
      + stream_arn       = (known after apply)
      + stream_label     = (known after apply)
      + stream_view_type = (known after apply)
      + tags_all         = (known after apply)
      + write_capacity   = (known after apply)

      + attribute {
          + name = "LockID"
          + type = "S"
        }

      + point_in_time_recovery {
          + enabled = true
        }

      + server_side_encryption {
          + enabled     = true
          + kms_key_arn = (known after apply)
        }
    }

  # aws_dynamodb_table.terraform_locks[8] will be created
  + resource "aws_dynamodb_table" "terraform_locks" {
      + arn              = (known after apply)
      + billing_mode     = "PAY_PER_REQUEST"
      + hash_key         = "LockID"
      + id               = (known after apply)
      + name             = "terraform_locks_fargate"
      + read_capacity    = (known after apply)
      + stream_arn       = (known after apply)
      + stream_label     = (known after apply)
      + stream_view_type = (known after apply)
      + tags_all         = (known after apply)
      + write_capacity   = (known after apply)
```

Amazon EKS Terraform Workshop

1. Workshop Introduction
2. Introduction to Kubernetes
3. Start the Workshop
4. Terraform Primer (Optional)
5. Creating a private EKS Cluster with Terraform
   1. EKS Terraform Scenario
      - EKS multi-part Build with Terraform
   2. Initial Setup for Terraform state
      - **Using Terraform to create the Terraform state bucket**
      - Terraform files explanation
   3. Setting up the Network
      - Using Terraform to create VPC and other Network related resources
      - Terraform files explanation
   4. IAM Roles and Policies for EKS
      - Using Terraform to create the IAM Roles


```
    + attribute {
        + name = "LockID"
        + type = "S"
      }

    + point_in_time_recovery {
        + enabled = true
      }

    + server_side_encryption {
        + enabled     = true
        + kms_key_arn = (known after apply)
      }
  }

# aws_kms_key.ekskey will be created
+ resource "aws_kms_key" "ekskey" {
    + arn                                = (known after apply)
    + bypass_policy_lockout_safety_check = false
    + customer_master_key_spec           = "SYMMETRIC_DEFAULT"
    + description                        = "EKS KMS Key 2 mycluster1"
    + enable_key_rotation                = false
    + id                                 = (known after apply)
    + is_enabled                         = true
    + key_id                             = (known after apply)
    + key_usage                          = "ENCRYPT_DECRYPT"
    + multi_region                       = (known after apply)
    + policy                             = (known after apply)
    + tags_all                           = (known after apply)
  }

# aws_s3_bucket.terraform_state will be created
+ resource "aws_s3_bucket" "terraform_state" {
    + acceleration_status         = (known after apply)
    + acl                         = (known after apply)
    + arn                         = (known after apply)
    + bucket                      = (known after apply)
    + bucket_domain_name          = (known after apply)
    + bucket_prefix               = (known after apply)
    + bucket_regional_domain_name = (known after apply)
    + force_destroy               = true
    + hosted_zone_id              = (known after apply)
    + id                          = (known after apply)
    + object_lock_enabled         = (known after apply)
    + policy                      = (known after apply)
    + region                      = (known after apply)
    + request_payer               = (known after apply)
```

1. Workshop Introduction
2. Introduction to Kubernetes
3. Start the Workshop
4. Terraform Primer (Optional)
5. Creating a private EKS Cluster with Terraform
1. EKS Terraform Scenario
EKS multi-part Build with Terraform
2. Initial Setup for Terraform state
Using Terraform to create the Terraform state bucket
Terraform files explanation
3. Setting up the Network
Using Terraform to create VPC and other Network related resources
Terraform files explanation
4. IAM Roles and Policies for EKS
Using Terraform to create the IAM Roles

```
    + tags_all                    = (known after apply)
    + website_domain              = (known after apply)
    + website_endpoint            = (known after apply)
  }

# aws_s3_bucket_public_access_block.pub_block_state will be created
+ resource "aws_s3_bucket_public_access_block" "pub_block_state" {
    + block_public_acls       = true
    + block_public_policy     = true
    + bucket                  = (known after apply)
    + id                      = (known after apply)
    + ignore_public_acls      = true
    + restrict_public_buckets = true
  }

# aws_s3_bucket_server_side_encryption_configuration.terraform_state will be created
+ resource "aws_s3_bucket_server_side_encryption_configuration" "terraform_state" {
    + bucket = (known after apply)
    + id     = (known after apply)

