

Hybrid Cloud Setup Using VPN

1. Project Overview

Project Name: Hybrid Cloud Setup Using VPN

Services Used: AWS Site-to-Site VPN, VPC, EC2, Route Tables, Virtual Private Gateway, Customer Gateway

Goal: Securely connect an on-premises *simulated environment* (VirtualBox VM) with AWS VPC using Site-to-Site VPN.

Skills Gained: Hybrid networking, routing, VPN concepts, secure connectivity, hybrid DNS basics.

2. Architecture Overview (Conceptual)

- On-Premises Network (VirtualBox VM)
 - Example CIDR: 192.168.1.0/24
 - Acts as Customer Network
 - AWS Cloud
 - VPC CIDR: 10.0.0.0/16
 - Public subnet with EC2 instance
 - Site-to-Site VPN tunnel connecting both networks
-

3. Prerequisites

On-Premises (Local System)

- VirtualBox installed
- Linux VM (Ubuntu preferred)
- VM Network Adapter: **Bridged** or **Host-only**
- Static private IP (example: 192.168.1.10)

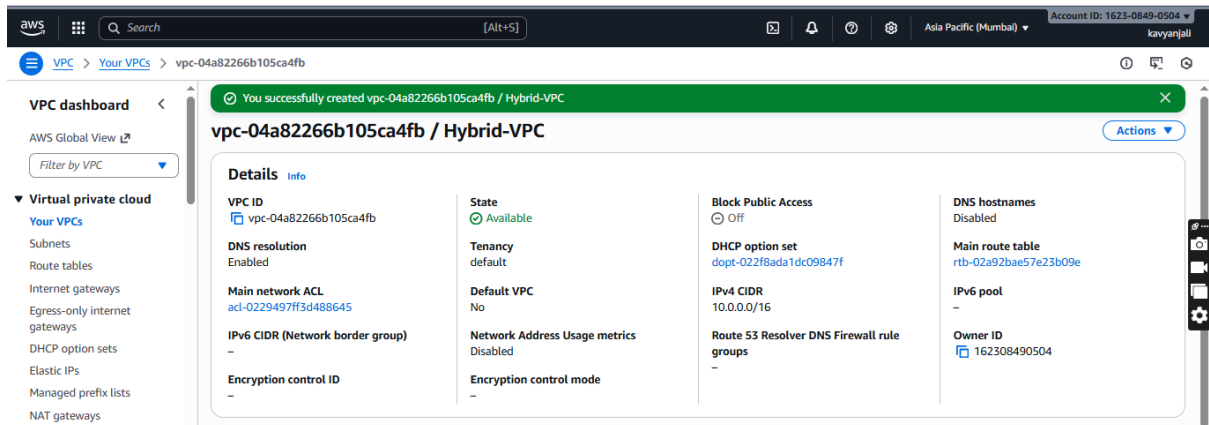
AWS Account Requirements

- AWS account
 - IAM user with VPC full access
 - Region selected (example: **ap-south-1**)
-

