Engineering Simplicity

Implementation Guide

vSRX VIRTUAL FIREWALL-BASED AWS TRANSIT VPC

TABLE OF CONTENTS

Introduction

This implementation guide describes prerequisites for deploying the Juniper Networks® vSRX Virtual Firewall as an application instance in the AWS Elastic Compute Cloud (EC2) used as a transit virtual private cloud (VPC) function. In this deployment, a Transit VPC links spoke VPCs, enabling network connectivity between EC2 instances in different VPCs without a hardware/physical device terminating Amazon Web Services (AWS) VPN connections from Virtual Gateways (VGWs).

The Transit VPC use case is based on a public cloud-centric compute model where one or more organizations use AWS or similar cloud compute resources. Transit VPC allows data flows to stay in the cloud, minimizing limitations such as bandwidth, latency, and availability. The vSRX is the central element of the capabilities and flexibility of the Transit VPC. The connection between all cloud resources in spoke VPCs can be aggregated via the Transit VPCs to simplify and reduce overhead when connecting physical resources with cloud-based resources.

Scope

This implementation guide describes the creation of a Transit VPC inside AWS using the Amazon Machine Image (AMI) and consisting of a pair of vSRX Virtual Firewall EC2 instances. It has been written for audiences with advanced networking and AWS skills. This includes network and systems engineers who possess practical knowledge of AWS services such as VPC, EC2, CloudFormation, and S3, and are familiar with the terminology and jargon associated with these services. Additionally, qualified engineers must also possess good working knowledge of the vSRX Virtual Firewall and the Juniper Networks Junos® operating system CLI environment.

Transit VPC Use Case

The Transit VPC use case is based on a public cloud-centric compute model where one or more organizations use AWS or similar cloud compute resources, and the collection of cloud resources needs to exchange data to meet the organization's goals. Before Transit VPC, these cloud-centric organizations would need to connect their virtual private cloud infrastructures using physical hardware assets such as firewalls to enable inter-VPC data flow. The Transit VPC allows data flow to stay in the cloud, minimizing limitations such as bandwidth, latency, and availability, which typically occur when introducing non-cloud-based resources to the data flow.

The vSRX inside the Transit VPC is the data flow hub between the other VPCs, which are spoke VPCs. These spoke VPCs provide the typical VPC functions: private address space, elastic public address space, network access control (NAC) using security groups, and Virtual Gateways (VGWs) to build IPsec VPN connections. The primary characteristic of this Transit VPC use case is that spoke VPC VGW VPN connections are established with the vSRX Virtual Firewall instances in the Transit VPC.

In addition to spoke VPC-to-spoke VPC data flow, this guide also describes how to tie the Transit VPC topology to a physical data center, where the vSRX in the Transit VPC uses standard generic routing encapsulation (GRE) tunnels to make the data center connections.

Figure 1 depicts the relationship between the Transit VPC, spoke VPCs, and the data center.

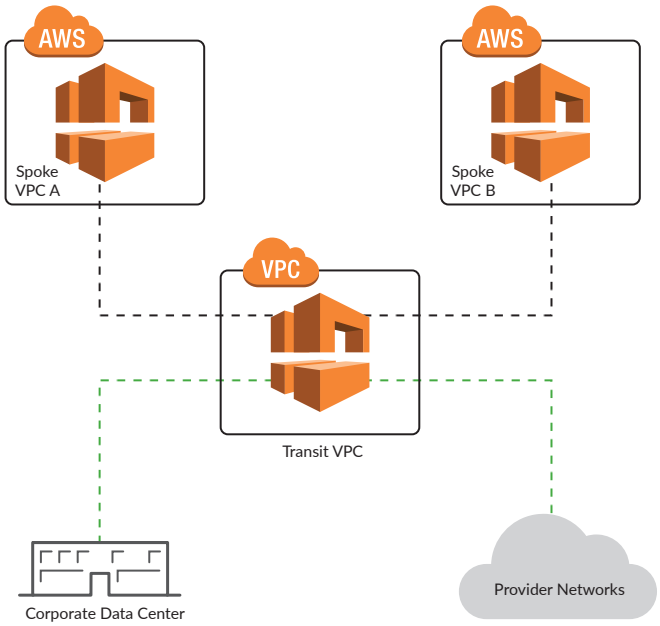

Figure 1: Transit VPC overview

Transit VPC Overview

To understand Transit VPC, it's important to first understand the general VPC concept.

"Virtual Private Cloud" is an AWS offering that enables compute and storage resources to be consumed as a service from a publically available pool of resources in a virtually private manner—that is, while the resources are part of a public cloud, they are private in the sense that the subscriber of the service controls access to the resources. The benefits of this type of service are scalability, availability, and flexibility.

Transit VPC is a specialized way of adding capability and flexibility to VPCs. The Transit VPC interconnects other VPCs, acting as the hub for data flow between spoke VPCs and potentially other on-premise customer resources.

Transit VPCs are different than other VPCs because they contain the Juniper Networks vSRX Virtual Firewall. The AMI is available within the AWS Marketplace as a "Bring Your Own License" (BYOL) option or as a bundled annual or hourly usage license. Transit VPC can be deployed in any region and availability zone that supports cloud formation and Lambda functions. Transit VPCs can be shared between AWS accounts, and connecting spoke VPCs to the Transit VPC is automated. VPC VGW tags trigger Lambda automation to configure the vSRX and spoke VPC VPN connections. Once the base vSRX elements are set up, the complexity of building this cloud-centric network are automated, including the management of dynamic BGP routing, organizing VPN connection policies, interface IP addressing, and inter-zone firewall security policies—all integrated and automated by the Juniper Networks Transit VPC stack.

The vSRX is the central element of the capabilities and flexibility of the Transit VPC. The connection between all cloud resources in spoke VPCs can be aggregated via the Transit VPCs to simplify and reduce overhead when connecting physical resources with cloud-based resources.

Transit VPC Architecture

For the purposes of this guide, a Transit VPC architecture consists of four components:

1. Transit VPC
2. vSRX Virtual Firewall
3. Spoke VPC
4. Physical data center

Transit VPC

The Transit VPC is like any other VPC, the key differences being that it doesn't need a VGW, only an Internet gateway (IGW). It needs two subnets, one for the vSRX management interface and another for a VPN-connection termination interface. Transit VPC must be deployed in an availability zone that supports CloudFormation, Lambda functions, S3, and KMS. Three Lambda functions are critical to the Transit VPC solution: Helper, Configurator, and Poller.

