https://github.com/JacobMenge/lern-unterlagen/blob/main/terraform/terraform-aws-aufgabe-00.md

Übung 4: AWS Ressourcenbereitstellung

Zusammenfassung: In dieser Übung validierst du deine AWS-Konfiguration und erstellst einen konformen S3-Bucket mit aktuellen Best Practices für Sicherheit und Compliance.
Wichtiger Hinweis: Für die Arbeit mit AWS benötigst du eine korrekt konfigurierte AWS CLI. Eine ausführliche Anleitung zur Einrichtung der AWS CLI findest du in Brians Tutorial:
https://github.com/BrianR-Back2Code/Terraform/blob/main/aws_connect_vscode.md

1. Stelle sicher, dass die AWS CLI installiert und konfiguriert ist:

```
aws --version
aws sts get-caller-identity
```

2.
3. Erstelle ein neues Projekt für die AWS-Ressourcen:

```
mkdir terraform-aws-test
cd terraform-aws-test
```

4.
5. Erstelle eine `provider.tf` Datei:

```
terraform {
  required_version = ">= 1.0.0"
  required_providers {
    aws = {
      source  = "hashicorp/aws"
      version = "~> 5.0"
    }
    random = {
      source  = "hashicorp/random"
      version = "~> 3.5"
    }
  }
}

provider "aws" {
  region = "eu-central-1"  # Anpassen an deine bevorzugte Region

  # Empfohlene Provider-Konfigurationen für Produktionsumgebungen
```

```
  # default_tags {
  #   tags = {
  #     Environment = "Test"
  #     ManagedBy   = "Terraform"
  #   }
  # }
}
```

4. Erstelle eine `main.tf` Datei für die S3-Ressource:

```
# Eindeutiger Suffix für den Bucket-Namen
resource "random_id" "bucket_suffix" {
  byte_length = 4
}

# S3-Bucket mit aktuellen Best Practices
resource "aws_s3_bucket" "test_bucket" {
  bucket = "terraform-test-${random_id.bucket_suffix.hex}"
}

# Bucket-Eigentum festlegen (empfohlen)
resource "aws_s3_bucket_ownership_controls" "bucket_ownership" {
  bucket = aws_s3_bucket.test_bucket.id
  rule {
    object_ownership = "BucketOwnerEnforced"
  }
}

# Public Access blockieren (sicherheitsempfehlung)
resource "aws_s3_bucket_public_access_block" "block_public_access" {
  bucket = aws_s3_bucket.test_bucket.id

  block_public_acls       = true
  block_public_policy     = true
  ignore_public_acls      = true
  restrict_public_buckets = true
}

# Server-seitige Verschlüsselung aktivieren
resource "aws_s3_bucket_server_side_encryption_configuration" "bucket_encryption" {
  bucket = aws_s3_bucket.test_bucket.id

  rule {
    apply_server_side_encryption_by_default {
      sse_algorithm = "AES256"
    }
  }
}
```

5. Erstelle eine `outputs.tf` Datei:

```
output "bucket_name" {
  description = "Name des erstellten S3-Buckets"
  value       = aws_s3_bucket.test_bucket.bucket
}
```

```
output "bucket_arn" {
  description = "ARN des erstellten S3-Buckets"
  value       = aws_s3_bucket.test_bucket.arn
}

output "bucket_region" {
  description = "Region des erstellten S3-Buckets"
  value       = aws_s3_bucket.test_bucket.region
}
```

6. Initialisiere das Projekt:

```
terraform init
```

7.

8. Validiere die Konfiguration:

```
terraform validate
```

9.

10. Führe eine Planung durch:

```
terraform plan
```

11.

12. Wende die Konfiguration an:

```
terraform apply
```

13.

14. Überprüfe die Erstellung in der AWS Management Console oder über die AWS CLI:

```
aws s3 ls
```

15.

16. Bereinige die Ressourcen, um Kosten zu vermeiden:

```
terraform destroy
```

17.

