Transit Network VPC
(Cisco CSR)

AWS Reference Deployment

Steve Morad

Bryan Miller

Sidhartha Chauhan

July 2016

Contents

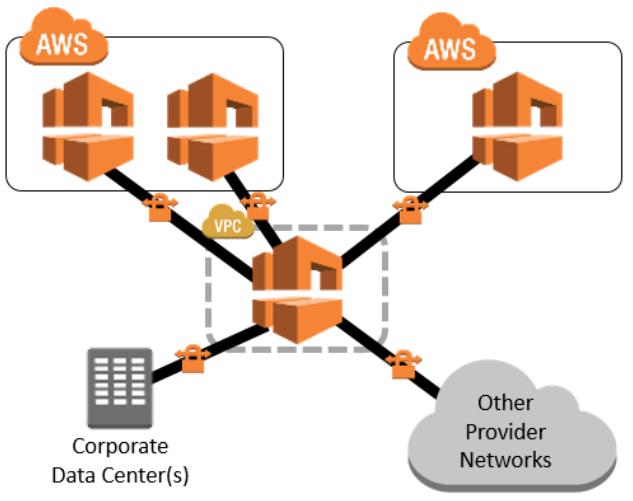
About This Guide

This reference implementation guide discusses architectural considerations and configuration steps for deploying a transit VPC on the Amazon Web Services (AWS) cloud. It includes links to AWS CloudFormation templates that launch, configure, and run the AWS compute, network, storage, and other services required to deploy this solution on AWS, using AWS best practices for security and availability.

The guide is intended for IT infrastructure architects, administrators, and DevOps professionals who have practical experience architecting on the AWS Cloud.

Overview

Amazon Virtual Private Cloud (Amazon VPC) provides customers with the ability to create as many virtual networks as they need, as well as different options for connecting those networks to each other and to non-AWS infrastructure. One common strategy for connecting multiple, geographically disperse VPCs and remote networks is to create a transit VPC that serves as a global network transit center. A transit VPC simplifies network management and minimizes the number of connections required to connect multiple VPCs and remote networks. This design can save time and effort and also 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.



This guide provides infrastructure and configuration information for planning and deploying a transit VPC that assumes a typical hub-and-spoke network topology, as depicted in the diagram to the left. In this design, remote VPCs access each other and remote networks through the central, transit VPC.

The AWS Cloud provides a suite of infrastructure services that enable you to deploy a transit VPC solution in a highly available, fault-tolerant, and affordable way. By integrating Cisco Cloud Services Router[1]

[1] https://aws.amazon.com/marketplace/pp/B00EV8VWWM
https://aws.amazon.com/marketplace/pp/B00OCG4Q4E

(CSR) with the AWS Cloud, you can take advantage of the functionality of enterprise-class networking services and VPN along with the flexibility and security of AWS.

The information in this guide assumes basic knowledge of highly available remote-network connectivity, IPsec VPNs, network addressing, subnetting, and routing. The following sections do not include general installation or configuration tasks for Cisco CSR. For additional general guidance, best practices, and licensing details consult the Cisco product documentation.

Cost and Licenses

You are responsible for the cost of the AWS services used while running this reference deployment. You are also responsible for the Cisco CSR licenses, which you can either purchase beforehand or request from the AWS Marketplace, depending on the deployment model you choose: Bring Your Own License (BYOL) or License Included.[2]

As of the date of publication, the cost for running a transit VPC with this solution's default settings in US East (N. Virginia) is as shown in the table below.

Transit VPC Deployment Size	BYOL Cost/Hour	License Included Cost/Hour
2 x 500 Mbps	$0.21	$4.35
2 x 1 Gbps	$0.84	$6.22
2 x 2 Gbps	$1.68	$8.40

Each spoke VPC connected to the transit network costs an additional $0.10/hour, plus network transit costs. Prices are subject to change. For full details, see the pricing webpage for each AWS service you will be using in this solution.

[2] See the Prerequisites section for detailed information.

Architecture Overview

Deploying this solution with the **default parameters** builds the following environment in the AWS Cloud.

