Transit Virtual Private Cloud Deployment Guide using Cisco CSR 1000v for Amazon Web Services

First Published: 2017-08-14

Last Modified: 2019-11-15

Americas Headquarters
Cisco Systems, Inc.
170 West Tasman Drive
San Jose, CA 95134-1706
USA
http://www.cisco.com
Tel: 408 526-4000
800 553-NETS (6387)
Fax: 408 527-0883

THE SPECIFICATIONS AND INFORMATION REGARDING THE PRODUCTS IN THIS MANUAL ARE SUBJECT TO CHANGE WITHOUT NOTICE. ALL STATEMENTS, INFORMATION, AND RECOMMENDATIONS IN THIS MANUAL ARE BELIEVED TO BE ACCURATE BUT ARE PRESENTED WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED. USERS MUST TAKE FULL RESPONSIBILITY FOR THEIR APPLICATION OF ANY PRODUCTS.

THE SOFTWARE LICENSE AND LIMITED WARRANTY FOR THE ACCOMPANYING PRODUCT ARE SET FORTH IN THE INFORMATION PACKET THAT SHIPPED WITH THE PRODUCT AND ARE INCORPORATED HEREIN BY THIS REFERENCE. IF YOU ARE UNABLE TO LOCATE THE SOFTWARE LICENSE OR LIMITED WARRANTY, CONTACT YOUR CISCO REPRESENTATIVE FOR A COPY.

The Cisco implementation of TCP header compression is an adaptation of a program developed by the University of California, Berkeley (UCB) as part of UCB's public domain version of the UNIX operating system. All rights reserved. Copyright © 1981, Regents of the University of California.

NOTWITHSTANDING ANY OTHER WARRANTY HEREIN, ALL DOCUMENT FILES AND SOFTWARE OF THESE SUPPLIERS ARE PROVIDED “AS IS" WITH ALL FAULTS. CISCO AND THE ABOVE-NAMED SUPPLIERS DISCLAIM ALL WARRANTIES, EXPRESSED OR IMPLIED, INCLUDING, WITHOUT LIMITATION, THOSE OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT OR ARISING FROM A COURSE OF DEALING, USAGE, OR TRADE PRACTICE.

IN NO EVENT SHALL CISCO OR ITS SUPPLIERS BE LIABLE FOR ANY INDIRECT, SPECIAL, CONSEQUENTIAL, OR INCIDENTAL DAMAGES, INCLUDING, WITHOUT LIMITATION, LOST PROFITS OR LOSS OR DAMAGE TO DATA ARISING OUT OF THE USE OR INABILITY TO USE THIS MANUAL, EVEN IF CISCO OR ITS SUPPLIERS HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

Any Internet Protocol (IP) addresses and phone numbers used in this document are not intended to be actual addresses and phone numbers. Any examples, command display output, network topology diagrams, and other figures included in the document are shown for illustrative purposes only. Any use of actual IP addresses or phone numbers in illustrative content is unintentional and coincidental.

All printed copies and duplicate soft copies of this document are considered uncontrolled. See the current online version for the latest version.

Cisco has more than 200 offices worldwide. Addresses and phone numbers are listed on the Cisco website at www.cisco.com/go/offices.

CONTENTS

CHAPTER 1

Introduction to Deploying Transit VPC for Amazon Web Services

This section contains the following topics:

Information About Transit VPC for Amazon Web Services

The Transit VPC design in the Amazon Web Services (AWS) marketplace uses multiple instances of the Cisco CSR 1000v. The Transit VPC design provides secure transit routing between spoke Virtual Private Clouds (VPCs) and the public internet or private data center. A transit VPC acts as a global network transit center, which allows a common strategy to be used to connect multiple, geographically disperse VPCs and remote networks. This can save time and effort and reduce costs, as it is implemented virtually without the traditional expense of establishing a physical presence in a colocation transit hub or deploying physical network gear. The Transit VPC design takes advantage of Cisco routing and security features.

Cisco DMVPN uses a centralized architecture to provide easier implementation and management for deployments that require granular access controls for diverse user communities, including mobile workers, telecommuters, and extranet users. Cisco DMVPN allows branch locations to communicate directly with each other over the public WAN or Internet, such as when using voice over IP (VOIP) between two branch offices. However, it doesn't require a permanent VPN connection between sites. Cisco DMVPN enables the zero-touch deployment of IPsec VPNs and improves network performance by reducing latency and jitter, while optimizing head office bandwidth utilization. The Cisco DMVPN solution is widely used to connect data centers and branches. The branches can exist in a public cloud environment such as AWS. The transit VPC design allows for automated DMVPN deployment between the public cloud and the private data center. If you use the Cisco CSR 1000v in the AWS cloud, you can take advantage of the functionality of enterprise-class networking services and VPNs that provide flexibility and security. The transit VPC network is treated as a spoke and is connected to a hub to form part of the DMVPN network. The transit VPC network communicates directly with other spokes, whether the spokes are in physical branch locations, or in private / public clouds.

The decision to use a transit VPC network is determined by your needs to provide connectivity between VPCs in the same account or different accounts and to provide connectivity between the public cloud, private data center and internet. The advantage of using a transit VPC network is that every time a new VPC is deployed, there is no need for manual intervention to provide connectivity—the VPC is automatically connected to the rest of the network. This document shows how you can deploy a new VPC using an AWS CloudFormation template—see Launching a Transit VPC Hub, on page 6.

The cost of using a Cisco transit VPC network design depends upon your choice of instance type, type of license and whether the Cisco CSR 1000v spoke VPCs are deployed in High Availability mode. For more information about the cost of instances, refer to the AWS website. You can buy a Bring Your Own License (BYOL) type license directly from Cisco and choose the licensing package that you need. As the transit VPC network requires IPsec, BGP and BFD, you must obtain either a Cisco Security or AX Technology Package License for each Cisco CSR 1000v.

A transit VPC network simplifies network management and minimizes the number of connections required to connect multiple VPCs and remote networks. Using a transit VPC design with Cisco CSR 1000v routers can save time, effort and money compared to using a network with physical networking gear in a colocation transit hub.

The three main components in the transit VPC design are summarized in the list below. (The processes for launching transit VPC hub, spoke VPC and DMVPN are described later in Deploying Transit VPC for Amazon Web Services, on page 5.)

1. Transit VPC hub—two Cisco CSR 1000v's are transit routers that connect to "spoke VPC" routers.

   The transit VPC hub controls outward traffic flow; for example, between a spoke VPC and another VPC or remote network. The hub has two Cisco CSR 1000v instances, which allow for VPN termination and routing. Each instance is in a separate Availability Zone.

   For details about launching the transit VPC hub, see Launching a Transit VPC Hub, on page 6. In the procedure, you use the AWS CloudFormation "transit-vpc-template" to enter values for bootstrapping the AWS infrastructure and automating the deployment of a transit VPC on the AWS Cloud. You can customize the network configuration by adjusting the template parameter values. For example, you can specify any of the available size options for the Cisco CSR 1000v, based on the required network bandwidth.

2. Spoke VPC—a Cisco CSR 1000v that is connects to the transit hub VPC using a dynamically routed VPN connection.

   The VPN connections of spoke VPCs allow the spoke VPCs to use routing and failover capabilities to maintain highly available network connections. IPSec tunnels provide connectivity between spoke VPCs.

3. DMVPN—dynamically routed VPN connections between private data center, branch networks and spoke VPCs.

   If you already have a DMVPN network with a hub on premise, with spokes for the branches and you would like to expand the branches into the public cloud, you can connect a transit VPC cloud network with your existing DMVPN network.

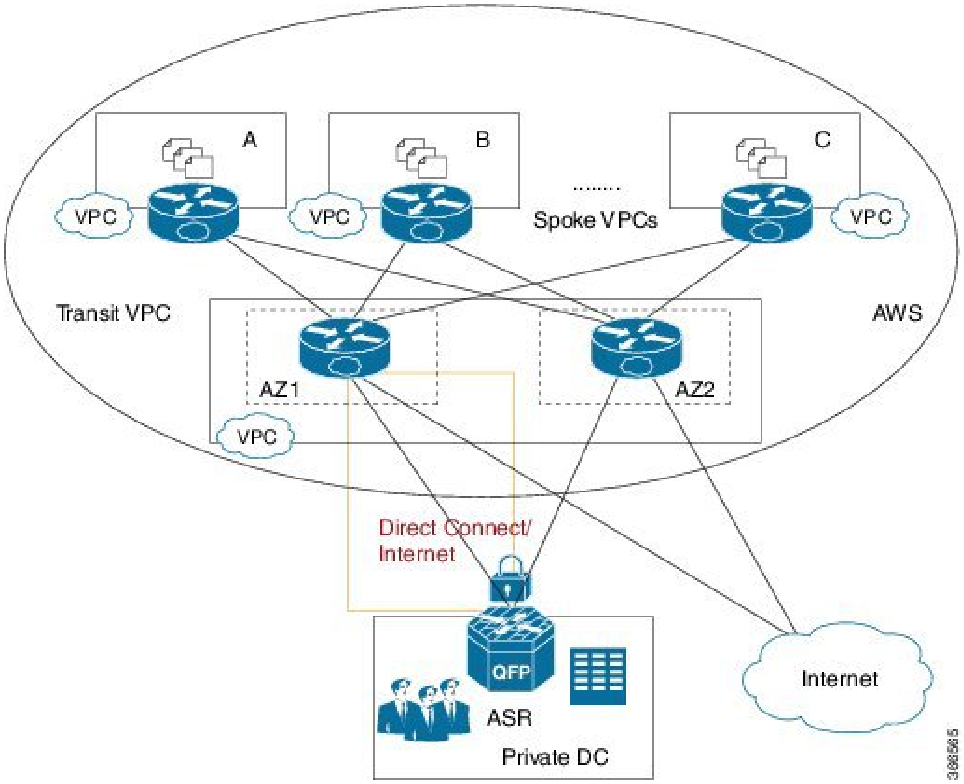
Transit VPC Hub and Spoke VPCs

The transit VPC hub, which uses two Cisco CSR 1000v's as transit routers, connects to spoke VPC's (Cisco CSR 1000v's). An example topology is shown in the figure below. The two transit VPC hub routers are shown in availability zones AZ1 and AZ2.

To deploy transit VPC hub and spoke VPC's, enter values in a template as described in Launching a Transit VPC Hub, on page 6.

The following figure shows an example transit VPC design in which a transit hub has two Cisco CSR 1000v's (CSR-A and CSR-B).

Figure 1: Transit VPC with Cisco CSR 1000v



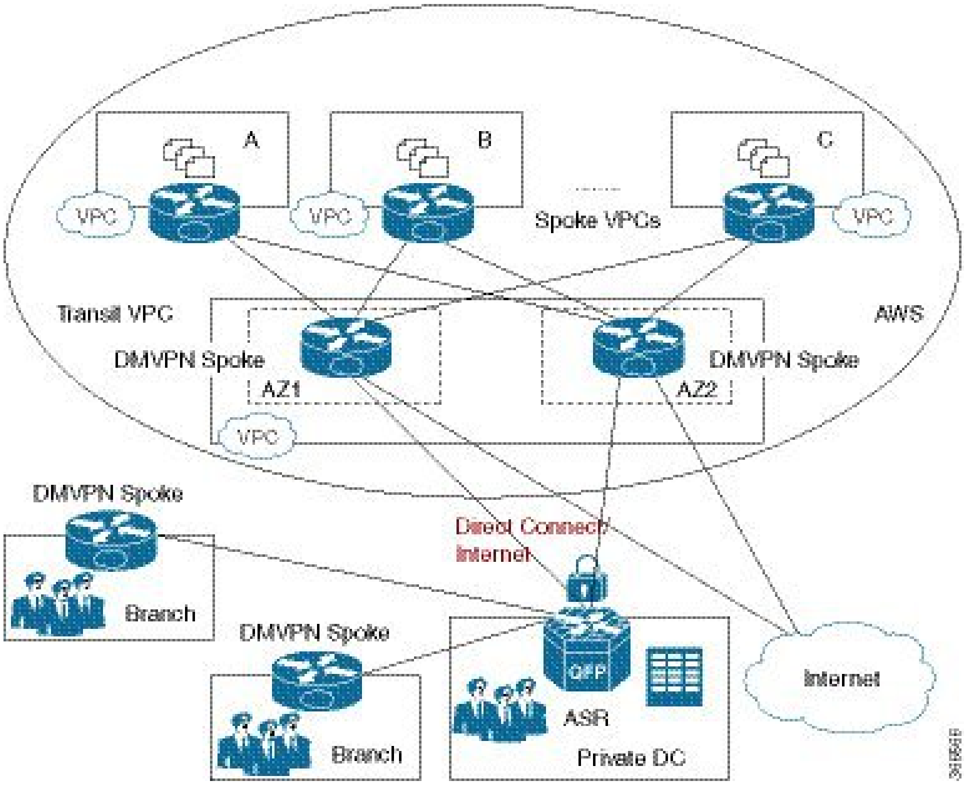
DMVPN Transit VPC

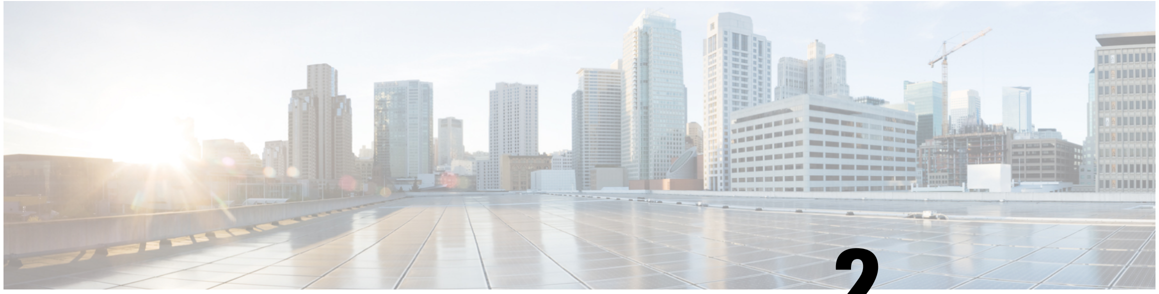
In a DMVPN transit VPC design, DMVPN is used to provide dynamically routed VPN connections between the data center, branch networks, transit VPC (hub) and spoke VPCs. Failover capabilities provide highly available network connections to transit VPC instances. Connectivity between the private data center and the public cloud is over the internet; however, it is protected by IPSec.

To deploy DMVPN, enter values in a template as described in Launching DMVPN for Transit VPC, on page 11.

The following figure shows an example DMVPN Transit VPC design, with a private data center hub, two branch networks (DMVPN hubs) and transit VPC/spoke VPCs.

Figure 2: DMVPN with Transit VPC





CHAPTER 2

Deploying Transit VPC for Amazon Web Services

This section contains the following topics:

Information About Deploying Transit VPC

Transit VPC acts as a hub for traffic flowing to another destination such as a VPC or a remote network. The following list summarizes the three main components for deploying the transit VPC design.