    + rule {
        + bucket_key_enabled = false

        + apply_server_side_encryption_by_default {
            + kms_master_key_id = (known after apply)
            + sse_algorithm     = "aws:kms"
          }
      }
  }

# aws_s3_bucket_versioning.terraform_state will be created
+ resource "aws_s3_bucket_versioning" "terraform_state" {
    + bucket = (known after apply)
    + id     = (known after apply)

    + versioning_configuration {
        + mfa_delete = (known after apply)
        + status     = "Enabled"
      }
  }

# aws_ssm_parameter.tf-eks-buck-name will be created
+ resource "aws_ssm_parameter" "tf-eks-buck-name" {
    + arn            = (known after apply)
    + data_type      = (known after apply)
    + description    = "The Terraform State bucket name for the workshop"
    + id             = (known after apply)
    + insecure_value = (known after apply)
```

1. Workshop Introduction
2. Introduction to Kubernetes
3. Start the Workshop
4. Terraform Primer (Optional)
5. Creating a private EKS Cluster with Terraform
  1. EKS Terraform Scenario
    EKS multi-part Build with Terraform
  2. Initial Setup for Terraform state
    Using Terraform to create the Terraform state bucket
    Terraform files explanation
  3. Setting up the Network
    Using Terraform to create VPC and other Network related resources
    Terraform files explanation
  4. IAM Roles and Policies for EKS
    Using Terraform to create the IAM Roles

```
    + key_id          = (known after apply)
    + name            = "/workshop/tf-eks/bucket-name"
    + tags            = {
        + "workshop" = "tf-eks-workshop"
      }
    + tags_all        = {
        + "workshop" = "tf-eks-workshop"
      }
    + tier            = (known after apply)
    + type            = "String"
    + value           = (sensitive value)
    + version         = (known after apply)
  }

# aws_ssm_parameter.tf-eks-cluster-name will be created
+ resource "aws_ssm_parameter" "tf-eks-cluster-name" {
    + arn             = (known after apply)
    + data_type       = (known after apply)
    + description     = "The EKS cluster name for the workshop"
    + id              = (known after apply)
    + insecure_value  = (known after apply)
    + key_id          = (known after apply)
    + name            = "/workshop/tf-eks/cluster-name"
    + tags            = {
        + "workshop" = "tf-eks-workshop"
      }
    + tags_all        = {
        + "workshop" = "tf-eks-workshop"
      }
    + tier            = (known after apply)
    + type            = "String"
    + value           = (sensitive value)
    + version         = (known after apply)
  }

# aws_ssm_parameter.tf-eks-id will be created
+ resource "aws_ssm_parameter" "tf-eks-id" {
    + arn             = (known after apply)
    + data_type       = (known after apply)
    + description     = "The unique id for the workshop"
    + id              = (known after apply)
    + insecure_value  = (known after apply)
    + key_id          = (known after apply)
    + name            = "/workshop/tf-eks/id"
    + tags            = {
        + "workshop" = "tf-eks-workshop"
      }
    + tags_all        = {
```

1. Workshop Introduction
2. Introduction to Kubernetes
3. Start the Workshop
4. Terraform Primer (Optional)
5. Creating a private EKS Cluster with Terraform
1. EKS Terraform Scenario
EKS multi-part Build with Terraform
2. Initial Setup for Terraform state
Using Terraform to create the Terraform state bucket
Terraform files explanation
3. Setting up the Network
Using Terraform to create VPC and other Network related resources
Terraform files explanation
4. IAM Roles and Policies for EKS
Using Terraform to create the IAM Roles

```
      + "workshop" = "tf-eks-workshop"
    }
  + tier           = (known after apply)
  + type           = "String"
  + value          = (sensitive value)
  + version        = (known after apply)
}

# aws_ssm_parameter.tf-eks-keyarn will be created
+ resource "aws_ssm_parameter" "tf-eks-keyarn" {
  + arn            = (known after apply)
  + data_type      = (known after apply)
  + description    = "The keyid for the workshop"
  + id             = (known after apply)
  + insecure_value = (known after apply)
  + key_id         = (known after apply)
  + name           = "/workshop/tf-eks/keyarn"
  + tags           = {
      + "workshop" = "tf-eks-workshop"
    }
  + tags_all       = {
      + "workshop" = "tf-eks-workshop"
    }
  + tier           = (known after apply)
  + type           = "String"
  + value          = (sensitive value)
  + version        = (known after apply)
}

