Cisco CSR 1000v Deployment Guide for Microsoft Azure

Last Modified: 2019-07-31

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

CONTENTS

PREFACE

CHAPTER 1

CHAPTER 2

CHAPTER 3

Preface

This section contains the following topics:

Objectives

This document provides an overview of the Cisco CSR 1000V Series Router deployment on Microsoft Azure. It is not intended as a comprehensive guide to all of the software features that can be run using the Cisco CSR 1000V Series router. For more information, see the Cisco CSR 1000V Series Cloud Services Router Software Configuration Guide.

For information on general software features that are also available on the Cisco CSR 1000V Series router, see the CSR 1000v Series Configuration Guides .

Revision History

The Revision History records technical changes to this document. The table shows the Cisco IOS XE software release number, the date of the change, and a brief summary of the change

Release	Date	Change Summary
Cisco IOS XE Release 16 and Cisco IOS XE 3.16	June 8, 2016	First release.

Document Conventions

This documentation uses the following conventions:

Convention	Description
^ or **Ctrl**	The ^ and **Ctrl** symbols represent the Control key. For example, the key combination **^D** or **Ctrl-D** means hold down the **Control** key while you press the **D** key. Keys are indicated in capital letters but are not case sensitive.
string	A string is a nonquoted set of characters shown in italics. For example, when setting an SNMP *community* string to public, do not use quotation marks around the string or the string will include the quotation marks.

Command syntax descriptions use the following conventions:

Convention	Description
bold	Bold text indicates commands and keywords that you enter exactly as shown.
italics	Italic text indicates arguments for which you supply values.
[x]	Square brackets enclose an optional element (keyword or argument).
|	A vertical line indicates a choice within an optional or required set of keywords or arguments.
[x | y]	Square brackets enclosing keywords or arguments separated by a vertical line indicate an optional choice.
{x | y}	Braces enclosing keywords or arguments separated by a vertical line indicate a required choice.
[x {y | z}]	Braces and a vertical line within square brackets indicate a required choice within an optional element.

Examples use the following conventions:

Convention	Description
`screen`	Examples of information displayed on the screen are set in Courier font.
`bold screen`	Examples of text that you must enter are set in Courier bold font.
< >	Angle brackets enclose text that is not printed to the screen, such as passwords.
!	An exclamation point at the beginning of a line indicates a comment line. (Exclamation points are also displayed by the Cisco IOS XE software for certain processes.)
[ ]	Square brackets enclose default responses to system prompts.

The following conventions are used to attract the attention of the reader:

Note Means *reader take note* . Notes contain helpful suggestions or references to materials that may not be contained in this manual.

⚠

Caution Means *reader be careful* . In this situation, you might do something that could result in equipment damage or loss of data.

Related Documentation

For related documentation, see Cisco CSR 1000v Documentation in the Documentation Roadmap for Cisco CSR 1000v Series, Cisco IOS XE 16 .

Obtaining Documentation and Submitting a Service Request

For information on obtaining documentation, submitting a service request, and gathering additional information, see the monthly *What's New in Cisco Product Documentation* , which also lists all new and revised Cisco technical documentation, at:

http://www.cisco.com/c/en/us/td/docs/general/whatsnew/whatsnew.html

Subscribe to the *What's New in Cisco Product Documentation* as a Really Simple Syndication (RSS) feed and set content to be delivered directly to your desktop using a reader application. The RSS feeds are a free service and Cisco currently supports RSS version 2.0.

CHAPTER 1

Information About Deploying Cisco CSR 1000v on Microsoft Azure

Overview of Cisco CSR 1000v on Microsoft Azure

The Cisco Cloud Services Router (CSR) 1000v is a full-featured Cisco IOS XE router, enabling IT departments to deploy enterprise-class networking services in the Microsoft Azure cloud. Most Cisco IOS XE features are also available on the virtual Cisco CSR 1000v.

You can choose to deploy Cisco CSR 1000v software on new or existing infrastructure, such as a virtual network.

The following VPN features are supported on the Cisco CSR 1000v: IPsec, DMVPN, FlexVPN, Easy VPN and SSLVPN. You can use dynamic routing protocols such as EIGRP, OSPF, and BGP to construct multi-tier architectures within Azure, and interconnect with corporate locations or other clouds.

You can secure, inspect, and audit hybrid cloud network traffic with application-aware Zone Based Firewall. You can also use IP SLA and Application Visibility and Control (AVC) to find out about performance issues, fingerprint application flows and export detailed flow data for real-time analysis and network forensics.

Prerequisites for Deploying Cisco CSR 1000v on Microsoft Azure

These are the main three prerequisites for deploying a Cisco CSR 1000v:

- You must have a user account/subscription with Microsoft Azure. For more information about creating an account with Microsoft Azure, see Get started with Azure.
- A number of resources must be deployed before, or during, the deployment of the Cisco CSR 1000v. For a description of the required resources, see Microsoft Azure Resources, on page 2.

- A software license must be obtained for the Cisco CSR 1000v.

Microsoft Azure Resources

To deploy a Cisco CSR 1000V instance on Microsoft Azure, the following resources are required. You must create the required resources during the deployment if they do not already exist in the Azure network.

- Resource group: The container for resources. Resources include virtual machines, interfaces, virtual networks, routing tables, public IP addresses, security groups and storage accounts.

  Note You must deploy a Cisco CSR 1000V with a Single Interface within an existing resource group only. The resource group can already contain other resources.

  If you create an object in a resource group that depends upon an object in a second resource group, the second resource group cannot be deleted until you delete your object in the first resource group. Create a new resource group for a new deployment. For more information about resource groups, see: Azure Resource Manager overview.

- Virtual network: A Cisco CSR 1000V with a 2-, 4-, or 8- Network Interface Cards (NICs). Requires a virtual network with a set of defined subnets. A Cisco CSR 1000V instance with a single interface requires a new or an existing virtual network with 1 subnet. For more information about virtual networks, see Azure Virtual Network.

- Route table: The user defined routes (UDRs) for subnetworks.

- Security group: The security rules for the virtual network.

- Public IP address: The IP address of the Cisco CSR 1000V instance.

- Storage account: The storage account for the Cisco CSR 1000V image, VM disk files, and boot diagnostics. The storage account type "`Standard_LRS`" is the only currently supported type. For more details about creating a storage account, see: About Azure storage accounts.

- Boot Diagnostics: The diagnostics used for debugging issues found during the operation of the Cisco CSR 1000V instance.

- Availability Set: A logical group of VMs. The VMs are separate and can run across multiple servers, racks, and switches in a data center. For more information on availability sets, see Information about Availability Sets, on page 5, in this document. Also search for "Availability Set" in the Microsoft Azure Documentation.

- Managed Disks: Provision to manage the storage accounts of VM disks. When you create a managed disk, specify the disk type (Premium or Standard) and the size of disk that you require. Azure Storage Service Encryption (SSE) is used by default for all the managed disks. For more information on managed disks, see Azure Managed Disks Overview.

- Interfaces: The network interfaces for a Cisco CSR 1000V VM with 2, 4, or 8 network interfaces. You can assign a public IP address to any interface. (Commonly, the public IP address is assigned to the first interface). All the Cisco CSR 1000V VM interfaces are in a private subnet. You can assign the IP address of each private interface using the **ip address dhcp** command in the interface configuration or assign a static IP address using the **ip address** command. For example, `ip address 1.1.1.1 255.255.255.0`. If you use a static IP address, ensure that the IP address is the same as the IP address assigned by Microsoft Azure. View the IP address of an interface by looking at the VM network settings in the Azure marketplace.

Cisco CSR 1000v Deployments in the Microsoft Azure Marketplace

Cisco has published a deployment solution templates in the Microsoft Azure marketplace to help create and manage resources. The following types of solution templates are supported:

- Full solution template - using this template, you can deploy a Cisco CSR 1000V with 2-, 4-, or 8- NICs, with other required resources.
- CSR 1000V-only template - using this template, you can deploy a Cisco CSR 1000V with a single interface, with pre-existing resources.

If you are deploying a Cisco CSR 1000V instance in a new network with no existing resources, it is recommended that you use a full solution template.

When you deploy a Cisco CSR 1000V instance with 2-, 4-, or 8- NICs solution template, many resources are automatically created. To know how to deploy the instance in this scenario, see Deploy a CSR 1000v with Multiple Interfaces on Microsoft Azure, on page 8.

To deploy a Cisco CSR 1000V instance and use the resources that already exist in Microsoft Azure, deploy the instance using a single interface template. For more information, see Deploy a CSR 1000v with a Single Interface on Microsoft Azure, on page 12. After you deploy a Cisco CSR 1000V instance with a single interface, you can manually add further interfaces using Powershell or by using Azure CLI commands.

Cisco CSR 1000v Public Cloud Deployments

The following 2, 4 and 8 NIC solution templates are currently offered in the Microsoft Azure marketplace in the public cloud:

Cisco IOS XE Release	Supported Instance Types/Max NICs supported
16.12.x, 17.1x, and 17.2.x releases	DS2_ v2/D2_v2 (2 NICs) DS3_v2/D3_v2 (4 NICs) DS4_v2/D4_v2 (8 NICs)
17.3.x release	DS2_ v2/D2_v2 (2 NICs) DS3_v2/D3_v2 (4 NICs) DS4_v2/D4_v2 (8 NICs) F16s_v2 (4 NICs) F32s_v2 (8NICs)

Cisco CSR1000V Government Cloud Deployments

The following 2, 4, and 8 NIC solution templates are currently offered in the Microsoft Azure marketplace in the government cloud:

Cisco CSR 1000V - XE 16.x with 2, 4 or 8 NICs

Cisco IOS XE releases 16.4, 16.5, 16.6, and 16.7 are supported.

Cisco CSR1000V Licensing

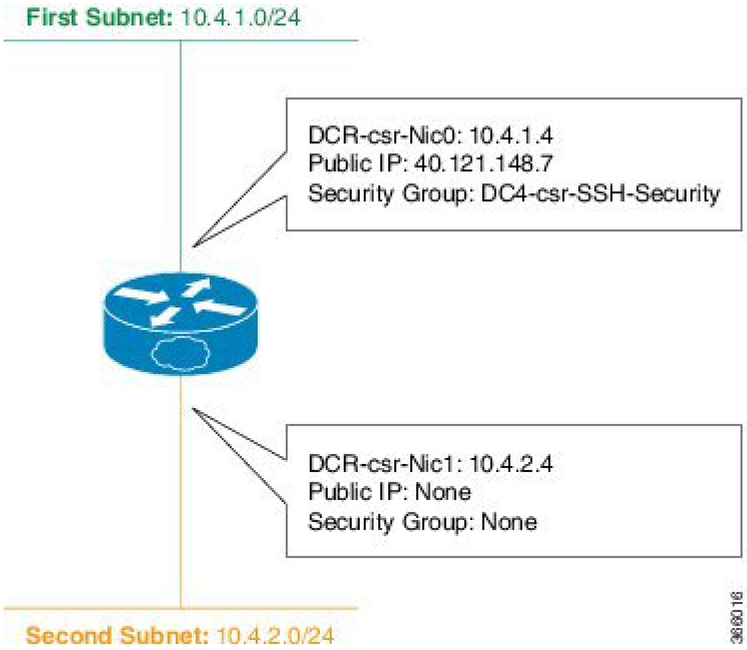
For a Cisco CSR1000V using the BYOL licensing model on Microsoft Azure, you must either have a conventional license or a smart license. The license determines the combinations of throughput level and technology packages that you can use.

For more information about obtaining a license, see Licensing for a Cisco CSR 1000v on Microsoft Azure, on page 6

Cisco CSR 1000v with 2 Network Interfaces—Example

This example shows the configuration that results after deploying a 2 network interface solution template from the Azure Marketplace.

A Cisco CSR 1000v virtual machine (2 vCPU, 7G RAM) is set up with 2 interfaces. There is a public IP address attached to the interface on the first subnet (NIC0). The first subnet (NIC0) has a security group with inbound rules for the interface. A default routing table is set up on the Microsoft Azure hypervisor router for the Cisco CSR 1000v. Note that a Cisco CSR 1000v can be deployed on a new or existing virtual network.



Subnetting Limits

The Cisco CSR 1000v on Microsoft Azure supports a subnet mask between /8 and /29 (CIDR definition).

The subnet /29 is the smallest available in Microsoft Azure, which supports 8 IP host addresses. 4 IP host addresses per subnet are reserved by Microsoft Azure. Therefore, for a /29 subnet, you have 4 IP host addresses available.

Information about Availability Sets

If you are deploying a Cisco CSR 1000v using a solution template for 2, 4 or 8 network interfaces from the Azure Marketplace, and choose to use the availability set feature, you must use a new availability set.

Availability sets are only available in solution templates for the public cloud (not for solution templates in the government cloud).

For more information, see Azure Managed Disks Overview.

Availability Sets for a Cisco CSR 1000v with 2, 4 or 8 Network Interfaces

The logical grouping of VM resources in an availability set helps to keep groups of VMs isolated from one another. The VMs in an availability set can run across multiple physical servers, compute racks, storage units, and network switches. If you use availability sets, and then a hardware or Microsoft Azure software failure occurs, only a subset of your VMs are affected. You must use a new availability set, if you are deploying a Cisco CSR 1000v using a solution template for 2, 4 or 8 network interfaces . An availability set is only available for Cisco CSR 1000v public cloud deployments. (An availability set is not available for Cisco CSR 1000v government cloud deployments.)

When you choose to use an availability set and you are deploying a Cisco CSR 1000v with 2, 4 or 8 network interfaces using a solution template, you are asked to enter the following parameters:

- **Availability Set Name**—name of the new availability set. You cannot use the name of an existing availability set.
- **Platform Fault Domain Count** —count of the fault domains. VMs that are in the same fault domain share common storage as well as a common power source and network switch. Value: 1 or 2 (2 is the default).
- **Platform Update Domain Count**—count of the update domains, which are a group of VMs and underlying physical hardware that can be rebooted simultaneously. Value: 1 to 20 (20 is the default).

Availability Sets for a Cisco CSR 1000v with a Single Interface

To use an existing availability set, you must deploy a Cisco CSR 1000v with a Single Interface.

Frequently Asked Questions About Cisco CSR 1000v Deployments on Microsoft Azure

1. When I search for CSR in Azure Marketplace, I am presented with a list of CSR 1000v solution templates/deployments. Which one should I pick?

The best practices for deciding whether to pick a solution template (for 2-, 4- or 8- NICs) or to pick an individual CSR 1000v, are as follows:

If you are creating a new virtual network, use one of the solution templates (for 2-, 4- or 8- NICs). This saves you the time and effort of manually creating all the resources.

If any of the following conditions are true, use an individual Cisco CSR 1000v (for example, **Cisco CSR 1000V Bring Your Own License - XE 16.7**) :

- You have an existing resource group which does not contain a Cisco CSR 1000v and you want to deploy a Cisco CSR 1000v in the resource group.
- You have an existing resource group which already contains a Cisco CSR 1000v and you want to deploy another one in the same availability set.

2. I want to create multiple CSR 1000v's in my subscription and I want them all to be deployed in a single availability set. How can I do this?

Perform the following steps:

1. Deploy the first Cisco CSR 1000v using a 2, 4, 8 NIC solution template; for example, **Cisco CSR 1000v – XE 16.6 Deployment with 2 NICs**. Create a new availability set for this Cisco CSR 1000v.
2. Deploy an individual Cisco CSR 1000v; for example, **Cisco CSR 1000V Bring Your Own License - XE 16.7**. Select the same availability set that you created in step 1. Using this "Bring Your Own License" individual Cisco CSR 1000v allows you to reuse existing resources in existing non-empty resource groups.
3. Repeat step 2 for all of the remaining Cisco CSR 1000v's.

Licensing for a Cisco CSR 1000v on Microsoft Azure

The Cisco CSR 1000v supports the following license model:

Bring Your Own License Model

The Bring Your Own License (BYOL) licensing model, for the Cisco CSR 1000v on Microsoft Azure, supports the following two types of license:

- Cisco Software License (CSL)—uses a traditional Product Authorization Key (PAK) licensing model. For further information on using a PAK, see Cisco Software Licensing (CSL).
- Cisco Smart Licensing—assigns a license to Cisco CSR1000v instances dynamically. This allows you to manage licenses across different CSR1000v instances without having to lock each license to a specific CSR1000v UDI serial number. For more information on Cisco Smart Licensing, see Smart Licensing.

Note In addition to paying for a Cisco CSR 1000v license, you will also need to pay for a Microsoft VM instance.

CHAPTER 2

How to Deploy a Cisco CSR 1000v on Microsoft Azure

Customize the Microsoft Azure Portal

You can customize the Microsoft Azure portal GUI by adding frequently used objects, such as virtual machines or virtual network to the left-hand side panel.

Note You only need to perform these optional steps if you are going to deploy a Cisco CSR 1000v using a single interface, where the resources need to be added manually. You do not need to create these resources manually, if you are going to deploy a Cisco CSR 1000v with 2, 4 or 8 interfaces using a solution template.

Before you begin

To customize the portal, you must have a Microsoft Azure subscription.

SUMMARY STEPS

1. Sign in to Microsoft Azure.
2. Click **Browse** and select an object to be added to the left hand side panel.
3. In the drop-down menu, click the star symbol for your chosen object.

DETAILED STEPS

	Command or Action	Purpose
Step 1	Sign in to Microsoft Azure.	

	Command or Action	Purpose
Step 2	Click **Browse** and select an object to be added to the left hand side panel.	
Step 3	In the drop-down menu, click the star symbol for your chosen object.	The details of this object are saved for future use. Repeat steps 2 and 3 to add a series of objects to the left-hand side panel.