1. **Launching a Transit VPC Hub**

   This procedure deploys the transit VPC hub, which acts as the central hub for traffic flowing to other destinations (other VPCs or remote networks). The transit VPC hub hosts two Cisco CSR 1000v instances, which allow for VPN termination and routing. For more information, see Launching a Transit VPC Hub, on page 6.

2. **Launching a Spoke VPC**

   This procedure creates a spoke VPC, which connects to the transit VPC hub through dynamically routed VPN connections. The VPN connections of spoke VPCs allow the spoke VPCs to use routing and failover capabilities to maintain highly available network connections. To know how to launch a Spoke VPC, see Launching a Spoke VPC, on page 9.

3. **Launching DMVPN for Transit VPC**

   (Optional) This procedure launches Dynamic Multipoint VPN (DMVPN), which connects the transit VPC network to a private DMVPN hub. DMVPN is a combination of GRE, NHRP, and IPsec. The transit VPC hub is treated as a DMVPN spoke. Follow the steps in the procedure: Launching DMVPN for Transit VPC, on page 11.

Note A Cisco CSR 1000V instance is deployed on AWS by using a CloudFormation template that attaches interfaces through ENIAttachment objects. When using a CloudFormation template to deploy a Cisco CSR 1000V instance on AWS, ensure that you attach the interfaces directly to the instance as part of the Instance object definition, rather than using an ENIAttachment object separately.

Note For the current version of all CSR transit VPC, if a CSR 1000v instance is down and a new spoke comes up, and is recovered, the new spoke in the transit VPC might not get configured with IPSec tunnel configuration. Do not stop a transit CSR instance voluntarily as the instances are deployed in pair to provide High Availability for your traffic through the cloud.

If a transit VPC CSR 1000v instance goes down, the autoscaling feature detects this and spins up a replacement CSR. To know more about enabling Autoscaler in a transit VPC solution, see Deploying Transit VPC with Autoscaling.

How to Deploy Transit VPC for DMVPN

Launching a Transit VPC Hub

This is the first procedure for launching the transit VPC for DMVPN—launching a transit VPC hub.

Before you begin

Note Before following the procedures below, the following two prerequisites are needed:

1. Review your current network architecture, configuration, and security, including any existing VPCs and DMVPN configurations.
2. Decide on which of the following two licensing models to use for each Cisco CSR 1000v.
   - The Bring Your Own License (BYOL) model—for maximum performance.
   - The "License Included" model—"Cisco Cloud Services Router (CSR) 1000v - AX Pkg. Max Performance".

     Under the "License Included" model, you can choose to have an "hourly" license. If you have an issue with an hourly license you first contact AWS and then AWS contacts Cisco (depending upon the severity of the issue).

Procedure

Step 1 Go to the following github location: https://github.com/csr1000v/transit_vpc_all_csr.

Step 2 In the "Readme" section, click **Launch Stack** under **Launching a Transit VPC Hub**.

Step 3 In the "Choose a Template" section, check **Specify an Amazon S3 template URL**. (Notice that a link to the S3 template is preconfigured.) Click **Next**.

Step 4 Enter the template parameters in the following list.

Table 1: Parameters for Launching a Transit VPC

Parameter	Description
Stack name	Name of this transit VPC or "stack".
CSR Throughput Requirements	Required throughput for the CSR 1000v instance. This determines the instance type to be launched. Default: 2 x 500 Mbps
SSH Key to access CSR	Public/private key pair which allows a secure connection to be made to a CSR 1000v instance after it has launched. You must enter a public/private key pair. (The key pair was created in your preferred region at the time when the AWS account was created.)
License Model	The license model can be either LicenseIncluded or BYOL. Default: LicenseIncluded
Enable Termination Protection	Termination protection for CSR 1000v instances helps to prevent accidental CSR 1000v termination. (This is recommended for production deployments.) Default: Yes
Prefix for S3 Objects	Text string to be used as a prefix when Amazon S3 objects are created. Default: **`vpnconfigs/`**
Additional AWS Account ID	Account ID of an AWS account to be associated with the transit network, which allows access to the S3 bucket and 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 permissions for the additional accounts.
Transit VPC CIDR Block	CIDR block for the transit VPC. Modify the VPC and subnet CIDR address ranges to avoid collisions with your network. Default: 100.64.127.224/27
1st Subnet Network	CIDR block for the transit VPC subnet created in AZ1 (See Figure 2. DMVPN with Transit VPC in DMVPN Transit VPC, on page 3). Default: 100.64.127.224/28

Parameter	Description
2nd Subnet Network	CIDR block for the transit VPC subnet created in AZ2. (See Figure 2. DMVPN with Transit VPC in DMVPN Transit VPC, on page 3). Default: 100.64.127.240/28
Transit VPC BGP ASN	BGP Autonomous System Number (ASN) for the transit VPC. Default: 64512
SendAnonymousData	Sends anonymous data to Amazon Web Services to help with understanding solution usage and achieve costs savings for customers. If you choose not to send this anonymous data, select "No". Default: Yes

Step 5 Click **Next**
The Options page appears.

Step 6 Specify tags (key-value pairs) for stack resources and additional options, then click **Next**.
The Review page appears.

Step 7 Review and confirm the settings. Note: Check the checkbox that acknowledges the template will create an AWS Identity and Access Management (IAM) resources.

Step 8 Click **Create** to deploy the stack.

Step 9 To view the status of the stack, look at the Status column in the AWS Cloud Formation console. If the deployment is successful, a status of "CREATE_COMPLETE" appears after a period of approximately five minutes.

Step 10 (Optional) To add another account, perform these sub-steps.

a. Select the stack that you created in steps 1 to 5.

b. Click "Actions" and select **update** from the drop-down menu .

c. Select "Use current template".

d. In the “Additional Account (Update Stack Allow)” field, select "Add additional account".

Note Do not change the other fields that are displayed in the template.

What to do next

To install a Cisco CSR 1000v as a network spoke, see Launching a Spoke VPC, on page 9. To connect a branch office or data center to the transit VPC hub, see Launching DMVPN for Transit VPC, on page 11.

Launching a Spoke VPC

Before you begin

Before launching a spoke VPC, you must launch a transit VPC hub—see Launching a Transit VPC Hub, on page 6.

Procedure

Step 1 Go to the following github location: https://github.com/csr1000v/transit_vpc_all_csr.

Step 2 In the "Readme" section, click **Launch Stack** under **Launching a Spoke VPC**.

Step 3 You have two options: (Option A) Enter the template parameters (see steps 4 and 5) or (Option B) Download, edit and upload the template file (see steps 6 and 7).

Step 4 (Option A) In the "Choose a Template" section, check the checkbox **Specify an Amazon S3 template URL** and click **Next**.

Step 5 (Option A) Enter the template parameters in the following list and go to step 8.

Table 2: Parameters for Launching a Spoke VPC

Parameter	Description
Stack name	Name of this spoke VPC.
CSR Throughput Requirements	Required throughput for the Cisco CSR 1000v instance. This determines the instance type to be launched. Default: 2 x 500 Mbps
SSH Key to access CSR	Public/private key pair which allows a secure connection to be made to a CSR 1000v instance after it has launched. You must enter a public/private key pair. (The key pair was created in your preferred region at the time when the AWS account was created.)
License Model	AWS license. Values: LicenseIncluded, BYOL. Default: LicenseIncluded
Enable Termination Protection	If enabled, termination protection for a Cisco CSR 1000v instance helps to prevent accidental Cisco CSR 1000v termination. (This is recommended for production deployments). Default: Yes

Parameter	Description
Enable High Availability	If High Availability is enabled, two Cisco CSR 1000v instances are created rather than one. These two Cisco CSR 1000v's run in high availability mode. (Additional costs apply.) Values: • NO—creates a single spoke Cisco CSR 1000v VPC. • YES—creates a two spoke Cisco CSR 1000v VPC, for high availability. Default: YES
Creates CSRs in a single availability Zone	Determines whether to create EC2 instances in one availability zone. Default: "No"
Prefix for S3 Objects	Text string to be used as a prefix when Amazon S3 objects are created. Default: **`vpnconfigs/`**
Transit VPC S3 Bucket	Name of the S3 bucket of the existing transit VPC hub, to which the spoke VPC will be connected.
Transit Prefer Path	Name of the preferred Cisco CSR 1000v instance to use for the active/passive paths through the transit network. Choose one of three options: NONE, CSR1, and CSR2. Default: NONE
Use existing VPC	Drop-down menu from which to choose an existing VPC as the the spoke VPC.
SendAnonymousData	Indicates whether to send anonymous data about the usage of this spoke VPC to Amazon Web Services. AWS uses the data to better understand how this transit VPC design is working and achieve costs savings for customers. If you do not want to send them this anonymous data, select "No". Default: Yes

Go to step 8.

Step 6 (Option B) Copy the template file from the URL shown in the text box in the "Choose a Template" section. Then download and edit the template file. Refer to the parameters listed in step 5.

Step 7 (Option B) Check “Upload a template to Amazon S3”, browse to your edited template file, and click **Next**.

Step 8 Click **Next**
The Options page appears.

Step 9 Specify tags (key-value pairs) for stack resources and additional options, then click **Next**.
The Review page appears.

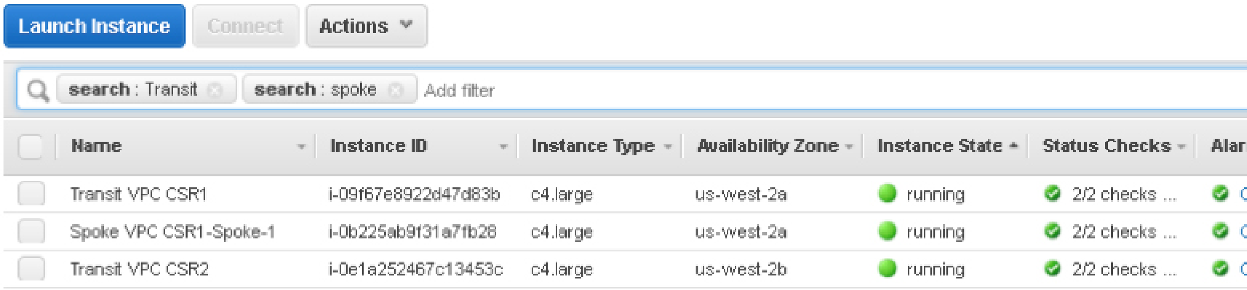
Step 10 Review and confirm the settings. Note: You must check the checkbox that acknowledges the template will create an AWS Identity and Access Management (IAM) resources.

Step 11 Click **Create** to deploy the stack.

Step 12 To view the status of the stack, look at the Status column in the AWS Cloud Formation console. A status of CREATE_COMPLETE should appear after a period of approximately five minutes.

Example:

The following example shows the AWS Cloud Formation console after a spoke is launched. The stack consists of a spoke and a transit VPC.



□	Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks	Alar
□	Transit VPC CSR1	i-09f67e8922d47d83b	c4.large	us-west-2a	● running	✓ 2/2 checks ...	✓ C
□	Spoke VPC CSR1-Spoke-1	i-0b225ab9f31a7fb28	c4.large	us-west-2a	● running	✓ 2/2 checks ...	✓ C
□	Transit VPC CSR2	i-0e1a252467c13453c	c4.large	us-west-2b	● running	✓ 2/2 checks ...	✓ C

What to do next

If required, to connect the transit hub to a private branch office or data center DMVPN network, see Launching DMVPN for Transit VPC, on page 11.

Launching DMVPN for Transit VPC

An AWS CloudFormation template is used to bootstrap the AWS infrastructure and automate the deployment of a DMVPN on the transit VPC. The transit VPC hub acts as a spoke to the DMVPN network.

To launch DMVPN, perform the following steps:

Before you begin

Make notes about the information about the private network's DMVPN configuration, to use in the following procedure.

Procedure

Step 1 Go to the following github location: https://github.com/csr1000v/transit_vpc_all_csr.

Step 2 In the "Readme" section, click **Launch Stack** under **Launching DMVPN for Transit VPC**.

Step 3 You have two options: (Option A) Enter the template parameters (see steps 4 and 5) or (Option B) Download, edit and upload the template file (see steps 6 and 7).

Step 4 (Option A) In the "Choose a Template" section, check the checkbox **Specify an Amazon S3 template URL** and click **Next**.

Step 5 (Option A) Enter the template parameters in the following list and go to step 8.

Table 3: Parameters for Launching DMVPN

Parameter	Description
DMVPN Profile	DMVPN Profile. Values: Spoke-to-OneHub–EIGRP, Spoke-to-MultiHub–EIGRP. Default: Spoke-to-OneHub–EIGRP
Create or Delete	Delete—deletes the DMVPN profile named in the DMVPN Profile field. Create—creates the DMVPN profile. Default: Create
Transit VPC S3 Bucket	Name of the Amazon S3 bucket for the existing Transit VPC.
Prefix for S3 Objects	Prefix name for Amazon S3 objects that are created during the process of deploying the transit VPC design. Default: `vpnconfigs/`
DMVPN tunnel CIDR	(Optional) DMVPN tunnel CIDR. Example: `10.101.0.0/16`
1st DMVPN Hub tunnel IP address	DMVPN hub tunnel IP address. Example: `10.101.0.1`
2nd DMVPN Hub tunnel IP address	(Optional) Second DMVPN hub tunnel IP address.
1st DMVPN Hub's IP address	Routable IP address of the first DMVPN Hub. (Use only for transit VPC as Spoke)
2nd DMVPN Hub's IP address	(Optional) Routable IP address of the second DMVPN Hub. (Use only for transit VPC as Spoke)
1st DMVPN spoke tunnel IP address	IP address of the first DMVPN spoke tunnel. Example: `10.101.0.3`
2nd DMVPN spoke tunnel IP address	IP address of the second DMVPN spoke tunnel. Example: `10.101.0.4`
Network ID for NHRP Protocol	Network ID for NHRP protocol—used under DMVPN tunnel interface. Example: `9898`

Parameter	Description
AuthString	Authentication string—used on an interface running NHRP. Example: `cisco123`
Tunnel Key	DMVPN Tunnel key—used in the hub. Example: 10
AS number for BGP/EIGRP	AS number of routing protocol—used in the hub Example: 10000
Choice of IPsec cipher algorithm	IPsec cipher algorithm. Select the alogithirm for possible values. Values: ESP-GCM, ESP-3DES, ESP-GMAC, ESP-DES, and ESP-AES. Default: ESP-AES
Choice of IPsec authentication algorithm	IPsec authentication algorithm. Values: ESP-SHA-HMAC, ESP-SHA256-HMAC, ESP-SHA384-HMAC, and ESP-SHA512-HMAC. Default: ESP-SHA256-HMAC
Shared Key	ISAKMP shared key, used in the IPsec algorithm.

Go to step 8.

Step 6 (Option B) Copy the template file from the URL shown in the text box in the "Choose a Template" section. Then download and edit the template file. Refer to the parameters listed in step 5.

Step 7 (Option B) Check “Upload a template to Amazon S3”, browse to your edited template file, and click **Next**.

Step 8 Click **Next**
The Options page appears.