Security group policy must allow vSRX management interfaces to be accessible from AWS Lambda, while vSRX VPN connection interfaces must be accessible from spoke VPC VGWs. Transit VPCs are associated with a VGW tag (Name:Value pair). Spoke VPC VGW with the specified tag will be automatically connected to the Transit VPC.

vSRX Virtual Firewall

vSRX is a fully functional virtual appliance deployed as an EC2 instance inside the Transit VPC. The vSRX uses interface fxp0 as its management interface and xe-0/0/0 through xe-0/0/7 for VPN connections and inter VPC data flow.

The CloudFormation template deploys two vSRX firewalls, each in different availability zones. Each vSRX uses two Elastic Network Interfaces (ENIs), one for its management interface (fxp0) and the other for its VPN-connection interface. Once CloudFormation has deployed the Transit VPC stack, vSRX management can be performed like any other Junos OS device, using SSH and interacting with the Junos OS CLI, although no manual configuration is necessary as the Transit VPC is a fully automated deployment.

Spoke VPC

The spoke VPC is typical of any VPC use case. Classless interdomain routing (CIDR) is defined and subnets within the CIDR are assigned. EC2 resources deployed in the spoke VPC are connected and associated with subnets. Spoke VPCs can have an IGW for external/public access, while spoke VPCs must have a VGW. Spoke VGWs intended to connect to a Transit VPC must have a tag configured to signal participation/inclusion with a Transit VPC. The Transit VPC VGW tag is specified in the CloudFormation stack deployment workflow.

Spoke VPC VGW VPN connections use dynamic protocol BGP, which will share spoke VPC subnet information with the vSRX at the Transit VPC. The vSRX will manage and control reachability information with all Transit VPC member spoke VPCs.

Physical Data Center

The physical data center can represent any physical hardware network element. Its intention is to demonstrate connecting the Transit VPC environment with a physical network. While inter-VPC data flow stays in the cloud, data can still be exchanged with the physical network and resources not in the cloud. The connection between the vSRX and the physical data center can be any tunneling scheme—commonly GRE or IPsec. In this guide, we will use GRE.

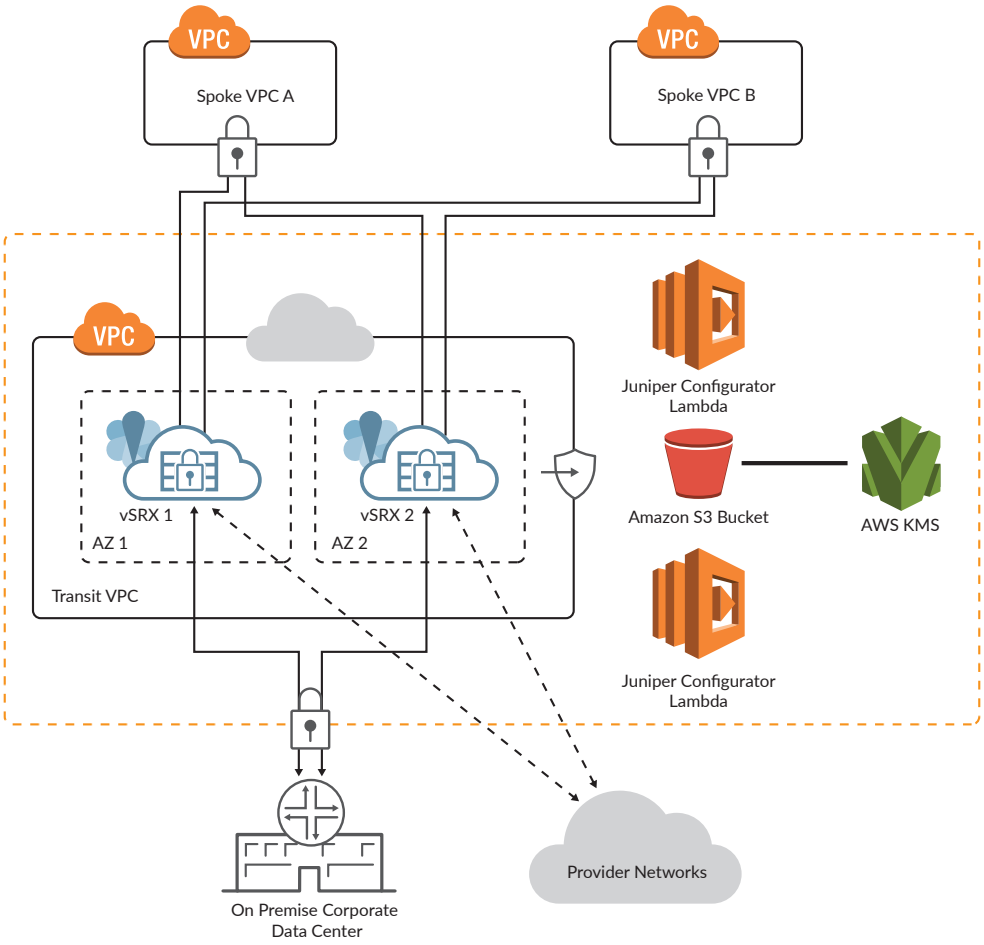

Figure 2: Transit VPC architecture.

Figure 2 represents the relationship between the four components of the Transit VPC architecture.

Connecting Spoke VPCs to Transit VPC

Once the CloudFormation template successfully creates the Transit VPC and its artifacts, the Lambda automation infrastructure is in place to connect the spoke VPC to the Transit VPC.

VGW Poller and the Juniper Configurator Lambda functions work together to discover VGWs and configure VPN connections to the vSRX.

As the Lambda Function's VGW Poller checks virtual private gateways (VGWs), it looks for the tag specified in the CloudFormation workflow and checks VGWs in the Transit VPC accounts based on the functions that trigger cron settings. The default cron setting is "every 1 minute." If the Poller finds a VGW with the Transit VPCs tag "name:value" pair, the Lambda Function Configurator defines the VPN connection and builds and commits the vSRX configuration.

The Lambda Poller function creates files in the S3 bucket specified during CloudFormation to signal to the configurator what to configure. All parameters and configurations associated with a VPN connection are saved in the S3 bucket.