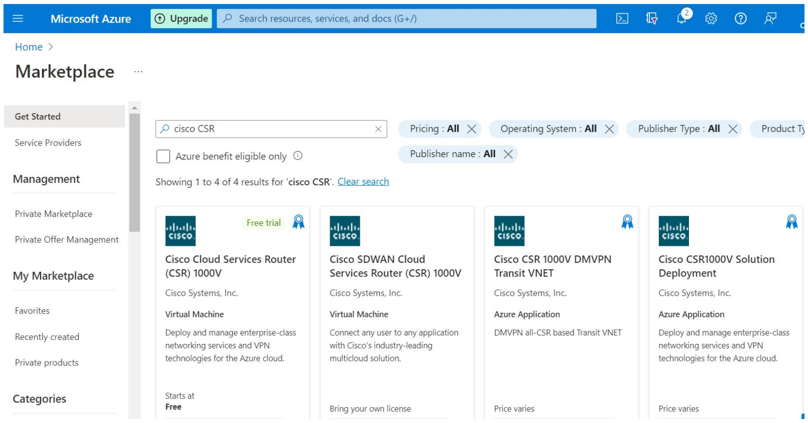
Deploy a CSR 1000v with Multiple Interfaces on Microsoft Azure

Perform the following steps to deploy a Cisco CSR1000V instance with multiple interfaces on Microsoft Azure.

Step 1 Log in to the Microsoft Azure Marketplace.

Step 2 On the Search bar, search for `Cisco CSR`.

Step 3 The system displays the various offerings under Cisco CSR1000V. Select **Cisco CSR1000V Solution Deployment**, and click **Create**.



cisco CSR

Showing 1 to 4 of 4 results for 'cisco CSR'. Clear search

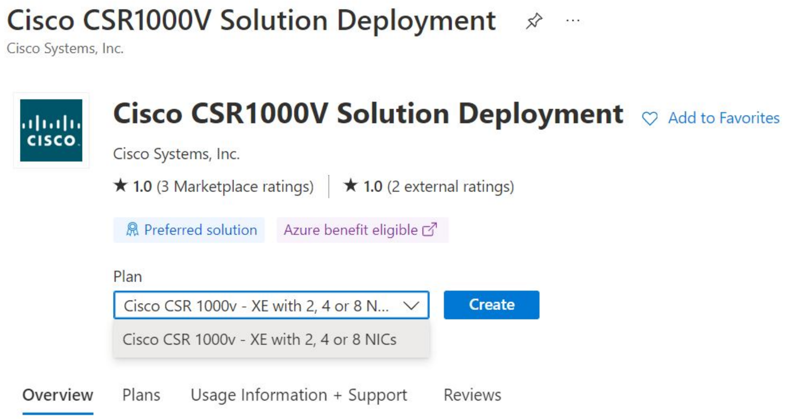
Cisco Cloud Services Router (CSR) 1000V — Cisco Systems, Inc. — Virtual Machine — Deploy and manage enterprise-class networking services and VPN technologies for the Azure cloud. — Starts at Free

Cisco SDWAN Cloud Services Router (CSR) 1000V — Cisco Systems, Inc. — Virtual Machine — Connect any user to any application with Cisco's industry-leading multicloud solution. — Bring your own license

Cisco CSR 1000V DMVPN Transit VNET — Cisco Systems, Inc. — Azure Application — DMVPN all-CSR based Transit VNET — Price varies

Cisco CSR1000V Solution Deployment — Cisco Systems, Inc. — Azure Application — Deploy and manage enterprise-class networking services and VPN technologies for the Azure cloud. — Price varies

Step 4 In the Cisco CSR1000V Solution Deployment page, the **Cisco CSR 1000v - XE with 2, 4 or 8 NICs** solution is available in the **Plan** drop-down field. Click **Create**.


Cisco CSR1000V Solution Deployment
Cisco Systems, Inc.
Cisco CSR1000V Solution Deployment
Add to Favorites
Cisco Systems, Inc.
★ 1.0 (3 Marketplace ratings)
★ 1.0 (2 external ratings)
Preferred solution
Azure benefit eligible
Create
Overview
Plans
Usage Information + Support
Reviews

Plan: Cisco CSR 1000v - XE with 2, 4 or 8 N...

Cisco CSR 1000v - XE with 2, 4 or 8 NICs

Step 5 In the Basics page, enter the following details:

a) **Subscription Name**: The name of your subscription. A default subscription name is available. You can modify the subscription name, if required.

b) **Resource Group**: A container that holds the resources for your solution. From this drop-down field, choose either **Create New** or **Select Existing**. You can create a Cisco CSR 1000V instance only in a new Resource Group or in a completely empty existing resource group. To remove a Resource Group, first delete the Cisco CSR 1000V VM and then delete the Resource Group.

c) **Region**: The region or location where you are performing this deployment. From this drop-down field choose your region.

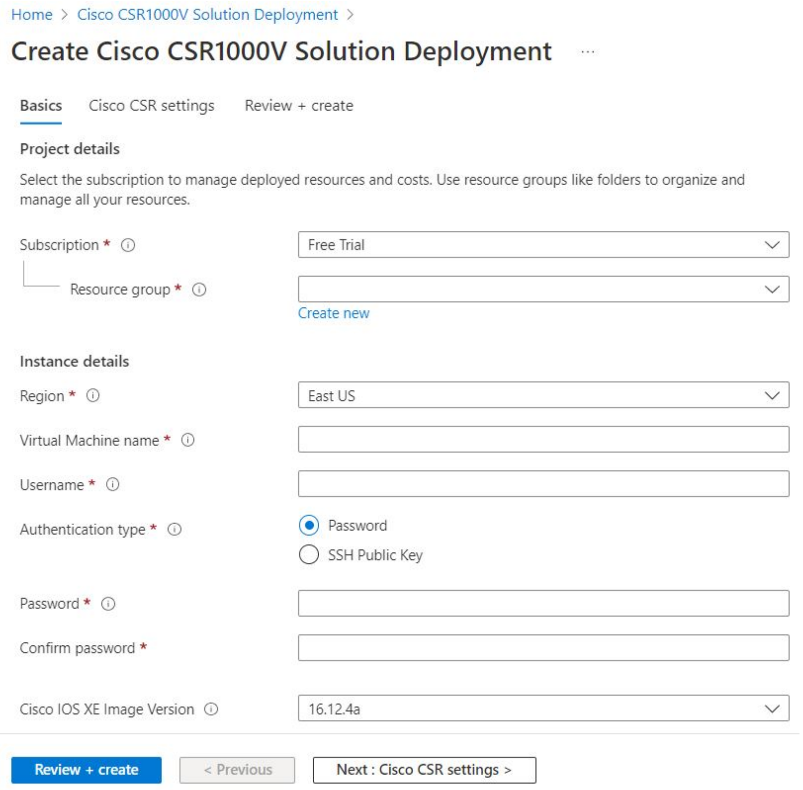
d) **Virtual Machine Name**: The name of the cloud-based network used by Microsoft Azure to represent a private network. Enter a name for the virtual machine.

e) **Username**: The username for your VM using which you can log in to the Cisco CSR 1000V instance. Enter a user name for your VM.

Note For Cisco IOS XE versions 3.16 and 16.4, if you're planning to choose SSH Key as an authentication type, enter the **Username** as **`azureuser`**.

f) **Authentication type**: The authentication type for the administrator account. You can use a username and password or an SSH key for authentication. If you select the **SSH Key** option, select the **SSH Public Key Source** and provide the **Key Pair Name**. If you select the **Password** opition, enter a password for authentication.

g) **Cisco IOS XE Image Version**: The version of the Cisco IOS XE software. From this drop-down field, choose your Cisco IOS XE version.



Home > Cisco CSR1000V Solution Deployment >

Create Cisco CSR1000V Solution Deployment ...

Basics | Cisco CSR settings | Review + create

Project details

Select the subscription to manage deployed resources and costs. Use resource groups like folders to organize and manage all your resources.

Subscription * ⓘ: Free Trial

Resource group * ⓘ:

Create new

Instance details

Region * ⓘ: East US

Virtual Machine name * ⓘ:

Username * ⓘ:

Authentication type * ⓘ: ● Password ○ SSH Public Key

Password * ⓘ:

Confirm password *:

Cisco IOS XE Image Version ⓘ: 16.12.4a

Step 6 Click **Next** and proceed to the Cisco CSR Settings page.

Step 7 In the **Cisco CSR Settings** page, enter the following details:

a) **Number of Network Interfaces in CSR**: The number of network interfaces you want to attach to the VM. From the drop-down list, choose the number of interfaces: 2, 4, or 8.

b) **License Type**: The license type. From this drop-down field, choose either **BYOL** or **PAYG** as the license type.

c) **Managed Disk**: The option that allows you to specify whether you want Azure to manage the disk for you. Select **Enabled**.

d) **Virtual machine size**: The size of the VM to provision. Select the appropriate virtual machine size. Based on the number of interfaces that you are using, select the appropriate virtual machine size. Microsoft Azure supports different image types with different performance expectations.

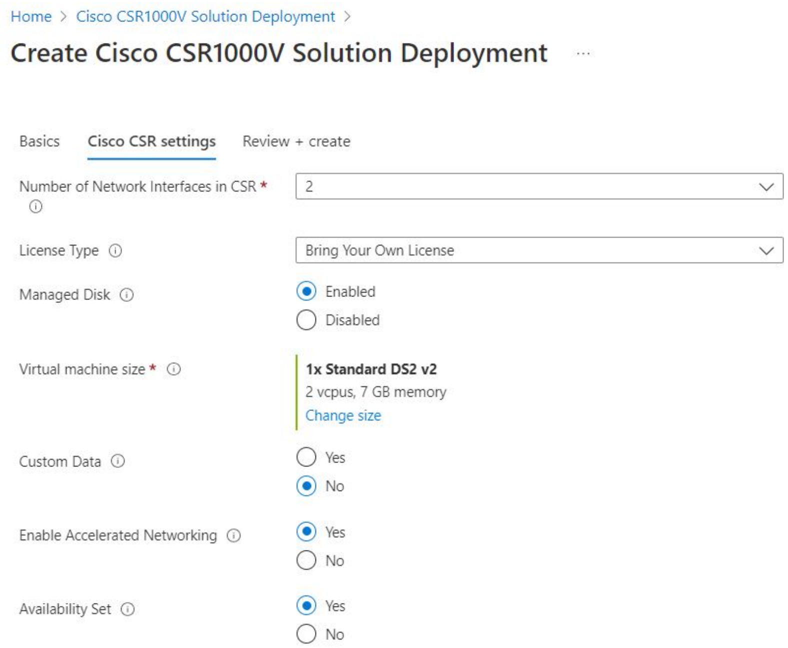
To view the supported instance types and the virtual machine sizes, see the following links:

- Dv2 and DSv2 series
- Fsv2 series

e) **Custom Data**: The provisioning configuration information for your VM. Select **Yes** if you want to provide a bootstrap configuration file for your Cisco CSR 1000V instance. For further information about providing a bootstrap configuration file for the Cisco CSR 1000V instance, see: Deploying a Cisco CSR 1000v VM on Microsoft Azure using a Day 0 Bootstrap File and customdata-examples.

f) **Enable Accelerated Networking**: The option to enable single root I/O virtualization (SR-IOV) to your VM. Select **Yes** to enable the accelerated networking feature.

g) **Availability Set**: The logical grouping of resources to create an availability set. Select **Yes** to create a new availability set.


Home > Cisco CSR1000V Solution Deployment >

Create Cisco CSR1000V Solution Deployment ...

Number of Network Interfaces in CSR * ⓘ: 2

License Type ⓘ: Bring Your Own License

Managed Disk ⓘ: ● Enabled ○ Disabled

Virtual machine size * ⓘ: 1x Standard DS2 v2
2 vcpus, 7 GB memory
Change size

Custom Data ⓘ: ○ Yes ● No

Enable Accelerated Networking ⓘ: ● Yes ○ No

Availability Set ⓘ: ● Yes ○ No

h) **Availability Set name**: The name of your availability set. Enter a name for your availability set.

i) **Availability Set Fault Domain Count**: The group of VMs that share a common power source and network switch. Availability sets arrange virtual machines across fault domains. In the field, enter the availability set fault domain count.

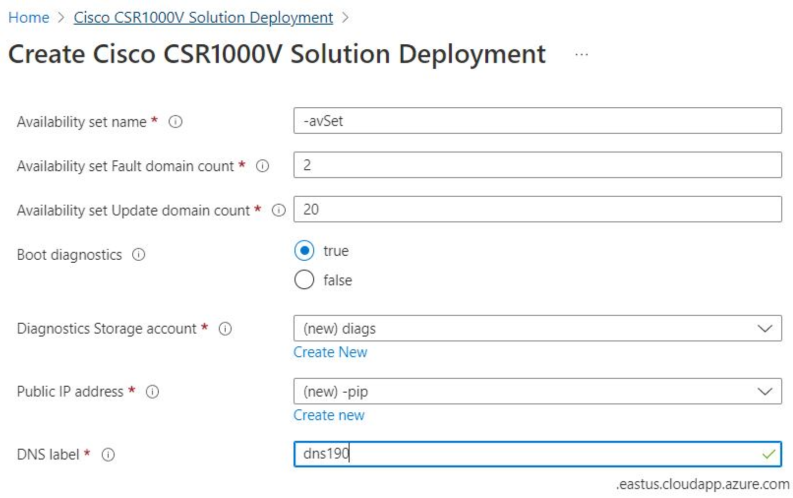
j) **Availability Set Update Domain Count**: A group of VMs and underlying physical hardware that can be rebooted at the same time. Enter the availability set update domain count.

k) **Boot diagnostics**: The option that enables you to capture the boot logs and screenshots of the virtual machine. Select **True** to enable boot diagnostics. For more information on boot diagnostics, see Microsoft Azure Resources, on page 2.

l) **Diagnostics Storage account**: The storage account for the boot diagnostics. Enter the storage account name. For more information on storage accounts, see Microsoft Azure Resources, on page 2.

m) **Public IP Address**: The public IP address name. For more information on the public IP address, see Microsoft Azure Resources, on page 2.

n) **DNS label**: The name of the public IP address to be assigned to the Cisco CSR 1000V instance. A default value for the DNS label is shown in the text box which is the VM name followed by "`-dns`". Change the name of the DNS label, if required.



Home > Cisco CSR1000V Solution Deployment >

Create Cisco CSR1000V Solution Deployment ...

Availability set name * : -avSet

Availability set Fault domain count * : 2

Availability set Update domain count * : 20

Boot diagnostics: ● true ○ false

Diagnostics Storage account * : (new) diags

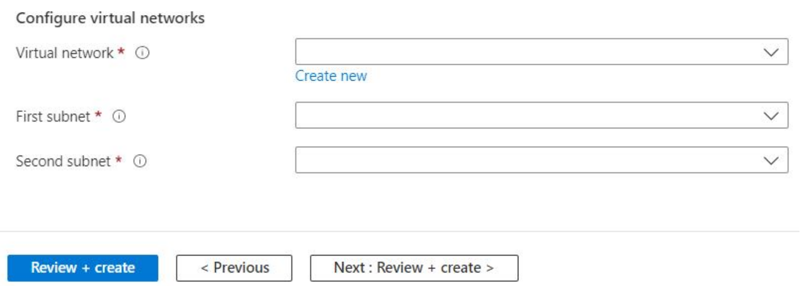
Public IP address * : (new) -pip

DNS label * : dns190 .eastus.cloudapp.azure.com

Step 8 In the Configure Virtual Networks section, specify the following details:

a) **Virtual network**: From the drop-down field, choose either **Create New** or **Use existing**. For a new virtual network, enter the name and IP address.

b) **First Subnet**/**Second Subnet**: The name and the IP address for your subnets. For more information on subnets, see "Interfaces" in Microsoft Azure Resources, on page 2.



Configure virtual networks

Virtual network *

First subnet *

Second subnet *

Step 9 Click **Next: Review + Create**.

Step 10 The system displays the summary of all your settings. Review your settings and then click **Next**.

Step 11 Click **Create**

The VM is created and the purchase is confirmed.

Step 12 To verify the successful creation of your VM, click **Virtual machines** in the left hand panel. After a few minutes, the status of the recently created VM changes from “Creating” to “Running”. Make a note of the Public IP address name.

Deploy a CSR 1000v with a Single Interface on Microsoft Azure

Perform the following steps to deploy a Cisco CSR 1000v with a single interface, on Microsoft Azure.

Note If you are deploying a Cisco CSR 1000v with a 2-, 4- or 8- NICs solution template, the following steps are not required. Instead, go to the Microsoft Azure portal and determine the public IP address of the Cisco CSR 1000v. Then, **ssh** into the Cisco CSR 1000v as described in Access the Cisco CSR 1000v CLI, on page 13.

Procedure

	Command or Action	Purpose
Step 1	Select **Virtual machines** in the left hand side panel.	
Step 2	Click **Add**.	
Step 3	Enter `csr`. A search starts, to find any Cisco CSR 1000v VM deployments in the Azure Marketplace.	
Step 4	Choose a deployment.	Example: **Cisco CSR 1000v Bring Your Own License - XE 16.7**.
Step 5	Click **Create**.	The **Basics** sub-menu is highlighted.
Step 6	**Name**—Enter the name of the virtual network.	The virtual network is a cloud-based network used by Microsoft Azure to represent the private network.
Step 7	**VM disk type**—Select a VM disk type.	The VM disk type is either SSD or HDD.
Step 8	**User name**	Username for the Cisco CSR 1000v virtual machine. This is the username that you will use to log into the Cisco CSR 1000v. For Cisco IOS XE version 3.16 and Cisco IOS XE 16.4, to enter an SSH public key to access the CSR set the “`Username`” field to “`azureuser`”.
Step 9	**Authentication type**—Enter a Password (default) or SSH public key.	
Step 10	**Subscription**—Select the name of a subscription.	A default name based on the name of the virtual machine is provided. You can change the default name.
Step 11	**Resource Group**—Create a new group by selecting **Create new** or select an existing group by selecting **Use existing**.	Specifies the name of a new or existing resource group.

