Deploying Transit VPC With Transit Gateway

Information About the Transit Gateway Solution

Amazon Virtual Private Cloud (Amazon VPC) provides you with the ability to create as many virtual networks as you need. AWS also provides different options for connecting these networks to each other and to non-AWS infrastructure, such as on-premises data centres, remote headquarters, or other offices.

When you deploy a Cisco Catalyst 8000V instance with the Transit VPC solution, you can build a hub-and-spoke topology on Amazon VPCs to centralize edge connectivity. Transit VPC allows you to implement shared services or packet inspection/replication in a VPC. It works across accounts and is easy to set up through an AWS CloudFormation stack. However, there is some level of complexity while adding a new spoke as this solution uses a VPN Gateway as opposed to the Transit Gateway.

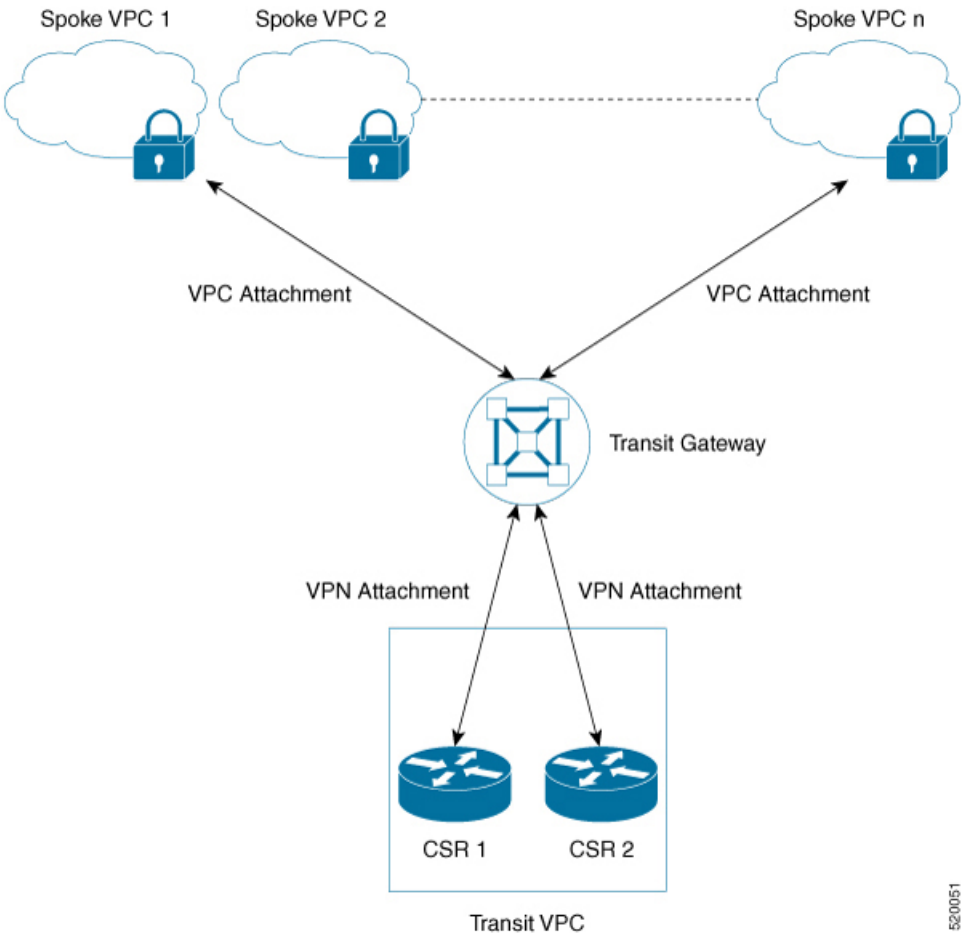
To overcome this limitation, you can now deploy a Cisco Catalyst 8000V Transit VPC with the Transit Gateway solution. A transit gateway is a regional network transit hub service provided by AWS to interconnect your VPCs in AWS cloud and on-premise network. In the Cisco Catalyst 8000V transit VPC with transit gateway solution, you use a transit gateway on the spoke side to provide connectivity between all the spoke VPCs in the same region. The transit gateway is attached to two Cisco Catalyst 8000V instances in the transit VPC using a VPN attachment. The Cisco Catalyst 8000V instance provides VPN connectivity to various on-premise branch locations.