# aws_ssm_parameter.tf-eks-keyid will be created
+ resource "aws_ssm_parameter" "tf-eks-keyid" {
  + arn            = (known after apply)
  + data_type      = (known after apply)
  + description    = "The keyid for the workshop"
  + id             = (known after apply)
  + insecure_value = (known after apply)
  + key_id         = (known after apply)
  + name           = "/workshop/tf-eks/keyid"
  + tags           = {
      + "workshop" = "tf-eks-workshop"
    }
  + tags_all       = {
      + "workshop" = "tf-eks-workshop"
    }
  + tier           = (known after apply)
  + type           = "String"
  + value          = (sensitive value)
  + version        = (known after apply)
```

Amazon EKS Terraform Workshop

1. Workshop Introduction
2. Introduction to Kubernetes
3. Start the Workshop
4. Terraform Primer (Optional)
5. Creating a private EKS Cluster with Terraform
   1. EKS Terraform Scenario
      - EKS multi-part Build with Terraform
   2. Initial Setup for Terraform state
      - **Using Terraform to create the Terraform state bucket**
      - Terraform files explanation
   3. Setting up the Network
      - Using Terraform to create VPC and other Network related resources
      - Terraform files explanation
   4. IAM Roles and Policies for EKS
      - Using Terraform to create the IAM Roles


```
  }

# aws_ssm_parameter.tf-eks-region will be created
+ resource "aws_ssm_parameter" "tf-eks-region" {
    + arn            = (known after apply)
    + data_type      = (known after apply)
    + description    = "The region for the workshop"
    + id             = (known after apply)
    + insecure_value = (known after apply)
    + key_id         = (known after apply)
    + name           = "/workshop/tf-eks/region"
    + tags           = {
        + "workshop" = "tf-eks-workshop"
      }
    + tags_all       = {
        + "workshop" = "tf-eks-workshop"
      }
    + tier           = (known after apply)
    + type           = "String"
    + value          = (sensitive value)
    + version        = (known after apply)
  }

# aws_ssm_parameter.tf-eks-version will be created
+ resource "aws_ssm_parameter" "tf-eks-version" {
    + arn            = (known after apply)
    + data_type      = (known after apply)
    + description    = "The EKS Version"
    + id             = (known after apply)
    + insecure_value = (known after apply)
    + key_id         = (known after apply)
    + name           = "/workshop/tf-eks/eks-version"
    + tags           = {
        + "workshop" = "tf-eks-workshop"
      }
    + tags_all       = {
        + "workshop" = "tf-eks-workshop"
      }
    + tier           = (known after apply)
    + type           = "String"
    + value          = (sensitive value)
    + version        = (known after apply)
  }

# null_resource.gen_backend will be created
+ resource "null_resource" "gen_backend" {
    + id       = (known after apply)
    + triggers = (known after apply)
```

Amazon EKS Terraform Workshop

1. Workshop Introduction
2. Introduction to Kubernetes
3. Start the Workshop
4. Terraform Primer (Optional)
5. Creating a private EKS Cluster with Terraform
   1. EKS Terraform Scenario
      - EKS multi-part Build with Terraform
   2. Initial Setup for Terraform state
      - **Using Terraform to create the Terraform state bucket**
      - Terraform files explanation
   3. Setting up the Network
      - Using Terraform to create VPC and other Network related resources
      - Terraform files explanation
   4. IAM Roles and Policies for EKS
      - Using Terraform to create the IAM Roles

```
    }


  # random_id.id1 will be created
  + resource "random_id" "id1" {
      + b64_std     = (known after apply)
      + b64_url     = (known after apply)
      + byte_length = 8
      + dec         = (known after apply)
      + hex         = (known after apply)
      + id          = (known after apply)
    }

Plan: 23 to add, 0 to change, 0 to destroy.