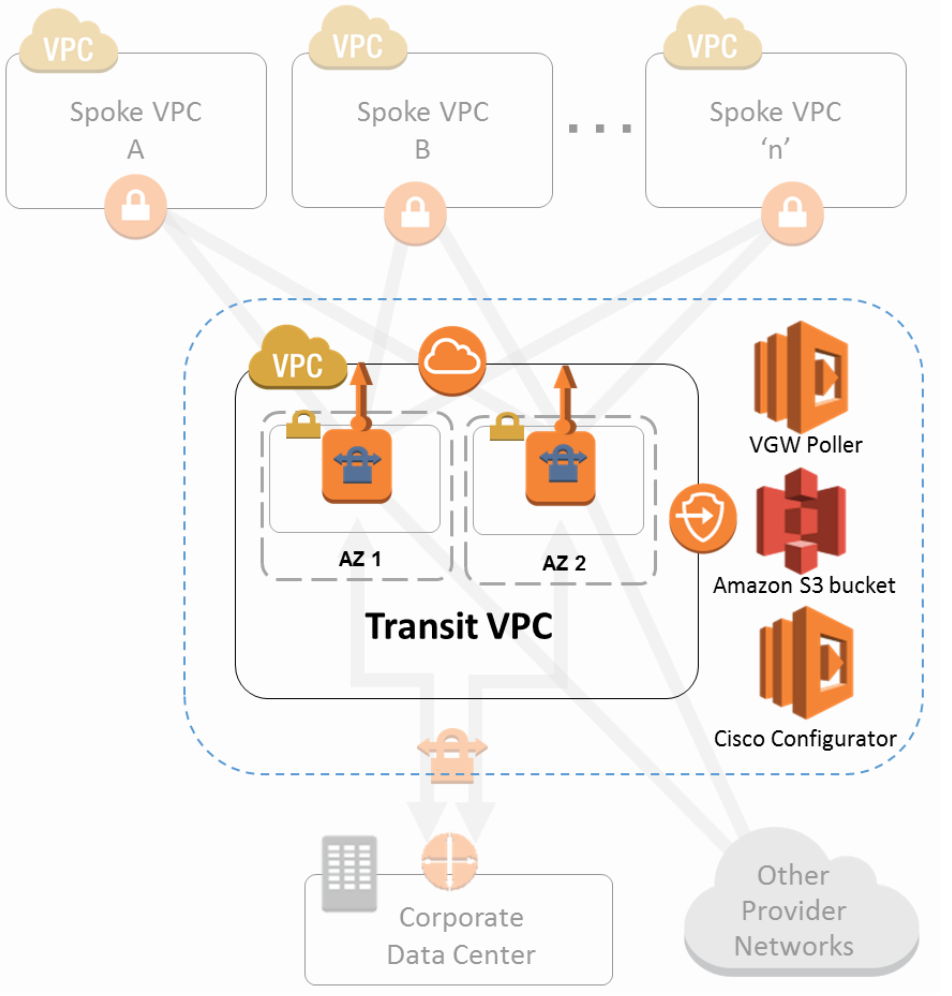


Figure 1: Transit VPC Solution Architecture on AWS

This highly available design deploys two VPN appliances (Cisco CSR 1000v instances) into separate Availability Zones of a dedicated transit VPC. Spoke VPCs are connected to the transit network through dynamically routed VPN connections between their virtual private gateways (VGWs) and the CSR instances. This design uses VPN connections to enable routing between any connected network, including external networks or spoke VPCs in other AWS Regions. (Note that VPC peering[3] connections, although convenient, are not available between different AWS Regions.) VPN connections also allows spoke VPC resources to

[3] http://docs.aws.amazon.com/AmazonVPC/latest/UserGuide/vpc-peering.html

leverage VGW capabilites for routing and failover in order to maintain highly available network connections to the transit VPC instances. Remote networks also connect to the transit VPC using redundant, dynamically routed VPN connections between their customer gateways and the CSR instances. This design supports dynamic routing protocols, which customers can use to automatically route traffic around potential network failures as well as to propagate network routes to remote networks.

Note that all communication with the CSR instances, including the VPN connections between corporate data centers or other provider networks and the transit VPC, uses the transit VPC Internet gateway and the instances' Elastic IP addresses. Each CSR instance has an associated Amazon CloudWatch alarm that enables automatic recovery[4] of the instance if the underlying EC2 hardware fails.

Along with providing direct network routing between VPCs and on-premises networks, this design also enables the transit VPC to implement more complex routing rules, such as network address translation (NAT) between overlapping network ranges, or to add additional network-level packet filtering or inspection. Although it is possible, we recommend that you consider implementing non-overlapping network ranges for your private networks to simplify the ability to route between remote networks. Although a transit network can be an excellent place to implement NAT rules to compensate for overlapping networks, this adds additional complexity to the network design.

The AWS-to-AWS VPN connectivity in this design relies on the capabilities of the Amazon VPC software VPN appliance and the Cisco CSR 1000v for AWS. It also relies on the capabilities of the hardware VPN device deployed on premises that connects to the software VPN appliance. While every effort has been made to come up with a solid configuration, each customer should verify the configuration and adapt it to their specific needs.

Solution Features

The automated transit VPC solution provides the following features:

- **AWS network connectivity**: You can connect any spoke VPCs that you wish—within the same AWS Region, across AWS Regions, and even from a second AWS account.
- **Remote network connectivity**: You can connect to your own data centers, other colocation providers, Managed Service Providers (MSPs), or even other cloud providers.
- **Automated configuration:** This solution leverages AWS Lambda to automatically configure VPN connections for the spoke VPCs you want to add to the transit network.

[4] http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ec2-instance-recover.html

Note that manual configuration of the CSR instances is required to connect remote networks to the transit VPC, and also to remove spoke VPCs.

- **Additional functionality:** You can customize this solution to provide additional network features:
  - IDS/IPS, Next-Generation Firewall
  - NAT to bridge overlapping networks
  - NAT to access VPC-E, EFS, or internal Amazon Route 53

Transit VPC Components

As described in the *Architecture Overview* section, at the core of the design is a VPC (the transit VPC) that acts as a central hub for traffic flowing to any other destination, whether it be another VPC or a remote network. The transit VPC hosts two CSR instances that allow for VPN termination and routing. This solution uses two AWS Lambda functions, the *VGW Poller* and the *Cisco Configurator*, to automatically configure VPN connections between these instances and spoke VPCs.

The following diagram gives an overview of the various components and steps involved in connecting spoke VPCs to the transit network. (See the appendix for detailed information.)