	Command or Action	Purpose
Step 12	Click **OK**.	The **Size** sub-menu is highlighted.
Step 13	Click **Virtual machine size**	For further information on virtual machine size, see Sizes for Windows virtual machines in Azure.
Step 14	Click **OK**.	The **Settings** sub-menu is highlighted.
Step 15	**High Availability**—Select an existing availability set or create a new availability set.	To use High Availability, select an existing availability set or create a new availability set.
Step 16	**Storage**—Enter the storage account name.	Enter the storage account name, if you are using Managed Disks to manage the storage accounts of VM disks.
Step 17	**Virtual network**—Enter the virtual network address.	Enter the address of the virtual network using Classless Inter-Domain Routing (CIDR) notation. Example: `10.4.1.0/16`
Step 18	**Subnet**——Enter the subnet IP address.	
Step 19	**Public IP address**—Enter the public IP address name.	The IP address is provided by Azure.
Step 20	**Network Security groups**—Enter the name of a network security group.	
Step 21	**Auto-shutdown**	To enable auto-shutdown, set Enable to "On". To disable auto-shutdown set Enable to "Off". For more information on auto-shutdown, search for "auto-shutdown" in the Microsoft Azure Documentation.
Step 22	(Optional) **Monitoring**—Select "Monitoring" to enable monitoring.	Enables Cisco CSR 1000v monitoring, using boot diagnostics. If you enable monitoring, you must also enter the boot diagnostics account name.
Step 23	Click **OK**.	The **4 Summary** sub-menu is highlighted. The summary details of the VM that is about to be deployed are displayed on the screen.
Step 24	Click **Create**.	The VM is created and the purchase is confirmed.
Step 25	Click **Virtual machines** on the left hand panel.	Verifies the VM status. After a few minutes, the VM status changes from “Creating” to “Running”. Make a note of the Public IP address name.

What to do next

Go to Access the Cisco CSR 1000v CLI, on page 13, which explains how to **ssh** into the Cisco CSR 1000v.

Access the Cisco CSR 1000v CLI

Access the CLI of the Cisco CSR 1000v VM via a terminal server.

Before you begin

Before you access the CLI, perform the steps in one of the preceding deployment procedures (Deploy a CSR 1000v with a Single Interface on Microsoft Azure, on page 12 or Deploy a CSR 1000v with Multiple Interfaces on Microsoft Azure, on page 8).

SUMMARY STEPS

1. Enter the **ssh** command using a command syntax from one of the two substeps below.
    - If you did not previously use an SSH public key (you did not specify a username of “azureuser”, then you can access the Cisco CSR 1000v CLI using the following command: **ssh –o ServerAliveInterval**=60 *username@csr_ip_address*
    - If you did previously use an SSH public key (you did specify a username of “azureuser”), then you can access the Cisco CSR 1000v CLI using the following command: **ssh –i***key***-o ServerAliveInterval**=60 azureuser@*csr_ip_address*

DETAILED STEPS

	Command or Action	Purpose
Step 1	Enter the **ssh** command using a command syntax from one of the two substeps below. • If you did not previously use an SSH public key (you did not specify a username of “azureuser”, then you can access the Cisco CSR 1000v CLI using the following command: **ssh –o ServerAliveInterval**=60 *username@csr_ip_address* • If you did previously use an SSH public key (you did specify a username of “azureuser”), then you can access the Cisco CSR 1000v CLI using the following command: **ssh –i***key***-o ServerAliveInterval**=60 azureuser@*csr_ip_address*	Enter the **ssh** command in a terminal server of your choice to access the CLI .

Example

In the following example, username=“azureuser”, public IP address = 40.121.148.7 and password=xxx are used as parameters in the **ssh** command, before other commands such as **show ip route**. No ssh public key was previously specified.)

```
$ ssh -o ServerAliveInterval=60  azureuser@40.121.148.7
              The authenticity of host '40.121.148.7 (40.121.148.7)' can't be established.

                RSA key fingerprint is 94:79:e9:d2:2e:85:93:d6:52:41:cc:a3:d9:14:7f:5f.
                Are you sure you want to continue connecting (yes/no)? yes
               Warning: Permanently added '40.121.148.7' (RSA) to the list of known hosts.

                Password: mypassword

# show ip int br
                Interface              IP-Address      OK? Method Status
  Protocol
                GigabitEthernet1       10.4.1.4        YES DHCP   up
  up
```

```
                GigabitEthernet2  unassigned     YES unset  administratively down     down


# configure terminal
                Enter configuration commands, one per line.  End with CNTL/Z.
                # interface g2
                # ip address dh
                # ip address dhcp
                # no shutdown
                # end
                # show run interface g2
                Building configuration...
                Current configuration : 69 bytes
                !
                interface GigabitEthernet2
                ip address dhcp
                negotiation auto
                end
                # show ip interface brief
                Interface              IP-Address      OK? Method Status
Protocol
                GigabitEthernet1       10.4.0.4        YES DHCP   up                    up
                GigabitEthernet2       10.4.1.4        YES DHCP   up                    up

                # show ip route
                <output snipped for brevity>
                Gateway of last resort is 10.4.1.1 to network 0.0.0.0

                S*    0.0.0.0/0 [1/0] via 10.4.1.1
                10.0.0.0/8 is variably subnetted, 4 subnets, 2 masks
                C        10.4.1.0/24 is directly connected, GigabitEthernet1
                L        10.4.1.4/32 is directly connected, GigabitEthernet1
                C        10.4.2.0/24 is directly connected, GigabitEthernet2
                L        10.4.2.4/32 is directly connected, GigabitEthernet2
                168.63.0.0/32 is subnetted, 1 subnets
                S        168.63.129.16 [254/0] via 10.4.1.1
```

CHAPTER 3

Running the Linux Azure Agent in the Cisco CSR 1000v

Information About the Linux Azure Agent

The primary requirement of a Linux-based virtual machine in the Azure cloud is to run the Microsoft Linux Azure Agent.

The Microsoft Azure Linux agent manages Linux provisioning and virtual machine interaction with the Azure Fabric Controller. It provides the following functionality for Linux deployments:

- Image Provisioning
- Networking
- Kernel
- Diagnostics
- System Center Virtual Machine Manager (SCVMM) Deployments
- VM Extensions

See the Microsoft Azure Linux VM Agent documentation or search for information about Microsoft Linux Azure Agent for more details.

Linux Azure Agent in the Cisco CSR 1000v

You can create a Cisco CSR 1000v in the Microsoft Azure cloud, running inside a Linux-based virtual machine. Although the Cisco CSR 1000v code runs on a Linux-based operating system, the Cisco CSR 1000v is not a generic Linux machine. Cisco IOS XE does not expose all the commands and functions of Linux to the user or to the Azure cloud.

In order for a Linux-based virtual machine to participate in various Azure management services, the virtual machine must support a minimal subset of Linux commands and functions.

Early in the initialization process for the Cisco CSR 1000v, it runs a minimalistic version of the Linux Azure Agent. This version only contains enough functionality to perform basic provisioning of the image. After completing this step, the Linux Agent loses its connectivity to the network, as ownership of all Cisco CSR 1000v interfaces are transferred to Cisco IOS XE.

This leaves the Cisco CSR 1000v in a state where it can no longer support the ongoing capabilities of the Linux Azure Agent. The Cisco CSR 1000v stops reporting its status on a periodic basis and cannot download/install any VM extensions. In order to restore this functionality, the Cisco CSR 1000v restarts the Linux Azure Agent in a guest shell container. The container provides an environment where all the Linux functions required by the agent are available.

Using the Guest Shell in the Azure Cloud

The guest shell container is the Linux host which represents the Cisco CSR 1000v virtual machine in the Azure cloud. When Azure servers and features interact with the Cisco CSR 1000v, they communicate with the Linux host in the guest shell container.

To install the Linux Azure Agent on a Cisco CSR 1000v, the guest shell container must be configured and enabled, see Information About the Guest Shell.

For Cisco IOS XE Fuji 16.8.1 and later, the guest shell container starts automatically during the initial configuration on the Cisco CSR 1000v. The Linux Azure Agent is downloaded, installed, and started.

For Cisco IOS XE Fuji 16.7 and Cisco IOS XE Everest 16.6, you must manually download and install the Linux Azure agent before starting the agent.

Manually Installing the Linux Azure Agent

Step 1 **guestshell**

Enters the guest shell from privileged mode.

Example:

```
Router# guestshell
[guestshell@guestshell ~]$
```

Example:

Step 2 **sudo pip install** csr_azure_guestshell

Download and install the package for Cisco CSR 1000v in the Azure cloud.

Example:

```
[guestshell@guestshell ~]$ sudo pip install csr_azure_guestshell
Collecting csr_azure_guestshell
/usr/lib/python2.7/site-packages/pip-8.1.2-py2.7.egg/pip/_vendor/requests/packages/urllib3/util/ssl_.py:318:

...
/usr/lib/python2.7/site-packages/pip-8.1.2-py2.7.egg/pip/_vendor/requests/packages/urllib3/util/ssl_.py:122:
```

```
...
Downloading csr_azure_guestshell-0.0.1.dev70.tar.gz (274kB)
100% |################################| 276kB 3.1MB/s
Installing collected packages: csr-azure-guestshell
  Running setup.py install for csr-azure-guestshell ... done
Successfully installed csr-azure-guestshell-1.0.0
```

Step 3 Verify that there is a new directory "azure" in directory /home/guestshell.

Example:

```
[guestshell@guestshell ~]$ ls
0_waagent.pid  azure  waagent.pid
```

Step 4 Display the two running processes.

The waagent daemon is the Linux Azure Agent.

Example:

In this example, apart from the waagent daemon, there is also a second process, which is also part of the agent and is used to download extensions to the virtual machine.

```
[guestshell@guestshell ~]$ ps -ef | grep waagent
root       110     1  0 13:32 ?        00:00:00 /usr/bin/python -u /usr/sbin/waagent -daemon
root       117   110  0 13:32 ?        00:00:00 python -u /usr/sbin/waagent -run-exthandlers
```

Step 5 Check on the status of the Linux Azure Agent (waagent.service).

Example:

```
[guestshell@guestshell]$ sudo systemctl status waagent.service
waagent.service - Azure Linux Agent
   Loaded: loaded (/usr/lib/systemd/system/waagent.service; disabled)
   Active: active (running) since Tue 2017-11-14 14:01:32 UTC; 15s ago
 Main PID: 161 (python)
   CGroup: /system.slice/libvirtd.service/system.slice/waagent.service
           ├─161 /usr/bin/python -u /usr/sbin/waagent -daemon
           └─164 python -u /usr/sbin/waagent -run-exthandlers
```

Restarting the Linux Agent and Guest Shell

Step 1 is optional and shows how to reinstall the Linux agent package if the guest shell has previously been destroyed and then re-enabled.

Step 2 shows how to reinstall the Linux agent service.

Step 1 (Optional) **sudo pip install** csr_azure_guestshell

(Optional) Perform this step if the guest shell has been destroyed and then re-enabled. Reinstalls the Linux Azure Agent package.

Step 2 **sudo systemctl start** waagent.service

Restarts the Linux Azure Agent service. The service must be restarted if the guest shell has been re-enabled after having been disabled or destroyed.

Microsoft Azure Guest Shell Package Scripts

The following script is included in the csr_azure_guestshell package under the azure directory `get-metadata.py`. The script retrieves and prints instance metadata from Microsoft Azure.

```
[guestshell@guestshell azure]$ ./get-metadata.py
{
  "compute": {
    "sku": "",
    "publisher": "",
    "name": "r167-csr1",
    "offer": "",
    "vmSize": "Standard_D2_v2",
    "vmId": "5121eb3b-6503-486e-b93b-dbae5cf12fe9",
    "platformUpdateDomain": "0",
    "platformFaultDomain": "0",
    "version": "",
    "location": "eastus",
    "osType": "Linux"
  },
  "network": {
    "interface": [
      {
        "mac": "000D3A199E46",
        "ipv4": {
          "subnet": [
            {
              "prefix": "24",
              "dnsservers": [],
              "address": "192.168.35.0"
            }
          ],
          "ipaddress": [
            {
              "publicip": "13.92.177.219",
              "ipaddress": "192.168.35.12"
            }
          ]
        },
        "ipv6": {
          "ipaddress": []
        }
      },
      {
        "mac": "000D3A1996E2",
        "ipv4": {
          "subnet": [
            {
              "prefix": "24",
              "dnsservers": [],
              "address": "192.168.36.0"
            }
          ],
          "ipaddress": [
            {
              "publicip": "",
              "ipaddress": "192.168.36.12"
            }
          ]
        },
        "ipv6": {
          "ipaddress": []
        }
      }
    ]
  }
}
Port 0
Mac is 000D3A199E46
Public ip is 13.92.177.219
Private ip is 192.168.35.12
subnet is 192.168.35.0/24
Port 1
Mac is 000D3A1996E2
Public ip is
Private ip is 192.168.36.12
subnet is 192.168.36.0/24
```

CHAPTER 4

Deploying a Cisco CSR 1000v VM on Microsoft Azure using a Day 0 Bootstrap File

Note Deploying a Cisco CSR 1000v VM using a Day 0 bootstrap file is supported on Cisco IOS XE Fuji 16.9.1 or later releases.

When you deploy a Cisco CSR 1000v VM instance on Microsoft Azure, you can optionally choose to use a "Day 0" bootstrap file to achieve a variety of automation goals. The Day 0 bootstrap file in Azure allows you to run Cisco IOS XE configuration commands, install Python packages in guestshell on Day0, run scripts in guestshell on Day0, and provide licensing information to boot CSR with a desired technology package.

To launch a CSR 1000v instance with Day 0 bootstrapping, perform the following steps:

- Editing the Day 0 Bootstrap File, on page 23
- Providing the Day 0 Bootstrap File, on page 26
- Verifying the Configuration after Uploading the Day 0 Bootstrap File, on page 27

Editing the Day 0 Bootstrap File

To edit the bootstrap file, configure these properties: IOS Configuration, Scripts, Script credentials, Python package, and Licensing. The properties can be placed in the bootstrap file in any order. Dependencies between the properties are noted in each of the following property descriptions. See the example bootstrap files at: https://github.com/csr1000v/customdata-examples.

Configuring the IOS Configuration Property

If you want to bootstrap certain IOS configuration on Day0, configure the “IOS Configuration” property. See the following example:

```
Section: IOS configuration
hostname CSR1
interface  GigabitEthernet1
description “static IP address config”
ip address 10.0.0.1 255.255.255.0
interface GigabitEthernet2
description “DHCP based IP address config”
ip address dhcp
```

After the first line that reads `Section: IOS configuration`, you can enter a list of Cisco IOS XE configuration commands to be run on the Cisco CSR 1000v router.

When you run this command, the above mentioned IOS configuration is applied to the CSR 1000v router on Day0.

Configuring the Scripts Property

Scripts property helps you to automate your deployment and achieve other automation goals. If you want to run a python or a bash script on Day0 under guestshell context, you can achieve the same by providing the public URL and arguments of the python or the bash script in Scripts property.

A script must include a piece of code that includes the shebang (!) character in the first line of the script. This line tells Cisco IOS-XE which script interpreter (Python or Bash) must be used to parse the script code. For example, the first line of a python script can contain `#!/usr/bin/env python`, while the first line of a bash script can contain `#!/bin/bash`. This line allows the Python or Bash script to run as executable code in a Linux environment.

When you execute the script, the script runs in the guestshell container of the Cisco CSR 1000v instance. To access the guestshell container, use the **guestshell** EXEC mode command. For more information on guestshell command, see the Programmability Configuration Guide.

To configure the Scripts property, follow the format given below:

```
Section: scripts
public_url <arg1> <arg2>
```

In this script, the first line of the property should read `Section: Scripts`.

In the second line of the property, enter the URL of the script and the script's arguments. The script can be either a python or a bash script. The script is run in guestshell in the first boot when the bootstrap file is uploaded when you create the CSR1000v instance.

To view more examples of the scrips, see "scripts" at: https://github.com/csr1000v/customdata-examples. Also refer to the following two examples:

Example 1

```
Section: Script
https://raw.githubusercontent.com/csr1000v/customdata-
examples/master/scripts/smartLicensingConfigurator.py --idtoken "<token_string>" --throughput
 <throughput_value>
```

The two lines in the scripts property retrieve the `smartLicensingConfigurator.py` script from the `customdata-examples` repository at the specified URL. The script runs in the guestshell container of the Cisco CSR 1000v with the arguments `idtoken` and `throughput`.

Example 2

```
Section: Scripts
ftp://10.11.0.4/dir1/dir2/script.py -a arg1 -s arg2
```

These two lines in the Scripts property retrieve the `script.py` script from the ftp server with the IP address 10.11.0.4, and runs the script with the `./script.py -a arg1 -s arg2` bash command in the guestshell container of the Cisco CSR 1000v using arguments arg1 and arg2.

Note If a script in the Scripts property requires a Python package that is not included in the standard CentOS Linux release (the CentOS Linux release that is used by the guestshell, which is currently CentOS Linux release 7.1.1503), you must include information about the Python package in the Python package property. For more information, see Configuring the Python package Property.