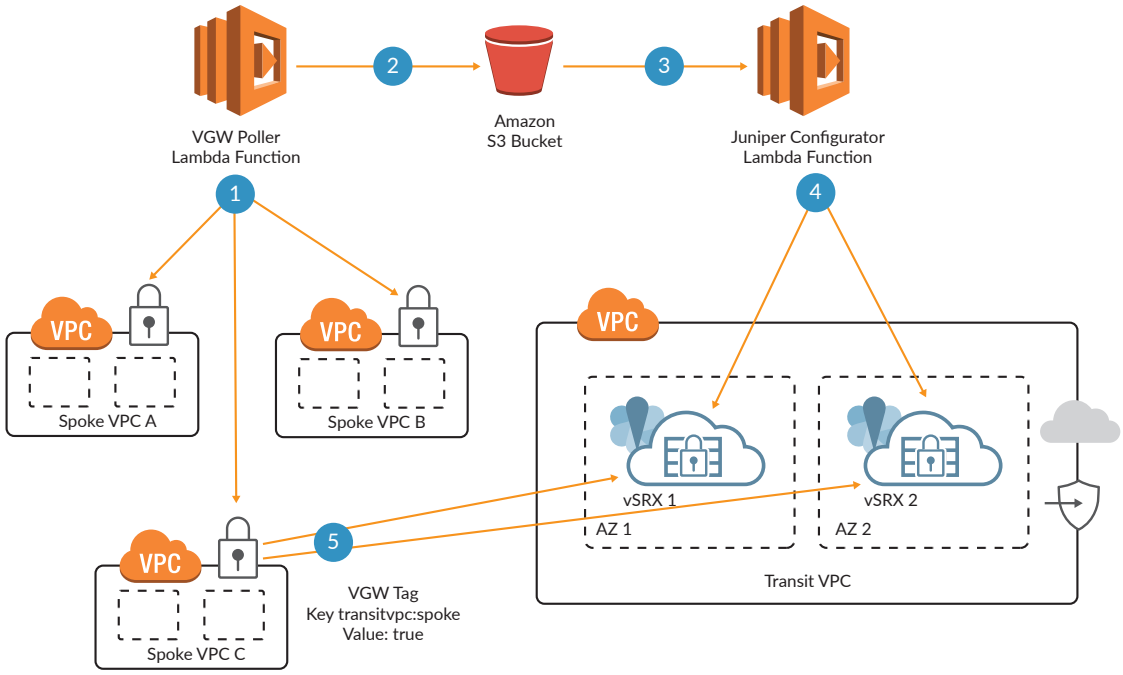

Figure 3: Connecting spoke VPCs to the transit VPC.

Figure 3 shows the VGW Poller and Configurator Lambda function interaction to automatically configure IPsec, BGP, and security policies on the VGW and vSRX.

Automation Logic When Adding a Spoke VPC

Step 1: At regular one minute intervals, the VGW Poller Lambda function is called by a CloudWatch event, which scans each region of the customer's AWS account, specifically searching for virtual gateways of spoke VPCs tagged with the key and value defined during the CloudFormation (CF) stack creation. This Lambda function Poller checks for VGWs which are not already associated with an existing VPN connection.

- Default tag key is "transitvpc:spoke," and the default tag value is "true."
- The tags for key and value can be changed as desired when the CloudFormation template is being used to create the full stack.
- Multiple Transit VPCs can operate simultaneously, each with their own tag, key/value pair within the AWS account or, in the case of a cross-account Transit VPC, within all accounts that participate in the Transit VPC.

Step 2: Once those VGWs have been identified with the correct tags applied and without any existing VPN connections associated with them, the Lambda function creates the corresponding customer gateways (CGWs), if required, for those spoke VPCs. The corresponding VPN connections to each vSRX in the Transit VPC are also created and the Elastic IP address associated with the eth1 (ge-0/0/0) interface of the vSRX instance is targeted. The Lambda function pushes this connection information to an S3 bucket using S3 SSE-KMS keys. All data objects stored in the S3 bucket are encrypted using a solution-specific AWS KMS-managed customer master key (CMK).

Step 3: A PUT event inside AWS S3 instantiates the Juniper Configurator Lambda function and parses the VPN connection information, generating the required Junos OS configuration statements for creating new VPN connections and related security and routing policies on the vSRX instances.

Step 4: The generated Junos OS configuration is pushed to the vSRX instances over an SSH connection.

Step 5: Once the Junos OS configuration is loaded on the vSRX instances and committed, the IPsec VPN tunnels come up. The establishment of IPsec VPN tunnels enables BGP peering sessions to connect with the spoke VPC's VGW, allowing routing information to propagate between spoke and Transit VPC.

Automation Logic When Removing a Spoke VPC

Step 1: CloudWatch calls the VGW Poller Lambda function in one minute intervals. The Poller Lambda function scans each region of the customer's AWS account, searching for VGWs from which the CGWs and VPN connections are to be removed.

Step 2: The Lambda Poller here looks for VGWs that have established/configured VPN connections towards the Transit VPC's vSRX instance, but do not have the specific spoke VPC tag because either the tag has been removed or the spoke VPC tag key/value has been changed from the value that the VGW Poller Lambda function expects. Example: If the value to the key "transitvpc:spoke" has been changed from "true" to "truly," or "affirmative."

Step 3: When a spoke VGW has been identified with tags changed, or it has been deleted, the VGW Poller Lambda function deletes the VPN connections to each vSRX. It saves the current connection information and parameters to the same S3 bucket using S3 SSE-KMS and deletes the CGWs. The object data stored in the S3 bucket is encrypted using solution-specific AWS KMS-managed CMK.

Step 4: S3 PUT action initiates the Juniper Configurator Lambda function. The Configurator Lambda parses VPN connection information, generating the required Junos OS configuration statements to delete the VPN connections.

Step 5: This candidate configuration is committed by the Configurator Lambda function on the vSRX.

Step 6: The Junos OS configuration statements remove the IPsec VPN tunnels, BGP peer, and security and routing policies.

Virtual Private Gateway (VGW) Poller

VGW Poller is a Lambda function that polls all of the VGWs across the user's AWS account, searching for a VGW tag (Name:Value) as defined during the CloudFormation Stack creation.

When it finds a VGW, it creates a pair of customer gateways (CGWs) referencing the Elastic IP address of the vSRX instance's VPN-Connection interface (ge-0/0/0), one for each vSRX in the Transit VPC. It also creates two VPN connections with each CGW, for a total of four VPN connections: two to vSRX1 and two to vSRX2. Connection parameters and configuration artifacts in XML format are uploaded to the AWS S3 bucket.

Figure 4 shows the process flow of the VGW Poller Lambda function when creating the CGW and VPN connections.