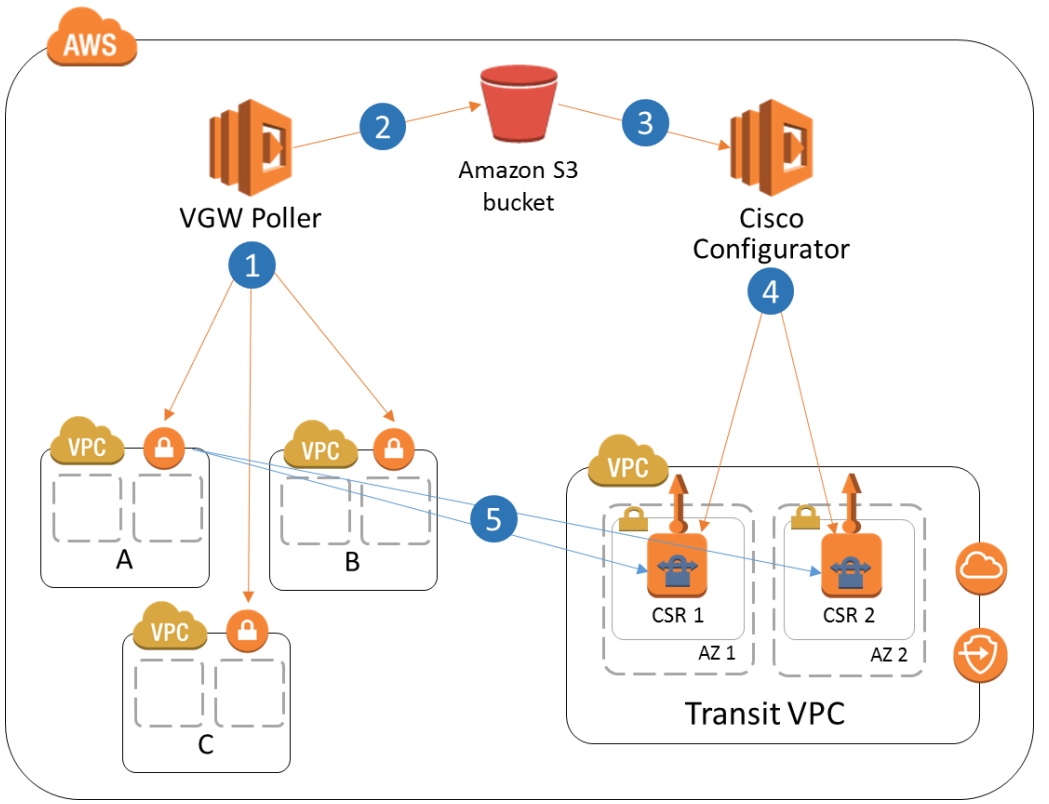


Figure 2: Connecting Spoke VPCs to the Transit Network

The process for adding a new spoke VPC is as follows:

1. Every (1) minute, an Amazon CloudWatch rule[5] invokes the VGW Poller Lambda function, which iterates through each AWS Region of a customer's account, searching for appropriately tagged spoke VGWs that do not have existing VPN connections.
2. When the VGW Poller identifies an applicable spoke VGW, it creates the corresponding customer gateways (if required) and VPN connections to each CSR, and then saves this connection information to an Amazon S3 bucket.
3. The S3 Put event invokes the Cisco Configurator Lambda function, which parses the VPN connection information and generates the necessary config files.
4. The Cisco Configurator pushes the configuration to the CSR instances using SSH.
5. As soon as the Cisco configuration is applied onto the CSR instances, the VPN tunnels come up and Border Gateway Protocol (BGP) neighbor relationships are established to the spoke VPCs.

AWS CloudFormation Templates

This solution uses AWS CloudFormation to bootstrap AWS infrastructure and automate the deployment of a transit VPC on the AWS Cloud from scratch. It includes the following CloudFormation templates, which you can download before deployment:

View template

transit-vpc-primary-account: This is the primary solution template you use to launch the transit VPC and all associated components, as described in the previous section. The default configuration offers three deployment size options based on common network bandwidth requirements, but you can also customize the template based on your specific network needs.

View template

transit-vpc-second-account: Use this template to connect a second AWS account to the transit network. It launches the VGW Poller Lambda function, which automatically searches and sets up VPN connections for appropriately tagged spoke VPCs from that account. Detailed instructions are in Step 4 of the *Automated Deployment*.

[5] See the CloudWatch section for more information.

[Note: template names in this doc don't match the links exactly. Ignore for now – links should work, though.]

Automated Deployment

Before you launch the automated deployment, please review the architecture, configuration, network security, and other considerations discussed in this guide. Follow the step-by-step instructions in this section to configure and deploy a transit VPC into your account.

Time to deploy: Approximately five (5) minutes

Prerequisites

You must decide on the licensing model for the Cisco Cloud Services Router (CSR) before launching the AWS CloudFormation template. If you choose to bring your own license (the *BYOL* model), you will need to sign up for the BYOL version of Cisco Cloud Services Router (CSR)
1000V, located here: https://aws.amazon.com/marketplace/pp/B00EV8VWWM
Otherwise, use the *License Included* model and sign up for the License Included version of Cisco Cloud Services Router (CSR) 1000V located here:
https://aws.amazon.com/marketplace/pp/B00OCG4Q4E

What We'll Cover

The procedure for deploying this architecture on AWS consists of the following steps. For detailed instructions, follow the links for each step.

Step 1. Accept the Cisco Software Terms

- Choose the proper Cisco Cloud Services Router (CSR) from the AWS Marketplace and click through the agreement.

Step 2. Launch the Stack

- Launch the AWS CloudFormation template into your AWS account.
- Enter values for required parameters: **Stack name**, **SSH Key to access CSR**, **License Model**.
- Review the other template parameters, and adjust if necessary.

Step 3. Tag the Spoke VPCs

- Tag the VGW of each VPC that you want to add to the transit network.

Step 4. Optional - Launch the VPC Poller in a Second Account

- Launch the *transit-vpc-second-account* AWS CloudFormation template in another AWS account that contains spoke VPCs.

Step 1. Accept the Cisco Software Terms

Before launching the Cisco CSR AMI from AWS Marketplace, you are required to accept the Cisco Software terms.

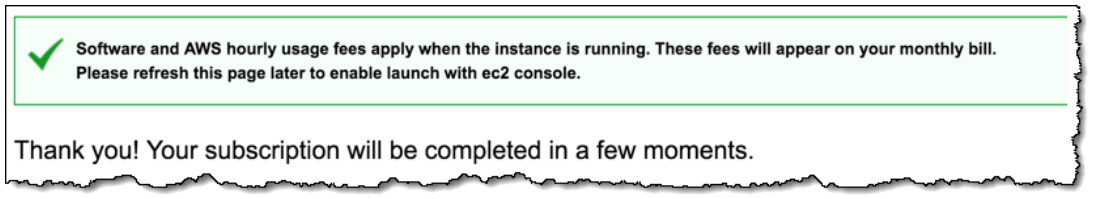
1. In AWS Marketplace, go to the page of the appropriate Cisco Software package based on your preference:

   - BYOL version of Cisco Cloud Services Router (CSR) 1000V: https://aws.amazon.com/marketplace/pp/B00EV8VWWM
   - License Included version of Cisco Cloud Services Router (CSR) 1000V: https://aws.amazon.com/marketplace/pp/B00OCG4Q4E

   Review the **Pricing Details** for the AWS Region where you place to deploy the transit VPC, and then choose **Continue**.