Prior to uploading the bootstrap file and running the bash or python script, we recommend that you test the URL that you intend to use in the Scripts property. You can test the `ftp://10.11.0.4/dir1/dir2/script.py -a arg1 -s arg2` URL by first running the curl software tool to download the script file. In the guestshell, enter the curl command, as shown in the following example:

```
curl -m 30 --retry 5 --user username:password
ftp://10.11.0.4/dir1/dir2/script_needs_credentials.py.
```

If the curl command is successful, a copy of the python script is downloaded, which verifies whether the URL is correct.

Configuring the Script credentials Property

If you have specified an FTP server in the Script property, and the server requires a username and password credentials, specify the credentials using the Script credentials property. If the FTP server can be accessed anonymously, you need not use the Script credentials property.

Configure the Scripts property with a URL and parameters that match those in the Script credentials property. To configure the Script credentials property, follow the format given below:

```
Section: Script credentials
public_url <username> <password>
```

Example 1

```
Section: Script credentials

ftp://10.11.0.4/dir1/dir2/script1.py userfoo foospass
```

The second line in the Script credentials property specifies the values of the username (`userfoo`) and password (`foospass`) credentials for the python script `script1.py`.

Include the name of the FTP server that is also in the Scripts property. An example line in the Scripts property is: `ftp://10.11.0.4/dir1/dir2/script1.py -a arg1 -s arg2`. See example 2 in the *Configuring Scripts Property* section.

Configuring the Python package Property

If a Python package is required by a script in the Scripts property and it is not part of the standard CentOS Linux release 7.1.1503, you must include information about the package in the Python package property. By including the Python package property in the bootstrap file, you ensure that the Cisco CSR 1000v downloads and installs the required Python package before running the script that you specified in the Scripts property.

To configure the Python package property, follow the format as specified below:

```
Section: Python package
package_name [ version ] [ sudo ] { [ pip_arg1 [ ..[ pip_arg9] ] ] }
```

The arguments: *version*, **sudo**, and *pip_arg1* to *pip_arg9* are optional. You must put the arguments to the pip command between “{“ and “}” braces.

If the *version* argument is specified, a specific version number is downloaded.

If the *sudo* argument is specified, the package is downloaded as a sudo user.

Example 1

In this example, the second line of the Python package property specifies that the *package_name* is "ncclient" and the *version* is "0.5.2". When the bootstrap file is uploaded, version 0.5.2 of the ncclient package is installed in the guestshell container of the Cisco CSR 1000v.

```
Section: Python package

ncclient 0.5.2
```

Example 2

```
Section: Python package

csr_azure_guestshell 1.1.2 sudo {--user}
```

In this example, the second line of the Python package property specifies that the *package_name* is "csr_azure_guestshell" and the *version* is "1.1.2". When the bootstrap file is uploaded, version 1.1.2 of the csr_azure_guestshell package is installed in the guestshell container of the Cisco CSR 1000v. The following command is executed as a sudo user: `sudo pip install csr_azure_guestshell==1.1.2 --user`.

Configuring the License property

Configure the license property to specify the license technology level for the Cisco CSR 1000v.

Enter the first line of the property: `Section: License`. Enter the second line of the property, which specifies the tech level of the license, using the following format: **TechPackage:***tech_level* .

Note There must be no spaces between "TechPackage:" and the *tech_level*. (*tech_level* values: ax, security, appx, or ipbase)

tech_level must be in lowercase.

Example 1

```
Section: License

TechPackage:security
```

Providing the Day 0 Bootstrap File

Provide the Day 0 bootstrap file, which creates a Cisco CSR 1000v VM, by performing the following Azure CLI command:

```
az vm create --name CSR-name --resource-group resource-group { [ arg1 [ ..[ arg9] ] ] }
--custom-data bootstrap-file
```

For further information on the **az vm create** command, see: https://docs.microsoft.com/en-us/cli/azure/vm?view=azure-cli-latest#az-vm-create.

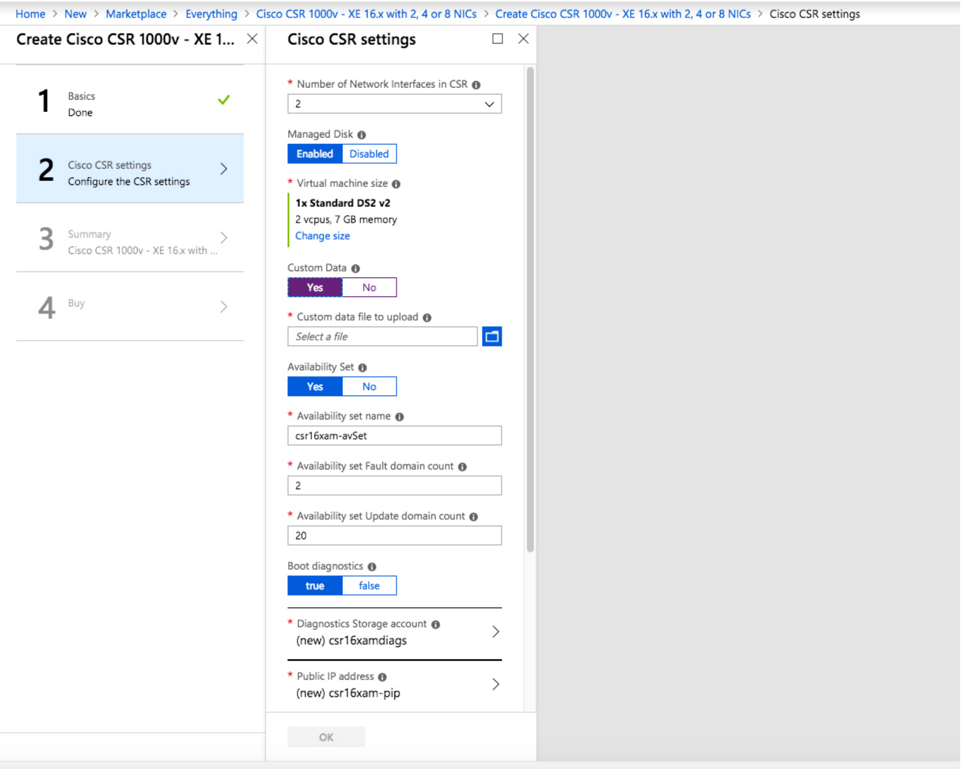
See the following example:

```
az vm create -n CSR-VM-Name -g MyResourceGroup --image
cisco:cisco-csr-1000v:16_6:16.6.120170804 --data-disk-sizes-gb 8 --availability-set myAvlSet
--nics nic1 nic2 nic3 nic4 --admin-username azureuser --admin-password "+Cisco123456"
--authentication-type password -l westus --size Standard_DS4_v2 --custom-data bootstrap.txt..
```

When you execute this command, a Cisco CSR 1000v VM is created. The router is configured using the commands in the bootstrap file: "bootstrap.txt".

If you are using a Cisco 16.x template to create a CSR 1000v instance, the custom data upload box is provided as shown in the following image:

Figure 1: Uploading Day0 Bootstrap File



Home > New > Marketplace > Everything > Cisco CSR 1000v - XE 16.x with 2, 4 or 8 NICs > Create Cisco CSR 1000v - XE 16.x with 2, 4 or 8 NICs > Cisco CSR settings

Create Cisco CSR 1000v - XE 1...

1 Basics
Done ✓

2 Cisco CSR settings
Configure the CSR settings

3 Summary
Cisco CSR 1000v - XE 16.x with ...

4 Buy

Cisco CSR settings

* Number of Network Interfaces in CSR: 2

Managed Disk: Enabled | Disabled

* Virtual machine size: 1x Standard DS2 v2, 2 vcpus, 7 GB memory, Change size

Custom Data: Yes | No

* Custom data file to upload: Select a file

Availability Set: Yes | No

* Availability set name: csr16xam-avSet

* Availability set Fault domain count: 2

* Availability set Update domain count: 20

Boot diagnostics: true | false

* Diagnostics Storage account: (new) csr16xamdiags

* Public IP address: (new) csr16xam-pip

Use the **Cisco CSR Settings** option to provide the custom data bootstrap config file.

For further information on managing Linux VMs, see: Tutorial: Create and Manage Linux VMs with the Azure CLI 2.0.

Verifying the Configuration after Uploading the Day 0 Bootstrap File

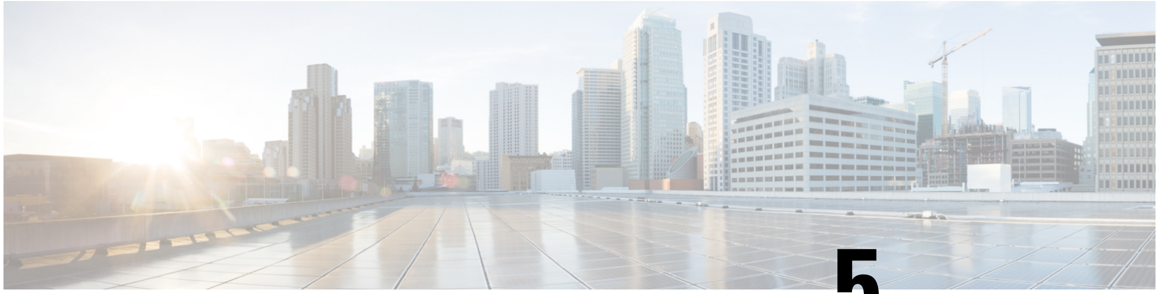
After the Day 0 bootstrap file is uploaded, the VM is created and configuration commands are executed. Perform the following commands to verify the configuration commands of each property.

To help determine if the license property worked, in Cisco IOS XE CLI on the CSR 1000v, enter the **show version** command. For example, you should see a reference to the security license.

To see if errors occurred after running commands in the scripts property, look at the customdata.log file in the /home/guestshell/customdata directory. The *scriptname*.log file stores any output sent to STDOUT by the script.

To check if the Python property worked, enter the **pip freeze | grep***package-name* command to view the currently installed python packages, searching for the package *package-name* in which you are interested.

To check the Cisco IOS XE commands were successful in the IOS Configuration property, enter the **show running-configuration** command.



CHAPTER 5

Configuring CSR1000v on Microsoft Azure

The following chapter tells you how to configure your CSR1000v instance for Microsoft Azure.

Note The Microsoft Azure serial console is supported only for Cisco CSR 1000v routers running on 16.9.1s release and later. If you want to use the serial console, upgrade the CSR 1000v to a 16.19.1s release or later

To configure your CSR1000v instance, see the following sections:

Update Route Tables

In Microsoft Azure, all VMs send packets to a hypervisor router, and the hypervisor forwards the packets based on the routing table associated with that subnet.

When a Cisco CSR 1000v VM is created, a route table is created for each subnet. For a 2 vNIC Cisco CSR 1000v VM, a default route is created for a second (internally facing) subnet that points to the CSR. All the VMs created on this subnet use the Cisco CSR 1000v as the default gateway. For Cisco CSR 1000v VMs that have more than two vNICs, you need to define the default routes and apply them to the subnets.

SUMMARY STEPS

1. Click **Route tables**
2. Navigate to the "Route tables" pane and select the target route table.
3. Click **All Settings**
4. In the Settings pane, click **Routes**

DETAILED STEPS

	Command or Action	Purpose
Step 1	Click **Route tables**	Expands the Settings pane.
Step 2	Navigate to the "Route tables" pane and select the target route table.	
Step 3	Click **All Settings**	
Step 4	In the Settings pane, click **Routes**	Add or modify routes.

Update Security Group

A Security Group controls which ports/destinations the hypervisor allows/denies for certain interfaces. When creating a Cisco CSR 1000v, a new Security Group is created for the first subnet inbound interface by default. For Cisco CSR1000v virtual machines, deployed through this deployment, the following ports are added for inbound internet traffic: TCP 22, UDP 500 and UDP 4500. Use of other ports is denied.

SUMMARY STEPS

1. Click Network security groups on the left hand side panel.
2. Click the target network security group.
3. Click **All Settings**.
4. Click Inbound security rules.
5. Click **Add** (under "Network security groups") to add additional rules.

DETAILED STEPS

	Command or Action	Purpose
Step 1	Click Network security groups on the left hand side panel.	The Network security groups pane appears, and shows a list of security groups.
Step 2	Click the target network security group.	A pane appears that shows the details of the security group.
Step 3	Click **All Settings**.	The Settings pane appears.
Step 4	Click Inbound security rules.	
Step 5	Click **Add** (under "Network security groups") to add additional rules.	

Configuring IPsec VPN for a Cisco CSR 1000v on Microsoft Azure

This example shows an IPsec VPN configured for Cisco CSR 1000v on Microsoft Azure.

```
crypto isakmp policy 1
 encr aes
 hash sha256
 authentication pre-share
 group 14
crypto isakmp key cisco123 address 0.0.0.0
crypto ipsec transform-set T1 esp-3des esp-md5-hmac
 mode transport
crypto ipsec profile P1
 set transform-set T1
interface Tunnel0
 ip address 3.3.3.1 255.255.255.0
 tunnel source GigabitEthernet1
 tunnel mode ipsec ipv4
 tunnel destination 104.45.154.184
 tunnel protection ipsec profile P1
end
!!!! To test, create loop back interface and static route!!!!!
interface Loopback1
 ip address 5.5.5.5 255.255.255.255
end
ip route 6.6.6.6 255.255.255.255 Tunnel0
```

Upgrading a Cisco IOS XE Image on Microsoft Azure

This procedure shows how to upgrade the image on a running CSR 1000v (Cisco IOS XE Fuji 16.7.1 and later).

Before you begin

To upgrade a Cisco CSR 1000v image for a Cisco CSR 1000v running in Microsoft Azure, the current version of Cisco IOS XE running on the Cisco CSR 1000v must be Cisco IOS XE Fuji 16.7.1 or later.

Note (Cisco IOS XE Everest 16.6 and earlier) On Microsoft Azure, you cannot use the Cisco CSR 1000v .bin file to upgrade a Cisco CSR1000v instance. You must re-deploy a new instance from the Microsoft Azure Portal and migrate your configuration and licenses.

Note You cannot downgrade a Cisco CSR 1000v image on Microsoft Azure to Cisco IOS XE Everest 16.6.2 or earlier. For example, if you are running Cisco IOS XE Fuji 16.7.1 or later you must not downgrade to Cisco IOS XE Everest 16.6.2 or earlier.

Note To upgrade or downgrade a Cisco CSR 1000v image on Microsoft Azure, you need to expand the `.bin` file and use `packages.conf` to upgrade to the new version.

Note The only currently available downgrade for a Cisco IOS XE Gibralter 16.10.1 image is to Cisco IOS XE Fuji 16.9.2. You cannot downgrade a Cisco CSR 1000v image on Microsoft Azure from Cisco IOS XE Gibralter 16.10 to Cisco IOS XE Fuji 16.9.1 or earlier.

Note If you are upgrading from an earlier release to 16.10.1 to achieve Accelerated Networking performance, ensure that you select the AZN variant of the 16.10.1 .bin image. The AZN variant contains the string "azn" in the filename. For example, `csr1000v-universalk9azn.2018-12-03_23.12_abcd4.SSA.bin`.

Check the version of Cisco IOS XE that is running on the Cisco CSR 1000v by using the `show version` command. Example:

```
Router# show version
Cisco IOS XE Software, Version 2017-11-08_14.44_user4
Cisco IOS Software [Fuji], Virtual XE Software (X86_64_LINUX_IOSD-UNIVERSALK9-M), Experimental
 Version 16.8.2017110
```

Procedure

	Command or Action	Purpose
Step 1	**scp** *upgrade-image azure-username@csr-public-ip-address:copied-upgrade-image* **Example:** The public IP address of the Cisco CSR 1000v used in the following example is 207.46.130.0. `scp UpgradeImage.bin azureusr1@207.46.130.0:upgrade.bin`	Copy the new image to the CSR 1000v (boot flash memory). You can choose any name for the copy of the image in bootflash; for example, `upgrade.bin`. **Note** If you are upgrading to Cisco IOS XE 16.10.1 or later, use a Microsoft Azure Accelerated Networking (AN) .bin image from www.cisco.com. For example: csr1000v-universalk9azn.16.10.x.SSA.bin

Step 2	**request platform software package expand file bootflash:upgrade.bin to bootflash:upgrade/**
	Example:
	`Router# request platform software package expand file bootflash:upgrade.bin to bootflash:upgrade/`
	`Nov  8 03:25:34.412 %INSTALL-5-OPERATION_START_INFO: R0/0: packtool: Started expand package bootflash:upgrade.bin`
	`Verifying parameters`
	`Expanding superpackage bootflash:upgrade.bin`
	`Validating package type`
	`Copying package files`
	`SUCCESS: Finished expanding all-in-one software package.`
		Expand the image that is in boot flash memory.
		Note If you have already performed an upgrade on this CSR instance before, use a different directory name when you perform the upgrade the second time. For example, if you used the `request platform software package expand file bootflash:upgrade.bin to bootflash:upgrade/` command when you performed the upgrade the first time, this command expands the bin file in the 'upgrade' directory of the blootflash and places the packages.conf file in the upgrade directory.
		When you perform an upgrade anytime after, ensure that you use a different directory name. For example, your upgrade command would read `request platform software package expand file bootflash:upgrade2.bin to bootflash:upgrade2/`. Note that the directory name is `upgrade2` in this instance.

Step 3	**configure terminal**	Enter global configuration mode.

	Command or Action	Purpose

Step 4	**boot system bootflash:upgrade/packages.conf**
	Example:
	The following example shows how to correctly enter a boot system entry:
	`Router(config)# boot system`
	`bootflash:upgrade/packages.conf`
		Add a boot system entry to the `packages.conf` file that was generated in step 2. For example, add the boot system entry to the `/bootflash/upgrade/packages.conf` file as shown in the example on the left.
		Note Do **not** add the boot system entry like this: `boot system bootflash:upgrade.bin`. This command tells the Cisco CSR 1000v to boot from `upgrade.bin`. However, the CSR 1000v boot fails if the file size of `upgrade.bin` is greater than the low memory size that is allowed by GRUB in Microsoft Azure.
		Note Ensure that you point the boot system command entry to the `packages.conf` file expanded in the upgrade directory as mentioned in step 2. You must use the same directory name that you have specified in step 2.