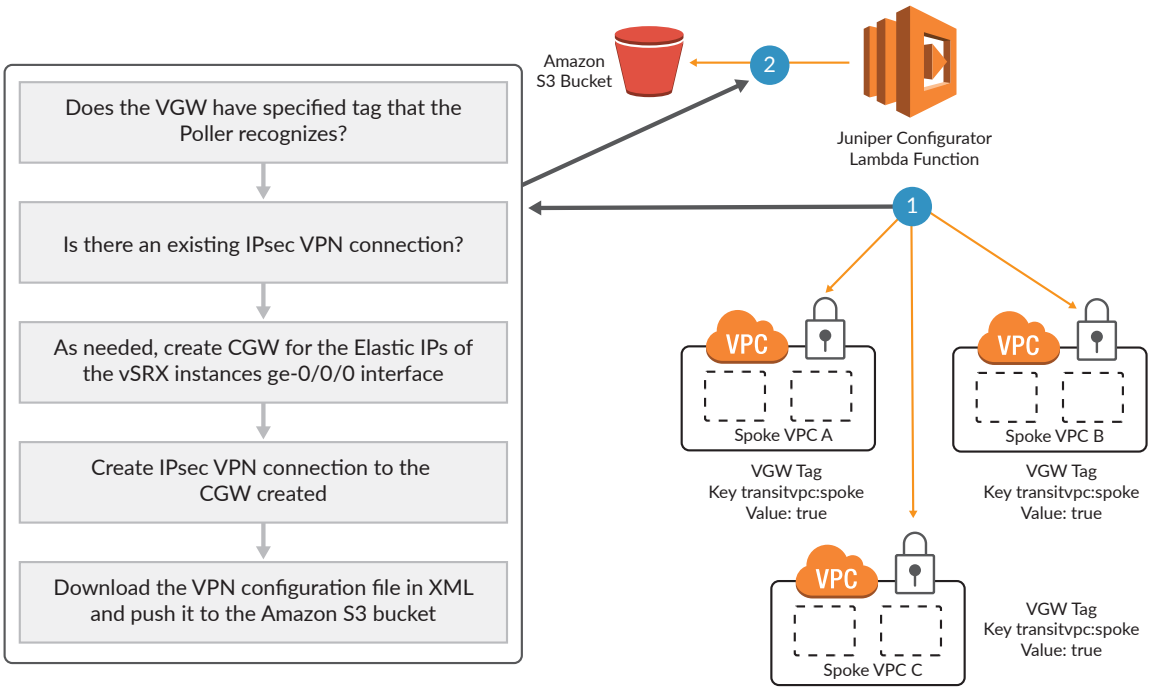

Figure 4: Component: Virtual Private Gateway (VGW) Poller.

Juniper Configurator

The Configurator Lambda function takes over where the VGW Poller stops. The S3 bucket "PUT" triggers the Configurator to read the XML configuration file that was saved by the VGW Poller, then extracts the parameters related to the IPsec VPN connections and the BGP dynamic routing.

The Configurator creates Juniper configuration in set-command format based on the extracted parameters. It logs in to the vSRX instances using the SSH key stored in the S3 bucket using the user "root." The Configurator then loads the candidate configuration and commits the changes on the vSRX.

Figure 5 shows the Configurator Lambda function process flow.

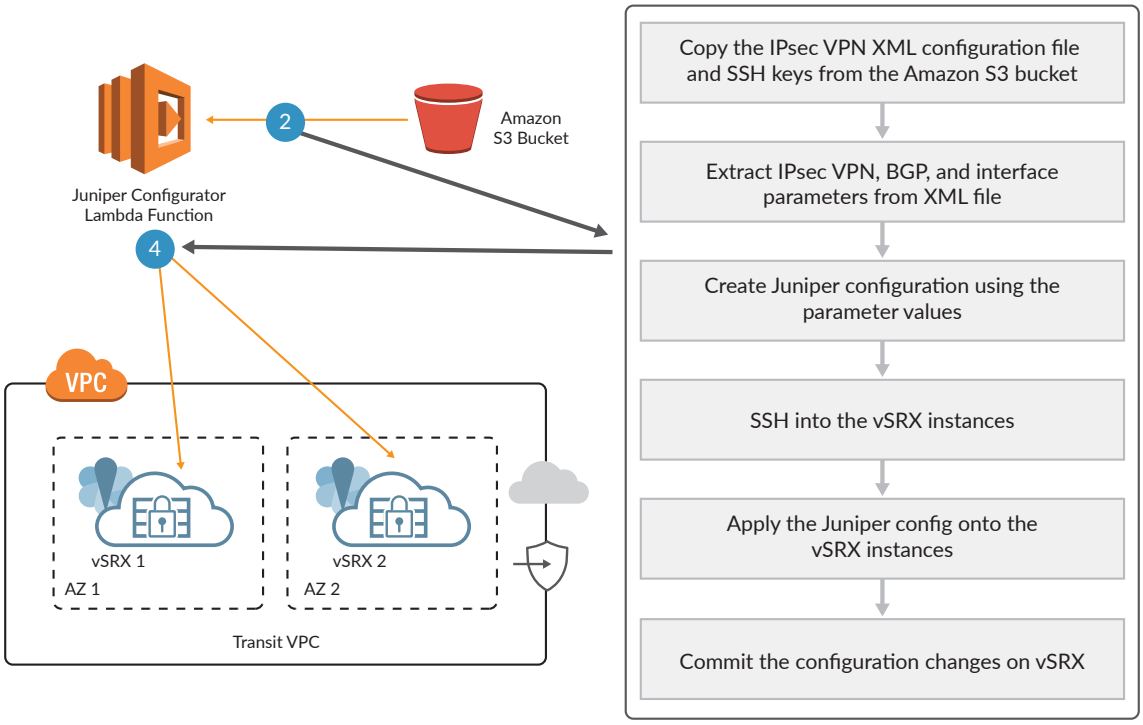

Figure 5: Component—Juniper configurator.

BGP Dynamic Routing over IPsec VPN

Border Gateway Protocol (BGP) peering sessions are established between spoke VPC VGWs and the Transit VPC vSRX. The BGP session is established over the IPsec VPN-Connection tunnels created by the VGW Poller and the Configurator Lambda functions.

Figure 6 shows the IPsec VPN connection and BGP relationship between spoke VPC and vSRX inside the Transit VPC.