2. Click **Manual Launch**. Be sure to read the End User License Agreement (EULA) and AWS Customer Agreement. If you accept these terms, choose **Accept Software Terms**.

   You should see a subscription confirmation message. You will also receive a confirmation email.



Step 2. Launch the Stack

This automated AWS CloudFormation template deploys a transit VPC on the AWS Cloud. Please make sure that you've selected the proper Cisco CSR in AWS Marketplace before launching the stack and have accepted the software terms. If you plan to add VPCs from a second AWS account to the transit network, make sure to note its account ID before you launch this stack.

> **Note**: You are responsible for the cost of the AWS services used while running this solution. See the Cost and Licenses section for more details. For full details, see the pricing webpage for each AWS service you will be using in this solution.

1. Log in to the AWS Management Console and click the button to the right to launch the *transit-vpc-primary-account* AWS CloudFormation template.

   Launch Full Solution

   You can also download the template as a starting point for your own implementation.
2. The template is launched in the US East (N. Virginia) Region by default. To launch the transit VPC in a different AWS Region, use the region selector in the console navigation bar.

   > **Note**: This solution uses the AWS Lambda service, which is currently available in specific AWS Regions only. Therefore, you must launch this solution an AWS Region where Lambda is available. [6]
3. On the **Select Template** page, verify that you selected the correct template and choose **Next**.
4. On the **Specify Details** page, assign a name to your transit VPC in the **Stack name** field.
5. Under **Parameters**, review the parameters for the template and modify them as necessary. This solution uses the following default values.

Parameter	Default	Description
CSR Throughput Requirements	`2x500Mbps`	A drop-down box with three options: `2x500Mbps` (c3.large), `2x1Gbps` (c3.2xlarge), and `2x2Gbps` (c3.4xlarge)
SSH Key to access CSR	<Requires input>	Public/private key pair, which allows you to connect securely to your instance after it launches. When you created an AWS account, this is the key pair you created in your preferred region.
License Model	`LicenseIncluded`	A drop-down box with two choices: `LicenseIncluded` and `BYOL`
Prefix for S3 Objects	`vpnconfigs/`	Any text string you want to use to specify the name of the prefix for the Amazon S3 objects that are created
Additional AWS Account ID	<Optional input>	Enter the account ID of an additional AWS account that you want to connect to the transit network. This is necessary to grant that account access to the S3 bucket.

[6] For the most current AWS Lambda availability by region, see https://aws.amazon.com/about-aws/global-infrastructure/regional-product-services/

Parameter	Default	Description
Transit VPC CIDR Block	`100.64.127.224/27`	CIDR block for the transit VPC. You can modify the VPC and subnet CIDR address ranges to avoid collisions with your network.
1st Subnet Network	`100.64.127.224/28`	CIDR block for the transit VPC subnet created in AZ1
2nd Subnet Network	`100.64.127.240/28`	CIDR block for the transit VPC subnet created in AZ2
Transit VPC BGP ASN	`64512`	BGP ASN to use for the transit VPC
Spoke VPC Tag Name	`transitvpc:spoke`	Tag name (key) to identify spoke VPCs to connect to the transit VPC. You can modify the tag name to align with any existing naming conventions. Use a name that is not likely to be used on VGWs for a different purpose to ensure you do not mistakenly add a VPC to the transit network.
Spoke VPC Tag Value	`true`	Tag value to determine which spoke VPCs to connect to the transit VPC. You can modify the tag value to align with any existing naming conventions. Be sure to use a value that is easy to understand and implement consistently.

For illustration in this document, we used the following values:

- **Stack name:** `TransitVPC`
- **SSH Key to access CSR:** (Selected a local key pair used for demo purposes)
- **License Model:** `LicenseIncluded`

We left the default value for all other fields in the **AWS Service Configuration** and **Network Configuration** sections.

6. Choose **Next**.
7. On the **Options** page, you can specify tags (key-value pairs) for resources in your stack and set additional options, and then choose **Next**.
8. On the **Review** page, review and confirm the settings. Be sure to check the box acknowledging that the template will create IAM resources.
9. Choose **Create** to deploy the stack.

   You can view the status of the stack in the AWS CloudFormation Console in the **Status** column. You should see a status of CREATE_COMPLETE in roughly five (5) minutes.

> **Note:** In addition to the VGW Poller (`poller`) and Cisco Configurator (`configurator`) Lambda functions, this solution includes the `solution-helper` Lambda function, which runs only during initial configuration or when resources are updated or deleted.

> When running this solution, you will see all Lambda functions in the AWS console, but only the two primary solution functions are regularly active. However, do not delete the `solution-helper` function as it is necessary to manage associated resources.

10. To see details for the stack resources, choose the **Outputs** tab. The following tables describes each of these outputs in more detail.