Step 9 Specify tags (key-value pairs) for stack resources and additional options, then click **Next**.
The Review page appears.

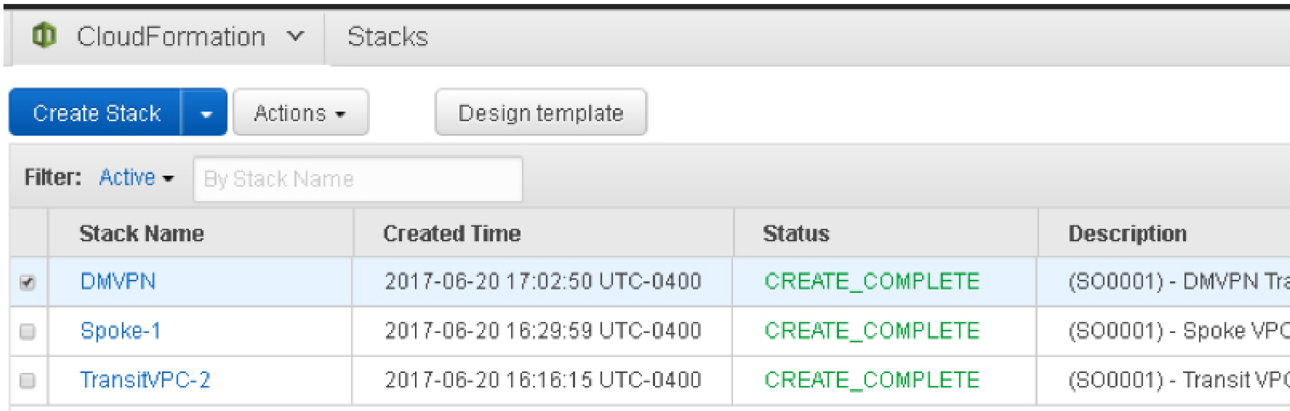
Step 10 Review and confirm the settings. Note: You must check the checkbox that acknowledges the template will create resources for AWS Identity and Access Management (IAM).

Step 11 Click **Create** to deploy the stack.

Step 12 To view the status of the stack, look at the Status of each stack in the AWS Cloud Formation console. A status of "CREATE_COMPLETE" should appear for a stack after a period of approximately five minutes.

Example:

The following example shows the AWS Cloud Formation console after launching DMVPN. The DMVPN, spoke and transit VPC stacks all show a status of "CREATE_COMPLETE".



	Stack Name	Created Time	Status	Description
☑	DMVPN	2017-06-20 17:02:50 UTC-0400	CREATE_COMPLETE	(SO0001) - DMVPN Tra
□	Spoke-1	2017-06-20 16:29:59 UTC-0400	CREATE_COMPLETE	(SO0001) - Spoke VPC
□	TransitVPC-2	2017-06-20 16:16:15 UTC-0400	CREATE_COMPLETE	(SO0001) - Transit VPC

Example Configurations for Transit VPC

Example 1

This example shows the output from the **show running-configuration** command after launching a transit VPC hub using the procedure Launching a Transit VPC Hub, on page 6 The transit VPC configuration includes VRFs to isolate the traffic from each of the spokes. There is a tunnel for each spoke.

```
# show running-config
Building configuration...

Current configuration : 5600 bytes
!
! Last configuration change at 13:48:53 UTC Mon Jun 5 2017 by automate
!
version 16.5
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
!
!
logging persistent size 1000000 filesize 8192 immediate
!
no aaa new-model
!
ip vrf vpn-vpc-a1c1ffc6-1
 rd 64512:1
 route-target export 64512:0
 route-target import 64512:0
!
ip vrf vpn0
 rd 64512:0
```

```
!
!
!
subscriber templating
!
!
multilink bundle-name authenticated
!
!
crypto pki trustpoint TP-self-signed-3523259110
 enrollment selfsigned
 subject-name cn=IOS-Self-Signed-Certificate-3523259110
 revocation-check none
 rsakeypair TP-self-signed-3523259110
!
!
crypto pki certificate chain TP-self-signed-3523259110
 certificate self-signed 01
  30820330 30820218 A0030201 02020101 300D0609 2A864886 F70D0101 05050030
  31312F30 2D060355 04031326 494F532D 53656C66 2D536967 6E65642D 43657274
  69666963 6174652D 33353233 32353931 3130301E 170D3137 30363035 31333235
  35315A17 0D323030 31303130 30303030 305A3031 312F302D 06035504 03132649
  4F532D53 656C662D 5369676E 65642D43 65727469 66696361 74652D33 35323332
  35393131 30308201 22300D06 092A8648 86F70D01 01010500 0382010F 00308201
  0A028201 0100DA69 00E57565 01537E34 288860E9 D619B338 EC9E98DA 3A0F876F
  C29C392F 1A6455E7 02F20233 340B5A8D D782321B 8F9B5EEC 7A282F83 98D1419F
  07710FE0 3EF6C2BF 7D73E229 F9118E4D 38C7CA13 368B1C4C 85EEA34D C1ADF679
  C5BC3713 30EF5F99 7FD57A14 41C74366 76939A99 5AB59D90 38A6494C B190BFAF
  ABE43A8C 98FD93F4 7EC238E3 D942E764 EE3069DA DB891977 12AA5280 47B85DEB
  B45A8E18 7105319C FC2A4DC4 E5700061 4581E540 2D3F661D E986DFBA 29734AF1
  C5BC1E52 2619DFED 229337B6 1F6F4E9C A44C2AA0 54FD5590 A1073803 EE9B1668
  BA6F96AC 7B75B1DE 27586013 227F0866 8239C0AD B3ED4553 FB63ACB6 99F28394
  87C194A5 3D3B0203 010001A3 53305130 0F060355 1D130101 FF040530 030101FF
  301F0603 551D2304 18301680 14933BEB 56E0BA3B 724C03DD 4992028A 423F9A66
  AC301D06 03551D0E 04160414 933BEB56 E0BA3B72 4C03DD49 92028A42 3F9A66AC
  300D0609 2A864886 F70D0101 05050003 82010100 D15CBF95 AE9301B7 3431B141
  0EFA2309 118D16E5 1AB45B67 E8DA140A 1A3624C9 7B42787A 87C5C8B1 145B751A
  CCCF5817 BC8526A7 2F1FFA9F 18B10DE8 4D7ECA46 E6A45DE8 30E85846 F00B3B3C
  DA7D2AD2 5E71CF00 05F2FD7F 41F6102A 504EDCA6 B70270BE 4D215CDA CEB5DF69
  7AC15D3F 9203EFAF DE9278B5 77042A4A F582B3D4 32749FB4 6CF62042 202D4520
  EFF7C2E1 3BB2C49E 2F38BA99 C50D6CCA E53CA1D7 8BFE5A78 5AE6129B 9F1DC35D
  D0487945 744AE38E 190739DD D321BABA D8175F32 8FAA1742 AD9DBB26 B1B86C73
  A0DD9B40 51807D18 A4E9CB18 DE7C1185 616D3B61 B2AB993B 9C0943BF 14EE373E
  7CBC65FC 6DCAEC9B 2725A155 6A85FAB3 473F065B
        quit
!
!
!
license udi pid CSR1000V sn 9AUMWWU9PUN
diagnostic bootup level minimal
!
spanning-tree extend system-id
!
!
username ec2-user privilege 15 secret 5 $1$kSop$DkIGMkrEfF25JjFKXguLf/
username automate privilege 15 password 7 130B4F422F0A081273333C3504043030
!
redundancy
!
!
crypto keyring keyring-vpn-vpc-a1c1ffc6-1-1
  local-address GigabitEthernet1
  pre-shared-key address 35.167.172.22 key xpXf7qkthnbUUbjdTOOCnd857wNXpwLC
!
```

```
!
crypto isakmp policy 200
 encr aes
 authentication pre-share
 group 2
 lifetime 28800
crypto isakmp keepalive 10 10
crypto isakmp profile isakmp-vpn-vpc-a1c1ffc6-1-1
   keyring keyring-vpn-vpc-a1c1ffc6-1-1
   match identity address 35.167.172.22 255.255.255.255
   local-address GigabitEthernet1
!
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
!
!
crypto ipsec profile ipsec-vpn-aws
 set transform-set ipsec-prop-vpn-aws
 set pfs group2
!
!
!
interface Tunnel1
 ip vrf forwarding vpn-vpc-a1c1ffc6-1
 ip address 169.254.98.158 255.255.255.252
 ip tcp adjust-mss 1387
 tunnel source GigabitEthernet1
 tunnel mode ipsec ipv4
 tunnel destination 35.167.172.22
 tunnel protection ipsec profile ipsec-vpn-aws
 ip virtual-reassembly
!
interface GigabitEthernet1
 ip address dhcp
 negotiation auto
 no mop enabled
 no mop sysid
!
router bgp 64512
 bgp log-neighbor-changes
 !
 address-family ipv4 vrf vpn-vpc-a1c1ffc6-1
  neighbor 169.254.98.157 remote-as 7224
  neighbor 169.254.98.157 timers 10 30 30
  neighbor 169.254.98.157 activate
  neighbor 169.254.98.157 as-override
  neighbor 169.254.98.157 soft-reconfiguration inbound
  neighbor 169.254.98.157 route-map rm-vpn-vpc-a1c1ffc6-1 out
 exit-address-family
!

threat-visibility
!
virtual-service csr_mgmt
 ip shared host-interface GigabitEthernet1
 activate
!
ip forward-protocol nd
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 ssh rsa keypair-name ssh-key
ip ssh version 2
ip ssh pubkey-chain
  username ec2-user
   key-hash ssh-rsa AB75A036D99B839B83054BD19A2FA911 ec2-user
  username automate
   key-hash ssh-rsa E1A59785E730AD9F790A8BEDCEDA5C49
ip ssh server algorithm encryption aes128-ctr aes192-ctr aes256-ctr
ip ssh server algorithm authentication publickey
ip ssh client algorithm encryption aes128-ctr aes192-ctr aes256-ctr
ip scp server enable
!
!
!
route-map rm-vpn-vpc-a1c1ffc6-1 permit 10
 set as-path prepend 64512
!
!
!
control-plane
!
!
!
line con 0
 stopbits 1
line vty 0 4
 login local
 transport input ssh
!
!
!
end
```

Example 2

This example shows the output from the **show running-configuration** command after adding a single spoke VPC using procedure Launching a Spoke VPC, on page 9. This configuration has two tunnels—a tunnel for each transit VPC.

```
ip-30-20-0-29# show running-config
Building configuration...

Current configuration : 6139 bytes
!
! Last configuration change at 13:49:19 UTC Mon Jun 5 2017 by automate
!
version 16.5
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-30-20-0-29
!
boot-start-marker
boot-end-marker
!
!
logging persistent size 1000000 filesize 8192 immediate
!
```

```
no aaa new-model
!
ip vrf vpn0
 rd 7224:0
!
!
subscriber templating
!
!
multilink bundle-name authenticated
!
!
!
crypto pki trustpoint TP-self-signed-1386236880
 enrollment selfsigned
 subject-name cn=IOS-Self-Signed-Certificate-1386236880
 revocation-check none
 rsakeypair TP-self-signed-1386236880
!
!
crypto pki certificate chain TP-self-signed-1386236880
 certificate self-signed 01
  30820330 30820218 A0030201 02020101 300D0609 2A864886 F70D0101 05050030
  31312F30 2D060355 04031326 494F532D 53656C66 2D536967 6E65642D 43657274
  69666963 6174652D 31333836 32333638 3830301E 170D3137 30363035 31333434
  31365A17 0D323030 31303130 30303030 305A3031 312F302D 06035504 03132649
  4F532D53 656C662D 5369676E 65642D43 65727469 66696361 74652D31 33383632
  33363838 30308201 22300D06 092A8648 86F70D01 01010500 0382010F 00308201
  0A028201 0100AEDE 2A0D11AC 749A3653 9167DD5C E2F3D734 F520CAD7 C25F043E
  4FFC8BC5 C2774656 B43E0184 03762DFE D28B99BF B7B4CE26 7C2045FF 562DBEFA
  5DE59C75 E0D478EB 0DEE310D 75C37EAD 6D011075 C5625DBE 90A62281 38D3CBEB
  39FF443C 4F66EB5A 2BB09E98 CFE092CF 92F0FFB6 29E65C72 AEB601D7 658EA1FA
  65D4DFC7 EA2088A7 3E7EED02 3149D8DC 29D9E774 AA73EB8B B404C899 2B9793F8
  1530913B 1B43D4B7 0A6E610B 2DF7ECC5 6B6EEC15 9AF6FE6E F83B3C3B 321F5C3B
  0843CBEE 1EFDC8EF 5B88AABF E78C5E48 8752EB26 0ACF2858 831A4558 AE4405FC
  A25F2989 38B56D00 EF6E2AAB D4C3F280 CACC4E74 9F724141 A7BC897C 94C78E7F
  57F66DE6 10830203 010001A3 53305130 0F060355 1D130101 FF040530 030101FF
  301F0603 551D2304 18301680 1430FBD0 ACC3496D 56D8DA0A 04C68E23 2DFDD68D
  16301D06 03551D0E 04160414 30FBD0AC C3496D56 D8DA0A04 C68E232D FDD68D16
  300D0609 2A864886 F70D0101 05050003 82010100 361BF93C 4F82F950 2E574DDE
  A88E5B1C 4108CF38 E44F15E8 8D6CFD27 4D91C825 DB33BB79 57BE21E9 4861BB7E
  34222C5A 55395427 C2F4CF58 8261C369 E9FD0FF6 B1CFDE77 C3903B7C 83261E46
  A2AA923A EEC1B39B 75EB447A 92D463B4 C2FF52B5 8C39DCA1 4625CBCF 6F0AF06F
  F47FD863 BB46AF60 3B0DD5D5 0057F036 81F42E1D 312DE686 8EC7886C 60C08D1E
  48FF2F30 1BC7A2A5 7B9B8903 E3813BD5 C8778BE2 A342E453 EC542749 687EF070
  A42251B3 5F08839C 057B9F42 52C0D431 6EFCAAFB 0F055EC5 DADF8741 F26ED316
  215986FC 6AFC1203 9099C888 A62E40A9 4677C397 E22F995B CBA18F14 855F0DA8
  AB6E4A03 071B1F5F 115474C5 4BA27036 B3D6D482
        quit
!
!
!
!
license udi pid CSR1000V sn 96XBJC62UE8
diagnostic bootup level minimal
!
spanning-tree extend system-id
!
!
username ec2-user privilege 15 secret 5 $1$YGPR$3jKBkXWAWfanoYJxzSRig0
username automate privilege 15 password 7 14404B221D5D3B73303E6B0001140B3E
!
redundancy
!
```

```
!
!
crypto keyring keyring-vpn-vpc-a1c1ffc6-1-2
  local-address GigabitEthernet1
  pre-shared-key address 52.35.65.47 key xpXf7qkthnbUUbjdTOOCnd857wNXpwLC
crypto keyring keyring-vpn-vpc-a1c1ffc6-1-1
  local-address GigabitEthernet1
  pre-shared-key address 35.166.108.244 key 2PwgEmI4y75GNiNFGuPL4OjPTEdnOmfw
!
crypto isakmp policy 200
 encr aes
 authentication pre-share
 group 2
 lifetime 28800
crypto isakmp keepalive 10 10
crypto isakmp profile isakmp-vpn-vpc-a1c1ffc6-1-1
   keyring keyring-vpn-vpc-a1c1ffc6-1-1
   match identity address 35.166.108.244 255.255.255.255
   local-address GigabitEthernet1
crypto isakmp profile isakmp-vpn-vpc-a1c1ffc6-1-2
   keyring keyring-vpn-vpc-a1c1ffc6-1-2
   match identity address 52.35.65.47 255.255.255.255
   local-address GigabitEthernet1
!
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
!
crypto ipsec profile ipsec-vpn-aws
 set transform-set ipsec-prop-vpn-aws
 set pfs group2
!
interface Tunnel1
 ip address 169.254.130.197 255.255.255.252
 ip tcp adjust-mss 1387
 tunnel source GigabitEthernet1
 tunnel mode ipsec ipv4
 tunnel destination 35.166.108.244
 tunnel protection ipsec profile ipsec-vpn-aws
 ip virtual-reassembly
!
interface Tunnel2
 ip address 169.254.98.157 255.255.255.252
 ip tcp adjust-mss 1387
 tunnel source GigabitEthernet1
 tunnel mode ipsec ipv4
 tunnel destination 52.35.65.47
 tunnel protection ipsec profile ipsec-vpn-aws
 ip virtual-reassembly
!
interface GigabitEthernet1
 ip address dhcp
 negotiation auto
 no mop enabled
 no mop sysid
!
router bgp 7224
 bgp log-neighbor-changes
 neighbor 169.254.98.158 remote-as 64512
 neighbor 169.254.98.158 timers 10 30 30
 neighbor 169.254.130.198 remote-as 64512
 neighbor 169.254.130.198 timers 10 30 30
 !
 address-family ipv4
  redistribute connected
  neighbor 169.254.98.158 activate
  neighbor 169.254.98.158 as-override
  neighbor 169.254.98.158 soft-reconfiguration inbound
  neighbor 169.254.130.198 activate
  neighbor 169.254.130.198 as-override
  neighbor 169.254.130.198 soft-reconfiguration inbound
 exit-address-family
!
threat-visibility
!
virtual-service csr_mgmt
 ip shared host-interface GigabitEthernet1
 activate
!
ip forward-protocol nd
ip http server
ip http authentication local
ip http secure-server
ip route 0.0.0.0 0.0.0.0 GigabitEthernet1 30.20.0.17
!
ip ssh maxstartups 2
ip ssh rsa keypair-name ssh-key
ip ssh version 2
ip ssh pubkey-chain
  username ec2-user
   key-hash ssh-rsa AB75A036D99B839B83054BD19A2FA911 ec2-user
  username automate
   key-hash ssh-rsa 2072B88CA0E5AC33E2E29638FDFD26E9
ip ssh server algorithm encryption aes128-ctr aes192-ctr aes256-ctr
ip ssh server algorithm authentication publickey
ip ssh client algorithm encryption aes128-ctr aes192-ctr aes256-ctr
ip scp server enable
!
control-plane
!
line con 0
 stopbits 1
line vty 0 4
 login local
 transport input ssh
!
end
```