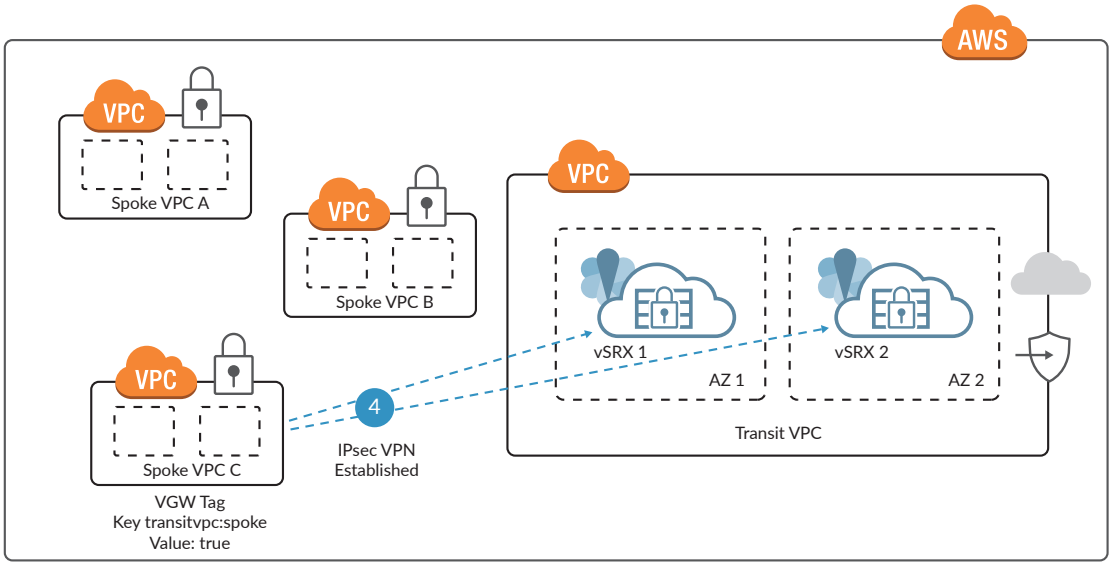

Figure 6: BGP relationship over IPsec VPN.

The vSRX instances inside the Transit VPC are deployed by the CloudFormation template as two standalone firewalls; they are not clustered, and session state is not maintained between them. BGP path selection is used to maintain symmetric data flow between spoke VPCs as data flows through the Transit VPC; the "preferred VPN endpoint" tag is used to select the vSRX path.

The preferred VPN endpoint tag key "transitvpc:preferred-path" will accept the following values, which can be set during CF stack creation or toggled afterwards:

1. Value = <blank>: Both vSRX instances are considered equal, and each spoke VPC independently chooses to send traffic through either of the vSRX instances.
2. Value = "vSRX1:" The transit VPC will configure vSRX1 as the preferred route through that particular spoke VGW. Autonomous System (AS) path prepending will make vSRX1 the chosen route over vSRX2.
3. Value = "vSRX2": The transit VPC will configure vSRX2 as the preferred route through that particular spoke VGW. AS path prepending will make vSRX2 the chosen route over vSRX1.

Transit VPC Integration with Physical Data Center

The final piece to getting the Transit VPC solution online is enabling the connectivity to the on-premise data center network. To accomplish this, GRE tunnels were configured between the vSRX instances and the on-premise router, and then BGP peering sessions were established over the GRE tunnel, which in turn will take care of dynamic routing. Figure 7 shows that the on-premise corporate data center's router is in a BGP peering relationship with the vSRX instances inside Transit VPC, encapsulated inside a GRE tunnel.

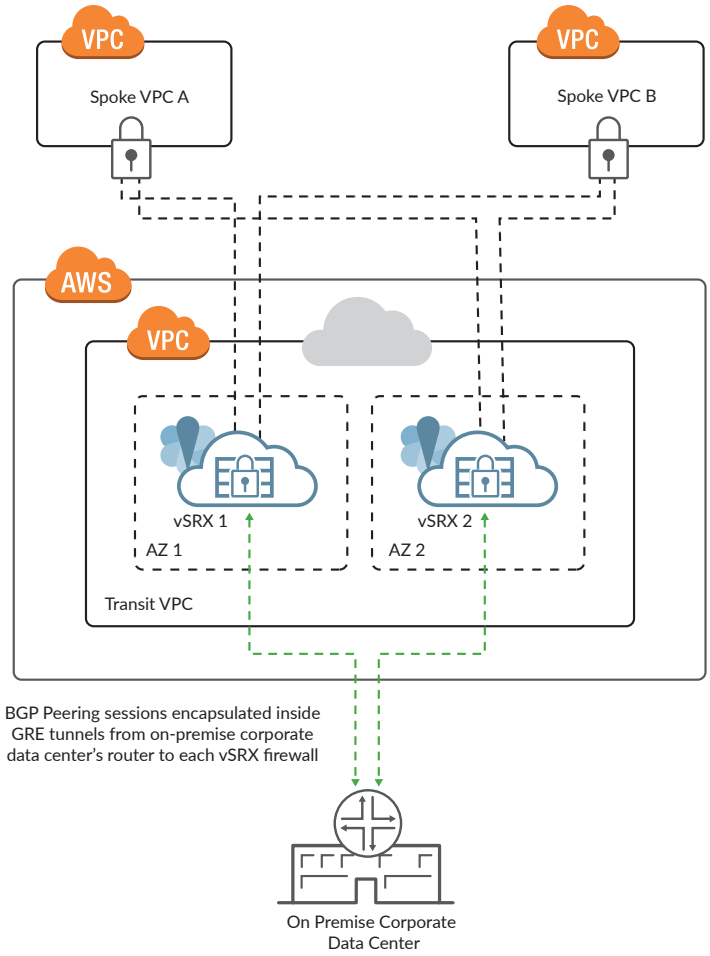

Figure 7: Integration with on-premise data center.

Preparing for Transit VPC Deployment

This section outlines the prerequisites that need to be completed before deploying the Transit VPC Stack, using the CloudFormation template.

To start, accept the terms and conditions for the vSRX AMI that will be used to spawn the vSRX instances inside the Transit VPC. Then verify the AWS service limits per service, per region, and verify IAM user privileges:

AWS Marketplace: T&Cs for vSRX AMI

The CloudFormation template has been designed to use "vSRX Next-Generation Virtual Firewall" as the base AMI to create vSRX instances inside the Transit VPC. You'll find this AMI in the AWS Marketplace. Accepting the terms and conditions for the vSRX Next-Generation Virtual Firewall AMI is a prerequisite to creating any vSRX-based instances.

Step 1: Visit the AWS Marketplace (https://aws.amazon.com/marketplace/) and search for "vSRX" in the search bar.