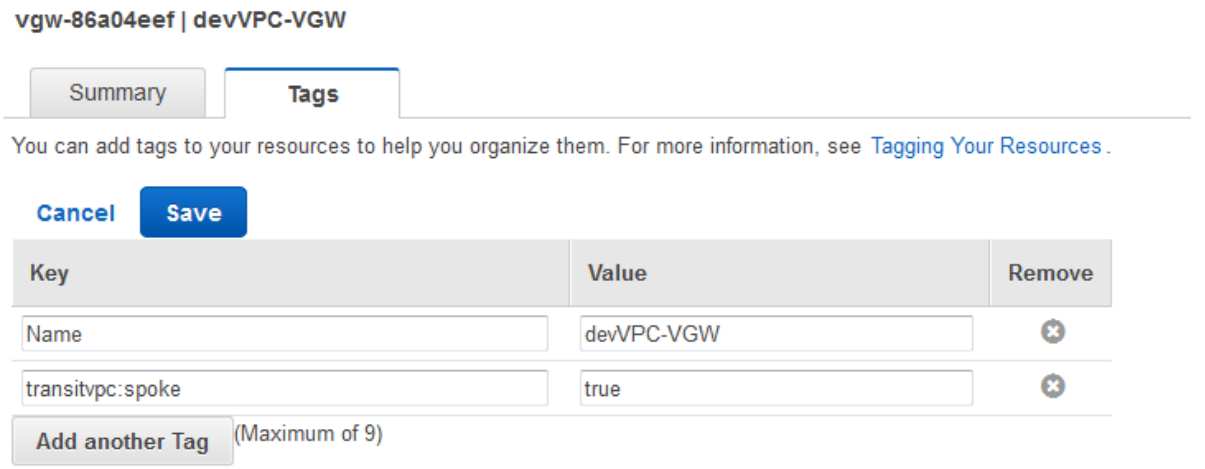
Key	Description
CSR1	Public IP address for CSR 1, which is necessary for manually configuring CSRs or connecting the transit VPC to remote networks
CSR2	Public IP address for CSR 2, which is necessary for manually configuring CSRs or connecting the transit VPC to remote networks
ConfigS3Bucket	S3 bucket created by this template, and used to store VPN connection information **Important:** If you plan on connecting a second AWS account to the transit VPC, you must note this value. You will enter the name of this S3 bucket as a parameter in the *transit-vpc-second-account* template that you launch in Step 4.
BucketPrefix	Should match the string you entered in the template's **Prefix for S3 Objects** value.
SpokeVPCTag	Should match the spoke VPC tag name you entered in the template.
SpokeVPCTagValue	Should match the spoke VPC tag value you entered in the template.

Step 3. Tag the Spoke VPCs

After the transit VPC stack launch completes, you can apply tags to existing VGWs that you wish to add to the transit network. Make sure to use the tag name and tag value that you specified during the stack creation. These VGWs will be automatically connected to the transit VPC within a few minutes.

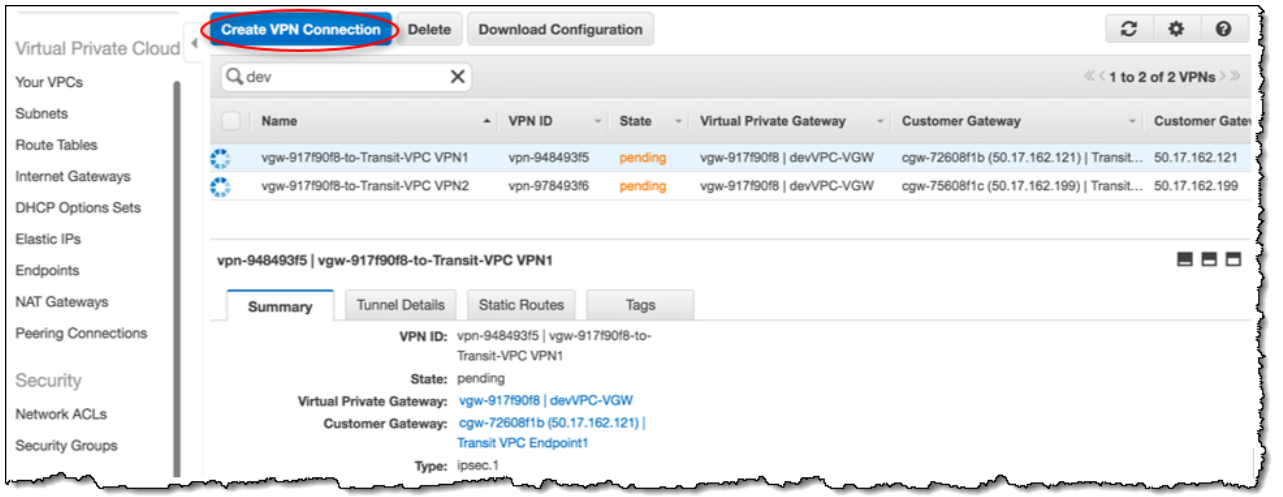
For illustration purposes, assume you want to add a developer-environment VPC to your transit, which has a VGW that you named `devVPC-VGW`.

1. In the left navigation pane of the Amazon VPC console, choose **Virtual Private Gateways.**
2. Select the VGW you want to modify, choose the **Tags** tab, and choose **Edit**.
3. Add the tag key and value that you defined in the AWS CloudFormation template. For our example, we didn't change the default values for these parameters, so we will enter the **Key** `transitvpc:spoke` and the **Value** `true`.



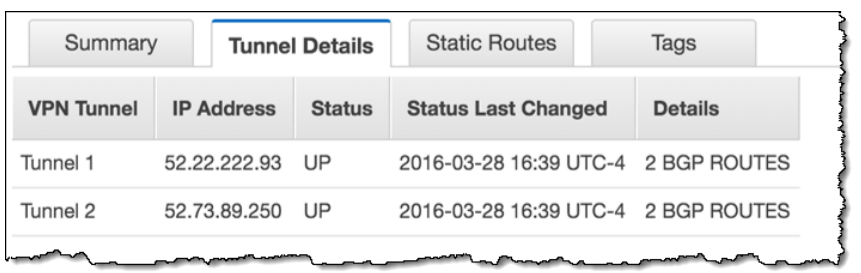
Within a minute or less, the VGW Poller Lambda function will find the tag and create a VPN connection from the spoke VGW to the CSR instances located in the transit VPC.

4. In the left navigation pane, choose **VPN Connections** to confirm the spoke VPC was successfully added to the transit network.



You should see two new VPN connections in `pending` state. This should soon change to `available`.

5. To see the tunnel status and BGP routes received, choose the **Tunnel Details** tab.



Step 4. Connect a Second AWS Account (Optional)

As explained in the AWS CloudFormation Templates section, the *transit-vpc-second-account* template is used to launch the VGW Poller Lambda function in a second AWS account, so that it can search for and add VPCs from that account to the transit network. During the initial launch of the transit VPC in the primary account (Step 2), if you specified an additional AWS account to participate in the transit network, you will need to launch the template in that account and tag the spoke VPCs accordingly.