Step 5	**end**
	Example:
	`Router(config)# end`
	`Router#`
		Exit global configuration mode and return to privileged EXEC mode.

Step 6	**show run | sec boot**
	Example:
	`Router# show run | sec boot`
	`boot-start-marker`
	`boot system bootflash:upgrade/packages.conf`
	`boot-end-marker`
	`diagnostic bootup level minimal`
		Verify the boot system entry.

Step 7	**copy running-configuration startup-configuration**
	Example:
	`Router# copy running-configuration`
	`startup-configuration`
	`Building configuration`
	`...`
	`[OK]`
		Save the configuration.

Step 8	Reload the router.	

Deploying CSR 1000v on Microsoft Azure vs Amazon Web Services

The differences between deploying Cisco CSR 1000v on Microsoft Azure and Amazon Web Services (AWS) are shown in the following table:

Table 1: Comparing Cisco CSR 1000v on Microsoft Azure and Amazon Web Services

Function	Cisco CSR 1000v on Microsoft Azure	Cisco CSR1000v on AWS
Number of Interfaces	1, 2, 4, or 8 Interfaces	3 or more Interfaces
Multiple IP addresses	Multiple IP addresses per vNIC	Multiple IP addresses per vNIC
GRE tunnel	GRE tunnel is unsupported	GRE tunnel is supported
Redundancy	Routing Redundancy is supported through 2 CSR instances	Routing Redundancy is supported through 2 CSR instances
Attachment/Detachment of interface on the running Cisco CSR 1000v	Not supported	Supported
Overlapping IP subnet	Supports overlapping IP subnets in different virtual networks.	Support overlapping IP subnet in different VPC

Best Practices and Caveats

1. Cisco recommends that you keep resources in a Resource Group. To clean up all the resources in a group, you can remove the relevant Resource Group.
2. When a Cisco CSR 1000v VM is deleted, not all the resources for the VM are deleted (route table, security group, public IP, network interfaces). Then, if you create a new Cisco CSR 1000v with the same name as before, the previous resources may be re-used. If you do not want to re-use these resources, choose one of the following actions:
   - Manually remove each individual resource.
   - Remove the Resource Group containing the individual resources.
   - Create a new Cisco CSR 1000v with a different name.
3. If you use the deployment template to create a Cisco CSR 1000v, make sure that the public IP address is configured as static on Microsoft Azure. (In Microsoft Azure, navigate to the public IP address and in the configuration settings, see if the address is shown as Dynamic or Static. Ensure that Static is selected (the default is Dynamic).

SSH Connectivity Issues

You may fail to establish an SSH connection to a Cisco CSR 1000v on Microsoft Azure after you initially deploy the Cisco CSR 1000v, or after you reload or restart the Cisco CSR 1000v. In the Azure portal, the Cisco CSR 1000v is in the running state. The following three scenarios suggest workarounds for when you fail to connect using SSH.

Scenario 1. Attempted SSH access soon after booting up CSR 1000v.

You may fail to establish an SSH connection if you tried to gain access to the Cisco CSR 1000v soon after boot up. After starting the deployment of a CSR 1000v, it takes about 5 minutes for SSH connectivity to become available.

Scenario 2. Binding problem in the Microsoft Azure Infrastructure.

Microsoft Azure support recommends that you perform the following steps:

1. On the Cisco CSR 1000v interface that has a public IP address, reassign the private IP address to a new static IP address within the subnet.

2. Open the PowerShell in the Azure portal.

3. Update the ARM VM.

   Refer to this Azure documentation: https://docs.microsoft.com/en-us/powershell/module/azurerm.compute/update-azurermvm?view=azurermps-5.6.0.

4. In the powershell, enter the following commands:

   $vm = Get-AzureRmVM -Name "reload-lnx" **-ResourceGroupName** "reload-rg"

   Update-AzureRmVM -VM $vm -ResourceGroupName "reload-rg"

5. Reset the network interface to which the public IP address is attached.

   For further information on resetting the network interface, see: https://docs.microsoft.com/en-us/azure/virtual-machines/windows/reset-network-interface.

6. Select **VM** > **Networking** and select the Network Interface.

7. Go to **IP configurations** and select the IP name.

8. If the private IP address that is assigned to the interface is statically configured, write down the address, for use in step **13**.

9. Under "Assignment", click **Static**.

10. In the IP address field, use an available IP address. Choose an available IP address within the subnet to which the network interface is connected.

11. Click **Save** and wait for the save to complete.

12. Retry connecting to the router using SSH.

13. After you add (or change) a static IP address and gain access to the VM, if the IP address that was originally assigned to this interface (see step **8.**) was statically configured, you can either change the IP address from static to dynamic, or you can reconfigure the IP address to the original address (the address you noted in step **8**).

Scenario 3. Misconfiguration of idle terminal timeouts.

When you start an SSH session to the CSR 1000v, ensure that you do not configure the terminal VTY timeout as infinite—do not configure: `exec-timeout 0 0`. Use a non-zero value for the timeout; for example, `exec-timeout 4 0` (this command specifies a timeout of four minutes and zero seconds).

The reason why the `exec-timeout 0 0` command causes an issue is as follows:

Azure enforces a timeout for the console idle period of between 4 and 30 minutes. When the idle timer expires, Azure disconnects the SSH session. However, the session is not cleared from the point of view of the CSR 1000v, as the timeout was set to infinite (by the `exec-timeout 0 0` configuration command). The disconnection causes a terminal session to be orphaned. The session in the CSR 1000v remains open indefinitely. If you try to establish a new SSH session, a new virtual terminal session is used. If this pattern continues to occur, the number of allowed simultaneous terminal sessions is reached and no new sessions can be established.

In addition to configuring the `exec-timeout` command correctly, it is also a good practice to delete idle virtual terminal sessions using the commands that are shown in the following example:

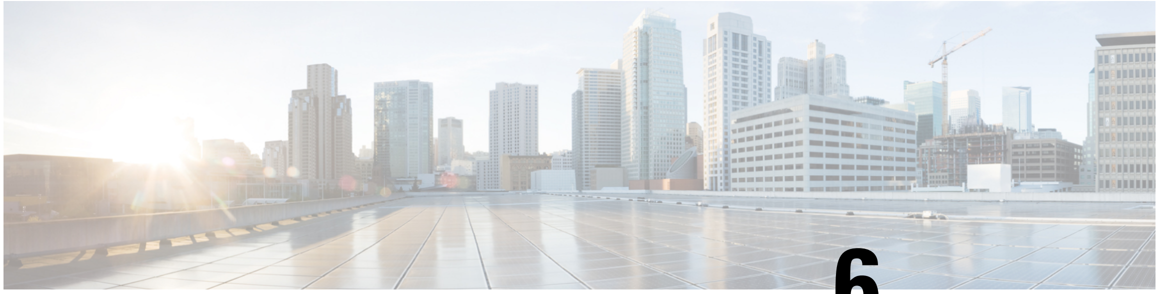
```
CSRA# show users
Line User Host(s) Idle Location
2 vty 0 cisco idle 00:07:40 128.107.241.177
* 3 vty 1 cisco idle 00:00:00 128.107.241.177

CSRA# clear line 2
```

If the workarounds in the preceding scenarios are ineffective, as a last resort, you can restart the Cisco CSR 1000v in the Azure portal.

Other Related Resources

DMVPN is supported on Microsoft Azure and AWS. The configuration for Microsoft Azure is similar to AWS. For further information, see the following white paper:

Extending Your IT Infrastructure Into Amazon Web Services Using Cisco DMVPN and the Cisco Cloud Services Router 1000V Series (PDF)



CHAPTER 6

Configuring Accelerated Networking on Microsoft Azure

Table 2: Feature History

Release	Description
Cisco IOS XE Gibraltar 16.10.1	Feature introduced. Support for accelerated networking with Mellanox 4.
Cisco IOS XE Gibraltar 16.12.1	Support for accelerated networking with Mellanox 5 introduced.
Cisco IOS XE Amsterdam 17.3.1	Support for Azure-PMD introduced.

Overview of Accelerated Networking

What is Accelerated Networking

Accelerated networking enables single root I/O virtualization (SR-IOV) on VMs such as a Cisco CSR 1000V VM. The accelerated networking path bypasses the virtual switch, increases the speed of network traffic, improves the networking performance, and reduces the network latency and jitter.

Usually, all the networking traffic in and out of the VM traverses the host and the virtual switch. However, with accelerated networking, the network traffic arrives at the virtual machine's network interface (NIC), and is then forwarded to the VM. Thus, all the network policies that the virtual switch applies are now offloaded and applied in the hardware.

For more information about the accelerated networking functionality that is available in Microsoft Azure, see Create a Linux VM With Accelerated Networking Using Azure CLI.

Accelerated networking is available in CSR 1000V public cloud deployments and in government cloud deployments for a Cisco CSR 1000V version Cisco IOS XE Gibraltar 16.10.1 or later.

If you are upgrading to Cisco IOS XE Gibraltar 16.10.x and 16.11.x, use a Microsoft Azure Accelerated Networking (AN) .bin image from www.cisco.com. For example, `csr1000v-universalk9azn.16.10.x.SSA.bin`.

Support for Azure-PMD

The Azure-PMD (Poll Mode Driver) functionality on Azure offers a faster, user-space packet processing framework for performance-intensive applications. This framework bypasses the virtual machine's kernel network stack. In a typical packet processing that uses the kernel network stack, the process is interrupt-driven. When the network interface receives the incoming packets, there is an interruption to the kernel to process the packet and a context switch from the kernel space to the user space. Azure-PMD eliminates the context switching and the interrupt-driven method in favor of a user-space implementation that uses poll mode drivers for fast packet processing.

Starting the Cisco IOS XE 17.3 release, you can enable the Azure-PMD functionality for CSR 1000V running on Microsoft Azure. This functionality increases the performance of the CSR 1000V when compared to the previous versions that use accelerated networking.

Supported VM Instance Types

The following VM instance types support the Accelerated Networking functionality:

IOS XE Version	Supported VM Instance Types
16.12.x	DS2_v2 / D2_v2 DS3_v2 / D3_v2 DS4_v2 / D4_v2
17.1.x	DS2_v2 / D2_v2 DS3_v2 / D3_v2 DS4_v2 / D4_v2
17.2.x	DS2_v2 / D2_v2 DS3_v2 / D3_v2 DS4_v2 / D4_v2
17.3.x	DS2_v2 / D2_v2 DS3_v2 / D3_v2 DS4_v2 / D4_v2 F16s_v2 F32s_v2

Support for Mellanox Hardware

Microsoft Azure cloud has two types of hardware that support the accelerated networking functionality. The following table specifies the Mellanox versions supported for the accelerated networking functionality.

Table 3: Compatibility Matrix of IOS Versions and Accelerated Networking

IOS XE Version	Support for Accelerated Networking	Support for MLX4	Support for MLX5	Support for Azure-PMD
16.10.x	Yes	Yes	No	No
16.11.x	Yes	Yes	No	No
16.12.x	Yes	Yes	Yes	No
17.1	Yes	Yes	Yes	No
17.2	Yes	Yes	Yes	No
17.3	Yes	Yes	Yes	Yes

Note MLX4 (Mellanox 4) is also referred to as ~ connectx3 = cx3, and MLX5 (Mellanox 5) is also referred as connectx4 = cx4.

You can't specify which NIC Azure uses MLX4 or MLX5 for your VM deployment. Cisco recommends that you upgrade to CSR 1000V 16.12 version or later to use the accelerated networking functionality.

Enable Accelerated Networking

To enable accelerated networking on a Cisco CSR 1000V, the instance must be running a Microsoft Azure AN variant image (applicable for Cisco IOS XE Gibraltar 16.10.x or 16.11.x images only), or a 16.12.x image, or later. Read on to know how to check for the variant image before you enable accelerated networking.

Check For Variant Image

Before you enable Microsoft Azure AN for a Cisco CSR 1000V VM in Microsoft Azure, ensure that you are running a Microsoft Azure AN variant image by executing the following Cisco IOS EXEC command: `router# show platform software system hypervisor`. This command displays specific details of your instance from the hypervisor.

The following example shows the output for a CSR1000V 16.10.x or 16.11.x image:

```
router# show platform software system hypervisor

Hypervisor: AZURE
Product Name : Virtual Machine in Azure
Manufacturer: Microsoft Corporation
Serial Number: 0000-0009-8597-0349-7291-4826-11
UUID: 17B4D488-BC82-F345-A829-F6279F54047D
image_variant : azn ===>> verifies you are running a Microsoft Azure AN variant image
Cloud Metadata
-------------------
Region: eastus
Zone: 1
Instance ID: cc60aff2-b7dc-4d81-9854-1c2be6eeacc2
Instance Type: Standard_DS2_v2
```

```
Version: 16.11.120210903
Image ID:
Publisher: cisco
Offer: cisco-csr1000v
SKU: 16_11_01a-byol

Interface Info
-------------------
Interface Number : 0
    IPv4 Public IP:
    IPv4 Private IP: 10.1.0.4
    IPv4 Subnet Mask: 24
    IPv4 Network: 10.1.0.0
    IPv4 Gateway: 10.1.0.1
    MAC Address: 000D3A180834
```

Note The version that is displayed in this command output is the initial deployment version of your Cisco CSR1000V instance. To view the current or upgraded version of your instance, run the **show version** command.

Note If you are upgrading from an earlier release to a 16.10.x or a 16.11.x image, select the AZN variant of the 16.x.x.bin image to achieve AN performance. The AZN variant contains the string "azn" in the file name. For example, csr1000v-universalk9azn.2018-12-03_23.12_abcd4.SSA.bin.

Enable Accelerated Networking

To enable accelerated networking, create or modify a vNIC using the **az network nic** command and the `--accelerated-networking` option. See the Microsoft Azure documentation for the az network nic command and also refer to the following examples.

Note Depending on how you created the CSR 1000V instance, accelerated networking might initially be disabled on the CSR NICs. If accelerated networking is disabled on the NIC, and you want to enable accelerated networking on an interface, use one of the commands as shown in the following examples.

Example 1

This example shows how to create a vNIC "mynic1" and enable accelerated networking using the **az network nic create** command with the `--accelerated-networking true` option.

```
az network nic create -n mynic1 -g "RG1" --accelerated-networking true -l "east us"
--vnet-name "vnetname" --subnet "subnet1"
```

Example 2

This example shows how to create a vNIC "mynic2" and enable accelerated networking using the **az network nic create** command with the `--accelerated-networking true option` option.

```
az network nic create -n "mynic2" -g "RG1" --accelerated-networking true -l "east us"
--vnet-name "vnetname" --subnet "subnet1"
```

Example 3

This example shows how to modify a vNIC "mynic3" to enable accelerated networking using the **az network nic update** command with the `--accelerated-networking true` option.

```
az network nic update -n mynic3 -g rg1 --accelerated-networking true
```

Caution Due to a Microsoft Azure limitation, enabling accelerated networking on all the interfaces of a Cisco CSR 1000V router might cause a significant performance drop if packets greater than 1500 bytes are sent across the Azure infrastructure. The performance degradation occurs because Azure starts fragmenting the packets at 1438 bytes and drops out the sequence packets. This is a known issue.

Verifying Accelerated Networking

After Enabling accelerated networking on the NICs, use the following IOS commands to verify whether accelerated networking is enabled on the NIC. The Azure infrastructure uses Mellanox NICs to achieve SR-IOV or accelerated networking.

You can use the following commands to verify CSR NICs by using the Mellanox kernel drivers as the NIC's I/O drivers to process the packets. In addition, the Mellanox NICs in the HyperV server of the Azure infrastructure presents a bonded interface to the CSR 1000V guest VM. This VM is used for accelerated networking, and the VM is in a bonded state whenever accelerated networking is enabled.

Run the following command to verify the accelerated networking status:

```
device#show platform software vnic-if database
vNIC Database
eth00_1539659125237802400
    Device Name : eth0
    Driver Name : mlx4_en  ==>> this verifies that AN is enabled on NIC and CSR is using
mellanox kernel modules/drivers as NIC's I/O driver.
    MAC Address : 000d.3a1e.11f9
    PCI DBDF    : 86ab:00:02.0
    UIO device  : no
    Management  : no
    Status      : bonded   ==>> this verifies that AN is enabled on the NIC and mellanox
kernel modules have recognized SR-IOV and kernel is presenting bonded interface mode to
CSR.
  eth01_1539659128292894000
    Device Name : eth1
    Driver Name : mlx4_en
    MAC Address : 000d.3a1e.1c73
    PCI DBDF    : 9354:00:02.0
    UIO device  : no
    Management  : no
    Status      : bonded
  eth_17__1539659131397365100
    Device Name : Gi1
    Driver Name : hv_netvsc
    MAC Address : 000d.3a1e.1c73
    PCI DBDF    : UNKNOWN
    UIO device  : no
    Management  : no
    Status      : supported
  eth_18__1539659134427961100
    Device Name : Gi2
    Driver Name : hv_netvsc
    MAC Address : 000d.3a1e.11f9
    PCI DBDF    : UNKNOWN
```

```
UIO device  : no
Management  : no
Status      : supported
```

Note The previous example is a configuration output from a release that supported the MLX4 driver. However, from the 16.12.1 release, this output might display MLX4 or MLX5, depending on the MLX driver in your Azure Infrastructure.