- Select "vSRX Next-Generation Virtual Firewall."

Step 2: Accept the terms and conditions for the vSRX Next-Generation Virtual Firewall.

- After selecting the desired product, you'll see a screen with details of the product.
- Review the Terms and Conditions and select "Accept Terms & Conditions" (if applicable).
- This process has to be done only once per account.

Note: This guide was written using version 15.1X49-D100. New versions are released regularly. It's likely that more recent versions will support all the functionalities needed for the scenario described in this implementation guide.

AWS Service Limits: Confirm and Verify

Step 1: Log in to your AWS Console using AWS Identity and Access Management (IAM) credentials.

Step 2: Visit Trusted Advisor under Support Center and check on the service limits for your account.

- The default service limits can be found at this URL: http://docs.aws.amazon.com/general/latest/gr/aws_service_limits.html

Step 3: If you must increase the service limits, create a new case under the AWS Support Center.

- Open the AWS Support Center page, sign in (if necessary), and then choose Create Case. URL: https://console.aws.amazon.com/support/home#/.
- Under Regarding, choose Service Limit Increase.
- Under Limit Type, choose the type of limit to increase, fill in the necessary fields in the form, and then choose your preferred method of contact.

Note: Service limits are defined per region and per account.

The next section dictates requirements, as per the Transit VPC artifacts list.

Transit VPC Inventory

For the purpose of creating a Transit VPC, the following service limits are required, *at a minimum:*

Transit VPC: One

- VPC CIDR: One (1)
- Subnets: Four (4)
  - Two (2) subnets in a separate availability zone each
  - Two (2) subnets per instance
- Elastic IPs: Four (4)
  - Two (2) EIPs per instance
- c4.xlarge instances: Two (2)
  - One (1) instance each in unique availability zone
- Security Groups: Three (3)
- Security Key Pair: One (1)
- Elastic Network Interfaces: Four (4)
  - Two (2) ENIs per instance
- Internet Gateway: One (1)
- Route Table: One (1)
- VPC Endpoint: One (1)
- VPC Network ACL: One (1)
- Virtual Private Gateway: 0 (not required for Transit VPC)
- Customer Gateway: 0 (not required for Transit VPC)
- VPN Connections: 0 (not required for Transit VPC)

- Lambda Functions:
  - Solutions-Helper
  - VGW Poller
  - Configurator

AWS IAM User Account Privileges

Step 1: Log in to the AWS Console as the root user, or as the IAM user with administrator level privileges.

- Navigate to Identity and Access Management (IAM) at https://console.aws.amazon.com/iam/home?#/home.

Step 2: Under IAM resources, select "Users."

Step 3: Select the user who will be deploying the CF template stack.

Step 4: Confirm that the user has power user privileges, or enough privileges to create, modify, and delete stacks using CloudFormation.

These are the permissions needed for CloudFormation:

```
{
    "Version":"2012-10-17",
    "Statement":[{
        "Effect":"Allow",
        "Action":[
            "cloudformation:DescribeStacks",
            "cloudformation:DescribeStackEvents",
            "cloudformation:DescribeStackResource",
            "cloudformation:DescribeStackResources"
        ],
        "Resource":"*"
    }]
}
```

- Reference: http://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/using-iam-template.html#w1ab2b7c16c15

Deploying Transit VPC with vSRX Virtual Firewall

This section of the implementation guide covers deploying the CloudFormation template and creating Transit VPC using vSRX instances.

Download CloudFormation Template Locally on Your Machine

Step 1: Visit GitHub (https://github.com/Juniper/vSRX-AWS) to download a modified/updated and tested version of the CloudFormation template to initiate and create a Transit VPC stack.

- CloudFormation filename: "transit-vpc-primary-account.template."

Step 2: Download the CloudFormation template to your local machine.

Step 3: Open the CloudFormation template in a text editor, in case it needs to be modified.

Create Transit VPC Stack Using the Verified CloudFormation Template

Step 1: Navigate to CloudFormation from inside the AWS Console. Click on "Create New Stack."

Step 2: Select the CF template from the local machine in "Upload a template to Amazon S3" field, under "Choose a template." Click "Next."

Step 3: Fill in the following fields on this page (where provided, default values are recommended):

- Specify Details > Stack name
  - Enter a unique name here.

Note: Add a timestamp to the CF template name, to make it identifiable. Example: <Name>-YYYYMMDD-HHMM

- Parameters > Juniper VSRX Configuration
  - VSRX Throughput Requirements (one option only: c4.xlarge)
  - SSH Key to access VSRX: here paste the content of the public key of your key pair. If you don't have a key pair, create one first (on a Macintosh, you can use the "ssh-keygen" utility). Please note that a valid public key must start with "ssh-rsa."
  - License model
    - License Included (Preferred)
    - BYOL
  - Enable Termination Protection: (User's choice)
    - Yes
    - No (okay for testing)
- Parameters > AWS Service Configuration
  - Prefix for S3 Objects: vpnconfigs/
  - Additional AWS Account ID (optional)
    - Optional only —> Enter AWS account number, if spoke VPCs are to be created inside another AWS account.
  - Lambda region:
    - Default option is us-east-1 region, with a dropdown selection menu available.
- Parameters > Network Configuration
  - Transit VPC CIDR Block: CIDR block for the Transit VPC.
  - vSRX1- First Subnet Network:
    - Subnet for fxp0 for vSRX1.
    - fxp0 is used for Management.
  - vSRX1- Second Subnet Network:
    - Subnet for ge-0/0/0 for vSRX1.
    - ge-0/0/0 is used for IKE/IPsec SA (tunnels).
  - vSRX2- First Subnet network:
    - Subnet for fxp0 for vSRX2.
    - fxp0 is used for management.
  - vSRX2- Second Subnet Network:
    - Subnet for ge-0/0/0 for vSRX2.
    - ge-0/0/0 is used for IKE/IPsec SA (tunnels).
  - Transit VPC BGP ASN: Any private Abstract Syntax Notation (ASN) should do.

  - Spoke VPC Tag Name: transitvpc:spoke.
    - This is user-defined at stack creation, and you can use any tag you'd like.
  - Spoke VPC Tag Value:
    - Define a unique value for the VGW Poller to invoke creation of CGWs and VPNs towards the vSRX firewalls in Transit VPC. Example: true

Note: Please use a unique tag when creating multiple Transit VPCs in the same account; this will help avoid VGW poller problems.