CHAPTER 3

Deploying Transit VPC with Autoscaling for Amazon Web Services (AWS)

The auto-scaling functionality (Autoscaler) manages transit VPC CSR instances by monitoring and performing scale-out and scale-in operations.

Information About Deploying Transit VPC with Autoscaler

Overview of Autoscaler

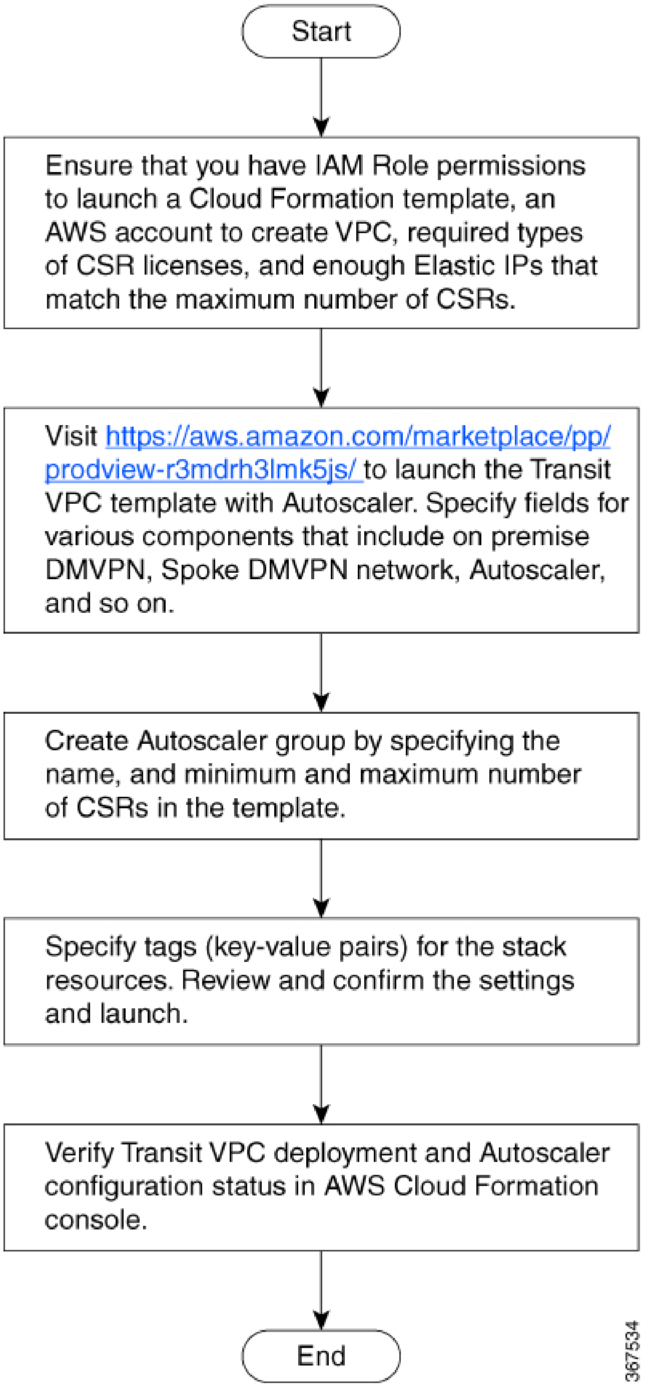
Autoscaler automatically performs scale in and scale out operations by adding and removing CSR instances depending on the volume of traffic in the transit VPC. Autoscaler optimizes CSR performance in transit VPC by effectively utilizing CSR instances in a cost-effective mode.

Scaling out refers to attaching additional CSR instances to increase capacity. Scaling in refers to detaching the CSR instances to reduce the extended capacity. When Autoscaler detects a load increase for a sustained period of time, it performs a scale-out by adding a new CSR instance to the Transit VPC capacity. Similarly, when Autoscaler detects a decrease in traffic for a sustained period of time, it performs scale-in action by terminating one of the instances in the Transit VPC. To handle varying loads, it configures and manages the appropriate number of CSR instances.

While performing scale-out, Autoscaler configures the CSR instance for all the existing on-premise VPN networks and the spoke VPC networks. Autoscaler cannot perform scaling out beyond the maximum number of instances and scaling in beyond the minimum number of instances defined.

Autoscaler performs scale-out and scale-in operations based on the metrics that are published on AWS CloudWatch. When Autoscaler detects that the metrics meet the pre-defined conditions, it takes appropriate action. To know about the conditions for performing scale out and scale in operations, see Scaling-Out and Scaling-In in Transit VPC, on page 22.

Figure 3: Transit VPC with Autoscaler Deployment Workflow


Start
Ensure that you have IAM Role permissions to launch a Cloud Formation template, an AWS account to create VPC, required types of CSR licenses, and enough Elastic IPs that match the maximum number of CSRs.
Visit https://aws.amazon.com/marketplace/pp/prodview-r3mdrh3lmk5js/ to launch the Transit VPC template with Autoscaler. Specify fields for various components that include on premise DMVPN, Spoke DMVPN network, Autoscaler, and so on.
Create Autoscaler group by specifying the name, and minimum and maximum number of CSRs in the template.
Specify tags (key-value pairs) for the stack resources. Review and confirm the settings and launch.
Verify Transit VPC deployment and Autoscaler configuration status in AWS Cloud Formation console.
End
367534

Scaling-Out and Scaling-In in Transit VPC

Autoscaler scales out by attaching CSR instances. Autoscaler scales out when these conditions are met:

- if the load on each CSR is higher than the base trigger value for a predefined period of time. For example, the load on each CSR in the group is higher than 50 percent of the license level for a period of 10 minutes.
- at least one CSR in the group is above the threshold for a predefined period of time. For example, at least one CSR in the group must exceed 80 percent of the license level for a period of 15 minutes.
- if the maximum number of configured instances has not been met
- if the configured debounce time for scale out has passed from the last scale out or scale in operation.
- If the maximum number of Out of Compliance instances has not been met

Autoscaler scales in when these conditions are met:

- If the load on all CSR instances in the group are below the trigger value. For example, the load on all CSR instances in the group are below 40 percent of the license level.
- at least any one CSR is lower than the threshold for a predefined period of time. For example, at least any one CSR is lower than 10 percent of the license level.
- if the configured debounce time for scale in has passed from the last scale-out or scale-in operation
- if the minimum number of configured instances has not been met

Monitoring in Transit VPC

Autoscaler monitors events in the Transit VPC CSRs through CloudWatch metrics. It continuously monitors and detects any changes in the load and determines the actions to be taken based on the parameters defined. Based on the statistics that are published on CloudWatch, it determines the appropriate time to scale-in or scale-out.

For example, if throughput value is higher than the threshold defined for a period of over 15 minutes, it scales out.

Benefits of Autoscaler

- Manages the changing requirements of a Transit VPC by adding or removing CSR instances to meet varying load demands. Whenever the load varies, there is no need for a manual intervention of adding or removing additional CSR instances.
- Performs automatic license activation for CSR instances for Bring Your Own License (BYOL) type.
- Effectively utilizes CSR instances in Transit VPC to save cost.

Prerequisites for Autoscaler

- An AWS account with the privileges to create CloudFormation stack
- Access to S3 Bucket and AMI in AWS
- Elastic IPs for each CSRs in the group
- CSR license types: BYOL or LicenseIncluded
- Ensure that you meet the following service limits for the specified resources in the transit the account for the transit VPC deployment:

- 1 VPC (for transit VPC deployment)
- 1 NAT gateway
- 12 EPIs
- 1 S3 bucket
- 4 lambda functions
- 4 Cloudwatch logs
- 8 EC2 instances
- 1 SQS queue

How to Deploy Transit VPC with Autoscaler

Launching Transit VPC Components

Transit VPC with Autoscaler deployment involves configuring various components such as Transit VPC Hub, Spoke VPC, DMVPN, CSR, AWS Service, Autoscaler, and IPSec Algo. You specify parameters for these components in the AWS CloudFormation window.

Procedure

Step 1 Go to https://aws.amazon.com/marketplace/pp/prodview-r3mdrh3lmk5js?qid=1574662124240&sr=0-1&ref_=srh_res_product_title, select **Create Stack** and choose the template, and then click **Next**.

Step 2 Specify tags (key-value pairs) for stack resources and additional options, then click **Next**.

Table 4: Parameters for Launching a Transit VPC

Parameter	Description
Stack name	Name of the Transit VPC.
CSR Throughput Requirements	Required throughput for the CSR 1000v instance. It determines the instance type to be launched. Default: 2 x 500 Mbps
SSH Key to access CSR	Public/private key pair that allows a secure connection to be made to a CSR 1000v instance after it has launched. You must enter a public/private key pair. (The key pair is created at your preferred location when the AWS account is created.)
License Model	The license model is BYOL.

Parameter	Description
Enable Termination Protection	Termination protection for CSR 1000v instances helps prevent accidental CSR 1000v termination. It is recommended for production deployments. Default: Yes
Prefix for S3 Objects	Text string to be used as a prefix when Amazon S3 objects are created. Default: **`vpnconfigs/`**
Transit VPC CIDR Block	CIDR block for the transit VPC. Modify the VPC and subnet CIDR address ranges to avoid collisions with your network. Default: 100.64.127.224/27
1st Subnet Network	CIDR block for the transit VPC subnet created in AZ1 (See Figure 2. DMVPN with Transit VPC in DMVPN Transit VPC, on page 3). Default: 100.64.127.224/28
2nd Subnet Network	CIDR block for the transit VPC subnet created in AZ2. (See Figure 2. DMVPN with Transit VPC in DMVPN Transit VPC, on page 3). Default: 100.64.127.240/28
Transit VPC BGP ASN	BGP Autonomous System Number (ASN) for the transit VPC. Default: 64512
Add/remove additional AWS account	(Optional) Enable this to add or remove additional AWS account.
Additional AWS Account ID	Account ID of an AWS account to be associated with the transit network, which allows access to the S3 bucket and 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 permissions for the additional accounts.
SendAnonymousData	Sends anonymous data to Amazon Web Services to help with understanding solution usage and achieve costs savings for customers. If you choose not to send this anonymous data, select NO. Default: Yes

Parameter	Description
Create Auto Scaling Group	Select Enable to deploy Transit VPC with Autoscaling functionality.
Group Name	Name of the Auto Scaling group for the Transit VPC CSR 1000v instances. This name is also used to tag the CSR instances.
Minimum number of Instances	The minimum number of CSR instances that the CSR group maintains all the time.
Maximum Number of Instances	The maximum number of CSR instances that the CSR group allows. Autoscaler cannot scale out beyond the maximum number of instances.
SNS Notification	(Optional) Enable this option to receive notifications related to Transit VPC events. An email address or SMS number can be specified to a subscribe notification.
License Model for Autoscaler	The license model supported is BYOL.
License Token ID	License token ID for smart licensing.
Technology Package	Select the technology package for licensing (AX or Security)
License Level	License level in megabytes for the Transit VPC CSR 1000v instances.
Email Address for License	Enter the email address for license.
Enable Scale In for Autoscaled Instances	Enable scale in for the transit VPC CSR 1000v instances. If Disabled, the Autoscaler will only scale out when needed.
Enable DMVPN	Select enable to configure DMVPN. Default: Enable
DMVPN Profile	DMVPN Profile. Values: Spoke-to-OneHub–EIGRP, Spoke-to-MultiHub–EIGRP. Default: Spoke-to-OneHub–EIGRP
Create or Delete	Create—creates the DMVPN profile. Delete—deletes the DMVPN profile named in the DMVPN Profile field. Default: Create
DMVPN tunnel CIDR	(Optional) DMVPN tunnel CIDR. Example: `10.101.0.0/16`

Parameter	Description
1st DMVPN Hub tunnel IP address	DMVPN hub tunnel IP address. Example: `10.101.0.1`
2nd DMVPN Hub tunnel IP address	(Optional) Second DMVPN hub tunnel IP address.
1st DMVPN Hub's IP address	Routable IP address of the first DMVPN Hub. (Use only for transit VPC as Spoke)
2nd DMVPN Hub's IP address	(Optional) Routable IP address of the second DMVPN Hub. (Use only for transit VPC as Spoke)
1st DMVPN spoke tunnel IP address	IP address of the first DMVPN spoke tunnel. Example: `10.101.0.3`
2nd DMVPN spoke tunnel IP address	IP address of the second DMVPN spoke tunnel. Example: `10.101.0.4`
Network ID for NHRP Protocol	Network ID for NHRP protocol—used under DMVPN tunnel interface. Example: `9898`
AuthString	Authentication string—used on an interface running NHRP. Example: `cisco123`
Tunnel Key	DMVPN Tunnel key—used in the hub. Example: 10
AS number for BGP/EIGRP	AS number of routing protocol—used in the hub Example: 10000
Choice of IPsec cipher algorithm	IPsec cipher algorithm. Select the alogithirm for possible values. Values: ESP-GCM, ESP-3DES, ESP-GMAC, ESP-DES, and ESP-AES. Default: ESP-AES
Choice of IPsec authentication algorithm	IPsec authentication algorithm. Values: ESP-SHA-HMAC, ESP-SHA256-HMAC, ESP-SHA384-HMAC, and ESP-SHA512-HMAC. Default: ESP-SHA256-HMAC
Shared Key	ISAKMP shared key, used in the IPsec algorithm.