4. Step-by-Step AWS Console Setup

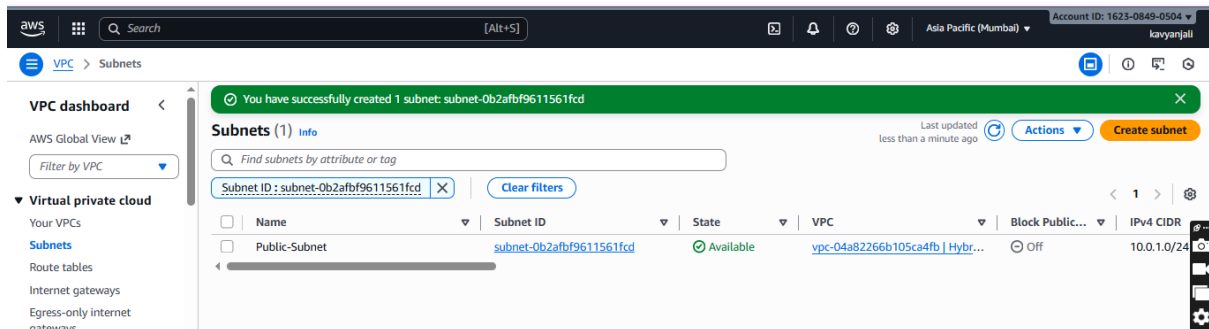
Step 1: Create a VPC

1. Go to **AWS Console** → **VPC**
2. Click **Create VPC**
3. Select **VPC only**
4. Configure:
 - Name: Hybrid-VPC
 - IPv4 CIDR: 10.0.0.0/16
5. Click **Create VPC**



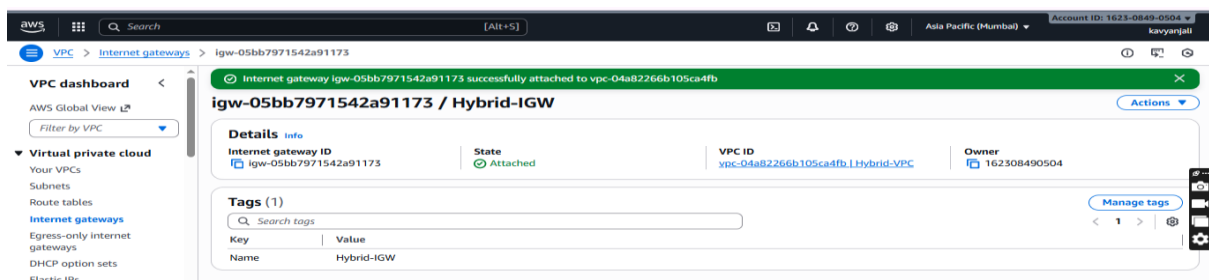
Step 2: Create a Public Subnet

1. VPC Dashboard → **Subnets** → Create subnet
2. Select VPC: Hybrid-VPC
3. Configure:
 - Subnet name: Public-Subnet
 - Availability Zone: ap-south-1a
 - CIDR: 10.0.1.0/24
4. Click **Create subnet**



Step 3: Create Internet Gateway

1. VPC → **Internet Gateways** → Create
2. Name: Hybrid-IGW
3. Click **Create internet gateway**
4. Attach to VPC:
 - Actions → Attach → Select Hybrid-VPC



Step 4: Create Route Table

1. VPC → **Route Tables** → Create
2. Name: Public-RT
3. VPC: Hybrid-VPC
4. Edit Routes:
 - Destination: 0.0.0.0/0
 - Target: Internet Gateway (Hybrid-IGW)
5. Associate with Public-Subnet

The screenshot shows the AWS Management Console interface for the 'Public-RT' route table. The left sidebar contains navigation links for VPC, Route tables, and Security. The main content area displays the 'Details' tab for the route table 'rtb-0b30b639a76bdbf59'. It shows the VPC as 'vpc-04a82266b105ca4fb | Hybrid-VPC' and the Owner ID as '162308490504'. Below the details, the 'Routes' tab is active, showing a table with two routes. The first route has a destination of '0.0.0.0/0' and a target of 'igw-05bb7971542a91173', with a status of 'Active'. The second route has a destination of '10.0.0.0/16' and a target of 'local', also with a status of 'Active'. The table includes columns for Destination, Target, Status, Propagated, and Route Origin.