18. Wenn du bis hierhin gelesen hast und die Übung erfolgreich durchgeführt hast, dann schreib mir den folgenden Code als Direktnachricht auf Slack ;)

```
X34t5
```

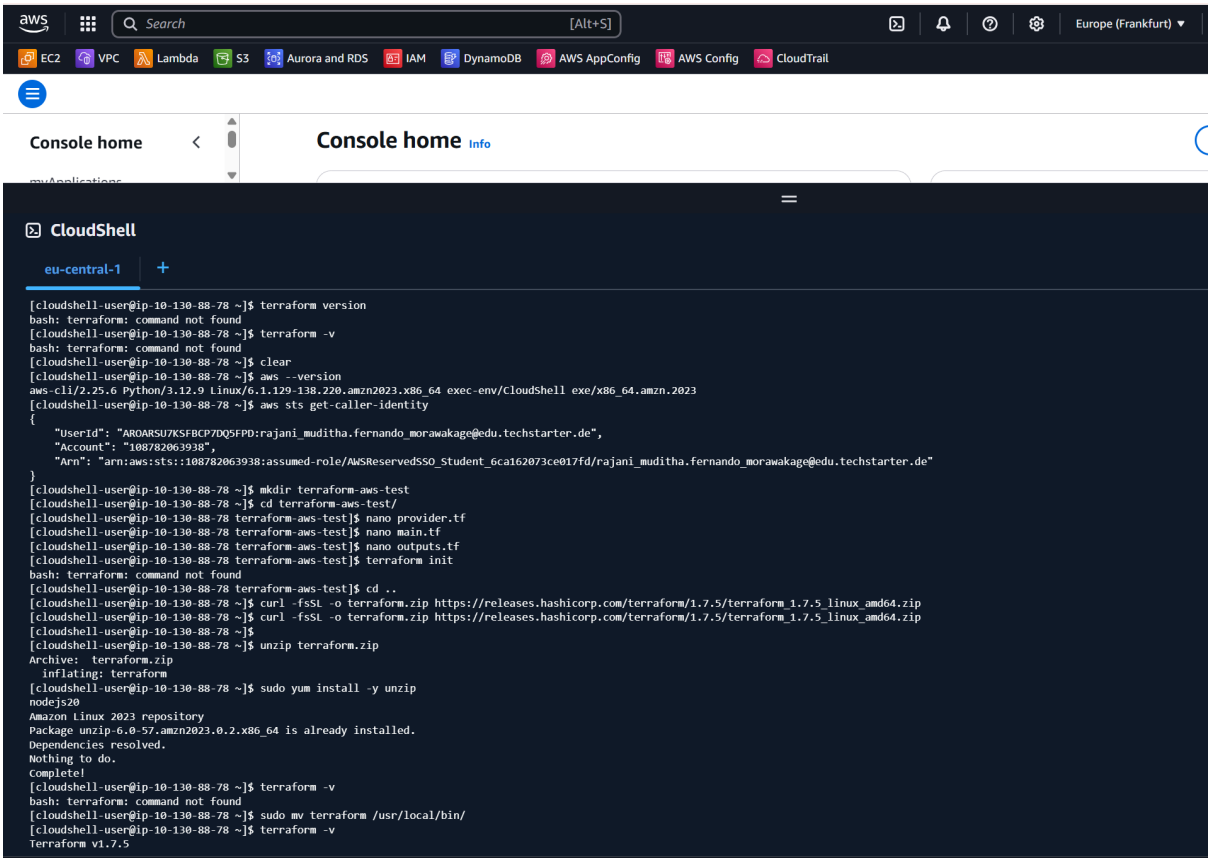
19.

Warum ist dieser Ansatz wichtig?
Diese Übung führt dich in die Erstellung von echten Cloud-Ressourcen mit Terraform ein und zeigt Best Practices für Sicherheit und Compliance:

- AWS Provider Konfiguration:
    - Die Definition der Provider-Version (~> 5.0) stellt sicher, dass du mit einer aktuellen, aber kompatiblen Version arbeitest.
    - Die Region-Angabe ist wichtig, da AWS-Ressourcen immer in einer bestimmten Region erstellt werden.
    - Die auskommentierten Tags zeigen bewährte Methoden für die Ressourcen-Organisation in Produktionsumgebungen.

