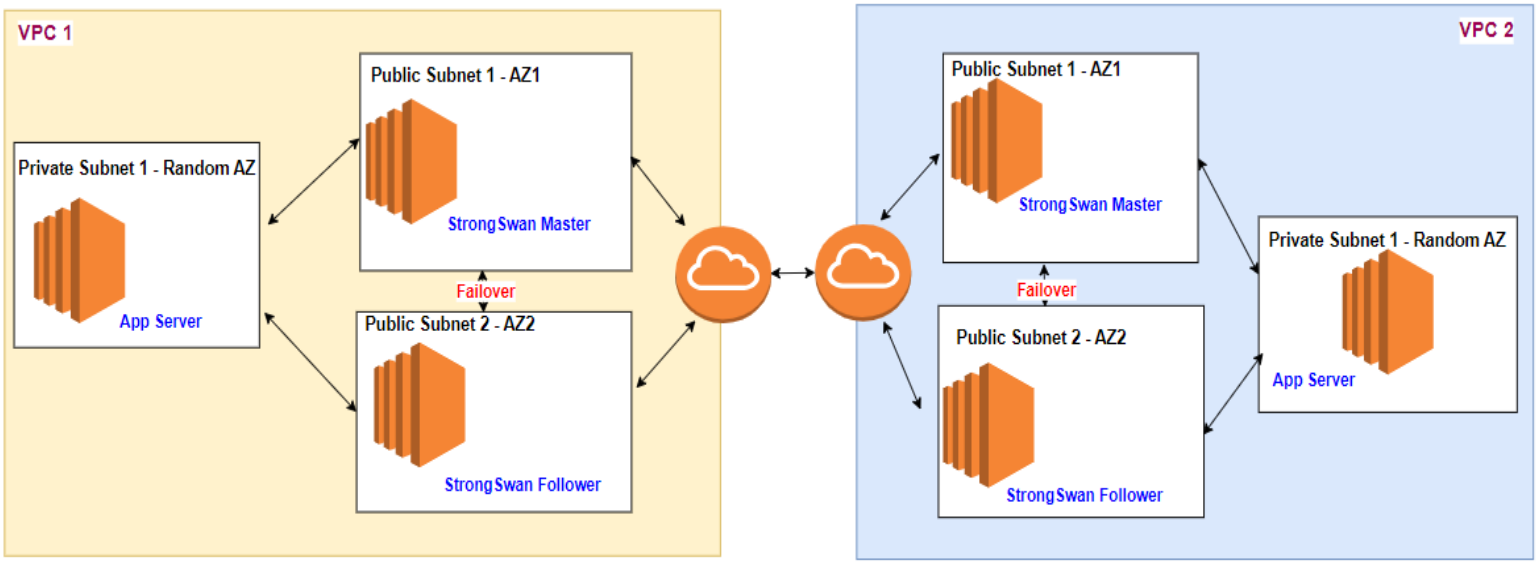
StrongSwan HA Cluster in AWS with Private IPs – Terraform Provisioning Guidelines (V2)

Janarthanan Kugathasan

Contents

Architecture



Resources Created

There are 2 Terraform modules provided.

1. VPCSite
    a. This module is used to create 1VPC with 2 Public Subnets, 1 Private Subnet
    b. Corresponding Route Tables (Public RT and Private RT)
2. EC2
    a. This module is used to create 2 EC2 Ubuntu based StrongSwan Servers (1 in each subnet – Public Subnet1, Public Subnet2)
    b. 1 EC2 in Private Subnet (Just for validating the IPsec VPN connectivity)
    c. Security group for EC2 (Allowing SSH/ICMP)
    d. Route Table to attach Strong Swan server as gateway to the other site network
    e. IAM instance profile for StrongSwan Servers (EC2) to allow them to update the route table and to run SSM document

Note- VPC Peering is created to enable/simulate connectivity between different VPC

Method

1. Fill the main.tf file
    a. Fill the provider section with the desired region and the IAM instance profile that you have created locally
    b. Since we need to create 2 VPCs for our testing, we have to call "**VPCSite**" and **"EC2**" module twice to create the required resources. Fill the parameters accordingly except in yellow highlighted
    c. Leave the resources "**openssl_random_password**" and "**aws_vpc_peering_connection**" as it is

Module Name	Variable Name	Purpose
VPCSite	source	Relative location of VPCSite Module
VPCSite	site_name	VPC Site Name
VPCSite	availability_zone1	AZ Name in that region
VPCSite	availability_zone2	AZ Name in that region
VPCSite	VPC_CIDR	VPC CIDR
VPCSite	Public_CIDR1	Public CIDR
VPCSite	Public_CIDR2	Public CIDR
VPCSite	Private_CIDR1	Private CIDR
EC2	source	Relative location of EC2 Module
EC2	site_name	Will be passed from previous module
EC2	EC2_Size	EC2 Size
EC2	AMI_ID	AMI ID (Must be Ubuntu) on that region
EC2	VPC_Id	Will be passed from previous module
EC2	Master_private_ip	Master Private IP of EC2 of primary site
EC2	Follower_private_ip	Follower Private IP of EC2 of primary site
EC2	Pre_Shared_Key	Will be passed from previous module
EC2	Public_SubnetID1	Will be passed from previous module
EC2	Public_SubnetID2	Will be passed from previous module
EC2	Private_SubnetID1	Will be passed from previous module
EC2	Primary_cidr	Will be passed from previous module
EC2	Secondary_cidr	Will be passed from previous module
EC2	Secondary_Master_private_ip	Master Private IP of EC2 of a secondary site
EC2	Secondary_Follower_private_ip	Follower Private IP of EC2 of a secondary site
EC2	Secondary_Site_name	Will be passed from previous module
EC2	Peering_ID	Will be passed from previous module

2. Run "**terraform init**"
3. Run "**terraform plan**" -> To check what all resources will be created
4. Run "**terraform apply -auto-approve**" to create all required resources
    a. It will take 2-3 minutes to create all resources
    b. But userdata script in EC2 instance will run for about 10-15 mins, **so its advisable to start your testing after 20 mins.**
    c. You can check the logs of userdata script using "sudo tail -f var/log/cloud-init-output.log"

    d. SSH is only possible via EC2-Connect method
5. Run "**terraform destroy -auto-approve**" to delete all resources. It will take 3-5 mins to finish deleting the resources

How Failover between StrongSwan takes place?

1. Keepalive daemon is configured to monitor the status of IPsec service
2. When IPsec service is down or EC2 is down, then the failover server automatically promoted to Master state
3. During Master state following actions are performed,
    a. SSM document is triggered to run on another site EC2 instances to update their IPSec config files and restart the IPSec service
    b. Private Route Table and Public Route Table will be updated with current instance id for routing