Destination	Target	Status	Propagated	Route Origin
0.0.0.0/0	igw-05bb7971542a91173	Active	No	Create Route
10.0.0.0/16	local	Active	No	Create Route Table

Step 5: Launch EC2 Instance (AWS Side)

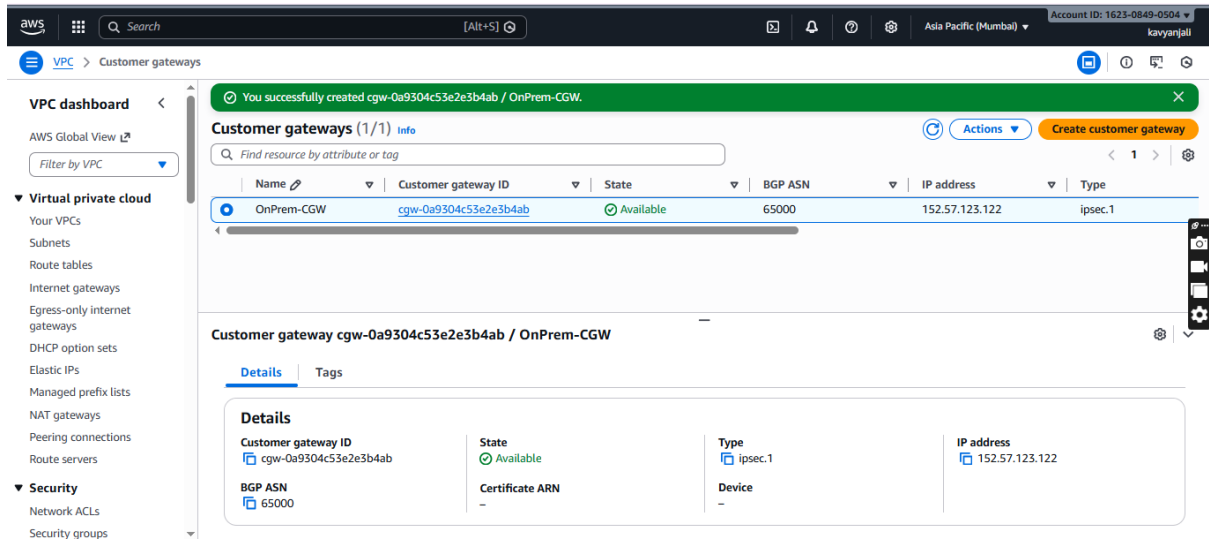
1. Go to **EC2** → **Launch Instance**
2. Choose AMI: Amazon Linux 2
3. Instance type: t2.micro
4. Network:
 - VPC: Hybrid-VPC
 - Subnet: Public-Subnet
 - Auto-assign public IP: Enabled
5. Security Group:
 - Allow SSH (22)
 - Allow ICMP (Ping)
6. Launch instance

The screenshot shows the AWS Management Console interface for the 'Instance summary' of a newly launched EC2 instance. The left sidebar contains navigation links for EC2, Instances, and Elastic Block Store. The main content area displays the 'Instance summary' for the instance 'i-0ecd62a5eb681fa0'. It shows the instance is in the 'Running' state, with a public IP address of '13.233.233.142'. The instance is associated with the 'vpc-04a82266b105ca4fb | Hybrid-VPC' and the 'subnet-0b2afb9611561fcd | Public-Subnet'. The instance type is 't2.micro'. The summary includes details about the instance ID, IP addresses, hostname, DNS names, VPC, subnet, and IAM role.

Instance ID	Public IPv4 address	Private IPv4 addresses
i-0ecd62a5eb681fa0	13.233.233.142 open address	10.0.1.40