- Eindeutige Ressourcennamen:
    - Der random_id-Provider erzeugt einen zufälligen Suffix, der sicherstellt, dass deine Bucket-Namen global eindeutig sind (ein S3-Bucket-Name muss AWS-weit einzigartig sein).
- Sicherheitspraktiken für S3:
    - Bucket Ownership Controls: Klarheit über das Eigentum an Objekten im Bucket.
    - Public Access Block: Verhindert versehentliche öffentliche Exposition des Buckets und seiner Daten - einer der häufigsten Sicherheitsfehler in AWS.
    - Verschlüsselung: Server-seitige Verschlüsselung schützt die Daten im Ruhezustand.
- Outputs für wichtige Ressourceninformationen:
    - Der ARN (Amazon Resource Name) ist ein eindeutiger Identifier, der für Berechtigungen und Cross-Service-Referenzen benötigt wird.
    - Name und Region sind grundlegende Informationen, die du für den Zugriff auf den Bucket benötigst.
- terraform validate:
    - Dieser Befehl prüft deine Konfiguration auf syntaktische Korrektheit und interne Konsistenz, bevor du Zeit mit der Ausführung verbringst.
- Aufräumen:
    - Der destroy-Befehl ist entscheidend, um unerwartete Kosten zu vermeiden, insbesondere bei Cloud-Diensten, die kontinuierlich Gebühren verursachen können.

Dieser Ansatz zeigt, wie du mit Terraform Cloud-Ressourcen nicht nur erstellen, sondern auch entsprechend aktueller Best Practices für Sicherheit und Betrieb konfigurieren kannst.