To know how to deploy the AWS Transit VPC with Transit Gateway solution, perform the configuration steps as mentioned in this chapter.

Transit VPC-Transit Gateway Components

The Transit Gateway solution has a transit gateway that acts as a hub for providing spoke-to-spoke VPC connectivity. The transit VPC is another core component that acts as the central hub for traffic flowing from any spoke VPC to a remote network. The transit VPC hosts two Cisco Catalyst 8000V instances that allow VPN termination and routing.

Figure 1: Sample Topology of the Transit Gateway Solution



This solution uses two AWS Lambda functions, the Solution Helper, and the Cisco Configurator to automatically configure the VPN connections between these instances and the spoke VPCs.

- **Solution Helper Lambda**: This component is triggered when you deploy the cloudformation template. This component creates the transit gateway, the VPN connections with the Cisco Catalyst 8000V instances, and the VPN attachment between the instances and the transit gateway. The lambda function then saves the VPN connection information to the Amazon S3 bucket using S3 SSE-KMS.
- **Cisco Configurator Lambda**: The S3 Put event invokes the Cisco Configurator Lambda function which parses the VPN connection information and generates the necessary configuration files to create new VPN connections. The Cisco Configurator Lambda pushes the IOS configuration to the Cisco Catalyst 8000V instances using SSH. As soon as the Cisco configuration is applied on the Cisco Catalyst 8000V instances, the VPN tunnels come up and the Border Gateway Protocol (BGP) neighbour relationships are established with the transit gateway.

Benefits of the AWS Transit Gateway Solution

- The Transit Gateway solution is scalable and resilient.
- The Transit Gateway solution is a managed service. That is, high availability and monitoring is built-in, and you can track the solution using metrics like CloudWatch.
- By using the Transit Gateway solution, you can simplify your network architecture, thereby reducing the operational cost.
- You can centrally manage your solution, including security.

Prerequisites to the AWS Transit Gateway Solution

- You must have sufficient Elastic IP, VPC, TGW and VPN connection limits.
- Ensure that you have IAM permission to manage the *cloudformation* service.

Limitations of the AWS Transit Gateway Solution

- Autoscaling is not supported with this version of the solution.
- You must manually add the spoke VPCs to the Transit Gateway through VPC attachments after you deploy this solution.

Configuring the AWS Transit Gateway Solution

Step 1 Log in to the Amazon Web Services Marketplace.

Step 2 Search the **Cisco Catalyst 8000V – Transit Network VPC** template and select the template.

Step 3 Launch the template in the appropriate region where you are located. The system displays the AWS Cloudformation Service page. Click **Next**.

Step 4 Specify the following **Stack Details**:

Parameter	Description
C8000V Throughput Requirements	The required throughput for the Cisco Catalyst 8000V instance. This determines the instance type to be launched. The default value is 2 x 500 Mbps.
SSH Key to access C8000V	The public/private key pair that allows a secure connection to be made to a Cisco Catalyst 8000V instance after you launch the instance.