  - Preferred VPN Endpoint Tag Name (this is not currently supported and is a placeholder for future releases):
    - Normally default is sufficient to get started.
    - Can choose to make one vSRX preferred by specifying "vsrx1" or "vsrx2."
  - Parameters > Anonymous Metrics Request:
    - SendAnonymousData —>Select Yes/No based on corporate policies.
    - Click Next.

Step 4: The "Options" page is presented next:

- (Optional) May keep the default values.
- Tags—if needed, specify particular tags with "Key" and "Value."
- Permissions:
  - Choose an IAM Role.
  - Or Enter the Role ARN.
- Advanced:
  - Notification options.
  - Timeout (in minutes).
  - Rollback on Failure: Choose "Yes."
  - Stack policy.
- Click "Next" to be taken to the review page for the CF stack.
- Review the sections of Template, Details, and Options.
- Check the box next to "I acknowledge that AWS CloudFormation might create IAM resources," and click "Create."
- Next, follow the CF stack creation progress.
  - This process will take several minutes.
  - No errors are expected in the logs.
- Stack creation takes several minutes. After stack creation completes, please wait 12 to 15 minutes for the vSRX Virtual Firewalls to boot up (until 2/2 checks on the EC2 dashboard show OK).
- Once the stack creation is completed successfully, if required, log in to the vSRX instances via SSH using their Elastic IP address assigned to the eth0 ENI (fxp0 inside Junos).
  - Use the private key of the key pair that you created for your Transit VPC.
  - Note: Never tag/untag a spoke VPC while in an active vSRX SSH session. To avoid potential race conditions, only one SSH session at a time is allowed.

Adding Spoke VPCs

The following section describes the steps to define, create, and configure two spoke VPCs, as well as a test instance inside each spoke VPC.

Prepare for Spoke VPCs

For the purpose of creating the spoke VPC, at a minimum, the following service limits are required to be available for use per each spoke VPC:

- Spoke VPC: One (1)
  - VPC CIDR: One (1)
  - Subnets: Two (2)
    - Both subnets in same availability zone
    - One (1) private subnet
    - One (1) public subnet
  - Elastic IPs: One (1)
  - t2.micro instances: One (1)
  - Security Groups: Two (2)
    - One (1) for private subnet
    - One (1) for public subnet
  - Security Key Pair: One (1)
  - Elastic Network Interfaces: Two (2)
    - Two (2) ENIs per instance
  - Internet Gateway: One (1)
  - Route Table: Two (2)
    - One (1) for private subnet
    - One (1) for public subnet
  - VPC Endpoint: One (1)
  - VPC Network ACL: One (1)
  - Virtual Private Gateway: One (1)
    - Tag to be applied on VGW: As plugged in during CF stack creation
  - Customer Gateway: Two (2)
    - One (1) CGW per vSRX in Transit VPC
  - VPN Connections: Two (2)
    - Two (2) IPsec tunnels running BGP per CGW

Create Artifacts Inside Spoke VPC

Step 1: Navigate to VPC under AWS Console.

Step 2: Create a new VPC in a region of your choice; this will preferably be a region different from where the Transit VPC was created:

- For the purpose of testing, create spoke VPC in a new region.
- For production rollouts, you want to have spoke VPCs in each of the AWS regions.

Step 3: Inside the new VPC, create two subnets. Both subnets will belong to the same availability zone (one will be the private subnet, and the other will be the public subnet).

Step 4: Create an Internet gateway and attach it to the above created spoke VPC.

Step 5: Create a Virtual Private Gateway and attach it to the above created spoke VPC.

- Do not add the Tag just yet.

Step 6: Create a routing table:

- Public RTB: Associate 0.0.0.0/0 with the IGW.
- Under "Subnet Associations," select the Public Subnet that was previously created.
- Private RTB (main route table which already exists): Associate RFC1918 routes with the VGW.
- Add routes towards 10.0.0.0/8 via the VGW.
- Under "Subnet Associations," confirm it is associated with the Private Subnet that was previously created.
- Enable route propagation, for the route table to learn routes via the VGW.

Step 7: Create two security groups:

- Public:
  - Inbound: Allow SSH from 0.0.0.0/0.
  - Outbound: Allow all traffic to 0.0.0.0/0.
- Private:
  - Inbound: Allow all traffic from RFC1918 addresses.
  - Outbound: Allow all traffic to RFC1918 addresses.

Step 8: Add the tag as referenced in the CF stack for the VGW:

- Key: "transitvpc:spoke"
- Value: "true" < or the value specified during Transit VPC creation >

Step 9: Check for the creation of two CGWs, with the Elastic IP addresses assigned to the eth1 interfaces of the vSRX1 and vSRX2 instances inside your Transit VPC.

- Go to the "Customer Gateway" section of the VPC navigation pane.

Step 10: Also check for the creation of two VPNs towards the above mentioned CGWs, one VPN connection each towards each CGW.

- Go to "Virtual Private Networks" section of the VPC navigation pane.

Note: Repeat Steps 1 through 10 to create one more spoke VPC (and all other artifacts), preferably in a different region than where the first spoke VPC is located.

Security Keys Per Region

Step 1: Navigate to Services > EC2.

- In the left navigation tree, under "Network & Security," go to "Key Pairs."
- For each region in which you wish to deploy the spoke VPCs, create a key pair.
- Click on "Create Key Pair."

Step 2: Give a unique name to each key pair; download it locally and store it in a safe place

Launch Test Instance Inside Spoke VPC

Step 1: From the AWS Console, go to Services > EC2.

Step 2: Launch a new Amazon Linux t2.micro instance in the spoke VPC you just created above. This instance should have two network interface cards associated with it, with each NIC being on a separate subnet:

- eth0 should be on public subnet of spoke VPC.

- eth1 should be on private subnet of spoke VPC.
- Disk space: 20 GB SSD

Step 3: Give a name to the instance in the pattern of "SpokeVPC<x>-Instance<n>".

Step 4: Place the instance in the Public Security Group that was created for the spoke VPC.

Step 5: Use the already created key pair for the region where the spoke VPC is located.

Step 6: Click "Launch" and wait for AWS Console to finish spawning the instance.

Step 7: Associate an Elastic IP address with eth0 of the new instance.

- Copy the Interface ID for eth0 of your instance to your clipboard.
- Go to Elastic IP section under "Network & Security."
- Allocate a new address; choose VPC.
- Associate the Elastic IP with the Network ID in your clipboard buffer.

Step 8: Log in to the instance via SSH on user "Ubuntu."