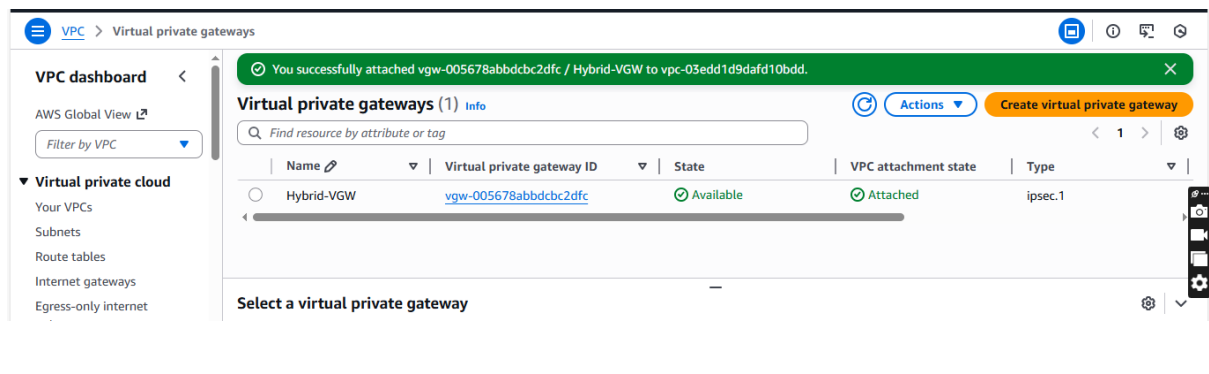
Step 6: Create Customer Gateway (On-Prem Representation)

1. VPC → **Customer Gateways** → Create
2. Configure:
 - Name: OnPrem-CGW
 - Routing: Static
 - IP Address: **Public IP of your local internet (ISP)**
3. Click **Create customer gateway**



Step 7: Create Virtual Private Gateway

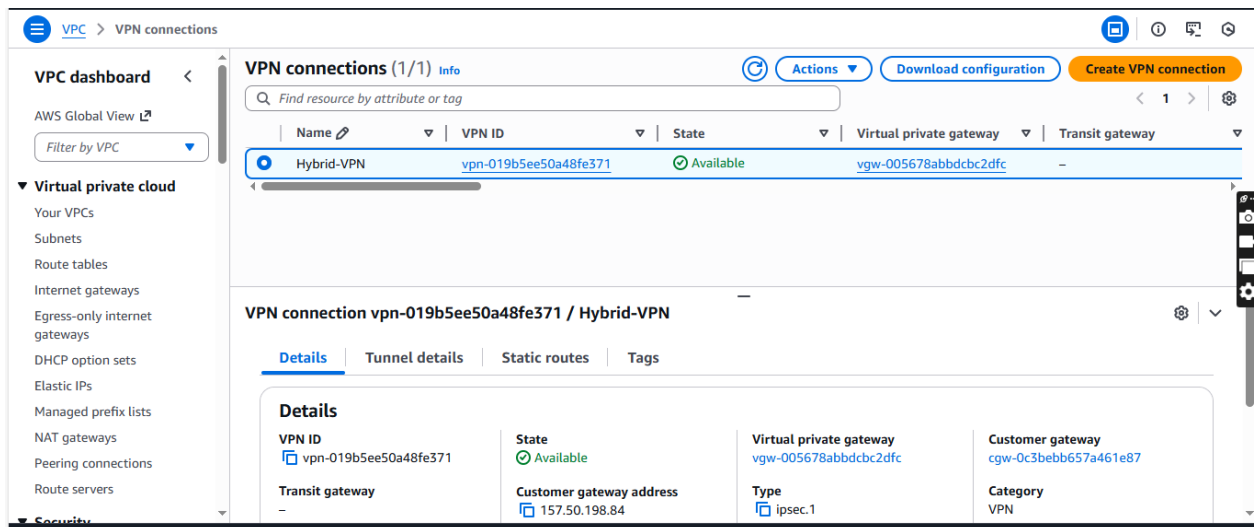
1. VPC → **Virtual Private Gateways** → Create
2. Name: Hybrid-VGW
3. ASN: Default
4. Create and Attach:
 - Attach to Hybrid-VPC