Parameter	Description
	You must enter a public/private key pair. The key pair is created in your preferred region at the time when you created the AWS account.
License Model	BYOL is the only license model that is currently supported.
Enable Termination Protection	Enable this field to enable termination protection for the Cisco Catalyst 8000V instances. This prevents accidental Cisco Catalyst 8000V termination. Cisco recommends that you enable this field for production deployments. By default, this field is set to **Yes**.
Prefix for S3 Objects	The text string that you need to use as a prefix when Amazon S3 objects are created. By default, the value is **vpnconfigs/**.
Additional AWS Account ID	The account ID of an AWS account associated with the transit network which allows access to the S3 bucket and the AWS KMS customer master key. **Note** You can only enter one additional AWS account ID in this field. If you want to connect more than one additional AWS account to the transit network, you must manually configure the permissions for the additional accounts.
Transit VPC CIDR Block	The CIDR block for the transit VPC. Modify the VPC and subnet CIDR address ranges to avoid collisions with your network. By default, the value is **100.64.127.224/27**.
1st Subnet Network	The CIDR block for the transit VPC subnet created in AZ1. By default, the value is **100.64.127.224/28**.
2nd Subnet Network	The CIDR block for the transit VPC subnet created in AZ2. By default, the value is **100.64.127.240/28**.
Transit VPC BGP ASN	The BGP Autonomous System Number (ASN) for the transit VPC. By default, the value is **64512**.
Spoke VPC Tag Name	The tag to use to identify the spoke VPCs to connect to the Transit VPC.
Preferred VPN Endpoint Tag Name	The tag to use to configure a preferred Cisco Catalyst 8000V VPN endpoint to control the traffic flow through the Transit VPC Cisco Catalyst 8000V instances. For example, when integrating with stateful on-prem firewalls.
Optional AZ configuration 1st Subnet	The availability Zone number for Public Subnet1.
Optional AZ configuration 2nd Subnet	The availability Zone number for Public Subnet2.

Step 5 Review and confirm the settings. Select the check box to acknowledge that resources might be created by the AWS Identity and Access Management (IAM) and CAPABILITY_AUTO_EXPAND capabilities might be required.

Step 6 Click **Create** to deploy the stack.
If the deployment is successful, the **Status** column in the AWS Cloud Formation console displays **CREATE_COMPLETE**.