1. Log in to the AWS Management Console of the applicable account, and click the button to the right to launch the *transit-vpc-second-account* AWS CloudFormation template.
   You can also download the template to use it as a starting point for your own implementation.

Launch VGW Poller in Second Account

2. The template is launched in the US East (N. Virginia) Region by default. To launch the transit VPC in a different AWS Region, use the region selector in the console navigation bar.

> **Note**: This solution uses the AWS Lambda service, which is currently available in specific AWS Regions only. Therefore, you must choose an AWS Region where Lambda is available. [7] Note that although you launch the VGW Poller in a single AWS Regions, it searches all AWS Regions of a customer account.

3. On the **Select Template** page, keep the default settings for **Stack** and **Template Source**.
4. On the **Specify Details** page, assign a name to your transit VPC in the **Stack name** field.
5. Under **Parameters**, review the parameters for the template and modify them as necessary. They include the following default values.

Parameter	Default	Description
BucketName	`transit-vpc<xyz>`	Use the exact bucket name that was an output from the Transit VPC template you deployed in Step 2: Launch the Stack. All transit VPC configuration files are stored in the same S3 bucket.

[7] For the most current AWS Lambda availability by region, see https://aws.amazon.com/about-aws/global-infrastructure/regional-product-services/

Parameter	Default	Description
BucketPrefix	`vpnconfigs/`	Use the exact string you entered for the **Prefix for S3 Objects** value when you launched the *transit-vpc-primary-account* template (Step 2).

6. Choose **Next**.
7. On the **Options** page, you can specify tags (key-value pairs) for resources in your stack and set additional options, and then choose **Next**.
8. On the **Review** page, review and confirm the settings. Be sure to check the box acknowledging that the template will create IAM resources.
9. Choose **Create** to deploy the stack.

   You can view the status of the stack in the AWS CloudFormation Console in the **Status** column. You should see a status of CREATE_COMPLETE in roughly five (5) minutes.
10. To see details for the stack resources, choose the **Outputs** tab. The following tables describes each of these outputs in more detail.

Key	Description
PollerFunction	The name of the Lambda poller function.
PollerFunctionARN	The ARN for the new Lambda poller function.

11. The VGW Poller Lambda function is now running in this second AWS account, and you can apply tags to the VGWs that you wish to add to the transit network (see Step 3. Tag the Spoke VPCs).

Security

The AWS Cloud provides a scalable, highly reliable platform that helps customers deploy applications and data quickly and securely.

When you build systems on AWS infrastructure, security responsibilities are shared between you and AWS. This shared model can reduce your operational burden as AWS operates, manages, and controls the components from the host operating system and virtualization layer down to the physical security of the facilities in which the services operate. In turn, you assume responsibility and management of the guest operating system (including updates and security patches), other associated applications, as well as the configuration of the AWS-provided security group firewall. For more information about security on AWS, visit the AWS Security Center.

Security Groups

The security groups created in this solution are designed to control and isolate network traffic between the AWS Lambda functions, CSR instances, and remote VPN endpoints. We recommend that you review the security groups and further restrict access as needed once the deployment is up and running.

In the transit VPC network, all VPN connections originate from the CSR instances. Therefore, no inbound traffic is necessary other than for access to the CSRs. This solution includes a security group rule that grants access to inbound SSH traffic from the Cisco Configurator Lambda function only.

Additional Security Settings

Password authorization is explicitly disabled. The Cisco Configurator Lambda function generates an SSH key pair, stores it securely in the Amazon S3 bucket, and uses that key pair for authentication to access the CSR instances. The Cisco Configurator Lambda function is configured to run inside the transit VPC only.

The S3 bucket is configured for AES-256 Server Side Encryption (SSE) for all files stored in the bucket. An Amazon S3 bucket policy controls which additional accounts can have access to the S3 bucket, therefore enabling those accounts to connect their VGWs to the transit VPC network. This policy may be modified manually to add additional accounts to the transit VPC network.

Amazon CloudWatch

This solution creates the following CloudWatch rules and logs to invoke and log associated Lambda functions.

CloudWatch Logs:

- `/aws/lambda/transit-vpc-poller` – Logged actions of the VGW Poller Lambda function
- `/aws/lambda/ transit-vpc-cisco-configurator` – Logged actions of the Cisco Configurator Lambda function
- `/aws/lambda/transit-vpc-solution-helper-XXXXXXXXXXXX` or `transit-vpc-poller-solution-helper-XXXXXXXXXXXX` – Created by the *solution-helper* customer resource when launching the *transit-vpc-primary-account* template or the *transit-vpc-poller* template.

CloudWatch Rule:

- `vgw-poller-1min`– This rule invokes the VGW Poller Lambda function every (1) minute.

Testing

This solution includes additional AWS CloudFormation templates that you can use to test the transit network in your AWS account.

Transit VPC Test with Tsunami UDP

The *transit-vpc-spoke-vpc-withec2-tsunamiudp* AWS CloudFormation template creates the transit VPC, automatically installs the server-side tsunami UDP protocol (http://tsunami-udp.sourceforge.net/), and downloads two test files from an Amazon S3 bucket. Use the following steps to test your transit network with this template.

1. Identify the spoke VPCs you want to use for the test and tag them as follows:
   - Tag name: `transitvpc:spoke`
   - Tag value: `true`

   Note: To quickly create a VPC with an Amazon EC2 instance, use the *transit-vpc-spoke-vpc-withec2* AWS CloudFormation template (see the next section).

2. Click the button to the right to launch the *transit-vpc-spoke-vpc-withec2-tsunamiudp* AWS CloudFormation template. You can also download the template.