Changes to Outputs:
  + keyid     = (known after apply)
  + region    = [
      + null,
    ]
  + s3_bucket = [
      + null,
    ]
  + tfid      = (known after apply)

------------------------------------------------------------------------------------------------------------------------

Saved the plan to: tfplan

To perform exactly these actions, run the following command to apply:
    terraform apply "tfplan"
```

Next "apply" the plan to build the infrastructure in AWS

```
1   terraform apply tfplan
```

```
random_id.id1: Creating...
random_id.id1: Creation complete after 0s [id=uNNKMcr3lKs]
aws_ssm_parameter.tf-eks-cluster-name: Creating...
```

Amazon EKS Terraform Workshop

1. Workshop Introduction
2. Introduction to Kubernetes
3. Start the Workshop
4. Terraform Primer (Optional)
5. Creating a private EKS Cluster with Terraform
   1. EKS Terraform Scenario
      - EKS multi-part Build with Terraform
   2. Initial Setup for Terraform state
      - **Using Terraform to create the Terraform state bucket**
      - Terraform files explanation
   3. Setting up the Network
      - Using Terraform to create VPC and other Network related resources
      - Terraform files explanation
   4. IAM Roles and Policies for EKS
      - Using Terraform to create the IAM Roles


```
aws_s3_bucket.terraform_state: Creating...
aws_kms_key.ekskey: Creating...
aws_ssm_parameter.tf-eks-region: Creating...
aws_ssm_parameter.tf-eks-version: Creating...
aws_ssm_parameter.tf-eks-id: Creating...
aws_ssm_parameter.tf-eks-version: Creation complete after 0s [id=/workshop/tf-eks/eks-version]
aws_ssm_parameter.tf-eks-id: Creation complete after 0s [id=/workshop/tf-eks/id]
aws_ssm_parameter.tf-eks-cluster-name: Creation complete after 0s [id=/workshop/tf-eks/cluster-name]
aws_ssm_parameter.tf-eks-region: Creation complete after 0s [id=/workshop/tf-eks/region]
aws_kms_key.ekskey: Creation complete after 0s [id=560f2ac4-d3c0-4815-95e7-e6aa713eeea7]
aws_ssm_parameter.tf-eks-keyarn: Creating...
aws_ssm_parameter.tf-eks-keyid: Creating...
aws_ssm_parameter.tf-eks-keyid: Creation complete after 1s [id=/workshop/tf-eks/keyid]
aws_ssm_parameter.tf-eks-keyarn: Creation complete after 1s [id=/workshop/tf-eks/keyarn]
aws_s3_bucket.terraform_state: Creation complete after 1s [id=tf-state-workshop-b8d34a31caf794ab]
aws_dynamodb_table.terraform_locks[7]: Creating...
aws_dynamodb_table.terraform_locks[3]: Creating...
aws_dynamodb_table.terraform_locks[2]: Creating...
aws_dynamodb_table.terraform_locks[1]: Creating...
aws_s3_bucket_public_access_block.pub_block_state: Creating...
aws_ssm_parameter.tf-eks-buck-name: Creating...
aws_dynamodb_table.terraform_locks[0]: Creating...
aws_dynamodb_table.terraform_locks[8]: Creating...
aws_dynamodb_table.terraform_locks[6]: Creating...
aws_dynamodb_table.terraform_locks[4]: Creating...
aws_ssm_parameter.tf-eks-buck-name: Creation complete after 0s [id=/workshop/tf-eks/bucket-name]
aws_s3_bucket_versioning.terraform_state: Creating...
aws_s3_bucket_public_access_block.pub_block_state: Creation complete after 0s [id=tf-state-workshop-b8d34a31caf7
aws_s3_bucket_server_side_encryption_configuration.terraform_state: Creating...
aws_s3_bucket_server_side_encryption_configuration.terraform_state: Creation complete after 1s [id=tf-state-work