Search
[Alt+S]
Europe (Frankfurt)
EC2
VPC
Lambda
S3
Aurora and RDS
IAM
DynamoDB
AWS AppConfig
AWS Config
CloudTrail
Console home
Console home Info
CloudShell
eu-central-1
[cloudshell-user@ip-10-130-88-78 ~]$ terraform version
bash: terraform: command not found
[cloudshell-user@ip-10-130-88-78 ~]$ terraform -v
bash: terraform: command not found
[cloudshell-user@ip-10-130-88-78 ~]$ clear
[cloudshell-user@ip-10-130-88-78 ~]$ aws --version
aws-cli/2.25.6 Python/3.12.9 Linux/6.1.129-138.220.amzn2023.x86_64 exec-env/CloudShell exe/x86_64.amzn.2023
[cloudshell-user@ip-10-130-88-78 ~]$ aws sts get-caller-identity
{
"UserId": "AROARSU7KSFBCP7DQ5FPD:rajani_muditha.fernando_morawakage@edu.techstarter.de",
"Account": "108782063938",
"Arn": "arn:aws:sts::108782063938:assumed-role/AWSReservedSSO_Student_6ca162073ce017fd/rajani_muditha.fernando_morawakage@edu.techstarter.de"
}
[cloudshell-user@ip-10-130-88-78 ~]$ mkdir terraform-aws-test
[cloudshell-user@ip-10-130-88-78 ~]$ cd terraform-aws-test/
[cloudshell-user@ip-10-130-88-78 terraform-aws-test]$ nano provider.tf
[cloudshell-user@ip-10-130-88-78 terraform-aws-test]$ nano main.tf
[cloudshell-user@ip-10-130-88-78 terraform-aws-test]$ nano outputs.tf
[cloudshell-user@ip-10-130-88-78 terraform-aws-test]$ terraform init
bash: terraform: command not found
[cloudshell-user@ip-10-130-88-78 terraform-aws-test]$ cd ..
[cloudshell-user@ip-10-130-88-78 ~]$ curl -fsSL -o terraform.zip https://releases.hashicorp.com/terraform/1.7.5/terraform_1.7.5_linux_amd64.zip
[cloudshell-user@ip-10-130-88-78 ~]$ curl -fsSL -o terraform.zip https://releases.hashicorp.com/terraform/1.7.5/terraform_1.7.5_linux_amd64.zip
[cloudshell-user@ip-10-130-88-78 ~]$
[cloudshell-user@ip-10-130-88-78 ~]$ unzip terraform.zip
Archive: terraform.zip
inflating: terraform
[cloudshell-user@ip-10-130-88-78 ~]$ sudo yum install -y unzip
nodejs20
Amazon Linux 2023 repository
Package unzip-6.0-57.amzn2023.0.2.x86_64 is already installed.
Dependencies resolved.
Nothing to do.
Complete!
[cloudshell-user@ip-10-130-88-78 ~]$ terraform -v
bash: terraform: command not found
[cloudshell-user@ip-10-130-88-78 ~]$ sudo mv terraform /usr/local/bin/
[cloudshell-user@ip-10-130-88-78 ~]$ terraform -v
Terraform v1.7.5

```
on linux_amd64

Your version of Terraform is out of date! The latest version
is 1.11.3. You can update by downloading from https://www.terraform.io/downloads.html
[cloudshell-user@ip-10-130-88-78 ~]$ cd terraform-aws-test/
[cloudshell-user@ip-10-130-88-78 terraform-aws-test]$ terraform -v
Terraform v1.7.5
on linux_amd64

Your version of Terraform is out of date! The latest version
is 1.11.3. You can update by downloading from https://www.terraform.io/downloads.html
[cloudshell-user@ip-10-130-88-78 terraform-aws-test]$ terraform init

Initializing the backend...

Initializing provider plugins...
- Finding hashicorp/aws versions matching "~> 5.0"...
- Finding hashicorp/random versions matching "~> 3.5"...
- Installing hashicorp/random v3.7.1...
- Installed hashicorp/random v3.7.1 (signed by HashiCorp)
- Installing hashicorp/aws v5.93.0...
- Installed hashicorp/aws v5.93.0 (signed by HashiCorp)

Terraform has created a lock file .terraform.lock.hcl to record the provider
selections it made above. Include this file in your version control repository
so that Terraform can guarantee to make the same selections by default when
you run "terraform init" in the future.

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.
[cloudshell-user@ip-10-130-88-78 terraform-aws-test]$ terraform validate
Success! The configuration is valid.
```

```
Success! The configuration is valid.

[cloudshell-user@ip-10-130-88-78 terraform-aws-test]$ terraform plan

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following symbols:
  + create

Terraform will perform the following actions:

  # aws_s3_bucket.test_bucket will be created
  + resource "aws_s3_bucket" "test_bucket" {
      + 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               = false
      + 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)
      + tags_all                    = (known after apply)
      + website_domain              = (known after apply)
      + website_endpoint            = (known after apply)
    }

  # aws_s3_bucket_ownership_controls.bucket_ownership will be created
  + resource "aws_s3_bucket_ownership_controls" "bucket_ownership" {
      + bucket = (known after apply)
      + id     = (known after apply)

      + rule {
          + object_ownership = "BucketOwnerEnforced"
        }
    }
```

```
  # aws_s3_bucket_public_access_block.block_public_access will be created
  + resource "aws_s3_bucket_public_access_block" "block_public_access" {
      + 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.bucket_encryption will be created
  + resource "aws_s3_bucket_server_side_encryption_configuration" "bucket_encryption" {
      + bucket = (known after apply)
      + id     = (known after apply)

      + rule {
          + apply_server_side_encryption_by_default {
              + sse_algorithm = "AES256"
            }
        }
    }

  # random_id.bucket_suffix will be created
  + resource "random_id" "bucket_suffix" {
      + b64_std     = (known after apply)
      + b64_url     = (known after apply)
      + byte_length = 4
      + dec         = (known after apply)
      + hex         = (known after apply)
      + id          = (known after apply)
    }

Plan: 5 to add, 0 to change, 0 to destroy.

Changes to Outputs:
```

```
Changes to Outputs:
  + bucket_arn    = (known after apply)
  + bucket_name   = (known after apply)
  + bucket_region = (known after apply)

---

Note: You didn't use the -out option to save this plan, so Terraform can't guarantee to take exactly these actions if you run "terraform apply" now.
[cloudshell-user@ip-10-130-88-78 terraform-aws-test]$ terraform apply

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following symbols:
  + create

Terraform will perform the following actions:

  # aws_s3_bucket.test_bucket will be created
  + resource "aws_s3_bucket" "test_bucket" {
      + 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               = false
      + 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)
      + tags_all                    = (known after apply)
      + website_domain              = (known after apply)
      + website_endpoint            = (known after apply)
    }

  # aws_s3_bucket_ownership_controls.bucket_ownership will be created
```

```
  # aws_s3_bucket_ownership_controls.bucket_ownership will be created
  + resource "aws_s3_bucket_ownership_controls" "bucket_ownership" {
      + bucket = (known after apply)
      + id     = (known after apply)

      + rule {
          + object_ownership = "BucketOwnerEnforced"
        }
    }

  # aws_s3_bucket_public_access_block.block_public_access will be created
  + resource "aws_s3_bucket_public_access_block" "block_public_access" {
      + 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.bucket_encryption will be created
  + resource "aws_s3_bucket_server_side_encryption_configuration" "bucket_encryption" {
      + bucket = (known after apply)
      + id     = (known after apply)

      + rule {
          + apply_server_side_encryption_by_default {
              + sse_algorithm = "AES256"
            }
        }
    }

  # random_id.bucket_suffix will be created
  + resource "random_id" "bucket_suffix" {
      + b64_std     = (known after apply)
      + b64_url     = (known after apply)
```