Configuration Example

The following is a configuration example of deploying the AWS Transit VPC with Transit Gateway solution:

```
ip-100-64-127-234#sh run
Building configuration...

Current configuration : 7284 bytes
!
! Last configuration change at 14:10:57 UTC Thu Oct 10 2020
!
version 17.4
service timestamps debug datetime msec
service timestamps log datetime msec
service password-encryption
platform qfp utilization monitor load 80
no platform punt-keepalive disable-kernel-core
platform console virtual
!
hostname ip-100-64-127-234
!
boot-start-marker
boot-end-marker
!
!
vrf definition GS
 rd 100:100
 !
 address-family ipv4
 exit-address-family
!
logging persistent size 1000000 filesize 8192 immediate
!
no aaa new-model
!
ip vrf vpn-0f56b2afc60b1d492
 rd 64525:1
 route-target export 64525:0
 route-target import 64525:0
!
ip vrf vpn0
 rd 64525:0
!
ip admission watch-list expiry-time 0
!
subscriber templating
!
multilink bundle-name authenticated
!
crypto pki trustpoint TP-self-signed-572041569
 enrollment selfsigned
subject-name cn=IOS-Self-Signed-Certificate-572041569
```

```
 revocation-check none
 rsakeypair TP-self-signed-572041569
!
!
crypto pki certificate chain TP-self-signed-572041569
 certificate self-signed 01
  3082032E 30820216 A0030201 02020101 300D0609 2A864886 F70D0101 05050030
  30312E30 2C060355 04031325 494F532D 53656C66 2D536967 6E65642D 43657274
  69666963 6174652D 35373230 34313536 39301E17 0D313931 30313031 34303631
  355A170D 33303031 30313030 30303030 5A303031 2E302C06 03550403 1325494F
  532D5365 6C662D53 69676E65 642D4365 72746966 69636174 652D3537 32303431
  35363930 82012230 0D06092A 864886F7 0D010101 05000382 010F0030 82010A02
  82010100 A974EDB7 292BBB6A 09026F6A 381F7852 714775E3 E25F1F89 CED40FCB
  F45204F9 2F2F5FEE C46A9D16 A8D7307A C5433234 10D3F709 B4B18B3D 009B4A7A
  85980EEB 1282D1F7 C3CD4429 16042D4D 544315F4 E3ABA673 21E66C52 187AD1E6
  6B21F98A F0537D0A 8171618E 6CDF3B70 E2C8B553 8096C2D6 B4CD1AE4 B6DFD615
  844924B8 83DBE166 3CBC90F1 889CB00F 1644ECCE F2E70D81 CA35B555 D9757BE4
  34440FD9 D15580FA C50181CD D646AB6C 22F707A7 1D9F98CA 19897AF4 7488762B
  35ECA78F D2B249C7 8079255F 72BE5CF8 214B5135 E97B1104 A9CB449E A4A1D996
  9B99EC0E 18EF94FE FE73706A BF417262 12771D33 FF61A325 4479CAFB 10D0EEAA
  810E3437 02030100 01A35330 51300F06 03551D13 0101FF04 05300301 01FF301F
  0603551D 23041830 16801476 E85FEE9B EAE114A4 74C542FD E923856D 6F17F830
  1D060355 1D0E0416 041476E8 5FEE9BEA E114A474 C542FDE9 23856D6F 17F8300D
  06092A86 4886F70D 01010505 00038201 010043A6 03287F7E 1F13A7D4 26D661FE
  D11FED41 FE195D3E 6ADEA111 267C534B 266F587A 6A2F395D C50F5894 4C01F62B
  A179B852 F5F8ED62 DFF35587 3CFF352C 523F8D3D 8A786E61 A73EA8BB C8FC0A8D
  C2F0C260 0BB25D28 01B26B2B 27D71A31 2CE81DA5 6296D4AA 756A6658 0ADB89FB
  52BE1E9F A8BF17AA B2A0379A 1921AF64 834455CF B6307205 CE12C83A 2D29AEF2
  D79B79F7 9701F86E EB51B8E2 95BA7D5A C67A05F8 2AA7A8E0 3626D155 FC2D79EC
  9506D897 D79B8E65 A1D89F8A 6EC21FD1 15BFBD79 8A6FEB77 15C10DEE 0A50A7A5
  C8109573 9C58A869 D2740BC4 61D953F2 7AA92870 69BF035C 08DA0EFB B4AB9AC1
  BD4DB053 66ADD9E3 B5957D2B 8E467A91 258A
   quit
!
license udi pid CSR1000V sn 9YGGWBVUY3N
no license smart enable
diagnostic bootup level minimal
!
spanning-tree extend system-id
!
username ec2-user privilege 15 secret 5 $1$Gf9p$OfANl/ujuCIvpunuRDwKi1
username automate privilege 15 secret 8
$8$g62y2e1pz0O4/n$M8DmVAM/G9yySvjbB1I2tBJAW4IWZRIc44Icent4bps
!
redundancy
!
crypto keyring keyring-vpn-0f56b2afc60b1d492-2
  local-address GigabitEthernet1
  pre-shared-key address 52.54.79.47 key lhvPlpTYxUTno.lNTbR25F9743HEguaH
crypto keyring keyring-vpn-0f56b2afc60b1d492-1