aws_dynamodb_table.terraform_locks[5]: Creating...
aws_s3_bucket_versioning.terraform_state: Creation complete after 2s [id=tf-state-workshop-b8d34a31caf794ab]
aws_dynamodb_table.terraform_locks[6]: Creation complete after 8s [id=terraform_locks_cicd]
aws_dynamodb_table.terraform_locks[3]: Creation complete after 8s [id=terraform_locks_c9net]
aws_dynamodb_table.terraform_locks[2]: Creation complete after 8s [id=terraform_locks_iam]
aws_dynamodb_table.terraform_locks[0]: Creation complete after 8s [id=terraform_locks_tf-setup]
aws_dynamodb_table.terraform_locks[7]: Creation complete after 8s [id=terraform_locks_sampleapp]
aws_dynamodb_table.terraform_locks[5]: Creation complete after 8s [id=terraform_locks_nodeg]
aws_dynamodb_table.terraform_locks[1]: Still creating... [10s elapsed]
aws_dynamodb_table.terraform_locks[8]: Still creating... [10s elapsed]
aws_dynamodb_table.terraform_locks[4]: Still creating... [10s elapsed]
aws_dynamodb_table.terraform_locks[8]: Creation complete after 10s [id=terraform_locks_fargate]
aws_dynamodb_table.terraform_locks[1]: Creation complete after 14s [id=terraform_locks_net]
aws_dynamodb_table.terraform_locks[4]: Creation complete after 15s [id=terraform_locks_cluster]
null_resource.gen_backend: Creating...
null_resource.gen_backend: Provisioning with 'local-exec'...
null_resource.gen_backend (local-exec): Executing: ["/bin/sh" "-c" "./gen-backend.sh"]
null_resource.gen_backend (local-exec): region=eu-west-1
```

Amazon EKS Terraform Workshop

1. Workshop Introduction
2. Introduction to Kubernetes
3. Start the Workshop
4. Terraform Primer (Optional)
5. Creating a private EKS Cluster with Terraform
   1. EKS Terraform Scenario
      - EKS multi-part Build with Terraform
   2. Initial Setup for Terraform state
      - **Using Terraform to create the Terraform state bucket**
      - Terraform files explanation
   3. Setting up the Network
      - Using Terraform to create VPC and other Network related resources
      - Terraform files explanation
   4. IAM Roles and Policies for EKS
      - Using Terraform to create the IAM Roles

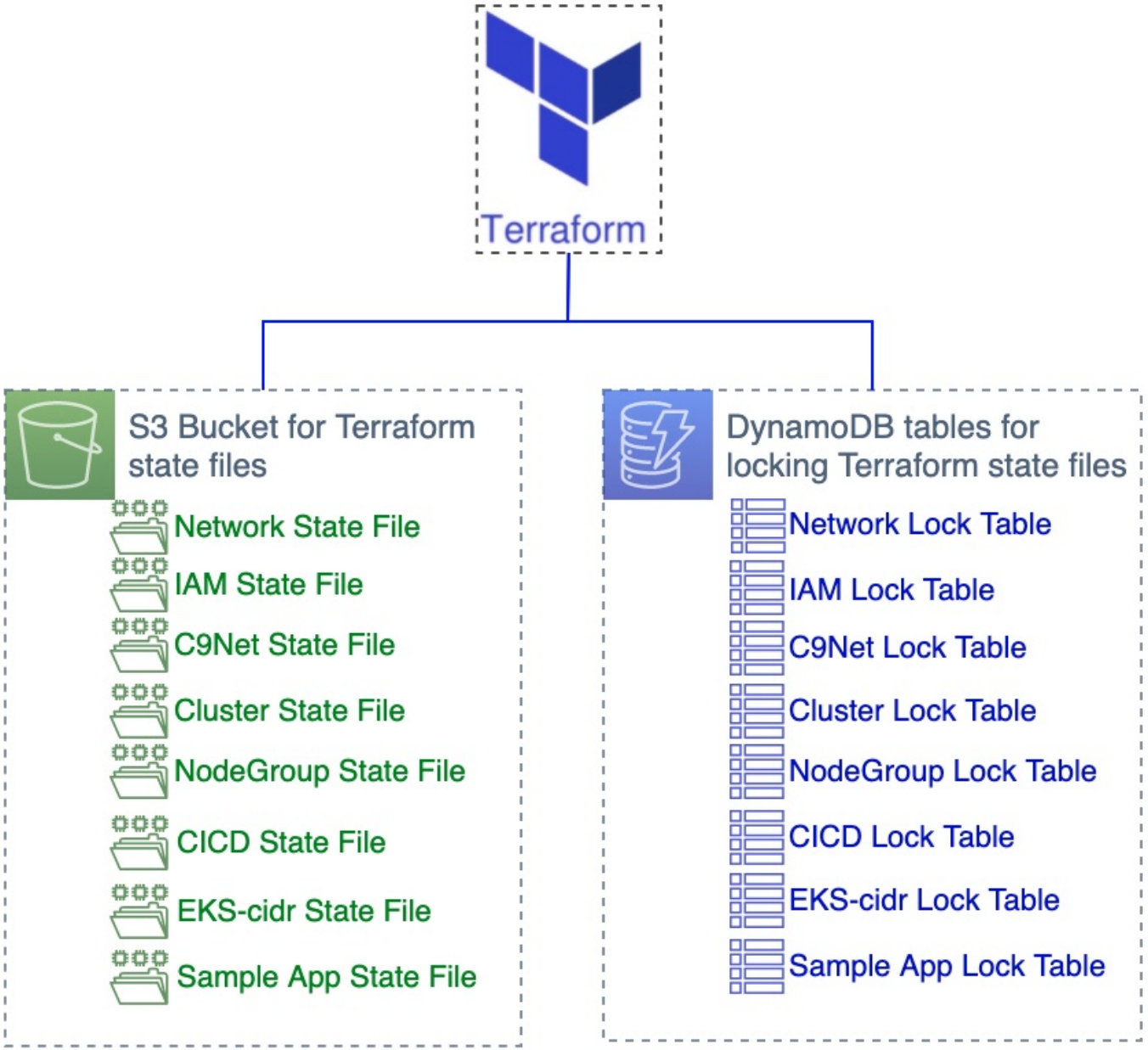
```
null_resource.gen_backend: Still creating... [10s elapsed]
null_resource.gen_backend (local-exec): tf-state-workshop-b8d34a31caf794ab terraform_locks_tf-setup
null_resource.gen_backend (local-exec): tf-state-workshop-b8d34a31caf794ab terraform_locks_net
null_resource.gen_backend (local-exec): tf-state-workshop-b8d34a31caf794ab terraform_locks_iam
null_resource.gen_backend (local-exec): tf-state-workshop-b8d34a31caf794ab terraform_locks_c9net
null_resource.gen_backend (local-exec): tf-state-workshop-b8d34a31caf794ab terraform_locks_cicd
null_resource.gen_backend: Still creating... [20s elapsed]
null_resource.gen_backend (local-exec): tf-state-workshop-b8d34a31caf794ab terraform_locks_cluster
null_resource.gen_backend (local-exec): tf-state-workshop-b8d34a31caf794ab terraform_locks_nodeg
null_resource.gen_backend (local-exec): tf-state-workshop-b8d34a31caf794ab terraform_locks_sampleapp
null_resource.gen_backend (local-exec): tf-state-workshop-b8d34a31caf794ab terraform_locks_fargate
null_resource.gen_backend (local-exec): tf-state-workshop-b8d34a31caf794ab terraform_locks_sampleapp
null_resource.gen_backend: Creation complete after 30s [id=7566236942982912112]