CloudShell Feedback

```
      + b64_url     = (known after apply)
      + byte_length = 4
      + dec         = (known after apply)
      + hex         = (known after apply)
      + id          = (known after apply)
    }

Plan: 5 to add, 0 to change, 0 to destroy.

Changes to Outputs:
  + bucket_arn    = (known after apply)
  + bucket_name   = (known after apply)
  + bucket_region = (known after apply)

Do you want to perform these actions?
  Terraform will perform the actions described above.
  Only 'yes' will be accepted to approve.

  Enter a value: yes

random_id.bucket_suffix: Creating...
random_id.bucket_suffix: Creation complete after 0s [id=6HVF2Q]
aws_s3_bucket.test_bucket: Creating...
aws_s3_bucket.test_bucket: Creation complete after 1s [id=terraform-test-e87545d9]
aws_s3_bucket_ownership_controls.bucket_ownership: Creating...
aws_s3_bucket_public_access_block.block_public_access: Creating...
aws_s3_bucket_server_side_encryption_configuration.bucket_encryption: Creating...
aws_s3_bucket_public_access_block.block_public_access: Creation complete after 0s [id=terraform-test-e87545d9]
aws_s3_bucket_server_side_encryption_configuration.bucket_encryption: Creation complete after 0s [id=terraform-test-e87545d9]
aws_s3_bucket_ownership_controls.bucket_ownership: Creation complete after 0s [id=terraform-test-e87545d9]

Apply complete! Resources: 5 added, 0 changed, 0 destroyed.

Outputs:

bucket_arn = "arn:aws:s3:::terraform-test-e87545d9"
bucket_name = "terraform-test-e87545d9"
bucket_region = "eu-central-1"
```

```
2025-04-03 11:24:07 terraform-test-e87545d9
[cloudshell-user@ip-10-130-88-78 terraform-aws-test]$ terraform destroy
random_id.bucket_suffix: Refreshing state... [id=6HVF2Q]
aws_s3_bucket.test_bucket: Refreshing state... [id=terraform-test-e87545d9]
aws_s3_bucket_ownership_controls.bucket_ownership: Refreshing state... [id=terraform-test-e87545d9]
aws_s3_bucket_server_side_encryption_configuration.bucket_encryption: Refreshing state... [id=terraform-test-e87545d9]
aws_s3_bucket_public_access_block.block_public_access: Refreshing state... [id=terraform-test-e87545d9]

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following symbols:
  - destroy

Terraform will perform the following actions:

  # aws_s3_bucket.test_bucket will be destroyed
  - resource "aws_s3_bucket" "test_bucket" {
      - arn                         = "arn:aws:s3:::terraform-test-e87545d9" -> null
      - bucket                      = "terraform-test-e87545d9" -> null
      - bucket_domain_name          = "terraform-test-e87545d9.s3.amazonaws.com" -> null
      - bucket_regional_domain_name = "terraform-test-e87545d9.s3.eu-central-1.amazonaws.com" -> null
      - force_destroy               = false -> null
      - hosted_zone_id              = "Z21DNDUVLTQW6Q" -> null
      - id                          = "terraform-test-e87545d9" -> null
      - object_lock_enabled         = false -> null
      - region                      = "eu-central-1" -> null
      - request_payer               = "BucketOwner" -> null
      - tags                        = {} -> null
      - tags_all                    = {} -> null

      - grant {
          - id          = "4bc00212979a348f3c976a0109b458c97277f64df882ffd12ed9a50d1db07342" -> null
          - permissions = [
              - "FULL_CONTROL",
            ] -> null
          - type        = "CanonicalUser" -> null
        }

      - server_side_encryption_configuration {
          - rule {
              - bucket_key_enabled = false -> null
```

```
          - enabled    = false -> null
          - mfa_delete = false -> null
        }
    }

  # aws_s3_bucket_ownership_controls.bucket_ownership will be destroyed
  - resource "aws_s3_bucket_ownership_controls" "bucket_ownership" {
      - bucket = "terraform-test-e87545d9" -> null
      - id     = "terraform-test-e87545d9" -> null

      - rule {
          - object_ownership = "BucketOwnerEnforced" -> null
        }
    }

  # aws_s3_bucket_public_access_block.block_public_access will be destroyed
  - resource "aws_s3_bucket_public_access_block" "block_public_access" {
      - block_public_acls       = true -> null
      - block_public_policy     = true -> null
      - bucket                  = "terraform-test-e87545d9" -> null
      - id                      = "terraform-test-e87545d9" -> null
      - ignore_public_acls      = true -> null
      - restrict_public_buckets = true -> null
    }

  # aws_s3_bucket_server_side_encryption_configuration.bucket_encryption will be destroyed
  - resource "aws_s3_bucket_server_side_encryption_configuration" "bucket_encryption" {
      - bucket = "terraform-test-e87545d9" -> null
      - id     = "terraform-test-e87545d9" -> null

      - rule {
          - bucket_key_enabled = false -> null

          - apply_server_side_encryption_by_default {
              - sse_algorithm = "AES256" -> null
            }
        }
    }
```