  local-address GigabitEthernet1
  pre-shared-key address 52.44.80.94 key Qq4fLolOMf1iW3d7gJhtzF8h8Tu3IlNT
!
crypto isakmp policy 200
 encr aes
 authentication pre-share
 group 2
 lifetime 28800
crypto isakmp keepalive 10 10 periodic
crypto isakmp profile isakmp-vpn-0f56b2afc60b1d492-1
   keyring keyring-vpn-0f56b2afc60b1d492-1
   match identity address 52.44.80.94 255.255.255.255
   local-address GigabitEthernet1
   rekey
```

```
crypto isakmp profile isakmp-vpn-0f56b2afc60b1d492-2
   keyring keyring-vpn-0f56b2afc60b1d492-2
   match identity address 52.54.79.47 255.255.255.255
   local-address GigabitEthernet1
   rekey
!
crypto ipsec security-association replay window-size 1024
!
crypto ipsec transform-set ipsec-prop-vpn-aws esp-aes esp-sha-hmac
 mode tunnel
crypto ipsec df-bit clear
no crypto ipsec nat-transparency udp-encapsulation
!
crypto ipsec profile ipsec-vpn-aws
 set transform-set ipsec-prop-vpn-aws
 set pfs group2
!
interface Tunnel1
 description vpn-0f56b2afc60b1d492 from TGW to cgw-00d8fbb76cc59295e for account 902347396780

 ip vrf forwarding vpn-0f56b2afc60b1d492
 ip address 169.254.185.70 255.255.255.252
 ip tcp adjust-mss 1387
 tunnel source GigabitEthernet1
 tunnel mode ipsec ipv4
 tunnel destination 52.44.80.94
 tunnel protection ipsec profile ipsec-vpn-aws
 ip virtual-reassembly
!
interface Tunnel2
 description vpn-0f56b2afc60b1d492 from TGW to cgw-00d8fbb76cc59295e for account 902347396780

 ip vrf forwarding vpn-0f56b2afc60b1d492
 ip address 169.254.232.90 255.255.255.252
 ip tcp adjust-mss 1387
 tunnel source GigabitEthernet1
 tunnel mode ipsec ipv4
 tunnel destination 52.54.79.47
 tunnel protection ipsec profile ipsec-vpn-aws
 ip virtual-reassembly
!
interface VirtualPortGroup0
 vrf forwarding GS
 ip address 192.168.35.101 255.255.255.0
 ip nat inside
 no mop enabled
 no mop sysid
!
interface GigabitEthernet1
 ip address 100.64.127.234 255.255.255.240
 ip nat outside
 negotiation auto
 no mop enabled
 no mop sysid
!
router bgp 64525
 bgp log-neighbor-changes
 !
 address-family ipv4 vrf vpn-0f56b2afc60b1d492
  neighbor 169.254.185.69 remote-as 64526
  neighbor 169.254.185.69 timers 10 30 30
  neighbor 169.254.185.69 activate
  neighbor 169.254.185.69 next-hop-self
  neighbor 169.254.185.69 default-originate
```

```
  neighbor 169.254.185.69 as-override
  neighbor 169.254.185.69 soft-reconfiguration inbound
  neighbor 169.254.232.89 remote-as 64526
  neighbor 169.254.232.89 timers 10 30 30
  neighbor 169.254.232.89 activate
  neighbor 169.254.232.89 next-hop-self
  neighbor 169.254.232.89 default-originate
  neighbor 169.254.232.89 as-override
  neighbor 169.254.232.89 soft-reconfiguration inbound
 exit-address-family
!
iox
ip nat inside source list GS_NAT_ACL interface GigabitEthernet1 vrf GS overload
ip forward-protocol nd
ip tcp window-size 8192
ip http server
ip http authentication local
ip http secure-server
ip route 0.0.0.0 0.0.0.0 GigabitEthernet1 100.64.127.225
ip route vrf GS 0.0.0.0 0.0.0.0 GigabitEthernet1 100.64.127.225 global
!
ip ssh rsa keypair-name ssh-key
ip ssh version 2
ip ssh pubkey-chain
  username ec2-user
   key-hash ssh-rsa F1B0DF92FC2E25F7D98A01B99FCE5F13 ec2-user
  username automate
   key-hash ssh-rsa ED4B0757CE2AC22C89B28BE55EDE7691
ip ssh server algorithm authentication publickey
ip scp server enable
!
ip access-list standard GS_NAT_ACL
 permit 192.168.35.0 0.0.0.255
!
control-plane
!
line con 0
 stopbits 1
line vty 0 4
 login local
 transport input ssh
!
app-hosting appid guestshell
 app-vnic gateway1 virtualportgroup 0 guest-interface 0
  guest-ipaddress 192.168.35.102 netmask 255.255.255.0
 app-default-gateway 192.168.35.101 guest-interface 0
 name-server0 8.8.8.8
end
```