Step 3 Click **Next**.

Step 4 Specify tags (key-value pairs) for the stack resources and click **Next**.

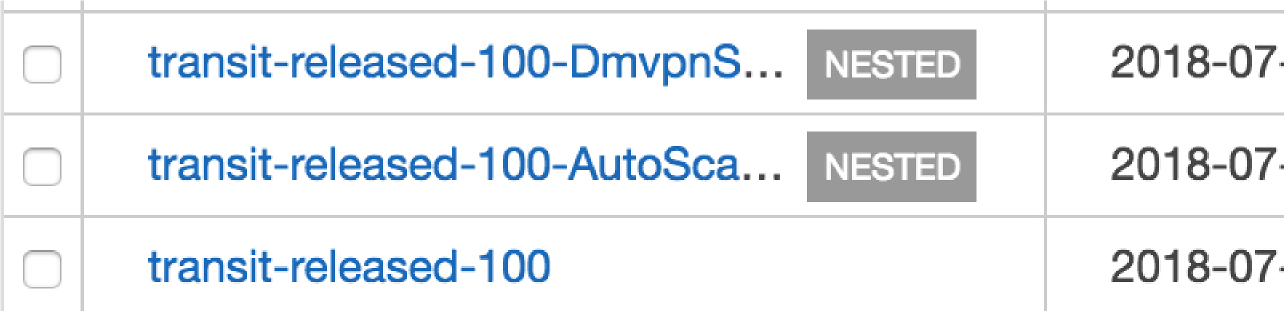
Step 5 Review and confirm the settings.

Step 6 Select the check box to create resources for AWS Identity and Access Management (IAM).

Step 7 Click **Create**.

Verifying Autoscaler Deployment

The AWS CloudFormation window allows you to view the deployment status of the Transit VPC with Autoscaler. A status CREATE_COMPLETE appears approximately in five minutes after the successful deployment. It shows the status of main template and the two nested templates separately as shown in the following image.



□	transit-released-100-DmvpnS... NESTED	2018-07-
□	transit-released-100-AutoSca... NESTED	2018-07-
□	transit-released-100	2018-07-

Note If Transit VPC launch fails, the stack event logs provide information that help you to identify the causes of the failure. You can also view CloudWatch logs and the router logs to detect connectivity issues that may have caused failure.

Configuration Example

This example shows the output from the **show running-configuration** command after configuring Transit VPC with Autoscaler.

```
Building configuration...

Current configuration : 11349 bytes
!
! Last configuration change at 01:14:34 UTC Wed Jul 4 2018 by automate
!
version 16.9
service timestamps debug datetime msec
service timestamps log datetime msec
service password-encryption
service internal
service call-home
platform qfp utilization monitor load 80
no platform punt-keepalive disable-kernel-core
platform console virtual
platform hardware throughput level MB 1000
!
hostname autoscaling-group-4
!
```

```
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
call-home
 ! If contact email address in call-home is configured as sch-smart-licensing@cisco.com
 ! the email address configured in Cisco Smart License Portal will be used as contact email
 address to send SCH notifications.
 contact-email-addr sch-smart-licensing@cisco.com
 profile "CiscoTAC-1"
  active
  destination transport-method http
  no destination transport-method email
  destination address http
http://pteodc-alphastg.cloudapps.cisco.com/its/service/oddce/services/DDCEService
!
ip vrf vpn-dmvpn-transit-vpc--Tun-3
 rd 64512:3
 route-target export 64512:0
 route-target import 64512:0
!
ip vrf vpn-vpc-9fcf81f9-81792
 rd 64512:1
 route-target export 64512:0
 route-target import 64512:0
!
ip vrf vpn0
!
!
ip admission watch-list expiry-time 0
!
!
subscriber templating
!
!
multilink bundle-name authenticated
!
crypto pki trustpoint TP-self-signed-2282641679
 enrollment selfsigned
 subject-name cn=IOS-Self-Signed-Certificate-2282641679
 revocation-check none
 rsakeypair TP-self-signed-2282641679
!
crypto pki trustpoint SLA-TrustPoint
 enrollment terminal
 revocation-check crl
!
!
crypto pki certificate chain TP-self-signed-2282641679
 certificate self-signed 01
  30820330 30820218 A0030201 02020101 300D0609 2A864886 F70D0101 05050030
  31312F30 2D060355 04031326 494F532D 53656C66 2D536967 6E65642D 43657274
  69666963 6174652D 32323832 36343136 3739301E 170D3138 30373034 30303539
  33305A17 0D333030 31303130 30303030 305A3031 312F302D 06035504 03132649
  4F532D53 656C662D 5369676E 65642D43 65727469 66696361 74652D32 32383236
```

```
    34313637 39308201 22300D06 092A8648 86F70D01 01010500 0382010F 00308201
    0A028201 0100AE02 C7451287 B91A57FF E8ECE6F9 A1AC83D6 016410D4 127E1725
    0F923728 014E91B6 0B1760C8 BE6FAAFF 46FBE5E0 36747B19 7C297232 981DBB31
    50CDB0C7 4CF3C8D9 49527FBC C950C718 198AFCA9 BA60165E 1327B8E5 B4FF0BBF
    56F56A65 31C0B1CD 069DB94B BB079EAA 70A6722B BC8B4272 08E42918 E6EA3487
    7CF7374D 190B23CC 5FE9C7B4 091E5C0F F30C5BA8 CBC4F54A B5AA44D2 33E7C14D
    F9AD72C4 4419DAB4 FF4E3528 C3E98324 7E7B3CE3 DDB8A859 BA9143A8 3AB595B7
    66281EB6 47743345 3CE5054B CE6E4E48 9A6D9FE5 DB06D14C DF24370B F503F252
    FD719163 F33353F1 9EBA399D 6E71E33E F266677C E4C0E791 B457E990 23076DC8
    BE91D4C6 A07D0203 010001A3 53305130 0F060355 1D130101 FF040530 030101FF
    301F0603 551D2304 18301680 146CA59F 42679C84 1276FA2E 01F72B49 13419063
    A4301D06 03551D0E 04160414 6CA59F42 679C8412 76FA2E01 F72B4913 419063A4
    300D0609 2A864886 F70D0101 05050003 82010100 757EA9A3 11398961 A6D2BEB2
    F9FA907F C6F9D658 ABE42351 DEF1C8D5 C1B9CE08 6AB01D0E CFC40E94 D433752C
    7853CF07 AAA0488B 70F7A4AB 69A87AE8 DABA2A6F 12511EF3 D0E63B24 2C336C7B
    C6CFFAF4 1F322EE1 F76A2D0D 8E8637BE 297C5673 ED8D7116 6E6B22BD 7C479518
    633F5684 FB362D4D 047053F1 F14249C4 F2BAD569 68B6E2E1 21878422 E2F542F1
    8BADB8A3 1376A191 34198F38 7A71BAF9 1E1E0CA2 69D6A685 0F0E09E5 1542F991
    FDE39C64 8687C830 20BC2F35 C5A429F9 4EE47919 2476F0B9 68684C4F 34F5CABA
    56168375 8DFD9DD5 A45C9FC4 6DBF48A4 1698599D C2D78AF1 107B9932 32A387F1
    A0450FB8 86441528 7F60D860 BE22D5B7 FEE03972
      quit
crypto pki certificate chain SLA-TrustPoint
 certificate ca 01
    30820321 30820209 A0030201 02020101 300D0609 2A864886 F70D0101 0B050030
    32310E30 0C060355 040A1305 43697363 6F312030 1E060355 04031317 43697363
    6F204C69 63656E73 696E6720 526F6F74 20434130 1E170D31 33303533 30313934
    3834375A 170D3338 30353330 31393438 34375A30 32310E30 0C060355 040A1305
    43697363 6F312030 1E060355 04031317 43697363 6F204C69 63656E73 696E6720
    526F6F74 20434130 82012230 0D06092A 864886F7 0D010101 05000382 010F0030
    82010A02 82010100 A6BCBD96 131E05F7 145EA72C 2CD686E6 17222EA1 F1EFF64D
    CBB4C798 212AA147 C655D8D7 9471380D 8711441E 1AAF071A 9CAE6388 8A38E520
    1C394D78 462EF239 C659F715 B98C0A59 5BBB5CBD 0CFEBEA3 700A8BF7 D8F256EE
    4AA4E80D DB6FD1C9 60B1FD18 FFC69C96 6FA68957 A2617DE7 104FDC5F EA2956AC
    7390A3EB 2B5436AD C847A2C5 DAB553EB 69A9A535 58E9F3E3 C0BD23CF 58BD7188
    68E69491 20F320E7 948E71D7 AE3BCC84 F10684C7 4BC8E00F 539BA42B 42C68BB7
    C7479096 B4CB2D62 EA2F505D C7B062A4 6811D95B E8250FC4 5D5D5FB8 8F27D191
    C55F0D76 61F9A4CD 3D992327 A8BB03BD 4E6D7069 7CBADF8B DF5F4368 95135E44
    DFC7C6CF 04DD7FD1 02030100 01A34230 40300E06 03551D0F 0101FF04 04030201
    06300F06 03551D13 0101FF04 05300301 01FF301D 0603551D 0E041604 1449DC85
    4B3D31E5 1B3E6A17 606AF333 3D3B4C73 E8300D06 092A8648 86F70D01 010B0500
    03820101 00507F24 D3932A66 86025D9F E838AE5C 6D4DF6B0 49631C78 240DA905
    604EDCDE FF4FED2B 77FC460E CD636FDB DD44681E 3A5673AB 9093D3B1 6C9E3D8B
    D98987BF E40CBD9E 1AECA0C2 2189BB5C 8FA85686 CD98B646 5575B146 8DFC66A8
    467A3DF4 4D565700 6ADF0F0D CF835015 3C04FF7C 21E878AC 11BA9CD2 55A9232C
    7CA7B7E6 C1AF74F6 152E99B7 B1FCF9BB E973DE7F 5BDDEB86 C71E3B49 1765308B
    5FB0DA06 B92AFE7F 494E8A9E 07B85737 F3A58BE1 1A48A229 C37C1E69 39F08678
    80DDCD16 D6BACECA EEBC7CF9 8428787B 35202CDC 60E4616A B623CDBD 230E3AFB
    418616A9 4093E049 4D10AB75 27E86F73 932E35B5 8862FDAE 0275156F 719BB2F0
    D697DF7F 28
      quit
!
license udi pid CSR1000V sn 9PEEBCTDYEL
license boot level security
license smart enable
diagnostic bootup level minimal
!
!
!
username ec2-user privilege 15 secret 5 $1$6tuC$/t0yPt7jlk13RTzdPaO2K.
username automate privilege 15 password 7 060F0B706F1730302A153E5C3D092F3F
!
redundancy
!
```