   Launch Test Transit VPC

3. Use the default settings. When the stack build completes, you will have a new transit VPC with a preinstalled Tsunami server and two test files.
4. Go to http://tsunami-udp.cvs.sourceforge.net/viewvc/tsunami-udp/docs/USAGE.txt and follow the instructions in *1. Very Quick Guide*. This walks you through a simple test using the `GET` command from client to server.
5. Then….?

Spoke VPC Templates

Use the following templates to quickly launch VPCs in your AWS account for your own testing protocol.

- **transit-vpc-spoke-vpc:** This template launches a basic VPC with a VGW and route table. Use this to create new VPCs in your accounts or as a starting point to retrofit your existing VPC templates with examples for creating transit VPC tagged spoke VGWs and related resources.
- **transit-vpc-spoke-vpc-withec2:** This template launches a more complete VPC along with a small EC2 instance. Use this template as a starting point for new VPCs or for testing connectivity between spoke VPCs

Additional Resources

AWS services

- AWS CloudFormation
  http://aws.amazon.com/documentation/cloudformation/
- Amazon VPC
  http://aws.amazon.com/documentation/vpc/
- AWS Lambda
  https://aws.amazon.com/documentation/lambda/
- Amazon EC2 user guide for Linux instances
  http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/concepts.html
- Amazon S3
  https://aws.amazon.com/documentation/s3/
- Amazon CloudWatch
  https://aws.amazon.com/documentation/cloudwatch/

Cisco products and documentation

- AWS Marketplace offerings:
  - Cisco Cloud Services Router (CSR) 1000V - Bring Your Own License (BYOL)
    https://aws.amazon.com/marketplace/pp/B00EV8VWWM
  - Cisco Cloud Services Router (CSR) 1000V - Security Pkg. Max Performance
    https://aws.amazon.com/marketplace/pp/B00OCG4Q4E

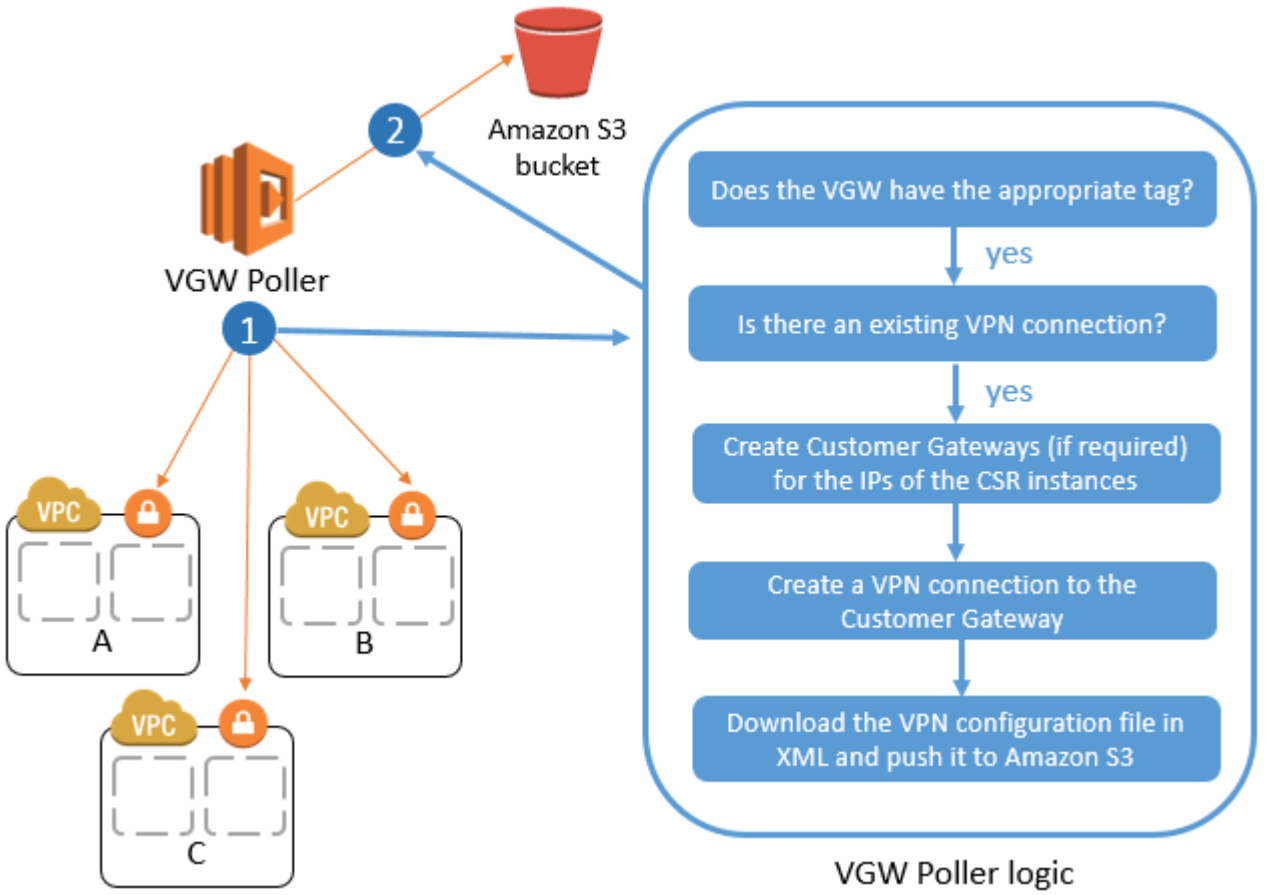
Related AWS Solution Briefs

- Multiple-VPC VPN Connection Sharing
  https://do.awsstatic.com/aws-answers/AWS_Multiple_VPC_VPN_Connection_Sharing.pdf
- Single Data Center HA Network Connectivity
  https://do.awsstatic.com/aws-answers/AWS_Single_Data_Center_HA_Network_Connectivity.pdf
- Multiple Data Center HA Network Connectivity
  https://do.awsstatic.com/aws-answers/AWS_Multiple_Data_Center_HA_Network_Connectivity.pdf