Verifying Accelerated Networking for CSR 1000V 16.10.x Through 17.2.x Images

In the following example, you can see that the interface eth1 is in use, is processing packets, and reflects the packet counters in the vf.

```
pdev1010anenabled-csr#show controllers
GigabitEthernet1 - Gi1 is mapped to eth_17_ on VXE
  DPIF Rx Drop 0 Packets 46339
  Driver Rx Stops 0 DPIF Rx Congestion Drop 0
Detailed interface statistics:
  tx_scattered 0
  tx_no_memory 0
  tx_no_space 0
  tx_too_big 0
  tx_busy 0
  tx_send_full 0
  rx_comp_busy 0
  vf_rx_packets 29750   ==>>> This verifies Accelerated networking is working properly.
this numbers should be moving if traffic is passing through CSR.
  vf_rx_bytes 32439581
  vf_tx_packets 48761
  vf_tx_bytes 7109172
  vf_tx_dropped 0
  tx_queue_0_packets 53
  tx_queue_0_bytes 7630
  rx_queue_0_packets 8554
  rx_queue_0_bytes 3577166
  tx_queue_1_packets 11
  tx_queue_1_bytes 1628
  rx_queue_1_packets 8056
  rx_queue_1_bytes 3121870
Bonded interface (eth1) statistics:
  rx_packets 29750
  tx_packets 48748
  rx_bytes 32856081
  tx_bytes 7108700
  rx_dropped 0
  tx_dropped 0
  tso_packets 0
  xmit_more 0
  queue_stopped 0
  wake_queue 0
  tx_timeout 0
  rx_alloc_pages 6144
  rx_csum_good 29748
  rx_csum_none 0
  rx_csum_complete 0
  tx_chksum_offload 0
  rx_xdp_drop 0
  rx_xdp_tx 0
  rx_xdp_tx_full 0
  tx0_packets 16
  tx0_bytes 1216
  tx1_packets 48732
  tx1_bytes 7107484
  rx0_packets 19051
  rx0_bytes 22867089
  rx0_dropped 0
  rx0_xdp_drop 0
  rx0_xdp_tx 0
  rx0_xdp_tx_full 0
  rx1_packets 10699
  rx1_bytes 9988992
  rx1_dropped 0
  rx1_xdp_drop 0
  rx1_xdp_tx 0
  rx1_xdp_tx_full 0
```

Verifying Accelerated Networking for CSR 1000V 17.3.x or Later (With Azure-PMD)

After enabling accelerated networking on the NICs, use the following IOS commands to verify whether accelerated networking with Azure-PMD is enabled on NIC. The Azure infrastructure uses Mellanox NICs to achieve SR-IOV or accelerated networking.

Use the following commands to verify the CSR 1000V NICs by using the Mellanox Azure-PMD drivers as the NIC's I/O drivers to process the packets. In addition, the Mellanox NICs in the HyperV server of the Azure infrastructure presents a bonded interface to the CSR 1000V guest VM. This VM is used for accelerated networking, and the VM is in a bonded state while accelerated networking is enabled. Note that the bonded interfaces share the same MAC address. The aggregate counters appear on Gi interfaces, while the non-accelerated packets counters appear on the net_tap interfaces. The accelerated packets counters appear on the net_mlx interfaces.

In the following example, the interface Gi2 indicates that a majority of the packets are flowing over the net_mlx interface.

```
csrANpmd#sh plat hard qfp act dat inf pmd controllers | inc NIC|good_packets
NIC extended stats for port 0  (Gi1) net_failsafe 000d.3a8f.1bf1 xstats count 13
  rx_good_packets: 411
  tx_good_packets: 326
NIC extended stats for port 1  (Bonded) net_mlx5 000d.3a8f.1bf1 xstats count 35
  rx_good_packets: 389
  tx_good_packets: 326
NIC extended stats for port 2  (Bonded) net_tap 000d.3a8f.1bf1 xstats count 13
  rx_good_packets: 22
  tx_good_packets: 0
NIC extended stats for port 3  (Gi2) net_failsafe 000d.3a8f.1040 xstats count 13
  rx_good_packets: 10638289
  tx_good_packets: 3634525
NIC extended stats for port 4  (Bonded) net_mlx5 000d.3a8f.1040 xstats count 35
  rx_good_packets: 10639534.  ==>>> This verifies Accelerated Networking is working properly
 for RX
  tx_good_packets: 3636099     ==>>> This verifies Accelerated Networking is working properly
 for TX
NIC extended stats for port 5  (Bonded) net_tap 000d.3a8f.1040 xstats count 13
  rx_good_packets: 291
  tx_good_packets: 0
NIC extended stats for port 6  (Gi3) net_failsafe 000d.3a8f.1a90 xstats count 13
  rx_good_packets: 3637187
  tx_good_packets: 10522981
NIC extended stats for port 7  (Bonded) net_mlx5 000d.3a8f.1a90 xstats count 35
  rx_good_packets: 3638631
  tx_good_packets: 10524554
NIC extended stats for port 8  (Bonded) net_tap 000d.3a8f.1a90 xstats count 13
```

```
  rx_good_packets: 28
  tx_good_packets: 0
```

Disable Accelerated Networking

To disable accelerated networking for the Cisco CSR 1000v, you can create or modify a vNIC using the **az network nic** command and the `--accelerated-networking` option.

For more information about the command, see the Microsoft Azure documentation for the az network nic command.

The following examples specify how to modify a vNIC.

Example

This example shows how to modify a vNIC "mynic1" to disable Accelerated Networking using the **az network nic update** command with the `--accelerated-networking false` option.

```
az network nic update -n "mynic1" -g rg1 --accelerated-networking false
```

CHAPTER 7

Usage Guidelines for User Defined Routes

Introduction to the Cisco CSR 1000v Route Tables

This section provides guidelines which will help you to decide user-defined routes to add to the route tables. When a Cisco CSR 1000v is deployed in a Virtual Network using the Microsoft Azure Marketplace template, a route table is created for each subnet to which the Cisco CSR 1000v has a network connection. For example, if you deploy a 4-NIC version of the Cisco CSR 1000v from the Microsoft Azure Marketplace, 4 subnets are created. Each subnet has an associated route table. No routes are automatically installed in the route table.

For further information on defining user-defined routes, also see the Microsoft Azure documentation: https://docs.microsoft.com/en-us/azure/, and search for "user defined routes".

User Defined Routes in the Same Virtual Network

By default, the Microsoft Azure network infrastructure provides a basic routing service which interconnects all the subnets within a virtual network. Packets can be passed between any virtual machines within the same virtual network without the assistance of the Cisco CSR 1000v.

However, if you need inter-subnet packets to be delivered to the Cisco CSR 1000v (to implement advanced services such as filtering and QoS), then you need to install a user defined route in the routing table for the subnet that designates the Cisco CSR 1000v as the next hop router.

Routing between Virtual Networks or On-Premises Networks

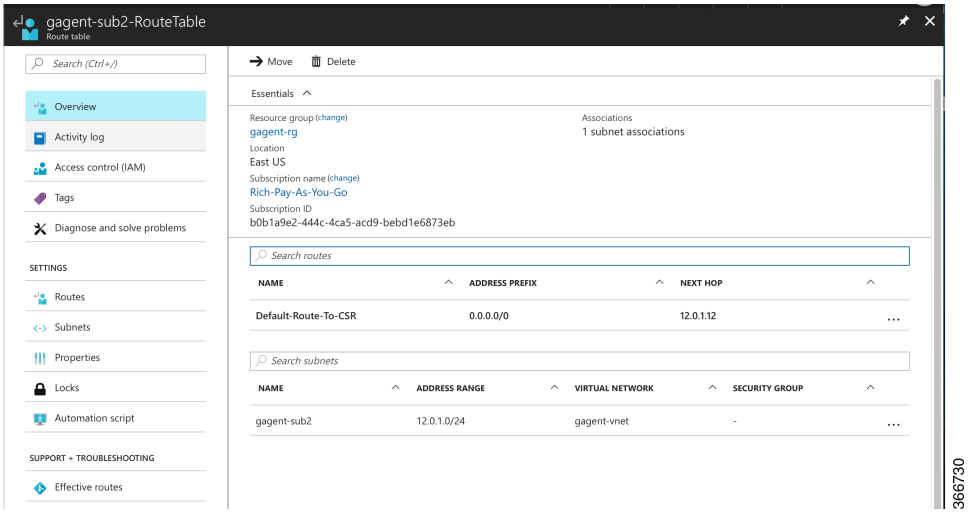
The Microsoft Azure network infrastructure does not by default interconnect different virtual networks or connect virtual networks to on-premises networks. To connect to these networks, you must create a user-defined route in each route table to specify the Cisco CSR 1000v as the next hop router to each remote network. The user-defined route can be either a default route or a specific destination route. To force traffic through the Cisco CSR 1000v, install either a default route or a specific destination route in the route table that points to the Cisco CSR 1000v. (Refer to the two examples below.)

Note If a default route is installed in a route table, all traffic is diverted to the specified next hop. This causes a problem if you have virtual machines with an allocated public IP address (used for management access to the VM). If you have a default route in the route table associated with the subnet, the virtual machine is not reachable via its public IP address.

Note Microsoft Azure supports a feature called VNET Peering, which can interconnect virtual networks as long as they are hosted in the same region. In order to use VNET Peering and utilize services within the Cisco CSR 1000v, you need to add a user-defined route to force traffic through the Cisco CSR 1000v.

The following example shows a default route pointing to the Cisco CSR 1000v.

Figure 2: Routing table in Microsoft Azure with a default route to the Cisco CSR 1000v



Resource group (change)
gagent-rg
Location
East US
Subscription name (change)
Rich-Pay-As-You-Go
Subscription ID
b0b1a9e2-444c-4ca5-acd9-bebd1e6873eb

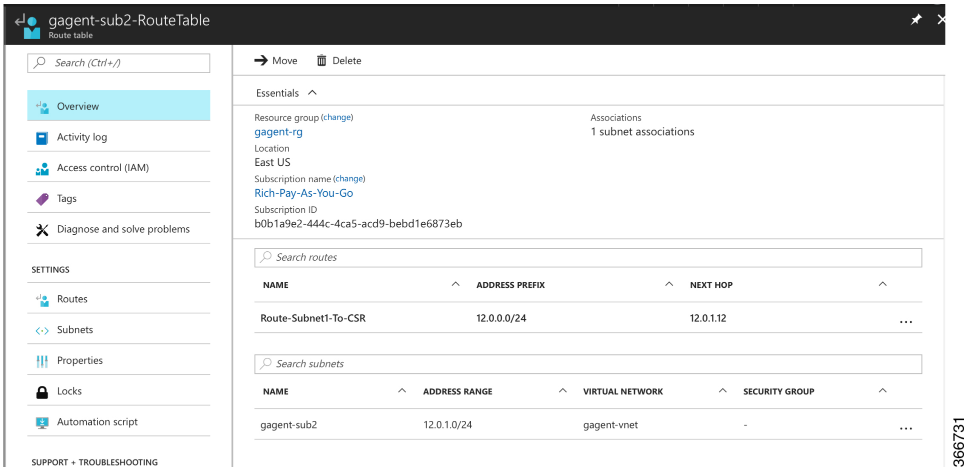
Associations
1 subnet associations

NAME	ADDRESS PREFIX	NEXT HOP
Default-Route-To-CSR	0.0.0.0/0	12.0.1.12

NAME	ADDRESS RANGE	VIRTUAL NETWORK	SECURITY GROUP
gagent-sub2	12.0.1.0/24	gagent-vnet	-

The following example shows a specific destination route pointing to the Cisco CSR 1000v.

Figure 3: Routing table in Microsoft Azure with a specific destination route to the Cisco CSR 1000v



Resource group (change)
gagent-rg
Location
East US
Subscription name (change)
Rich-Pay-As-You-Go
Subscription ID
b0b1a9e2-444c-4ca5-acd9-bebd1e6873eb

Associations
1 subnet associations

NAME	ADDRESS PREFIX	NEXT HOP
Route-Subnet1-To-CSR	12.0.0.0/24	12.0.1.12

NAME	ADDRESS RANGE	VIRTUAL NETWORK	SECURITY GROUP
gagent-sub2	12.0.1.0/24	gagent-vnet	-

User Defined Routes for High Availability

You can deploy two Cisco CSR 1000v's in the same virtual network to provide 1:1 redundancy for high availability. A Cisco CSR 1000v, configured with high availability, monitors the reachability of its peer router. If the Cisco CSR 1000v believes that the peer router has gone down, it installs its own IP address in the route table. This causes traffic to be routed through the "working" Cisco CSR 1000v.

When you configure user defined routes, you need to decide if you want the entries in the route table to be updated when there is a failure of one of the Cisco CSR 1000v peer routers. You must configure a redundancy node for each user-defined route table if the route table is one in which the high availability feature needs to redirect traffic to the “working” Cisco CSR 1000v.

For Cisco IOS XE Everest 16.6, all the routes in the route table specified by a redundancy node are updated in the case of a Cisco CSR 1000v peer failure.

CHAPTER 8

Configuring High Availability on the Cisco CSR 1000v

High Availability refers to the ability to establish redundancy of networking functionality and configuration data between two peer routers.

Information About High Availability on the Cisco CSR 1000v

Overview of High Availability for the CSR 1000v on Microsoft Azure

You can deploy the Cisco CSR 1000v between the front-end and back-end subnets. The Cisco CSR 1000v represents a single point of failure for access to back-end resources. For example, you can configure two Cisco CSR 1000vs and connect them in parallel between the front and back end subnets. Each of these Cisco CSR 1000vs is known a peer router.

The routing table for the back-end subnet contains entries that point to the next hop router, which is one of the two Cisco CSR 1000vs. The routing protocol that is configured on the Cisco CSR 1000v determines the path of the downstream traffic.

The peer Cisco CSR 1000v routers communicate with one another over a tunnel using the Bi-directional Forwarding Detection (BFD) protocol. The BFD generates an event if connectivity between the peer routers is lost. This event causes the active Cisco CSR 1000v to update the entries in the route table to point to itself as the default router. The routing table controls the upstream traffic of the Cisco CSR 1000v. An active router is the next-hop router for either an individual route or all routes in the user-defined route table.

In cloud environments, it is common for virtual networks to implement a simplistic mechanism for routing, which is based on a centralized route table. You can create multiple route tables. Each route table has a subnet that is assigned, which is a source of route information. In Microsoft Azure, the route table is populated automatically and includes one or more individual routes depending upon the network topology. You can configure the routes in the route table. You can also configure using the GUI on the Microsoft Azure portal web site, entering Azure CLI commands, or by using the programmatic APIs (for example, a REST API).

Using a centralized route table for a subnet allows a pair of Cisco CSR 1000v routers to operate in a redundant fashion. You can deploy two Cisco CSR 1000vs can be deployed in the same virtual network and have interfaces that are directly connected to subnets within the virtual network. You can add routes to the route table to point to one of the two redundant CSR 1000vs. So at any given time, one of the two CSRs is the next hop router for a subnet. This router is called the active router for the subnet and the other (peer) router is the passive router.

A subnet has a centralized route table, which allows two Cisco CSR 1000v routers to operate in a redundant mode. You can deploy two Cisco CSR 1000vs in the same virtual network with their interfaces that are directly connected to subnets in the virtual network. You can add routes to the route table to point to one of the two redundant CSR 1000vs. At any given time, one of the two Cisco CSR 1000v's serves as the next hop router for a subnet. This router is called the active router for the subnet. The peer router is referred to as the passive router. The “active” Cisco CSR 1000v is the next hop for a given route destination.

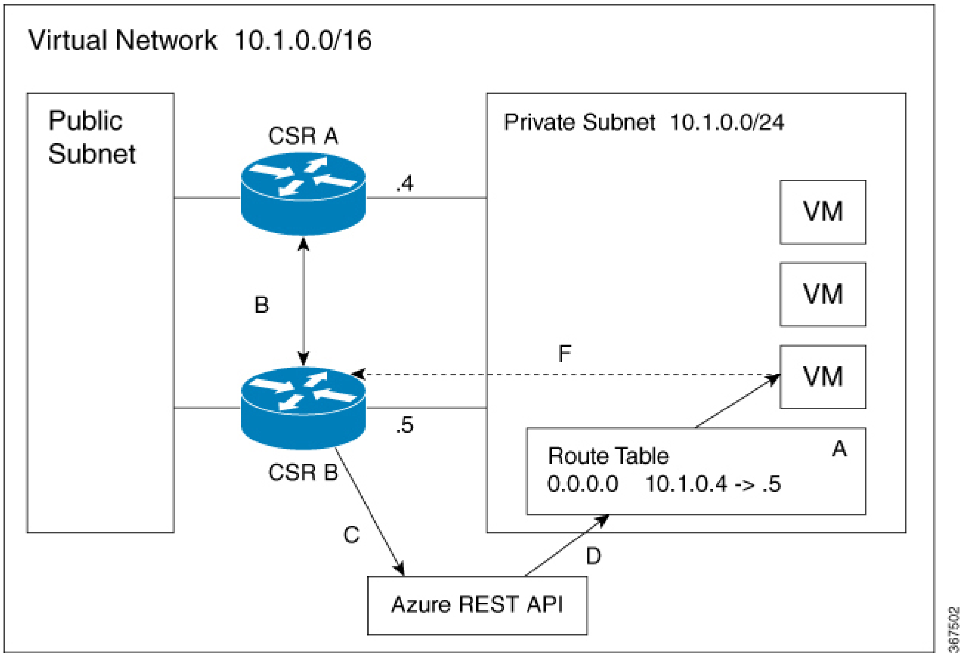
Failure detection is a mechanism that is used by a Cisco CSR 1000v to detect whether its peer Cisco CSR 1000v is operating properly. The Bidirectional Failure Detection (BFD) protocol is used. An IP tunnel is created between the two peer routers. Each router periodically sends a BFD protocol message to the other router. If one router fails to receive a BFD message from its peer for some period, it concludes the peer router has failed.