Apply complete! Resources: 23 added, 0 changed, 0 destroyed.

Outputs:

keyid = "560f2ac4-d3c0-4815-95e7-e6aa713eeea7"
region = [
  "eu-west-1",
]
s3_bucket = [
  "tf-state-workshop-b8d34a31caf794ab",
]
tfid = "b8d34a31caf794ab"
```

The above performed the following actions:

- Initializes Terraform in the tf-setup directory.
- Runs Terraform (plan and apply) which:
  - Creates a Randon ID for our project stores it in `var-tfid.tf`
  - Creates a s3 bucket
  - Creates the DynamoDB tables for terraform locks
  - Runs the the `gen-backend.sh` script from a Terraform "null resource" `null_resource.tf`

Amazon EKS Terraform Workshop

1. Workshop Introduction
2. Introduction to Kubernetes
3. Start the Workshop
4. Terraform Primer (Optional)
5. Creating a private EKS Cluster with Terraform
   - 1. EKS Terraform Scenario
     - EKS multi-part Build with Terraform
   - 2. Initial Setup for Terraform state
     - **Using Terraform to create the Terraform state bucket**
     - Terraform files explanation
   - 3. Setting up the Network
     - Using Terraform to create VPC and other Network related resources
     - Terraform files explanation
   - 4. IAM Roles and Policies for EKS
     - Using Terraform to create the IAM Roles



The **gen-backend.sh script** generates these terraform files for use in other sections and copies them into place:

- generated/backend-{section}.tf *(For each section this defines where our Terraform state file is located)*

Amazon EKS Terraform Workshop

1. Workshop Introduction
2. Introduction to Kubernetes
3. Start the Workshop
4. Terraform Primer (Optional)
5. Creating a private EKS Cluster with Terraform
    1. EKS Terraform Scenario
        EKS multi-part Build with Terraform
    2. Initial Setup for Terraform state
        Using Terraform to create the Terraform state bucket
        Terraform files explanation
    3. Setting up the Network
        Using Terraform to create VPC and other Network related resources
        Terraform files explanation
    4. IAM Roles and Policies for EKS
        Using Terraform to create the IAM Roles