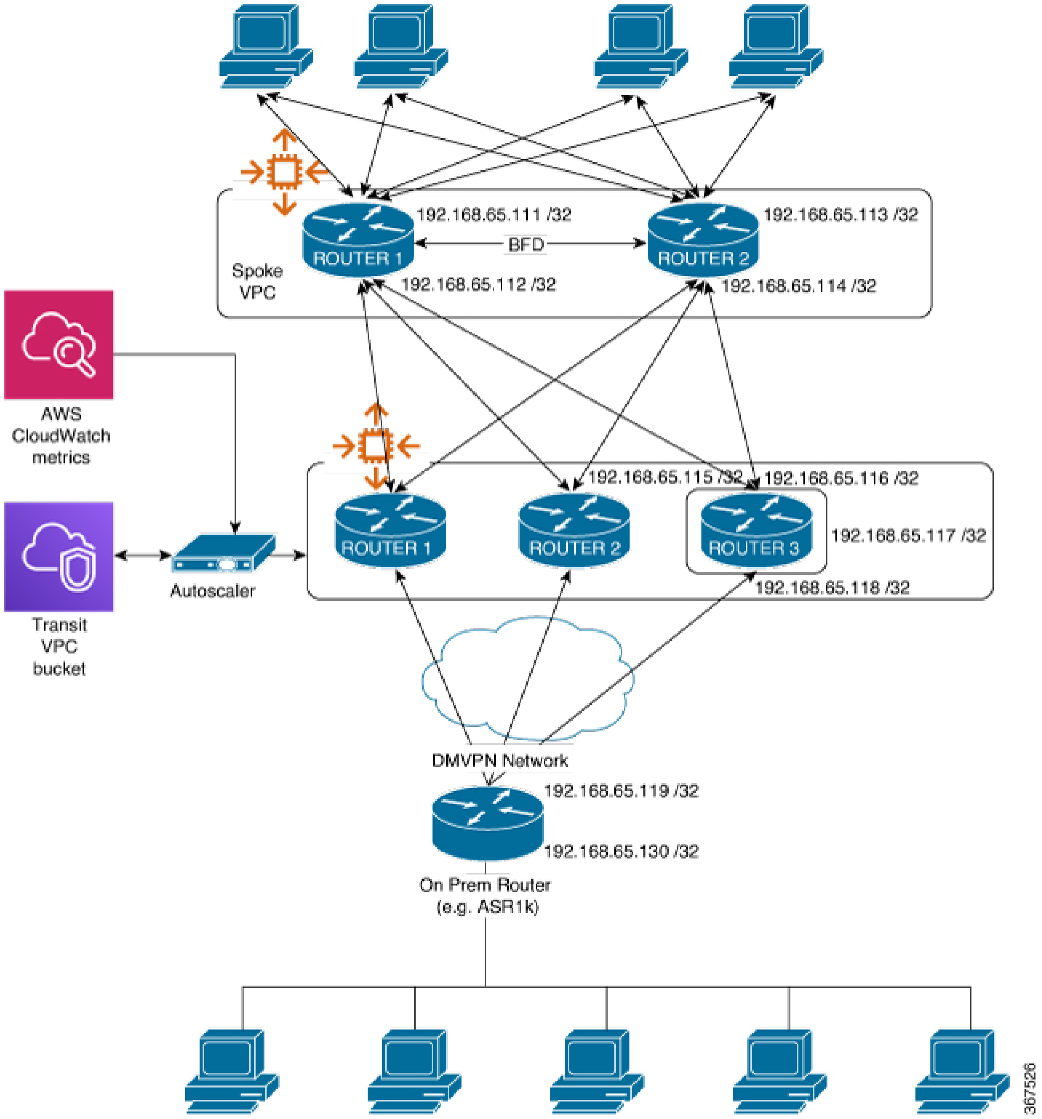
```
!
class-map match-all autoscaler-class-map
 match access-group 50
!
policy-map autoscaler-policy-map
 class autoscaler-class-map
  set cos 5
!
!
crypto keyring keyring-transit-vpc--Tun-3
  local-address GigabitEthernet1
  pre-shared-key address 0.0.0.0 0.0.0.0 key cisco
crypto keyring keyring-vpn-vpc-9fcf81f9-81792-2
  local-address GigabitEthernet1
  pre-shared-key address 52.214.77.11 key 0WdAFsN33XkfsBZOukL3zqxihq6OCrgZ
crypto keyring keyring-vpn-vpc-9fcf81f9-81792-1
  local-address GigabitEthernet1
  pre-shared-key address 34.242.251.176 key Y1EPGZDdW8eMKvp9R5966rgX7RM3lOgr
!
!
crypto isakmp policy 200
 encr aes
 authentication pre-share
 group 2
 lifetime 28800
!
crypto isakmp policy 300
 encr aes
 authentication pre-share
 group 5
crypto isakmp keepalive 30 5
crypto isakmp profile isakmp-vpn-vpc-9fcf81f9-81792-1
   keyring keyring-vpn-vpc-9fcf81f9-81792-1
   match identity address 34.242.251.176 255.255.255.255
   local-address GigabitEthernet1
crypto isakmp profile isakmp-vpn-vpc-9fcf81f9-81792-2
   keyring keyring-vpn-vpc-9fcf81f9-81792-2
   match identity address 52.214.77.11 255.255.255.255
   local-address GigabitEthernet1
!
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 transform-set transform-transit-vpc--Tun-3 esp-aes esp-sha-hmac
 mode transport
crypto ipsec df-bit clear
!
!
crypto ipsec profile ipsec-vpn-aws
 set transform-set ipsec-prop-vpn-aws
 set pfs group2
!
crypto ipsec profile profile-transit-vpc--Tun-3
 set transform-set transform-transit-vpc--Tun-3
!
!
!
!
interface Tunnel1
 ip vrf forwarding vpn-vpc-9fcf81f9-81792
 ip address 169.254.99.142 255.255.255.252
 ip tcp adjust-mss 1387
 tunnel source GigabitEthernet1
 tunnel mode ipsec ipv4
 tunnel destination 34.242.251.176
 tunnel protection ipsec profile ipsec-vpn-aws
 ip virtual-reassembly
!
interface Tunnel2
 ip vrf forwarding vpn-vpc-9fcf81f9-81792
 ip address 169.254.181.190 255.255.255.252
 ip tcp adjust-mss 1387
 tunnel source GigabitEthernet1
 tunnel mode ipsec ipv4
 tunnel destination 52.214.77.11
 tunnel protection ipsec profile ipsec-vpn-aws
 ip virtual-reassembly
!
interface Tunnel3
 ip vrf forwarding vpn-dmvpn-transit-vpc--Tun-3
 ip address 172.31.1.12 255.255.0.0
 no ip redirects
 ip mtu 1400
 ip nhrp authentication cisco123
 ip nhrp network-id 9898
 ip nhrp nhs 172.31.1.5 nbma 34.254.75.62 multicast
 ip tcp adjust-mss 1360
 tunnel source GigabitEthernet1
 tunnel mode gre multipoint
 tunnel key 0
 tunnel protection ipsec profile profile-transit-vpc--Tun-3
!
interface VirtualPortGroup0
 vrf forwarding GS
 ip address 192.168.35.101 255.255.255.0
 ip nat inside
!
interface GigabitEthernet1
 ip vrf select source
 ip vrf receive vpn-vpc-9fcf81f9-81792
 ip address dhcp
 ip nat outside
 negotiation auto
 service-policy output autoscaler-policy-map
!
!
router eigrp 10000
 !
 address-family ipv4 vrf vpn-dmvpn-transit-vpc--Tun-3
  redistribute bgp 64512 metric 10000 100 255 1 1500
  network 172.31.0.0
  autonomous-system 10000
 exit-address-family
!
router bgp 64512
 bgp router-id interface GigabitEthernet1
 bgp log-neighbor-changes
 !
 address-family ipv4 vrf vpn-dmvpn-transit-vpc--Tun-3
  redistribute eigrp 10000
 exit-address-family
 !
 address-family ipv4 vrf vpn-vpc-9fcf81f9-81792
  neighbor 169.254.99.141 remote-as 7224
  neighbor 169.254.99.141 timers 10 30 30
  neighbor 169.254.99.141 activate
  neighbor 169.254.99.141 as-override
  neighbor 169.254.99.141 soft-reconfiguration inbound
  neighbor 169.254.181.189 remote-as 7224
  neighbor 169.254.181.189 timers 10 30 30
  neighbor 169.254.181.189 activate
  neighbor 169.254.181.189 as-override
  neighbor 169.254.181.189 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 13.13.1.1
ip route vrf GS 0.0.0.0 0.0.0.0 GigabitEthernet1 13.13.1.1 global
!
ip ssh rsa keypair-name ssh-key
ip ssh version 2
ip ssh pubkey-chain
  username ec2-user
   key-hash ssh-rsa 3214AF3ED1A55040F7CB7BDCB17633FA ec2-user
  username automate
   key-hash ssh-rsa 10705A66BA5D14D2C4E2C3A46F6F2CF5
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
access-list 50 permit 13.13.1.35
!
!
control-plane
!
!
line con 0
 stopbits 1
line vty 0 4
 login local
 transport input ssh
line vty 5 20
 login local
 transport input ssh
!
!
event manager applet PUSH_CSR_METRICS
 event timer cron cron-entry "*/1 * * * *"
 action 1.0 cli command "enable"
 action 2.0 cli command "guestshell run get-stat-drop.py"
!
!
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
 name-server1 8.8.4.4
 name-server2 8.8.8.8
 name-server3 8.8.4.4
end
```

Use Cases

Transit VPC Network

The following figure illustrates a Transit VPC network architecture and its interconnections. These interconnections comprise of multiple virtual networks in cloud such as VPCs and on premise networks such as corporate network or data center networks. This` topology illustrates the Transit VPC (a virtual network in customer's AWS account) acting as a transit hub for transiting traffic among the different networks from the multiple geographical locations.

Figure 4: Transit VPC in AWS

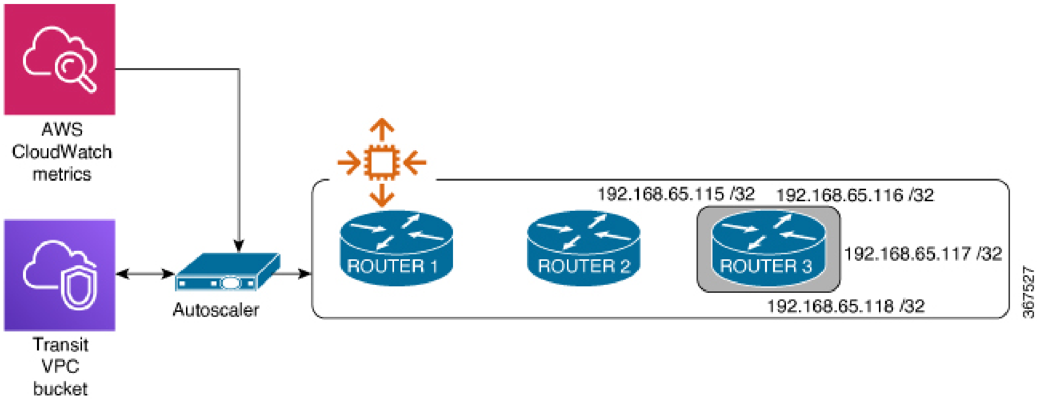


Autoscaling in Transit VPC

The following topology presents a Transit VPC network with autoscaling functionality enabled. A minimum of two CSRs are always in service with little or no traffic. The topology shows the 'ROUTER 3' is launched and configured by Autoscaler whenever there is high traffic that meets the pre-defined threshold.

The topology shows that Autoscaler monitors the CSR metrics published on AWS CloudWatch and takes appropriate action, if required. The topology also presents configuration files stored in S3 bucket are applied to newly launched CSR when Autoscaler triggers a scale out action.

Figure 5: Autoscaler in Transit VPC



Troubleshooting Autoscaler Issues

If you are facing issues with the transit VPC solution with Autoscaling, see the following troubleshooting mechanisms:

Accessing the Autoscale status.jsonFile

The autoscale_status.json file is your first stop for investigating why an Autoscaler action either occurred, or did not occur. In this file, you can find the following details:

- The current state of the Autoscaler solution.
- The cloud configuration for all the CSR 1000v instances in the group such as IP address, subnet information, security group, AMI ID and so on.
- The metrics that were last read for each watcher.

To view the autoscale_status.json file:

1. Navigate to the transit VPC vpn bucket in S3.
2. Select the appropriate Autoscaler directory.
3. Download the autoscale_status.json file.

   The autoscale_status.json file resides in the `AutoScaler/autoscale_status.json` directory under the transit VPC S3 bucket.

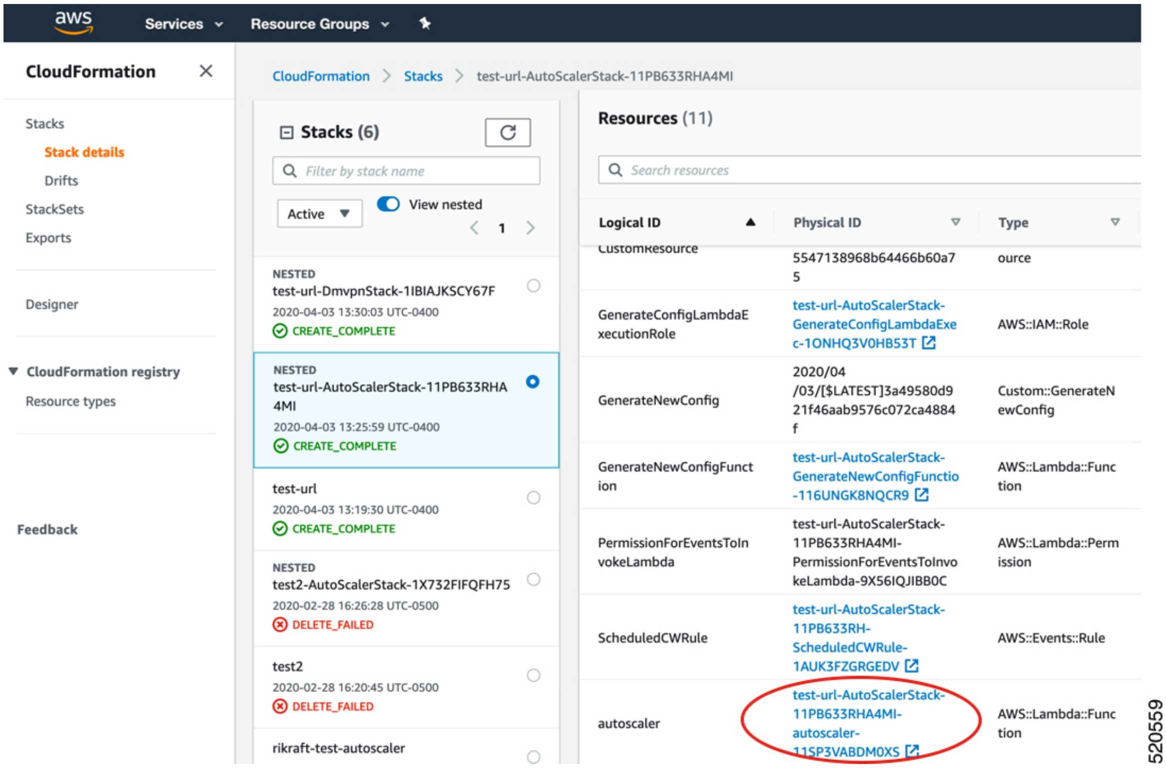
Debugging using CloudWatch Logs

The CloudWatch logs from Autocaler’s lambda functions will show what state the Autoscaler is in, for each lambda invocation. Each log features the metrics which are retrieved and compared against the respective lambda function.

To view the lambda log files, perform the following steps:

1. From the CloudFormation dashboard, click **Stacks** > **Stack Details**. Select the AutoScalerStack for your solution.

2. Select the **Resources** tab. In the bottom of the list, under the **Logical ID column** > **Autoscaler** option, click the link under the **Physical ID** option.