Step 8: Create Site-to-Site VPN Connection

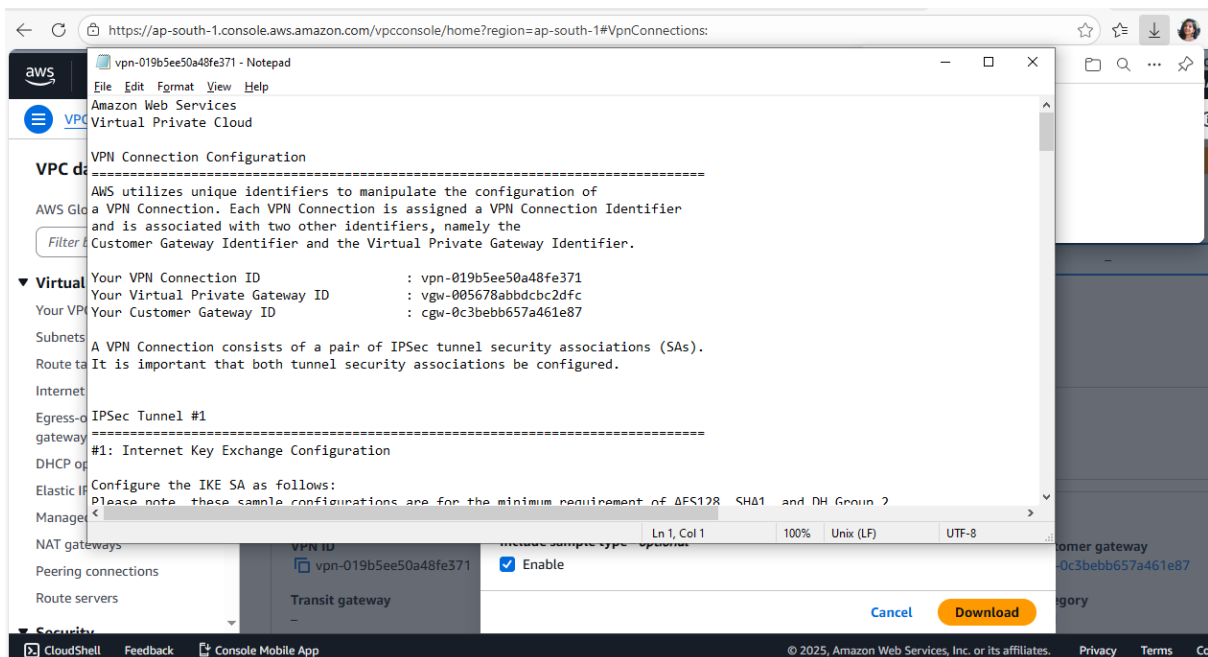
1. VPC → **Site-to-Site VPN Connections** → Create
2. Configure:
 - Name: Hybrid-VPN
 - Target Gateway Type: Virtual Private Gateway
 - Virtual Private Gateway: Hybrid-VGW
 - Customer Gateway: Existing (OnPrem-CGW)
 - Routing Options: Static

