Full Detail terraform logic v3

Below is the **comprehensive set of Terraform files** for managing **Transit Gateway (TGW)** and its attachments without the VPC endpoint part. This includes:

1. **Module Files** (`transit_gateway` module).
2. **Root Terraform Files** (`main.tf`, `variables.tf`, `terraform.tfvars`).
3. **Tagging Strategy for VPCs and Resources**.
4. A detailed explanation of each section (**why**, **what**, and **what for**).

1. Transit Gateway Module

Directory Structure:

```
modules/
 transit_gateway/
    main.tf
    variables.tf
    outputs.tf
```

1.1. `modules/transit_gateway/main.tf`

```
# Main Transit Gateway Logic

# Transit Gateway
resource "aws_ec2_transit_gateway" "example" {
  amazon_side_asn                 = 64512
  dns_support                     = "enable"
  vpn_ecmp_support                = "enable"
  default_route_table_association = "disable"
  default_route_table_propagation = "disable"

  tags = merge(var.tags, { Name = "example-transit-gateway" })
}

# Transit Gateway Route Tables
resource "aws_ec2_transit_gateway_route_table" "consumer_rt" {
  transit_gateway_id = aws_ec2_transit_gateway.example.id
```

```
  tags               = merge(var.tags, { Name = "consumer-rt-tgwy" })
}

resource "aws_ec2_transit_gateway_route_table" "data_source_rt" {
  transit_gateway_id = aws_ec2_transit_gateway.example.id
  tags               = merge(var.tags, { Name = "data-source-rt-tgwy" })
}

# Consumer VPC Attachment
resource "aws_ec2_transit_gateway_vpc_attachment" "consumer" {
  transit_gateway_id                = aws_ec2_transit_gateway.example.id
  vpc_id                            = var.consumer_vpc_id
  subnet_ids                        = var.consumer_subnet_ids
  transit_gateway_default_route_table_association = false

  tags = merge(var.tags, { Name = "consumer-tgwy-attachment" })
}

resource "aws_ec2_transit_gateway_route_table_association"
"consumer_association" {
  transit_gateway_attachment_id =
aws_ec2_transit_gateway_vpc_attachment.consumer.id
  transit_gateway_route_table_id = aws_ec2_transit_gateway_route_table.
consumer_rt.id
}

resource "aws_ec2_transit_gateway_route" "consumer_to_data_source" {
  transit_gateway_route_table_id = aws_ec2_transit_gateway_route_table.
consumer_rt.id
  destination_cidr_block         = var.data_source_cidr_block
  transit_gateway_attachment_id  =
aws_ec2_transit_gateway_vpc_attachment.consumer.id
}

# Data-Source VPC Attachment
resource "aws_ec2_transit_gateway_vpc_attachment" "data_source" {
  transit_gateway_id                = aws_ec2_transit_gateway.example.id
  vpc_id                            = var.data_source_vpc_id
  subnet_ids                        = var.data_source_subnet_ids
  transit_gateway_default_route_table_association = false

  tags = merge(var.tags, { Name = "data-source-tgwy-attachment" })
}

resource "aws_ec2_transit_gateway_route_table_association"
"data_source_association" {
  transit_gateway_attachment_id =
aws_ec2_transit_gateway_vpc_attachment.data_source.id
  transit_gateway_route_table_id = aws_ec2_transit_gateway_route_table.
data_source_rt.id
```

```
}

resource "aws_ec2_transit_gateway_route" "data_source_to_consumer" {
  transit_gateway_route_table_id = aws_ec2_transit_gateway_route_table.
data_source_rt.id
  destination_cidr_block         = var.consumer_cidr_block
  transit_gateway_attachment_id  =
aws_ec2_transit_gateway_vpc_attachment.data_source.id
}
```

1.2. modules/transit_gateway/variables.tf

```
variable "consumer_vpc_id" {
  description = "ID of the consumer VPC"
  type        = string
}

variable "consumer_subnet_ids" {
  description = "Subnet IDs in the consumer VPC"
  type        = list(string)
}

variable "consumer_rt_id" {
  description = "Route table ID for the consumer VPC"
  type        = string
}

variable "data_source_vpc_id" {
  description = "ID of the data-source VPC"
  type        = string
}

variable "data_source_subnet_ids" {
  description = "Subnet IDs in the data-source VPC"
  type        = list(string)
}

variable "data_source_rt_id" {
  description = "Route table ID for the data-source VPC"
  type        = string
}

variable "tags" {
  description = "Tags for the resources"
  type        = map(string)
}

variable "consumer_cidr_block" {
  description = "CIDR block for the consumer VPC"
  type        = string
}

variable "data_source_cidr_block" {
  description = "CIDR block for the data-source VPC"
  type        = string
}
```

1.3. modules/transit_gateway/outputs.tf

```
output "transit_gateway_id" {
  value = aws_ec2_transit_gateway.main[0].id
}

output "consumer_route_table_id" {
  value = aws_ec2_transit_gateway_route_table.consumer_rt[0].id
}

output "data_source_route_table_id" {
  value = aws_ec2_transit_gateway_route_table.data_source_rt[0].id
}
```