CloudFormation > Stacks > test-url-AutoScalerStack-11PB633RHA4MI

Stacks (6)

Filter by stack name

Active ▼ View nested 1

NESTED
test-url-DmvpnStack-1IBIAJKSCY67F
2020-04-03 13:30:03 UTC-0400
CREATE_COMPLETE

NESTED
test-url-AutoScalerStack-11PB633RHA4MI
2020-04-03 13:25:59 UTC-0400
CREATE_COMPLETE

test-url
2020-04-03 13:19:30 UTC-0400
CREATE_COMPLETE

NESTED
test2-AutoScalerStack-1X732FIFQFH75
2020-02-28 16:26:28 UTC-0500
DELETE_FAILED

test2
2020-02-28 16:20:45 UTC-0500
DELETE_FAILED

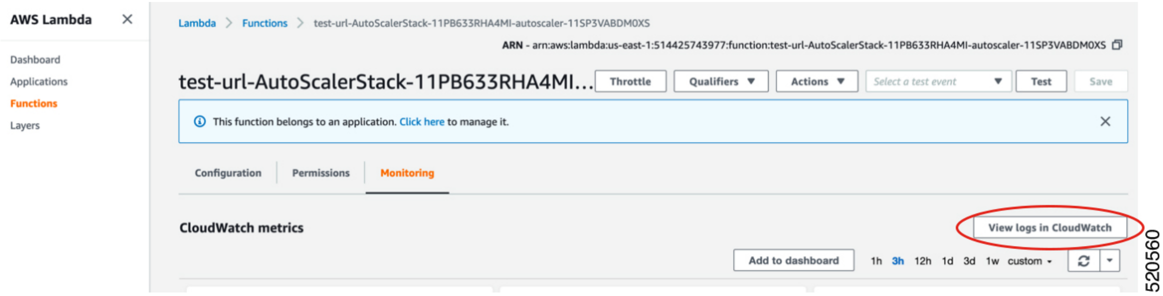
rikraft-test-autoscaler

Resources (11)

Search resources

Logical ID	Physical ID	Type
CustomResource	5547138968b64466b60a7 5	ource
GenerateConfigLambdaExecutionRole	test-url-AutoScalerStack-GenerateConfigLambdaExec-1ONHQ3V0HB53T	AWS::IAM::Role
GenerateNewConfig	2020/04 /03/[$LATEST]3a49580d921f46aab9576c072ca4884f	Custom::GenerateNewConfig
GenerateNewConfigFunction	test-url-AutoScalerStack-GenerateNewConfigFunctio-116UNGK8NQCR9	AWS::Lambda::Function
PermissionForEventsToInvokeLambda	test-url-AutoScalerStack-11PB633RHA4MI-PermissionForEventsToInvokeLambda-9X56IQJIBB0C	AWS::Lambda::Permission
ScheduledCWRule	test-url-AutoScalerStack-11PB633RH-ScheduledCWRule-1AUK3FZGRGEDV	AWS::Events::Rule
autoscaler	test-url-AutoScalerStack-11PB633RHA4MI-autoscaler-11SP3VABDM0XS	AWS::Lambda::Function

3. From the **Functions** dashboard, select the **Monitoring** tab. Click the **View logs in CloudWatch** button.



Lambda > Functions > test-url-AutoScalerStack-11PB633RHA4MI-autoscaler-11SP3VABDM0XS

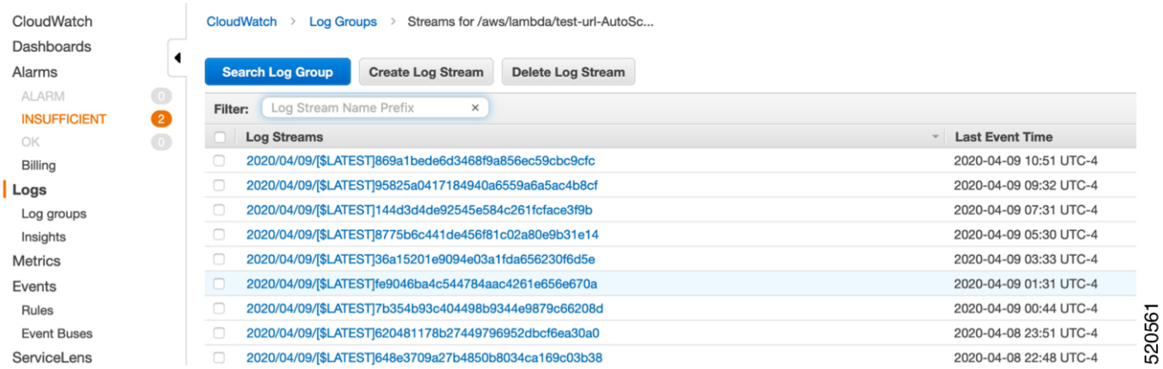
ARN - arn:aws:lambda:us-east-1:514425743977:function:test-url-AutoScalerStack-11PB633RHA4MI-autoscaler-11SP3VABDM0XS

test-url-AutoScalerStack-11PB633RHA4MI...

This function belongs to an application. Click here to manage it.

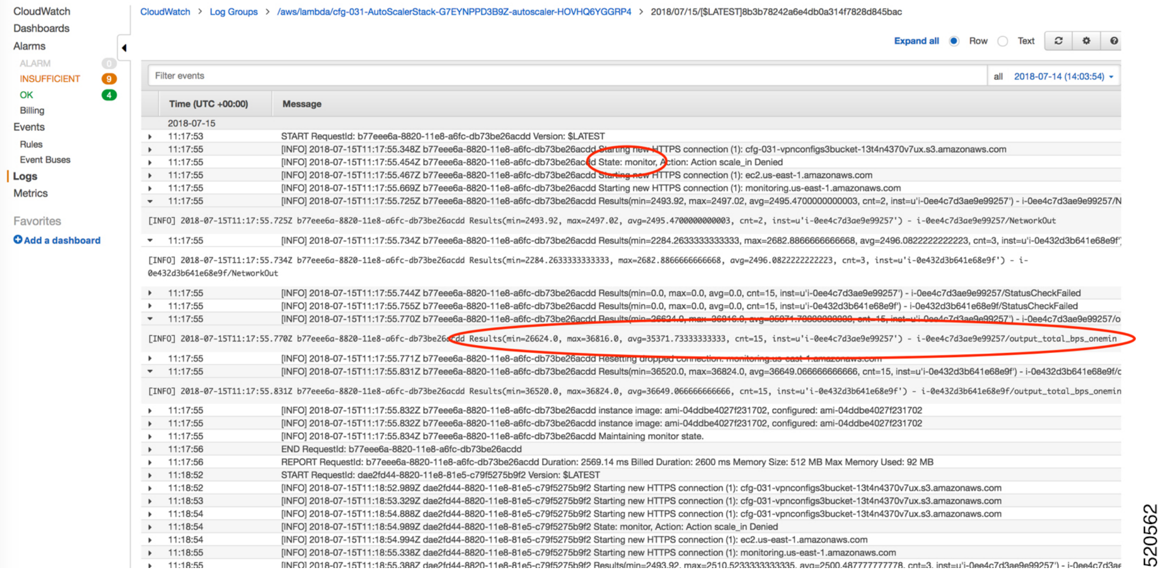
CloudWatch metrics

4. Here, you can select and view the log streams for your AutoScalerStack’s lambda functions.



Log Streams	Last Event Time
2020/04/09/[$LATEST]869a1bede6d3468f9a856ec59cbc9cfc	2020-04-09 10:51 UTC-4
2020/04/09/[$LATEST]95825a0417184940a6559a6a5ac4b8cf	2020-04-09 09:32 UTC-4
2020/04/09/[$LATEST]144d3d4de92545e584c261fcface3f9b	2020-04-09 07:31 UTC-4
2020/04/09/[$LATEST]8775b6c441de456f81c02a80e9b31e14	2020-04-09 05:30 UTC-4
2020/04/09/[$LATEST]36a15201e9094e03a1fda656230f6d5e	2020-04-09 03:33 UTC-4
2020/04/09/[$LATEST]fe9046ba4c544784aac4261e656e670a	2020-04-09 01:31 UTC-4
2020/04/09/[$LATEST]7b354b93c404498b9344e9879c66208d	2020-04-09 00:44 UTC-4
2020/04/09/[$LATEST]620481178b27449796952dbcf6ea30a0	2020-04-08 23:51 UTC-4
2020/04/09/[$LATEST]648e3709a27b4850b8034ca169c03b38	2020-04-08 22:48 UTC-4

5. Select a log stream link to view its contents.


CloudWatch
Dashboards
Alarms
ALARM 0
INSUFFICIENT 9
OK 4
Billing
Events
Rules
Event Buses
Logs
Metrics
Favorites
Add a dashboard
CloudWatch > Log Groups > /aws/lambda/cfg-031-AutoScalerStack-G7EYNPPD3B9Z-autoscaler-HOVHQ6YGGRP4 > 2018/07/15/[$LATEST]8b3b78242a6e4db0a314f7828d845bac
Expand all
● Row
○ Text
Filter events
all 2018-07-14 (14:03:54)
520562

```
Time (UTC +00:00)  Message
2018-07-15
11:17:53  START RequestId: b77eee6a-8820-11e8-a6fc-db73be26acdd Version: $LATEST
11:17:55  [INFO] 2018-07-15T11:17:55.348Z b77eee6a-8820-11e8-a6fc-db73be26acdd Starting new HTTPS connection (1): cfg-031-vpnconfigs3bucket-13t4n4370v7ux.s3.amazonaws.com
11:17:55  [INFO] 2018-07-15T11:17:55.454Z b77eee6a-8820-11e8-a6fc-db73be26acdd State: monitor, Action: Action scale_in Denied
11:17:55  [INFO] 2018-07-15T11:17:55.467Z b77eee6a-8820-11e8-a6fc-db73be26acdd Starting new HTTPS connection (1): ec2.us-east-1.amazonaws.com
11:17:55  [INFO] 2018-07-15T11:17:55.669Z b77eee6a-8820-11e8-a6fc-db73be26acdd Starting new HTTPS connection (1): monitoring.us-east-1.amazonaws.com
11:17:55  [INFO] 2018-07-15T11:17:55.725Z b77eee6a-8820-11e8-a6fc-db73be26acdd Results(min=2493.92, max=2497.02, avg=2495.4700000000003, cnt=2, inst=u'i-0ee4c7d3ae9e99257') - i-0ee4c7d3ae9e99257/N
[INFO] 2018-07-15T11:17:55.725Z b77eee6a-8820-11e8-a6fc-db73be26acdd Results(min=2493.92, max=2497.02, avg=2495.4700000000003, cnt=2, inst=u'i-0ee4c7d3ae9e99257') - i-0ee4c7d3ae9e99257/NetworkOut
11:17:55  [INFO] 2018-07-15T11:17:55.734Z b77eee6a-8820-11e8-a6fc-db73be26acdd Results(min=2284.2633333333333, max=2682.8866666666668, avg=2496.0822222222223, cnt=3, inst=u'i-0e432d3b641e68e9f')
[INFO] 2018-07-15T11:17:55.734Z b77eee6a-8820-11e8-a6fc-db73be26acdd Results(min=2284.2633333333333, max=2682.8866666666668, avg=2496.0822222222223, cnt=3, inst=u'i-0e432d3b641e68e9f') - i-
0e432d3b641e68e9f/NetworkOut
11:17:55  [INFO] 2018-07-15T11:17:55.744Z b77eee6a-8820-11e8-a6fc-db73be26acdd Results(min=0.0, max=0.0, avg=0.0, cnt=15, inst=u'i-0ee4c7d3ae9e99257') - i-0ee4c7d3ae9e99257/StatusCheckFailed
11:17:55  [INFO] 2018-07-15T11:17:55.755Z b77eee6a-8820-11e8-a6fc-db73be26acdd Results(min=0.0, max=0.0, avg=0.0, cnt=15, inst=u'i-0e432d3b641e68e9f') - i-0e432d3b641e68e9f/StatusCheckFailed
11:17:55  [INFO] 2018-07-15T11:17:55.770Z b77eee6a-8820-11e8-a6fc-db73be26acdd Results(min=26624.0, max=36816.0, avg=35371.73333333333, cnt=15, inst=u'i-0ee4c7d3ae9e99257') - i-0ee4c7d3ae9e99257/o
[INFO] 2018-07-15T11:17:55.770Z b77eee6a-8820-11e8-a6fc-db73be26acdd Results(min=26624.0, max=36816.0, avg=35371.73333333333, cnt=15, inst=u'i-0ee4c7d3ae9e99257') - i-0ee4c7d3ae9e99257/output_total_bps_onemin
11:17:55  [INFO] 2018-07-15T11:17:55.771Z b77eee6a-8820-11e8-a6fc-db73be26acdd Resetting dropped connection: monitoring.us-east-1.amazonaws.com
11:17:55  [INFO] 2018-07-15T11:17:55.831Z b77eee6a-8820-11e8-a6fc-db73be26acdd Results(min=36520.0, max=36824.0, avg=36649.066666666666, cnt=15, inst=u'i-0e432d3b641e68e9f') - i-0e432d3b641e68e9f/c
[INFO] 2018-07-15T11:17:55.831Z b77eee6a-8820-11e8-a6fc-db73be26acdd Results(min=36520.0, max=36824.0, avg=36649.066666666666, cnt=15, inst=u'i-0e432d3b641e68e9f') - i-0e432d3b641e68e9f/output_total_bps_onemin
11:17:55  [INFO] 2018-07-15T11:17:55.832Z b77eee6a-8820-11e8-a6fc-db73be26acdd instance image: ami-04ddbe4027f231702, configured: ami-04ddbe4027f231702
11:17:55  [INFO] 2018-07-15T11:17:55.832Z b77eee6a-8820-11e8-a6fc-db73be26acdd instance image: ami-04ddbe4027f231702, configured: ami-04ddbe4027f231702
11:17:55  [INFO] 2018-07-15T11:17:55.834Z b77eee6a-8820-11e8-a6fc-db73be26acdd Maintaining monitor state.
11:17:56  END RequestId: b77eee6a-8820-11e8-a6fc-db73be26acdd
11:17:56  REPORT RequestId: b77eee6a-8820-11e8-a6fc-db73be26acdd Duration: 2569.14 ms Billed Duration: 2600 ms Memory Size: 512 MB Max Memory Used: 92 MB
11:18:52  START RequestId: dae2fd44-8820-11e8-81e5-c79f5275b9f2 Version: $LATEST
11:18:52  [INFO] 2018-07-15T11:18:52.989Z dae2fd44-8820-11e8-81e5-c79f5275b9f2 Starting new HTTPS connection (1): cfg-031-vpnconfigs3bucket-13t4n4370v7ux.s3.amazonaws.com
11:18:53  [INFO] 2018-07-15T11:18:53.329Z dae2fd44-8820-11e8-81e5-c79f5275b9f2 Starting new HTTPS connection (1): cfg-031-vpnconfigs3bucket-13t4n4370v7ux.s3.amazonaws.com
11:18:54  [INFO] 2018-07-15T11:18:54.888Z dae2fd44-8820-11e8-81e5-c79f5275b9f2 Starting new HTTPS connection (1): cfg-031-vpnconfigs3bucket-13t4n4370v7ux.s3.amazonaws.com
11:18:54  [INFO] 2018-07-15T11:18:54.989Z dae2fd44-8820-11e8-81e5-c79f5275b9f2 State: monitor, Action: Action scale_in Denied
11:18:54  [INFO] 2018-07-15T11:18:54.994Z dae2fd44-8820-11e8-81e5-c79f5275b9f2 Starting new HTTPS connection (1): ec2.us-east-1.amazonaws.com
11:18:55  [INFO] 2018-07-15T11:18:55.338Z dae2fd44-8820-11e8-81e5-c79f5275b9f2 Starting new HTTPS connection (1): monitoring.us-east-1.amazonaws.com
11:18:55  [INFO] 2018-07-15T11:18:55.388Z dae2fd44-8820-11e8-81e5-c79f5275b9f2 Results(min=2493.92, max=2510.5233333333335, avg=2500.487777777778, cnt=3, inst=u'i-0ee4c7d3ae9e99257') - i-0ee4c7d3ae
```

If the CSR Metrics is not being pushed to CloudWatch, check whether the Guestshell EEM Applet (def: get-stat-drop.py) is running. To do so, recreate the guestshell environment by:

Destroying the guestshell. In the CLI, enter **guestshell destroy** command.

Recreate the guestshell by executing the **guestshell enable** command.

Reinstall the guestshell package by executing the **sudo pip install csr_aws_guestshell** command.

Note If your CSR 1000v instance is reachable from the Autoscaler, and the guestshell is not operating properly due to the unavailability metrics, the Autoscaler automatically performs the above steps.