- Static IP Prefix: 192.168.1.0/24
3. Create VPN Connection



Step 9: Download VPN Configuration

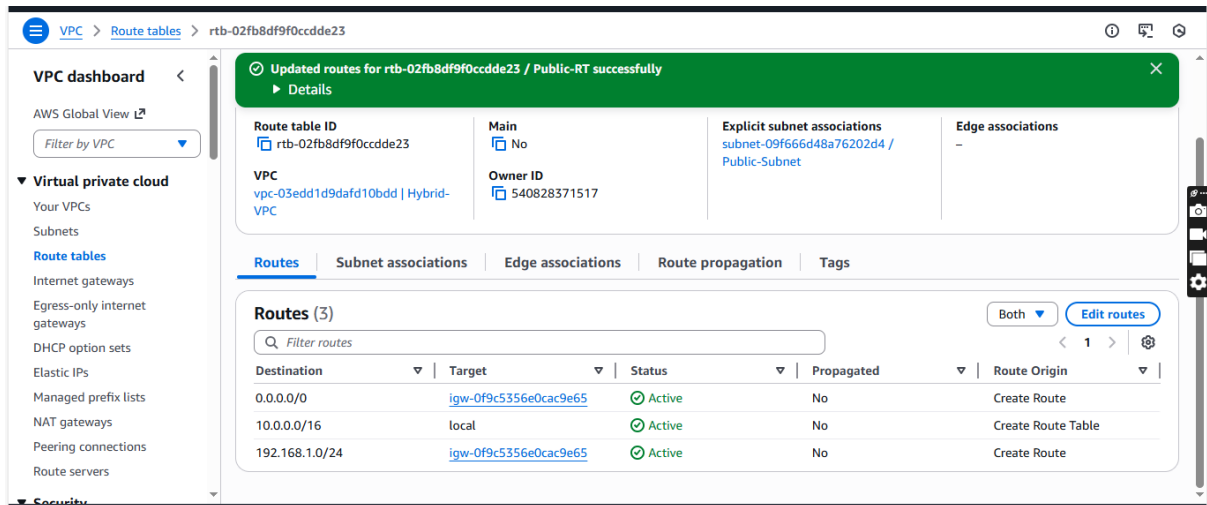
1. Select VPN connection
2. Click **Download configuration**
3. Vendor: Generic
4. Save the configuration file



Step 10: Update VPC Route Table for VPN

1. VPC → Route Tables → Select Public-RT
2. Edit routes:

- Destination: 192.168.1.0/24
 - Target: Virtual Private Gateway (Hybrid-VGW)
3. Save routes



Step 11: Verify VPN Status

Purpose

To check whether the **Site-to-Site VPN tunnel** between AWS and on-premises network is **established and working**.

Console Steps

1. Login to **AWS Management Console**
2. Go to **Services** → **VPC**
3. In the left menu, click **Site-to-Site VPN Connections**
4. Select the VPN connection named **Hybrid-VPN**
5. Scroll down to the **Tunnel details** section

What You Will See

AWS creates **2 VPN tunnels** for high availability:

Tunnel Status

Tunnel 1 UP / DOWN

Tunnel 2 UP / DOWN

Status Meaning

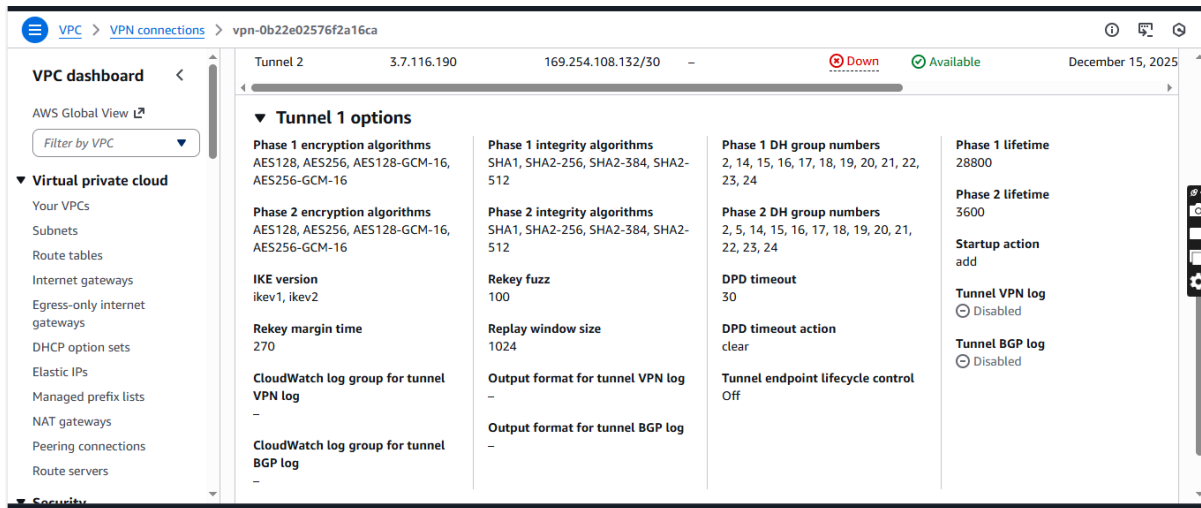
✓UP

- VPN tunnel is successfully established
- AWS can reach the on-premises gateway
- Routing and encryption are working