If the active router fails, the route table for the subnet can be dynamically updated to change the next hop address for one or more routes so that they refer to the passive router. If the peer router detects the failure of the active router, the peer Cisco CSR 1000v router uses the programmatic API to update the route table entries.

Note The Microsoft Azure route table updates may take up to one minute to take effect and may cause a delay in the High Availability failover.

For a route table entry, configure which of the two Cisco CSR 1000v routers is the primary router. The other router is the passive router if it is configured as a secondary router. By default, all routes are configured as secondary.

Consider the simple network diagram that is shown in the following figure. The topology in this figure is an example of 1-for-1 redundancy. For more information, see the Redundancy Topologies, on page 51 section.



The Private Subnet has an address block of 10.1.0.0/24. CSR A and CSR B provide a redundant path for traffic leaving this leaf subnet. The subnet has a route table that provides the route information to virtual machines which are attached to the Private Subnet. Initially the default route in the route table records the IP address of the next hop router as 10.1.0.4 (CSR A). All traffic leaving the subnet goes through CSR A. CSR A is currently the active router for the default route. Then, CSR A fails and CSR B detects the failure because it stops receiving BFD protocol messages from CSR A. CSR B writes to the route table via a REST API to change the default route to the interface of CSR B on the 10.1.0.0/24 subnet, which is IP address 10.1.0.5. CSR B becomes the active router for the default route.

Step	Description
A	CSR A with address 10.1.0.4 is the active router for the 15.0.0.0 network.
B	CSR A fails. CSR B detects the failure using the BFD protocol.
C	CSR B uses an HTTP request to the Azure REST API.
D	Azure updates the 15.0.0.0 route in the user-defined route table to the IP address of CSR B.
E	Virtual machines see the route table update.
F	Packets from the virtual machines are now directed to CSR B.

Redundancy Topologies

Two different redundancy topologies are supported:

1-for-1 redundancy topology If both of the Cisco CSR 1000v routers have a direct connection to the same subnet, the routers provide 1-for-1 redundancy. All the traffic that is intended for a Cisco CSR 1000v only goes to the active router. The active router is the next-hop router for a subnet. The other router is the passive router for all the routes.

Load sharing topology In this topology, both the Cisco CSR 1000v routers have direct connections to different subnets within the same virtual network. Traffic from subnet A goes to router A and traffic from subnet B goes to router B. Each of these subnets is bound to different route tables. If router A fails, the route table for subnet A is updated. Instead of router A being the next hop, the route entry is changed to router B as the next hop. If router B fails, the route table for subnet B is updated. In the same manner if router B fails, the route table now includes router A as the next hop router.

Redundancy Nodes

A redundancy node is a set of configuration parameters that specifies an entry in a route table. The next hop of a route is updated when an active router fails. Configuring a redundancy node requires the following information:

Table 4: Redundancy Node Parameters

Parameter	Description
Route Table	The route may include the following details: • Name or identifier for the table • The region or group in which the table was created • Identifier for the creator/owner of the table • (Optional) Individual Route. If an individual route is not specified, the redundancy node represents all of the routes in the route table.
Credentials	Authentication for the Cisco CSR 1000v, which gives it the authorization to update entries in the route table. Each cloud provider may handle the process of obtaining and specifying the credentials differently.
Next Hop	Next hop address that is written to the routing table entry when a trigger event occurs. It is usually the next hop address of the interface for the Cisco CSR 1000v, on the subnet that is being protected.
Peer Router	Identifies the redundant router that forwards traffic for this route, after a failure occurs on this router.
Router Role	Identifies whether the redundancy node serves in a primary or secondary role. This is an optional parameter. If not specified, the router role defaults to a secondary role.

Event Types

The high availability feature recognizes and responds to three types of events:

Peer Router Failure

When the peer route fails, it is detected as a Peer Router Failure event. In response to this event, the event handler writes the route entry with the next hop address that is defined in the redundancy node. To enable this event to be generated, configure the BFD protocol to a peer router and associate the BFD peer with one or more redundancy nodes.

Revert to Primary Router

After a router recovers from a failure, the Revert to Primary Router event occurs. The purpose of the event is to ensure that in the route table entry for the redundancy node has this router defined as the primary router.

Redundancy Node Verification

Use the Redundancy Node verification event to verify the ability of the event handler to execute functions. The event handler detects a Redundancy Node verification event and reads the route entry that is specified by the redundancy node. The event handler writes the same data back to the route entry. The Redundancy Node verification event is triggered by executing a script (manually or programatically). For further information about verification events, see User-Defined Triggers, on page 80 in the Advanced Programming for High Availability on Microsoft Azure, on page 80 section.

New and Updated Information About High Availability

The first version of high availability in the Azure cloud—HA Version 1—was introduced in Cisco IOS XE Everest 16.5.1.

Note The second version of high availability --HA Version 2-- is available using Cisco IOS XE Fuji 16.9.2 or later. HA Version 2 supports several new features, and a new configuration and deployment mechanism. It is recommended that you migrate to HA version 2, as support for HA version 1 will be removed from a future IOS release.

The following functions are introduced in HA Version 2:

Active–Active Operation

You can configure both Cisco CSR 1000vs to be active simultaneously, which allows for load sharing. In this mode of operation, each route in a route table has one of the two routers serve as the primary router and the other router serves as a secondary router. To enable load sharing, take all the routes and split them between the two Cisco CSR 1000vs.

Reversion to Primary CSR After Fault Recovery

You can designate a Cisco CSR 1000v as the primary router for a given route. While this Cisco CSR 1000v is up and running, it is the next hop for the route. If the Cisco CSR 1000v fails, the peer Cisco CSR 1000v takes over as the next hop for the route, maintaining network connectivity. When the original router recovers from the failure, it reclaims ownership of the route and is the next hop router.

New Configuration and Deployment Mechanism

In HA version 2, the implementation of HA has been moved out of the Cisco IOS XE code and runs in the guestshell container. For further information on the guestshell, see the "Guest Shell" section in the Programmability Configuration Guide. In HA version 2, the configuration of redundancy nodes is performed in the guestshell using a set of Python scripts.

Authentication by Microsoft Managed Service Identity

The Azure Active Directory(AAD) must authenticate a Cisco CSR 1000v to access and update route tables in the Azure network. (In HA version 1, authentication is achieved by creating a Cisco CSR 1000v application in Azure Active Directory (AAD) to represent the Cisco CSR 1000v.)

Microsoft has a Managed Service Identity (MSI) service that automates the creation of an application for a virtual machine. For more information on MSI, see: https://docs.microsoft.com/en-us/azure/active-directory/managed-service-identity/overview. HA version 2 uses the MSI service to authenticate the Cisco CSR 1000v. You do not need to manually create the Cisco CSR 1000v application.

Note HA version 2 also continues to support authentication in Azure Active Directory (AAD).

User-Supplied Scripts

The guestshell is a container in which you can deploy your own scripts. HA Version 2 exposes a programming interface to user-supplied scripts, so you can write scripts that can trigger both failover and reversion events. You can develop your own algorithms and triggers to control which Cisco CSR 1000v provides the forwarding services for a given route.

Migrating from High Availability Version 1 to Version 2

Choose one of the following deployment options for High Availability(HA) on Microsoft Azure, on Cisco IOS XE Fuji 16.9.x:

- HA Version 1—continues to be supported in Cisco IOS XE Fuji 16.9.x. However, this will be deprecated in all releases later than Cisco IOS XE Fuji 16.9.x.
- HA Version 2 with Redundancy Node Configuration in Cisco IOS XE. Configuration steps using Cisco IOS XE CLI commands is supported in Cisco IOS XE Fuji 16.9.1 and provides access to the new features in HA version 2. This allows you to continue using your existing redundancy node configurations. However, this deployment option will be deprecated in all releases later than Cisco IOS XE Fuji 16.9.x.
- HA Version 2 with Redundancy Node Configuration in the guestshell. Configuration steps using guestshell-based Python scripts is supported in Cisco IOS XE Fuji 16.9.1. This is the preferred deployment method. If you currently use Redundancy Node Configuration in Cisco IOS XE, we recommend that, during the period of time that you are using Cisco IOS XE Fuji 16.9.x., you migrate to using Redundancy Node Configuration in the guestshell.

Note By default, in Cisco IOS XE Fuji 16.9.x, the Cisco CSR 1000v on Microsoft Azure runs HA Version 1. To run HA Version 2, you must manually install the "csr_azure_ha" package in the guestshell.

Differences in High Availability across Cisco IOS XE Releases

The following table shows some of the differences between running high availability in various IOS releases.

Table 5: Support of HA Functions for Different Cisco IOS XE Releases

Aspect	Cisco IOS XE 16.5.x to IOS XE 16.8.x	Cisco IOS XE 16.9.x	Cisco IOS XE 16.10.x or later
HA Version 1	Yes	Yes	No
HA Version 2	No	Yes	Yes
Redundancy node configuration in Cisco IOS XE	YES	Yes	Yes
Redundancy node configuration in guestshell	No	Yes	Yes
Revert back to primary router after recovery	No	Yes	Yes
CSR 1000v authentication by Azure Active Directory	Yes	Yes	Yes
CSR 1000v authentication by Managed Service Interface	No	Yes	Yes
User scripts to update routes	No	Yes	Yes

Comparison in the Configuration of HA Version 1 and HA Version 2

The following configuration procedures are unchanged between HA Version 1 and HA Version 2:

- Configuring a Tunnel Between Cisco CSR 1000v Routers
- Configuring EIGRP over Virtual Tunnel Interfaces, on page 57

For configuration of HA Version 1, see Configuring High Availability Version 1 for the CSR 1000v on Microsoft Azure, on page 81.

Configuring High Availability for the CSR 1000v on Microsoft Azure

Configuring High Availability in Cisco IOS XE

Configuring IOX and the Guestshell on Cisco IOS XE

The following Cisco IOS XE configuration shows the commands that are required to access the guestshell. You need not configure these prerequisites as they are included automatically in the startup-config file.

However, if you are upgrading the Cisco IOS XE, the configuration is not applied automatically. In this case, you must configure the prerquisites manually.

```
iox
ip nat inside source list GS_NAT_ACL interface GigabitEthernet1 vrf GS overload
ip route vrf GS 0.0.0.0 0.0.0.0 GigabitEthernet1 192.168.35.1 global
interface VirtualPortGroup0
 vrf forwarding GS
 ip address 192.168.35.101 255.255.255.0
  ip nat inside
 no mop enabled
 no mop sysid
ip access-list standard GS_NAT_ACL
 permit 192.168.35.0 0.0.0.255
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
```

An Event Manager Applet detects that the guestshell reaches the "up" state. The applet then performs an action - it installs the guestshell package by issuing the command: `pip install csr_azure_guestshell~=1.1 --user` . If the guestshell package is not installed, or you want to ensure you have the latest package installed, then you can manually install the package by issuing the `pip install csr_azure_guestshell~=1.1 --user` command at the guestshell prompt. See Installing the csr_azure_guestshell Package, on page 60.

Note Configure each Cisco CSR 1000v to use a DNS server. In this example, the Cisco CSR 1000v uses the DNS server in the "name-server 8.8.8.8" command. Refer to the IP Addressing: DNS Configuration Guide.

Configuring a Tunnel Between Cisco CSR 1000v Routers

Configure an IPsec tunnel or a VxLAN tunnel between Cisco CSR 1000v routers as shown in the following two configuration examples. A tunnel uses either the EIGRP or the BGP routing protocol. The tunnel uses Bidirectional Forwarding Detection (BFD) to detect peer failures.

Configuration of an IPsec Tunnel

```
crypto isakmp policy 1
 encr aes 256
 authentication pre-share
crypto isakmp key cisco address 0.0.0.0
!
!
crypto ipsec transform-set uni-perf esp-aes 256 esp-sha-hmac
 mode tunnel
!
!
crypto ipsec profile vti-1
 set security-association lifetime kilobytes disable
 set security-association lifetime seconds 86400
 set transform-set uni-perf
 set pfs group2
!
!
interface Tunnel1
 ip address 192.168.101.1 255.255.255.252
 load-interval 30
 tunnel source GigabitEthernet1
 tunnel mode ipsec ipv4
 tunnel destination 23.96.91.169
 tunnel protection ipsec profile vti-1
 bfd interval 500 min_rx 500 multiplier 3
```

Note We recommend that you specify a BFD interval of 500 ms or more. You can increase the BFD timers to account for the varying latency in different regions.

Configuration of a VxLAN Tunnel

```
interface Tunnel100
 ip address 192.168.101.1 255.255.255.0
 shutdown
 bfd interval 500 min_rx 500 multiplier 3
 tunnel source GigabitEthernet1
 tunnel mode vxlan-gpe ipv4
 tunnel destination 40.114.93.164
 tunnel vxlan vni 10000
```

Note We recommend that you specify a BFD interval of 500 ms or more. You can increase the BFD timers to account for the varying latency in different regions.

Configuring EIGRP over Virtual Tunnel Interfaces

Configure EIGRP over the virtual tunnel interfaces using the following steps.

Note Other than using EIGRP, which is the protocol used in the following steps, you also have the option of using either BGP, or OSPF.

Before you begin

Configure either a VxLAN or IPsec tunnel between the Cisco CSR 1000v routers.

SUMMARY STEPS

1. **router eigrp** *as-number*
2. **network** *ip-address subnet-mask*
3. **bfd all-interfaces**
4. **end**
5. **show bfd neighbors**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**router eigrp** *as-number* **Example:** `Device(config)# router eigrp 1`	Enables the EIGRP routing process and enters router configuration mode.
Step 2	**network** *ip-address subnet-mask* **Example:** `network 192.168.101.0 0.0.0.255`	Share the network of the tunnel using EIGRP.
Step 3	**bfd all-interfaces** **Example:** `Device(config-router)# bfd all-interfaces`	Enables BFD globally on all interfaces that are associated with the EIGRP routing process.

Step 4	**end**
	Example:
	`Device(config-router)# end`
		Exits router configuration mode and returns the router to privileged EXEC mode.

Step 5	**show bfd neighbors**
	Example:
	`Device# show bfd neighbors`
	`IPv4 Sessions`
	`NeighAddr      LD/RD        RH/RS    State  Int`
	`192.168.101.2  4097/4097    Up       Up     Tu100`
		Verifies that the BFD neighbor is active and displays the routing protocols that BFD has registered.

Configuring BFD Binding Between Routers

Before you Begin

Configure a tunnel between Cisco CSR 1000v routers and configure EIGRP or BGP over the tunnel interface.

Procedure

```
redundancy
cloud provider azure 2        << 2 is the node index
bfd 172.13.1.2
```

The cloud provider can be `azure` (commercial cloud) or `azusgov` (U.S. Government cloud). The node index matches the redundancy node that is configured in the guestshell. The IP address in the **bfd** command is that of the peer Cisco CSR 1000v.

Note If you make a mistake when you configure the BFD peer address, delete the address entry by executing the `no bfd peer 172.13.1.2` command. Then, enter the correct BFD peer IP address.

Ensure that you do no not run the `no cloud provider azure 2` command, as this command deletes the entire redundancy node. If you delete the entire redundancy node, the redundancy node configured in guestshell with index 2 is also deleted.

Configuring the Console Timeout

When you start an SSH session to the Cisco CSR 1000v, ensure that you do not configure the terminal VTY timeout as infinite - do not configure: **exec-timeout** 0 0. Use a non-zero value for the timeout; for example, **exec-timeout** 4 0 (this command specifies a timeout of four minutes and zero seconds). The reason why the **exec-timeout** 0 0 command causes an issue is as follows: Azure enforces a timeout for the console idle period of between 4 and 30 minutes. When the idle timer expires, Azure disconnects the SSH session. However, the session is not cleared from the point of view of the Cisco CSR 1000v, as the timeout was set to infinite (by the **exec-timeout** 0 0 configuration command). The disconnection causes a terminal session to be orphaned. The session in the Cisco CSR 1000v remains open indefinitely. If you try to establish a new SSH session, a new virtual terminal session is used. If this pattern continues to occur, the number of allowed simultaneous terminal sessions is reached and no new sessions can be established. In addition to configuring the exec-timeout command correctly, it is also a good practice to delete idle virtual terminal sessions using the commands that are shown in the following example:

```
CSRA# show users
Line User Host(s) Idle Location
2 vty 0 cisco idle 00:07:40 128.107.241.177
* 3 vty 1 cisco idle 00:00:00 128.107.241.177

CSRA# clear line 2
```

If the workarounds in the preceding scenarios are ineffective, as a last resort, you can restart the Cisco CSR 1000v in the Azure portal.

Configuring High Availability in the Guest Shell

Installing the csr_azure_guestshell Package

To allow the Cisco CSR 1000v instances to use Microsoft Azure resources and services, install the csr_azure_guestshell Python package. This package is automatically installed in the guestshell when a Cisco CSR 1000v instance is created on the Azure cloud.

However, if this package is not automatically installed, you can do so by performing the following procedure:

Before you Begin

Download the csr_azure_guestshell Python package from pypi.org. Ensure that you are not on the SUDO mode when you use the guestshell Python package.

Procedure

```
Router# guestshell enable      << enable the guestshell
Router# guestshell                       << enter the guestshell
(guestshell) pip install csr_azure_guestshell~=1.1 --user
```

Installing the csr_azure_ha Package

Before you Begin

Download the csr_azure_guestshell Python package from www.cisco.com to ensure that the Cisco CSR 1000v is running the latest version of the package. The package is not automatically installed in the guestshell; it must be be explicitly installed so that you can run HA version 2.