Disaster Recovery

In the event of a disaster, or when your instances go down, one or all of the following happens:

The CSR 1000v instances in a transit VPC solution should be backed up with IPSec tunnels attached to the instances in the Spoke VPC. If DMVPN is enabled, the CSR 1000v instances in the transit VPC solution joins the DMVPN hub again. The network traffic is re-established from the Spoke VPCs to the on-prem network via the CSR 1000v instances in the transit VPC solution.

The Autoscaler controller lambda function enters the Monitoring state. This component initiates the required scaling action based on the network traffic and the autoscaler configuration file.

In most cases, the network recovery and autoscaler recovers to a stable state in an hour or so. However, this action and the time taken differs based on how you've configured your solution.

Regional Failure Recovery

To protect against regional failures Cisco recommends that you have back up regions with the same deployment. In the event of a failure, the traffic from the on-premise network could be routed through the secondary region using a routing protocol such as BGP to preserve operational continuity.

Related Documentation

To see licensing related information, see the following:

- Smart Licensing Guide and Access and Edge Routers.
- Installing CSR 1000v Licenses
- For the latest information on evaluation licenses and throughput, see the latest CSR 1000v Release Notes.

  To know more about installing software and updates, see Upgrading the Cisco IOS XE Software.

CHAPTER 4

Verifying Transit VPC on Amazon Web Services

This section contains the following topics:

Verifying Transit VPC

If the launch of transit VPC hub/spoke or DMVPN fails, stack events provide information that may be useful in determining why the stack failed to launch successfully. If connectivity issues exist, you can collect data that is held in the cloud watch logs and router logs.

If you have Cisco licensing issues or need other technical support, send your questions in an email to ask-asr1k-pm@cisco.com or tac@cisco.com .

If you have issues with a non-Cisco license, contact Amazon Web Services technical support; for example, see https://aws.amazon.com/contact-us/.

CHAPTER 5

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 CSR 1000v 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.

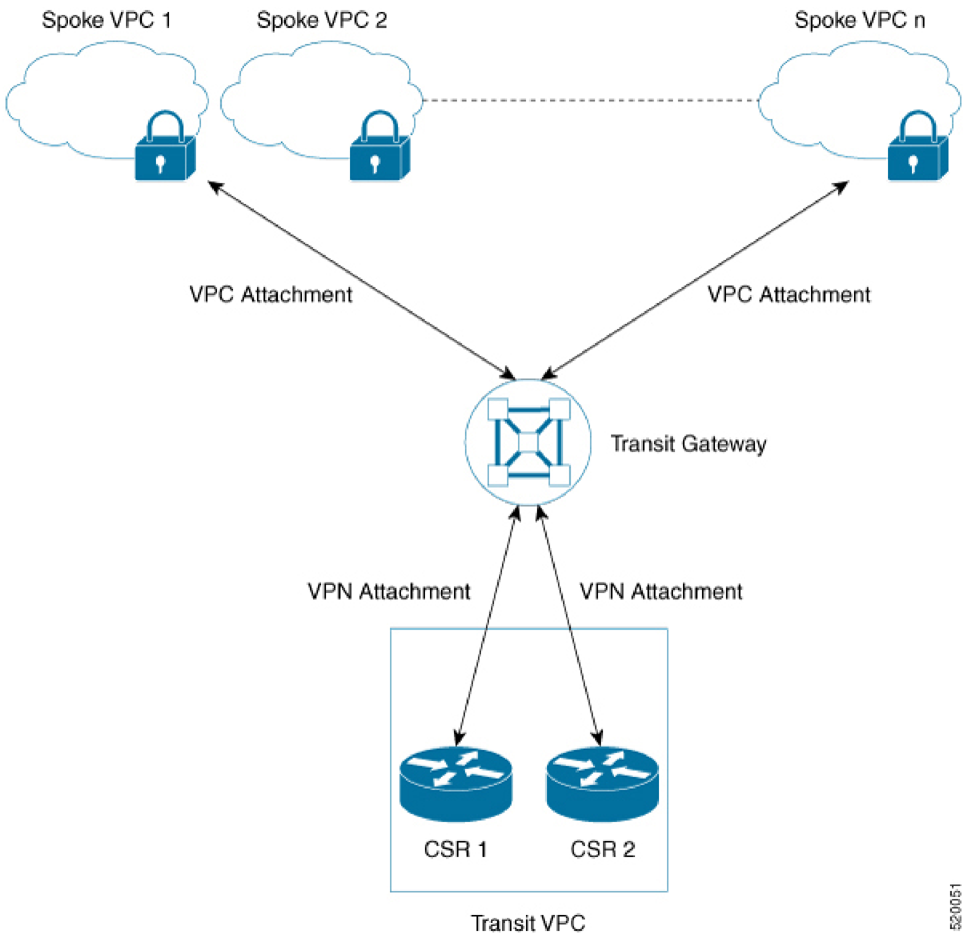
To overcome this limitation, you can now deploy a CSR 1000v Transit VPC with the Transit Gateway solution. A transit gateway is a regional network transit hub service provided by AWS to interconnect your VPCs in AWS cloud and on-premise network. In the Cisco CSR1000v transit VPC with transit gateway solution, you use a transit gateway on the spoke side to provide connectivity between all spoke VPCs in the same region. The transit gateway is attached to two CSR 1000v instances in the transit VPC using a VPN attachment. The CSR1000v 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 CSR1000v instances that allow for VPN termination and routing.

Figure 6: 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 CSR 1000v 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 CSR 1000v instances using SSH. As soon as the Cisco configuration is applied onto the CSR 1000v 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.

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.

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.

Configuring the AWS Transit Gateway Solution

Procedure

Step 1 Log in to the Amazon Web Services Marketplace.

Step 2 Search the **Cisco CSR1000V– Transit VPC with Transit Gateway** template and select the template.

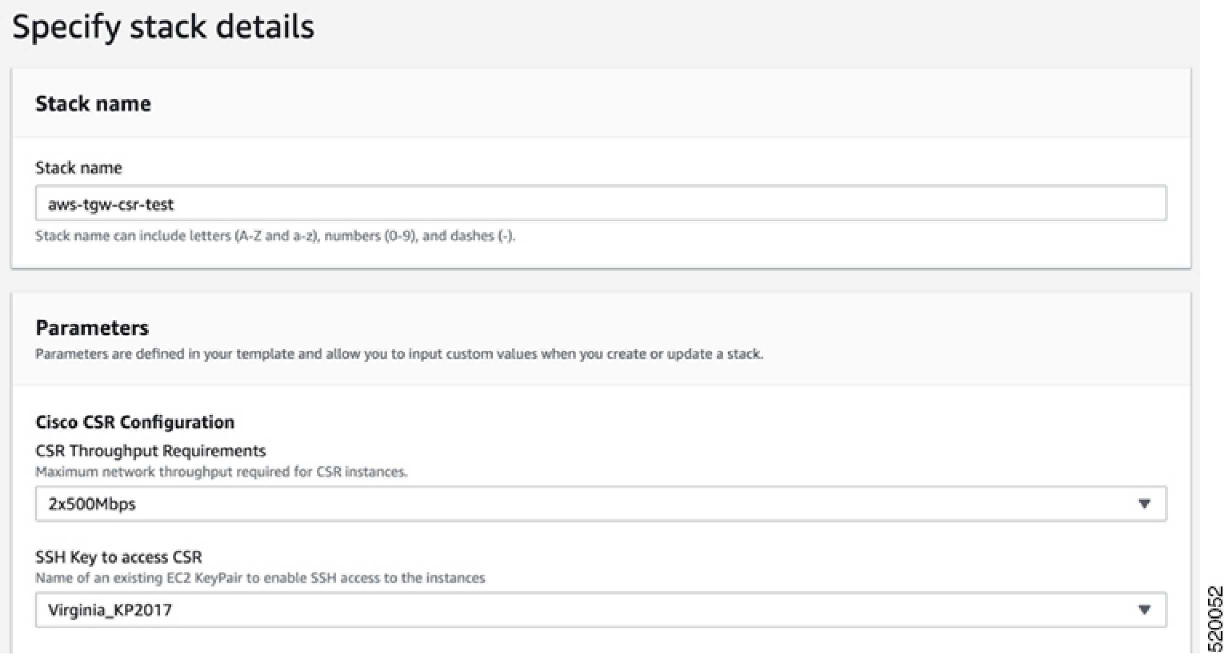
Step 3 Launch the template in the appropriate region where you are located. The system displays the AWS Cloudformation Service page. Click **Next**.

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

Parameter	Description
CSR Throughput Requirements	The required throughput for the CSR 1000V instance. This determines the instance type to be launched. The default value is 2 x 500 Mbps.
SSH Key to access CSR	The public/private key pair that allows a secure connection to be made to a CSR 1000V 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 CSR 1000V instances. This prevents accidental CSR 1000V termination. Cisco recommends 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 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**.
Transit GW BGP ASN	The BGP Autonomous System Number (ASN) for the transit gateway. 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 CSR VPN endpoint to control the traffic flow through the Transit VPC CSR 1000v instances. For example, when integrating with stateful on-prem firewalls.

Parameter	Description
Optional AZ configuration 1st Subnet	The availability Zone number for Public Subnet1.
Optional AZ configuration 2nd Subnet	The availability Zone number for Public Subnet2.



Specify stack details

Stack name

Stack name

aws-tgw-csr-test

Stack name can include letters (A-Z and a-z), numbers (0-9), and dashes (-).

Parameters

Parameters are defined in your template and allow you to input custom values when you create or update a stack.

Cisco CSR Configuration

CSR Throughput Requirements

Maximum network throughput required for CSR instances.

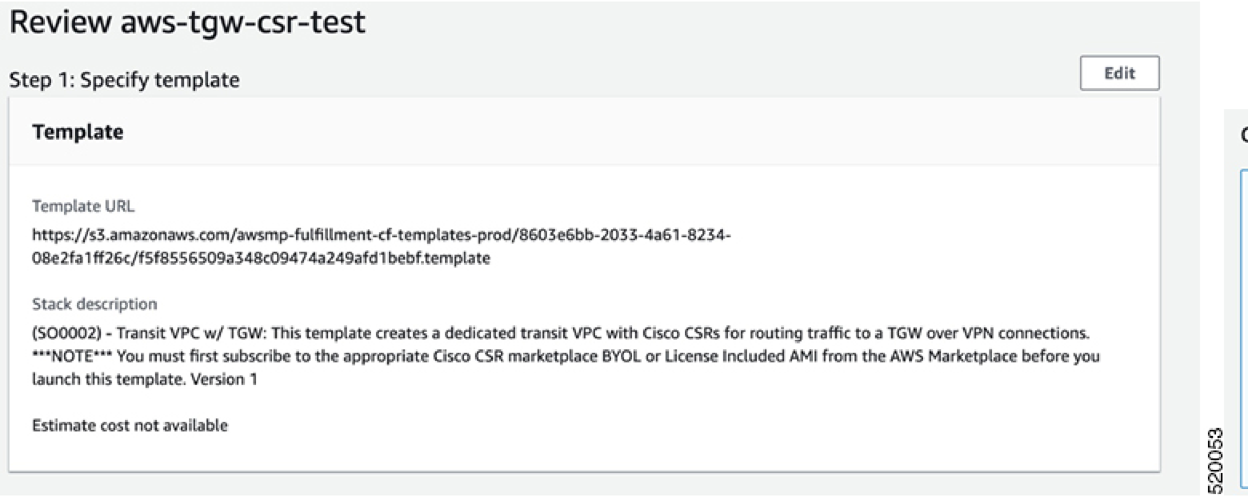
2x500Mbps ▼

SSH Key to access CSR

Name of an existing EC2 KeyPair to enable SSH access to the instances

Virginia_KP2017 ▼

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.



Review aws-tgw-csr-test

Step 1: Specify template

Template

Template URL

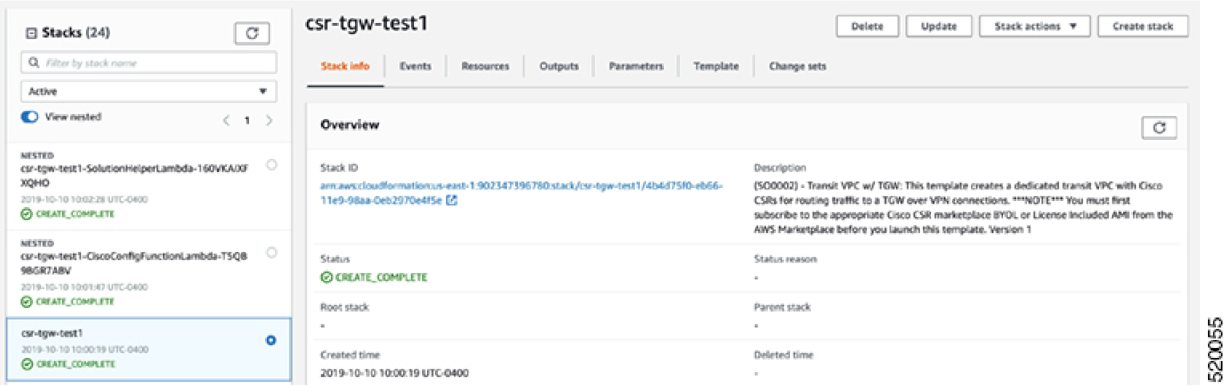
https://s3.amazonaws.com/awsmp-fulfillment-cf-templates-prod/8603e6bb-2033-4a61-8234-08e2fa1ff26c/f5f8556509a348c09474a249afd1bebf.template

Stack description

(SO0002) - Transit VPC w/ TGW: This template creates a dedicated transit VPC with Cisco CSRs for routing traffic to a TGW over VPN connections. ***NOTE*** You must first subscribe to the appropriate Cisco CSR marketplace BYOL or License Included AMI from the AWS Marketplace before you launch this template. Version 1

Estimate cost not available

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**.



NESTED
csr-tgw-test1-SolutionHelperLambda-160VKAJXFXQHO
2019-10-10 10:02:28 UTC-0400
CREATE_COMPLETE

NESTED
csr-tgw-test1-CiscoConfigFunctionLambda-T5QB9BGR7ABV
2019-10-10 10:01:47 UTC-0400
CREATE_COMPLETE

csr-tgw-test1
2019-10-10 10:00:19 UTC-0400
CREATE_COMPLETE

csr-tgw-test1

Stack ID
arn:aws:cloudformation:us-east-1:902347396780:stack/csr-tgw-test1/4b4d75f0-eb66-11e9-98aa-0eb2970e4f5e

Description
(SO0002) - Transit VPC w/ TGW: This template creates a dedicated transit VPC with Cisco CSRs for routing traffic to a TGW over VPN connections. ***NOTE*** You must first subscribe to the appropriate Cisco CSR marketplace BYOL or License Included AMI from the AWS Marketplace before you launch this template. Version 1

Status
CREATE_COMPLETE

Status reason
-

Root stack
-

Parent stack
-

Created time
2019-10-10 10:00:19 UTC-0400

Deleted time
-

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 2019
!
version 16.9
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
```