- Hybrid connectivity is **active**

✗DOWN

- VPN tunnel is **not established**
- AWS cannot reach the on-premises VPN device
- Common reasons:
 - On-prem VPN software not configured
 - Wrong public IP in Customer Gateway
 - ISP blocks IPsec traffic
 - Tunnel not initiated yet



12. Verification & Testing

From On-Prem VM

ping 10.0.0.10

ssh [ec2-user@10.0.0.10](#)

From EC2 Instance

ping 192.168.1.10

Successful ping confirms hybrid connectivity.

```
unbound-anchor-1.17.1-1.amzn2023.0.10.x86_64      unbound-libs-1.17.1-1.amzn2023.0.10.x86_64

Complete!
[ec2-user@ip-10-0-1-89 ~]$ ipsec --version
Libreswan 4.12
[ec2-user@ip-10-0-1-89 ~]$ sudo systemctl enable ipsec
sudo systemctl start ipsec
sudo systemctl status ipsec
Created symlink /etc/systemd/system/multi-user.target.wants/ipsec.service → /usr/lib/systemd/system/ipsec.service.
● ipsec.service - Internet Key Exchange (IKE) Protocol Daemon for IPsec
   Loaded: loaded (/usr/lib/systemd/system/ipsec.service; enabled; preset: disabled)
   Active: active (running) since Mon 2025-12-15 14:49:58 UTC; 70ms ago
     Docs: man:ipsec(8),
           man:pluto(8),
           man:ipsec.conf(5)
   Process: 28241 ExecStartPre=/usr/libexec/ipsec/addconn --config /etc/ipsec.conf --checkconfig (code=exited, status=0/SUCCESS)
   Process: 28242 ExecStartPre=/usr/libexec/ipsec/_stackmanager start (code=exited, status=0/SUCCESS)
   Process: 28625 ExecStartPre=/usr/sbin/ipsec --checknss (code=exited, status=0/SUCCESS)
   Process: 28630 ExecStartPre=/usr/sbin/ipsec --checknflag (code=exited, status=0/SUCCESS)
  Main PID: 28641 (pluto)
    Status: "Startup completed."
      Tasks: 2 (limit: 1120)
    Memory: 7.6M
       CPU: 504ms
   CGroup: /system.slice/ipsec.service
           └─28641 /usr/libexec/ipsec/pluto --leak-detective --config /etc/ipsec.conf --nofork

Dec 15 14:49:58 ip-10-0-1-89.ap-south-1.compute.internal pluto[28641]: listening for IKE messages
Dec 15 14:49:58 ip-10-0-1-89.ap-south-1.compute.internal pluto[28641]: Kernel supports NIC esp-hw-offload
Dec 15 14:49:58 ip-10-0-1-89.ap-south-1.compute.internal pluto[28641]: adding UDP interface enx0 10.0.1.89:500
```

13. Hybrid DNS

- On-prem DNS resolves local names
 - AWS Route 53 (optional) resolves cloud resources
 - Use conditional forwarding for hybrid DNS
-

14. Security Best Practices

- Restrict Security Group rules
 - Use strong VPN pre-shared keys
 - Enable CloudWatch logs for VPN
 - Rotate keys periodically
-

15. Project Outcome

- ✓ Secure hybrid connectivity established
 - ✓ On-prem and AWS networks communicate privately
 - ✓ Real-world enterprise hybrid cloud setup simulated
-

16. Conclusion

This project demonstrates a **real-world Hybrid Cloud architecture** using AWS Site-to-Site VPN. It is widely used in enterprises to extend on-prem infrastructure securely into AWS without exposing services to the public internet.