Appendix: Component Details

VGW Poller

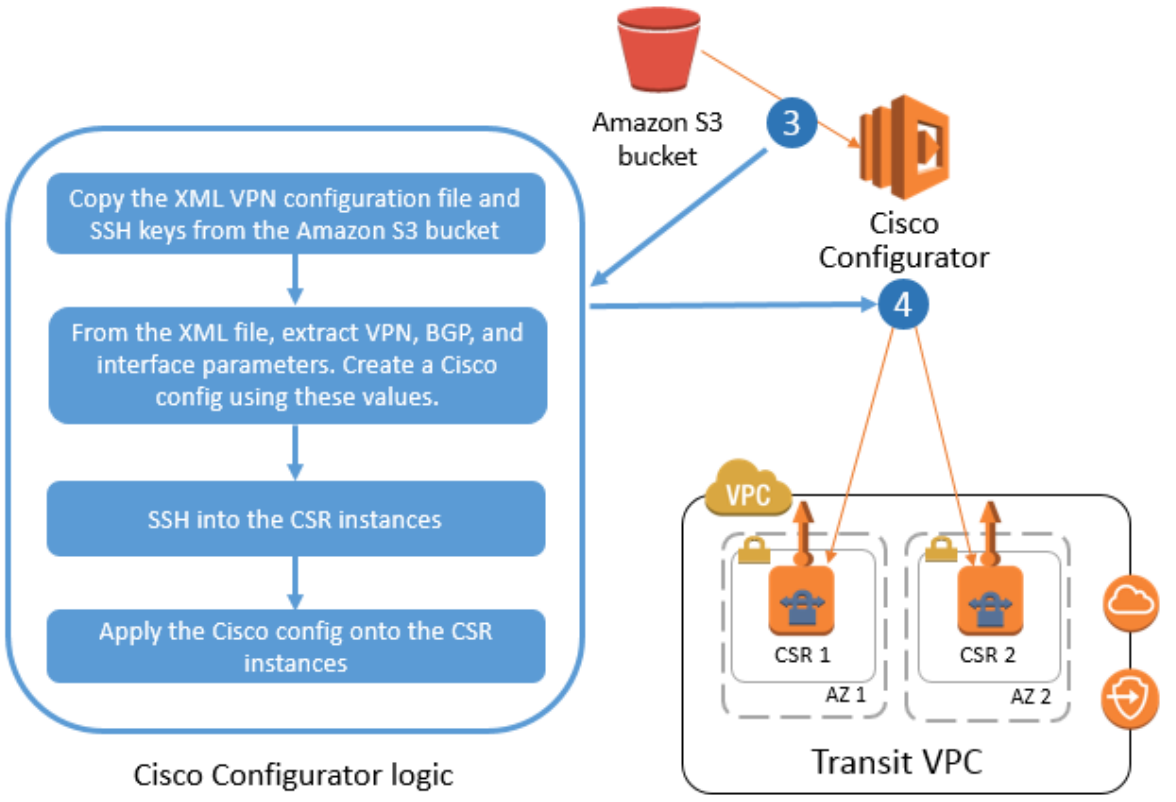
An Amazon CloudWatch rule invokes the VGW Poller Lambda function every minute. The VGW Poller is configured to iterate through each AWS Region of a customer's account, searching for appropriately tagged spoke VGWs that do not have existing VPN connections to the transit VPC. Once a spoke VGW is identified, the function creates corresponding customer gateways (if required) and VPN connections to each CSR. After creating VPN connections, the function retrieves the VPN configuration and saves it to an Amazon S3 bucket. The following diagram describes the VGW Poller logic in more detail.



VGW Poller logic

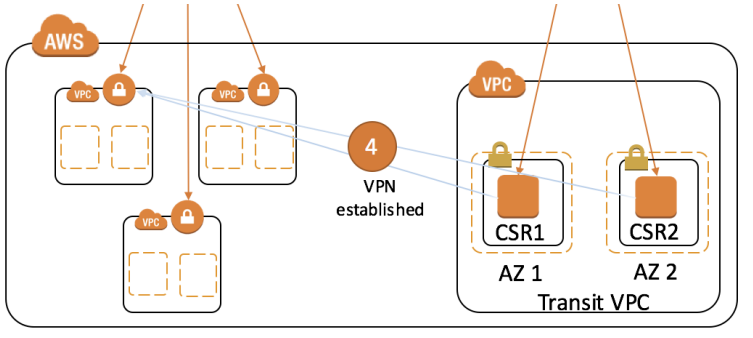
Cisco Configurator

The Cisco Configurator is an AWS Lambda function that is invoked by Amazon S3 Put events to the solution's S3 bucket. After the VGW Poller function writes new VPN connection details to the S3 bucket, the Cisco Configurator function parses the VPN connection information and generates the necessary config files to establish VPN connections to spoke VPCs. It then pushes the configuration to the CSR instances using SSH. The following diagram describes the Cisco Configurator logic in more detail.



BGP and Failover

As soon as the Cisco configuration is applied onto the CSR instances, the VPN tunnels come up and Border Gateway Protocol (BGP) neighbor relationships are established.



Send Us Feedback

We welcome your questions and comments. Please post your feedback on the [Solution Builder Forum COMING SOON].

You can visit our [GitHub repository COMING SOON] to download the templates and scripts for this solution, and to share your customizations with others.

Document Revisions

Date	Change	In sections
<month> 2016	*Brief description of change. Formatting and minor text changes don't warrant any mention; major additions and changes do.*	*Links to revised sections*

Notices

This document is provided for informational purposes only. It represents AWS's current product offerings and practices as of the date of issue of this document, which are subject to change without notice. Customers are responsible for making their own independent assessment of the information in this document and any use of AWS's products or services, each of which is provided "as is" without warranty of any kind, whether express or implied. This document does not create any warranties, representations, contractual commitments, conditions or assurances from AWS, its affiliates, suppliers or licensors. The responsibilities and liabilities of AWS to its customers are controlled by AWS agreements, and this document is not part of, nor does it modify, any agreement between AWS and its customers.