- ssh -i <file location> ec2-user<Public-IP>

Step 9: Add a route to the private subnet of the other spoke VPC that you have created via the instance's private interface.

- Repeat steps 1 through 9 to launch another test instance in the other spoke VPC.
- Confirm that you can ping the private IP address of the other test instance (which is located in the other spoke VPC) from the test instance in one of the spoke VPCs.
- Confirm that from the test instance in one of the spoke VPCs, you can ping the private IP address of the other test instance (which is located in the other spoke VPC).

Connecting Transit VPC to the Data Center

The following section describes the steps to connect the Transit VPC to customers' on-premise data center for using the vSRX instances as the hub for both spoke VPC to on-premise data center, and also on-premise data center to spoke VPC communications.

Create GRE Tunnels Between vSRX and On-Premise Router: vSRX Side

Step 1: Create a GRE tunnel from vSRX instances towards on-premise data center. Log in to Junos OS CLI for vSRX1, and enter configuration mode.

Step 2: Add the following inside the configuration mode:

```
top edit interfaces gr-0/0/0.0
set tunnel source <IP-Address-of-ge-0/0/0>
set tunnel destination <Destination-IP-Address-on-DC-FW>
set tunnel routing-instance destination AWS
set family inet address 192.168.200.1/30

top edit routing-instances DC
set instance-type virtual-router
set interface gr-0/0/0.0
set routing-options static route 0.0.0.0/0 next-table transit.inet.0
set routing-options static route <Destination-IP-Address-on-DC-FW>/32 next-table AWS.
inet.0
```

```
top edit security zones security-zone trust
set tcp-rst
set host-inbound-traffic system-services lsping
set host-inbound-traffic system-services all
set host-inbound-traffic protocols all
set interfaces gr-0/0/0.0
delete interfaces ge-0/0/1.0
```

- Commit check
- Commit confirmed 5
- Commit

Step 3: Repeat steps 1 through 2 for the vSRX2 instance.

Create GRE Tunnels Between vSRX and On-Premise Router: On-Premise Router Side

To create two GRE tunnels from on-premise router towards vSRX1 and vSRX2:

Step 1: Log in to Junos OS CLI for the on-premise router, and enter configuration mode.

Step 2: Add the following inside the configuration mode:

```
top edit interfaces gr-0/0/0.0
set tunnel source <IP-assigned-to-Egress-Interface>
set tunnel destination <Public-IP-of-vSRX1-ge-0/0/0>
set family inet address 192.168.200.2/30

top edit security zones security-zone trust
set host-inbound-traffic system-services all
set host-inbound-traffic protocols all
set interfaces gr-0/0/0.0

top edit routing-options static
set route <subnet> next-hop gr-0/0/0.0
top edit interfaces gr-0/0/1.0
set tunnel source <IP-assigned-to-Egress-Interface>
set tunnel destination <Public-IP-of-vSRX2-ge-0/0/0>
set family inet address 192.168.2.200/24

top edit security zones security-zone trust
set host-inbound-traffic system-services all
set host-inbound-traffic protocols all
set interfaces gr-0/0/1.0

top edit routing-options static
set route <subnet> next-hop gr-0/0/1.0
```

- commit check
- commit confirmed 5
- commit

Create BGP Peering Sessions Using the GRE Tunnel Created Above

To create BGP peering session over the above created GRE tunnel, from vSRX instances towards on-premise data center:

Step 1: Log in to Junos OS CLI for vSRX1, and enter configuration mode.

Step 2: Add the following inside the configuration mode:

```
edit routing-instances DC
set protocols bgp group OnPrem-DC-Router-Remote type internal
set protocols bgp group OnPrem-DC-Router-Remote neighbor 192.168.200.2 local-address
192.168.200.1
set protocols bgp group OnPrem-DC-Router-Remote neighbor 192.168.200.2 hold-time 30
set protocols bgp group OnPrem-DC-Router-Remote neighbor 192.168.200.2 export EXPORT-
DEFAULT
set protocols bgp group OnPrem-DC-Router-Remote neighbor 192.168.200.2 peer-as 64512
set protocols bgp group OnPrem-DC-Router-Remote neighbor 192.168.200.2 local-as 64512
```

- Commit check
- Commit confirmed 5
- Commit

On-Premise Router Side

To create two BGP peering sessions from On-Premise Router towards vSRX1 and vSRX2:

Step 1: Log in to Junos OS CLI for the on-premise router, and enter configuration mode.

Step 2: Add the following inside the configuration mode:

```
set protocols bgp group vSRX1-Remote neighbor 192.168.200.1 local-address
192.168.200.2
set protocols bgp group vSRX1-Remote neighbor 192.168.200.1 hold-time 30
set protocols bgp group vSRX1-Remote neighbor 192.168.200.1 export CUSTOMER-1_Export
set protocols bgp group vSRX1-Remote neighbor 192.168.200.1 peer-as 64512
set protocols bgp group vSRX1-Remote neighbor 192.168.200.1 local-as 64512
set protocols bgp group vSRX2-Remote neighbor 192.168.200.65 local-address
192.168.200.66
set protocols bgp group vSRX2-Remote neighbor 192.168.200.65 hold-time 30
set protocols bgp group vSRX2-Remote neighbor 192.168.200.65 export CUSTOMER-1_Export
set protocols bgp group vSRX2-Remote neighbor 192.168.200.65 peer-as 64512
set protocols bgp group vSRX2-Remote neighbor 192.168.200.65 local-as 64512
```

- Commit check
- Commit confirmed 5
- Commit

About Juniper Networks

Juniper Networks brings simplicity to networking with products, solutions and services that connect the world. Through engineering innovation, we remove the constraints and complexities of networking in the cloud era to solve the toughest challenges our customers and partners face daily. At Juniper Networks, we believe that the network is a resource for sharing knowledge and human advancement that changes the world. We are committed to imagining groundbreaking ways to deliver automated, scalable and secure networks to move at the speed of business.

Corporate and Sales Headquarters
Juniper Networks, Inc.
1133 Innovation Way
Sunnyvale, CA 94089 USA
Phone: 888.JUNIPER (888.586.4737)
or +1.408.745.2000
Fax: +1.408.745.2100
www.juniper.net

APAC and EMEA Headquarters
Juniper Networks International B.V.
Boeing Avenue 240
1119 PZ Schiphol-Rijk
Amsterdam, The Netherlands
Phone: +31.0.207.125.700
Fax: +31.0.207.125.701

JUNIPER NETWORKS | Engineering Simplicity