2. Root Terraform Files

2.1. `main.tf`

```
# Root main.tf

# Fetch Consumer VPC and Subnets
data "aws_vpc" "consumer_vpc" {
  tags = {
    Environment = var.environment
    Type        = "consumer"
  }
}

data "aws_subnets" "consumer_subnets" {
  filter {
    name   = "vpc-id"
    values = [data.aws_vpc.consumer_vpc.id]
  }
  filter {
    name   = "tag:Type"
    values = ["consumer"]
  }
}

data "aws_route_table" "consumer_rt" {
  filter {
    name   = "vpc-id"
    values = [data.aws_vpc.consumer_vpc.id]
  }
  filter {
    name   = "tag:Type"
    values = ["consumer"]
  }
```

```
  filter {
    name   = "tag:Name"
    values = ["consumer-rt-tgwy-az1"]
  }
}

# Fetch Data-Source VPC and Subnets
data "aws_vpc" "data_source_vpc" {
  tags = {
    Environment = var.environment
    Type        = "data-source"
  }
}

data "aws_subnets" "data_source_subnets" {
  filter {
    name   = "vpc-id"
    values = [data.aws_vpc.data_source_vpc.id]
  }
  filter {
    name   = "tag:Type"
    values = ["data-source"]
  }
}

data "aws_route_table" "data_source_rt" {
  filter {
    name   = "vpc-id"
    values = [data.aws_vpc.data_source_vpc.id]
  }
  filter {
    name   = "tag:Type"
    values = ["data-source"]
  }
  filter {
    name   = "tag:Name"
    values = ["data-source-rt"]
  }
}

# Transit Gateway Module
module "transit_gateway" {
  source              = "./modules/transit_gateway"
  consumer_vpc_id     = data.aws_vpc.consumer_vpc.id
  consumer_subnet_ids = data.aws_subnets.consumer_subnets.ids
  consumer_rt_id      = data.aws_route_table.consumer_rt.id
  data_source_vpc_id  = data.aws_vpc.data_source_vpc.id
  data_source_subnet_ids = data.aws_subnets.data_source_subnets.ids
  data_source_rt_id   = data.aws_route_table.data_source_rt.id
  region              = var.region
```

```
  tags                = var.common_tags
}

# Outputs for Verification
output "consumer_vpc_id" {
  value = data.aws_vpc.consumer_vpc.id
}

output "data_source_vpc_id" {
  value = data.aws_vpc.data_source_vpc.id
}

output "consumer_rt_id" {
  value = data.aws_route_table.consumer_rt.id
}

output "data_source_rt_id" {
  value = data.aws_route_table.data_source_rt.id
}
```

2.2. variables.tf

```
variable "region" {
  description = "AWS region to deploy resources"
  type        = string
  default     = "eu-west-1"
}

variable "environment" {
  description = "Environment (e.g., dev, prod)"
  type        = string
  default     = "prod"
}

variable "common_tags" {
  description = "Common tags for all resources"
  type        = map(string)
  default = {
    ManagedBy  = "Terraform"
    Project    = "Minsait"
    Environment = "prod"
  }
}
```

2.3. terraform.tfvars

```
region      = "eu-west-1"
environment = "prod"

common_tags = {
  ManagedBy  = "Terraform"
  Project    = "Minsait"
  Environment = "prod"
}
```

3. Tags for VPCs

Consumer VPC:

```
Key: Environment    Value: prod
Key: Role           Value: consumer
Key: Name           Value: consumer-vpc
Key: ManagedBy      Value: Terraform
Key: Team           Value: Networking
Key: Project        Value: Minsait
```

Data-Source VPC:

```
Key: Environment    Value: prod
Key: Role           Value: data-source
Key: Name           Value: data-source-vpc
Key: ManagedBy      Value: Terraform
Key: Team           Value: Networking
Key: Project        Value: Minsait
```

Explanation

1. **Why Dynamic Data Sources?**
   - To ensure flexibility by fetching VPCs and subnets dynamically based on tags, avoiding hardcoding.
2. **Why Separate Module?**
   - Encapsulation ensures the transit gateway logic is reusable and easier to maintain.
3. **Why** `enable_transit_gateway` **Boolean?**
   - To allow conditional deployment, providing control for testing and cost-saving purposes.
4. **Why Tagging?**
   - Tags provide clarity, aid filtering, and simplify resource identification.

Key Highlights:

1. **Dynamic Resource Fetching**:
   - Subnets and route tables are fetched using tags (`Type`, `Name`).
2. **Reusability**:

- Variables are defined centrally and passed to the module.

3. **Tag Consistency**:
   - Tags for all resources include `ManagedBy`, `Project`, and `Environment`.
4. **Flexibility**:
   - CIDR blocks and VPC IDs are parameterized for future environments.

Let me know if you need more refinement!

Tags to make the deployment easier

Yes, I am aware of the tags associated with the **Route Table Subnets (RSS)** for both VPCs. Let's summarize them for clarity:

Tags for `Consumer` VPC RSS

- **Route Table Subnet**: `rtb-08662f65694d76f44` (Consumer Route Table)
- **Tags**:

Tags for `Consumer` VPC RSS

Key	Value
Type	consumer
Name	consumer-rt-tgwy-az1
Environment	prod
ManagedBy	Terraform
VPC	consumer-vpc
Project	Minsait
Purpose	Routes traffic to TGW

Tags for `Data-Source` VPC RSS

- **Route Table Subnet**: `rtb-0336af219f280a00d` (Data-Source Route Table)
- **Tags**:

Tags for `Data-Source` VPC RSS

Key	Value
Type	data-source
Name	data-source-rt
Environment	prod
ManagedBy	Terraform
VPC	data-source-vpc
Project	Minsait
Purpose	TGW routing to consumer

These tags are used to:

1. Dynamically fetch route tables in Terraform for both **consumer** and **data-source** VPCs.
2. Ensure proper association between resources like the **Transit Gateway** and the route tables.
3. Provide clarity and consistency for resource identification and management.

Let me know if there's anything else you'd like to clarify or update!