CloudShell Feedback

```
        }
    }

  # random_id.bucket_suffix will be destroyed
  - resource "random_id" "bucket_suffix" {
      - b64_std     = "6HVF2Q==" -> null
      - b64_url     = "6HVF2Q" -> null
      - byte_length = 4 -> null
      - dec         = "3899999705" -> null
      - hex         = "e87545d9" -> null
      - id          = "6HVF2Q" -> null
    }

Plan: 0 to add, 0 to change, 5 to destroy.

Changes to Outputs:
  - bucket_arn    = "arn:aws:s3:::terraform-test-e87545d9" -> null
  - bucket_name   = "terraform-test-e87545d9" -> null
  - bucket_region = "eu-central-1" -> null

Do you really want to destroy all resources?
  Terraform will destroy all your managed infrastructure, as shown above.
  There is no undo. Only 'yes' will be accepted to confirm.

  Enter a value: yes

aws_s3_bucket_public_access_block.block_public_access: Destroying... [id=terraform-test-e87545d9]
aws_s3_bucket_server_side_encryption_configuration.bucket_encryption: Destroying... [id=terraform-test-e87545d9]
aws_s3_bucket_ownership_controls.bucket_ownership: Destroying... [id=terraform-test-e87545d9]
aws_s3_bucket_server_side_encryption_configuration.bucket_encryption: Destruction complete after 0s
aws_s3_bucket_public_access_block.block_public_access: Destruction complete after 1s
aws_s3_bucket_ownership_controls.bucket_ownership: Destruction complete after 1s
aws_s3_bucket.test_bucket: Destroying... [id=terraform-test-e87545d9]
aws_s3_bucket.test_bucket: Destruction complete after 0s
random_id.bucket_suffix: Destroying... [id=6HVF2Q]
random_id.bucket_suffix: Destruction complete after 0s

Destroy complete! Resources: 5 destroyed.
[cloudshell-user@ip-10-130-88-78 terraform-aws-test]$
```