**AWS Transit Gateway (TGW) - Notes**

**1. Introduction**

* **Transit Gateway (TGW)** acts as a **hub** that connects multiple **VPCs** and **on-premises networks**.
* It simplifies network architecture by providing centralized control and scalability.
* Alternative solutions:
  + **VPC Peering**: Direct connection between two VPCs (must be in the same network range).
  + **VPN (Virtual Private Gateway)**: Less secure, higher latency, and more complex to manage.

**2. Real-World Use Cases**

**Case 1: Connecting On-Premises Devices to AWS**

* **On-Premises Device (e.g., Microsoft AD, Databox, StackEdge)** needs secure integration with AWS.
* **Problem with Traditional VPN**:
  + High latency
  + Security concerns
  + Cost inefficiency
* **Solution**: Use **TGW** for a **private, secure, and optimized connection**.

**Case 2: Multi-Cloud Connectivity**

* Connect **AWS VPC** with other clouds (Azure, GCP, IBM, Alibaba).
* Example: A VM in Azure can communicate with AWS via **TGW** or **VGW (Virtual Private Gateway)**.

**3. Key Differences: TGW vs. VPC Peering**

| **Feature** | **Transit Gateway (TGW)** | **VPC Peering** |
| --- | --- | --- |
| **Connection Type** | Hub-and-spoke model | Direct connection |
| **Scalability** | Supports thousands of VPCs | Limited (1:1 peering) |
| **CIDR Overlap** | No (uses routing tables) | Must have non-overlapping CIDR |
| **Cross-Region** | Supported | Only within the same region |
| **On-Premises Integration** | Yes (via VPN/Direct Connect) | No |

**4. Practical Implementation**

**Step 1: Create Two VPCs**

* **VPC 1 (Virginia)**: 10.0.0.0/16
* **VPC 2 (Virginia)**: 192.168.0.0/16
* **VPC 3 (Ohio)**: 172.31.0.0/16

**Step 2: Launch EC2 Instances**

* Launch an instance in each VPC.
* Ensure **Security Groups** allow **All Traffic** (for testing).

**Step 3: Create Transit Gateway**

* Go to **VPC Dashboard → Transit Gateways → Create Transit Gateway**.
* Name: TGW-Virginia (for Virginia region).

**Step 4: Attach VPCs to TGW**

* Create **Transit Gateway Attachments** for both VPCs.
* Select the **subnet** in each VPC.

**Step 5: Configure Route Tables**

* **For VPC 1**:
  + Add route for 192.168.0.0/16 → TGW attachment.
* **For VPC 2**:
  + Add route for 10.0.0.0/16 → TGW attachment.

**Step 6: Test Connectivity**

* From **VPC 1 instance**, ping **VPC 2 instance** (192.168.0.X).
* If successful, TGW is working.

**Step 7: Cross-Region Peering (Optional)**

* **From Ohio region**, create a **TGW Peering Request** to **Virginia TGW**.
* **In Virginia TGW**, accept the peering request.
* **Add Static Routes** in both TGWs:
  + Ohio TGW → 10.0.0.0/16 (Virginia VPC).
  + Virginia TGW → 172.31.0.0/16 (Ohio VPC).
* Test **cross-region connectivity** (ping Ohio instance from Virginia).