Procedure

```
guestshell enable
guestshell
pip install csr_azure_ha~=1.0 --user             << execute this command in the guestshell
```

Setting the Path Environment Variable

Update the path environment in the guest shell to specify the location of the configuration scripts.

Procedure

Run the following command in the guestshell:

```
source ~/.bashrc
```

Configuring Redundancy Nodes

Create and modify redundancy nodes using a set of Python scripts. See "Creating a Redundancy Node" and further sections below. Python scripts use parameters that are shown in the following tables:

- Redundancy Node Parameters: required by Python scripts to create, modify, or delete redundancy nodes.
- AAD Redundancy Node Parameters: required by Python scripts if you are authenticating the Cisco CSR 1000v using an application that is defined in the Azure Active Directory.

Table 6: Redundancy Node Parameters

Parameter Name	Switch	Description
Node index	-i	Index that is used to uniquely identify this node. Valid values: 1–255.
Cloud provider	-p	Specifies the type of Azure cloud: azure, azusgov, or azchina.
Subscription ID	-s	Azure subscription id.
Resource group name	-g	Name of the resource group that contains the route table.
Route table name	-t	Name of the route table to be updated.
Route	-r	IP address of the route to be updated in CIDR format. Can be IPv4 or IPv6 address. If a route is unspecified, then the redundancy node is considered to apply to all routes in the routing table of type "virtual appliance".
Next hop address	-n	IP address of the next hop router. Use the IP address that is assigned to this CSR 1000v on the subnet which utilizes this route table. Can be an IPv4 or IPv6 address.
Mode	-m	Indicates whether this router is the primary or secondary router for servicing this route. Default value is secondary.

Table 7: AAD Redundancy Node Parameters

Parameter Name	Switch	Description
Tenant ID	-d	Identifies the AAD instance.
Application ID	-a	Identifies the application in AAD.

Parameter Name	Switch	Description
Application key	-k	Access key that is created for the application. **Note** For High Availability version 2, the key must be URL unencoded. For High Availability version 1, the key must be URL encoded.

Creating a Redundancy Node

Run the following script to create a redundancy node and add it to the database.

create_node.py { *switch value* } [… [{ *switch value* }]

switch—See: Configuring Redundancy Nodes.

A valid redundancy node must have the following parameters configured:

- Node index
- Cloud provider
- Subscription ID
- Resource group name
- Route table name

Example

```
create_node.py -i 10 -p azure -s b0b1a9e2-4444-4ca5-acd9-bebd1e6873eb -g ds-rg -t
ds-sub2-RouteTable
```

```
-r 15.0.0.0/8 -n 192.168.7.4
```

If successful, the script returns a value of zero.

Setting Redundancy Node Parameters

If you need to change the value of parameters in an existing redundancy node, run the following script.

set_params.py{ *switch value* } [… [{ *switch value* }]

Example

```
set_params.py -i 10 -r 15.0.0.0/16 -n 192.168.7.5
```

The index parameter (`-i`) is required. This command sets the values of any specified parameters. If the specified parameter has already been defined for the redundancy node, then the value of the parameter is updated.

When a node index value of zero is specified, the values that are provided by the command for the specified parameters are treated as the default values for these parameters.

If successful, the script returns a value of zero.

Clearing Redundancy Node Parameters

If you want to clear the value of specified parameters, for an existing redundancy node, run the following script.

clear_params.py -i *value* { *value* } [...[{ *switch* *value* }]

Example

In this example, the clear_params script clears both the route and next hop address parameters.

```
clear_params.py -i 10 -r -n
```

Only the index parameter is required. The values of any additional specified parameters are cleared.

If you specify a node index value of zero, the parameter values that are provided are treated as the default values for the parameters.

If successful, the script returns a value of zero.

Show Node

Run the following script to display the parameter values of an existing redundancy node.

show_node.py{ -i *value* }

Example

```
show_node.py -i 10
```

-i specifies the index of the redundancy node (0–255).

If the script is successful, it displays the parameters of the specified node. If the script is unsuccessful, it displays an error message.

Delete Node

Run the following script to delete an existing redundancy node.

delete_node.py{ -i *value* }

Example

```
delete_node.py -i 10
```

-i specifies the index of the redundancy node (0–255).

The `-i` parameter must be specified. The node is expected to exist in the database. If the client tries to delete a node that is not in the database, an error is not generated. If successful, the script returns a value of zero and it displays the parameters of the specified node. If the script is unsuccessful, a non-zero return value is produced and an error message is written to the log file.

If the script is successful, it displays an error message.

Note The database of redundancy nodes is maintained on a virtual disk that is allocated to the guestshell. If you issue the following Cisco IOS XE command: **guestshell destroy**, then the virtual disk is deleted and all node configurations are lost.

If you want to recover from the loss of a deleted virtual disk, manually perform the following steps:

1. Enable the guestshell.
2. Install the "csr_azure_guestshell" Python package.
3. Install the "csr_azure_ha" Python package.
4. Run Python scripts to reconfigure all of the redundancy nodes.

Configuring High Availability in Microsoft Azure

Before You Begin

Before configuring High Availability for CSR 1000v on Microsoft Azure, you require:

- A virtual network setup in Microsoft Azure with two subnets.
- Two Cisco CSR 1000v VMs.
- Licenses for each Cisco CSR 1000v:

  (Cisco IOS XE Everest 16.6.1 or later) Enable the AX or SEC license.

  (Cisco IOS XE Everest 16.5.1 or earlier) Enable the AX license.

Methods for Configuring Microsoft Azure

The following methods can be used for configuring Microsoft Azure:

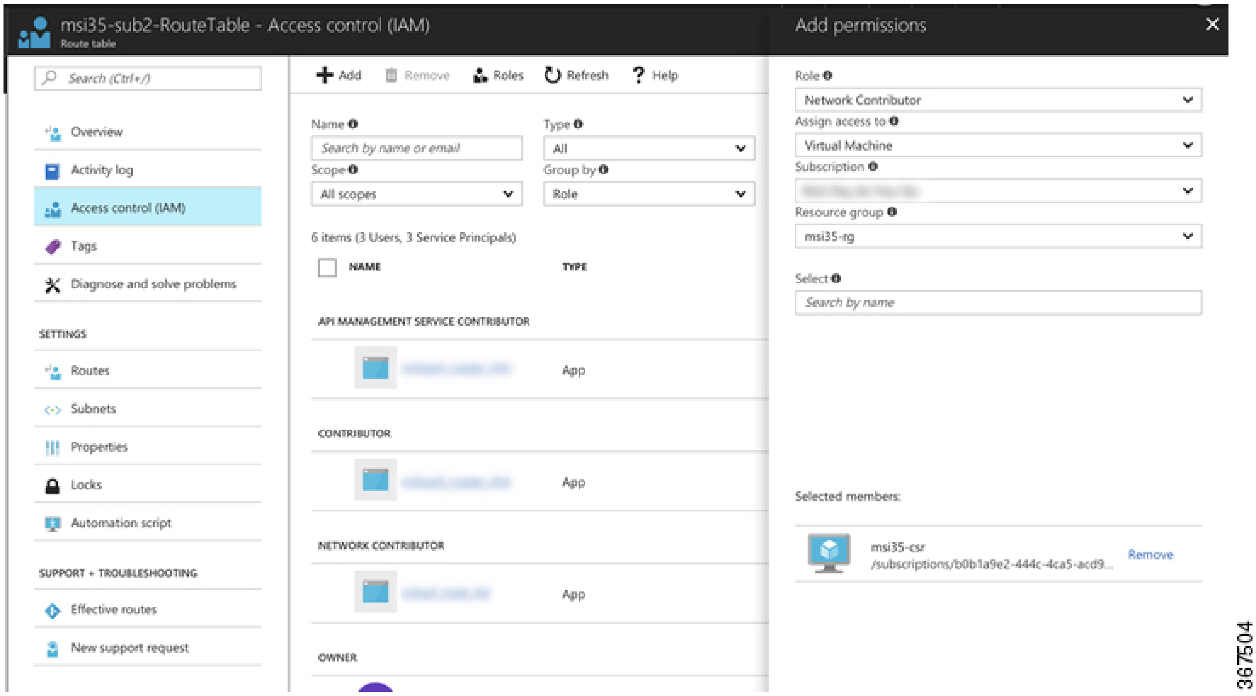
- Microsoft Azure CLI commands
- Powershell commands
- Microsoft Azure Portal https://portal.azure.com/

Configuring IAM for the Route Table

This section explains how you can configure the route table of a subnet to allow a VM to modify the Cisco CSR 1000v route table.

Step 1 To add an application into an existing network, in the **All resources** pane, choose a private side subnet in the left pane; for example, "subnet2-CSR-RouteTable".

Example:



Role: Network Contributor

Assign access to: Virtual Machine

Subscription:

Resource group: msi35-rg

Select: Search by name

Selected members:

msi35-csr /subscriptions/b0b1a9e2-444c-4ca5-acd9... Remove

Step 2 Select "Access control (IAM)" and click **+Add**.

Step 3 In the "Role" textbox, choose **Network contributor**.

Step 4 In the "Assign access to" checkbox, select "Virtual Machine".

Step 5 Enter your subscription ID.

Step 6 Enter the Resource group.

Step 7 A list of your Cisco CSR 1000v appears in the list of machines under "Selected members". Select the VM for which you want to assign permissions to change the route table.

Note If you expect to see a Cisco CSR 1000v under "Selected members" but it is not displayed, then check that you entered a valid subscription ID, resource group and also check that your Cisco CSR 1000v is running.

Step 8 Click **Save**.

Configuring the Network Security Group

If you have a network security group attached to interface NIC0 of the router, configure an inbound and outbound security rule for the network security group. Each rule must allow ports 4789 and 4790 to pass BFD protocol traffic. To set up a security rule on a group, see: https://docs.microsoft.com/en-us/azure/virtual-machines/windows/nsg-quickstart-portal.

Configuring an Application in Azure Active Directory

In HA version 2, you need not explicitly create an application in the Azure Active Directory and grant it permission to access the route tables. An application representing the CSR 1000v is automatically created in the Azure Active Directory via the Managed Service Identity service. However, you can continue to use a manually configured application (as used in HA Version 1). For more information, see Create an Application in a Microsoft Azure Active Directory, on page 81.

Configuring High Availability for the CSR 1000v on Microsoft Azure: Examples

Single Route Table and Two Secondary Routers—Example

In this example, the two peer routers are connected to the same subnet. One of the two routers is active for the specified redundancy node. Since both redundancy nodes are configured in secondary mode, the active router does not change after the recovery of a failed CSR.

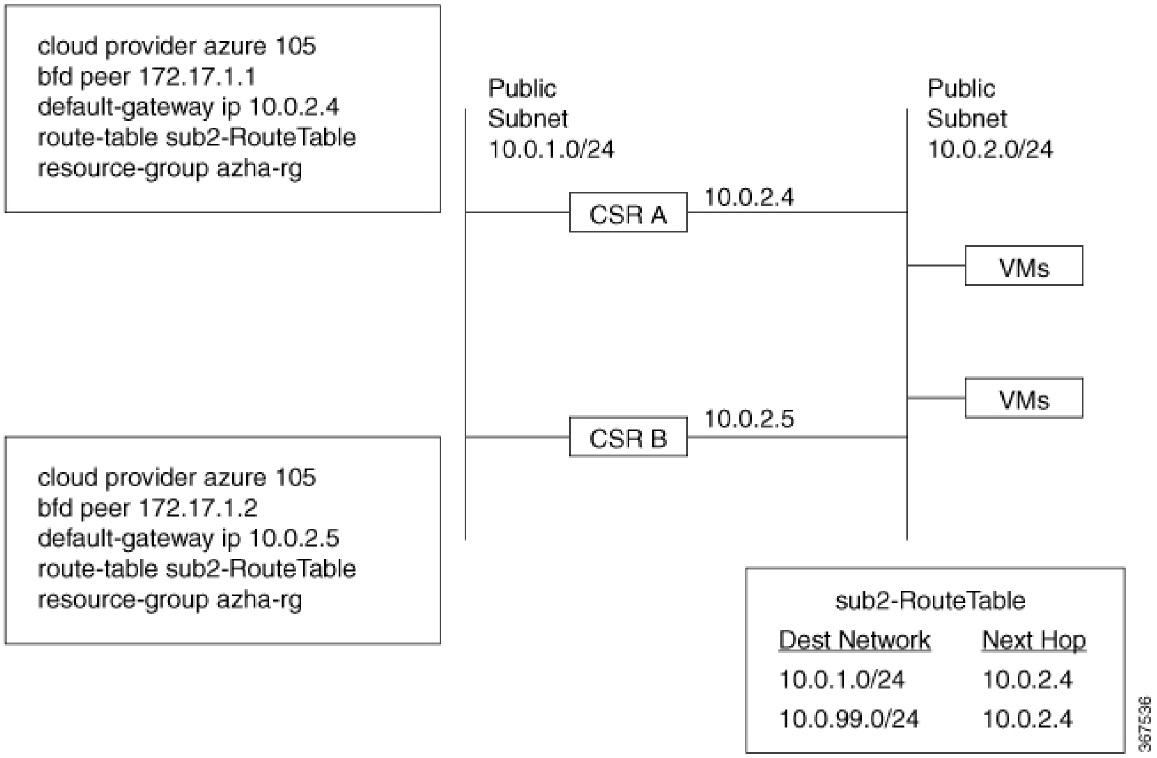


Table 8: CSR A Node Parameters

Parameter	Value
Node index	105
BFD Peer	172.17.1.1
Next Hop IP	10.0.2.4 (IP address of CSR A)
Route Table	sub2-RouteTable
Resource Group	azha-rg

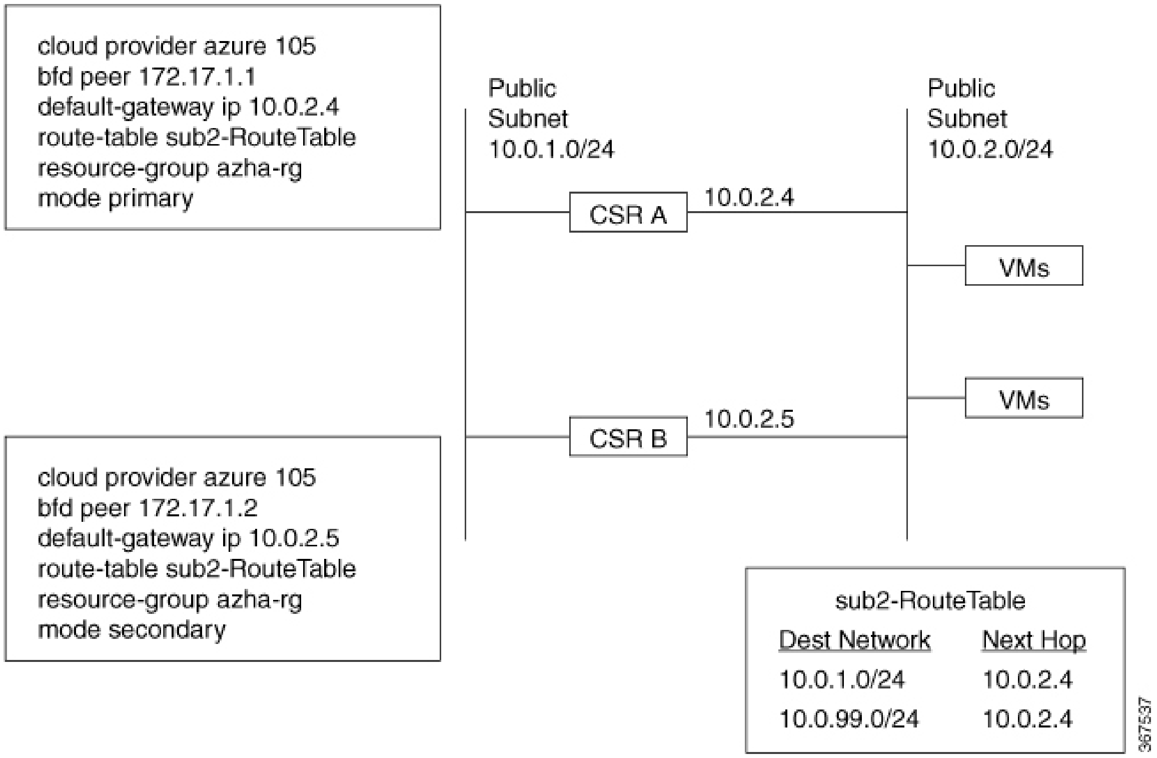
Table 9: CSR B Node Parameters

Parameter	Value
Node Index	105
BFD Peer	172.17.1.1
Next Hop IP	10.0.2.5 (IP address of CSR B)
Route Table	sub2-RouteTable
Resource Group	azha-rg

Note The route name was not configured for either node, which causes all the routes in the sub2-RouteTable to be updated with the Next Hop IP after a peer failure event.

Single Route Table, One CSR Primary

In this example, the two peer routers are connected to the same subnet. One of the two routers is active for the specified redundancy node. CSR A is configured to be the primary router for this subnet. In the absence of a failure of CSR A, it is the active router for all routes in the route table. If CSR A fails, CSR B temporarily becomes the active router. After CSR A recovers from the failure, it becomes the active router.



```
cloud provider azure 105
bfd peer 172.17.1.1
default-gateway ip 10.0.2.4
route-table sub2-RouteTable
resource-group azha-rg
mode primary
```

```
cloud provider azure 105
bfd peer 172.17.1.2
default-gateway ip 10.0.2.5
route-table sub2-RouteTable
resource-group azha-rg
mode secondary
```

sub2-RouteTable

Dest Network	Next Hop
10.0.1.0/24	10.0.2.4
10.0.99.0/24	10.0.2.4

Table 10: CSR A Node Parameters

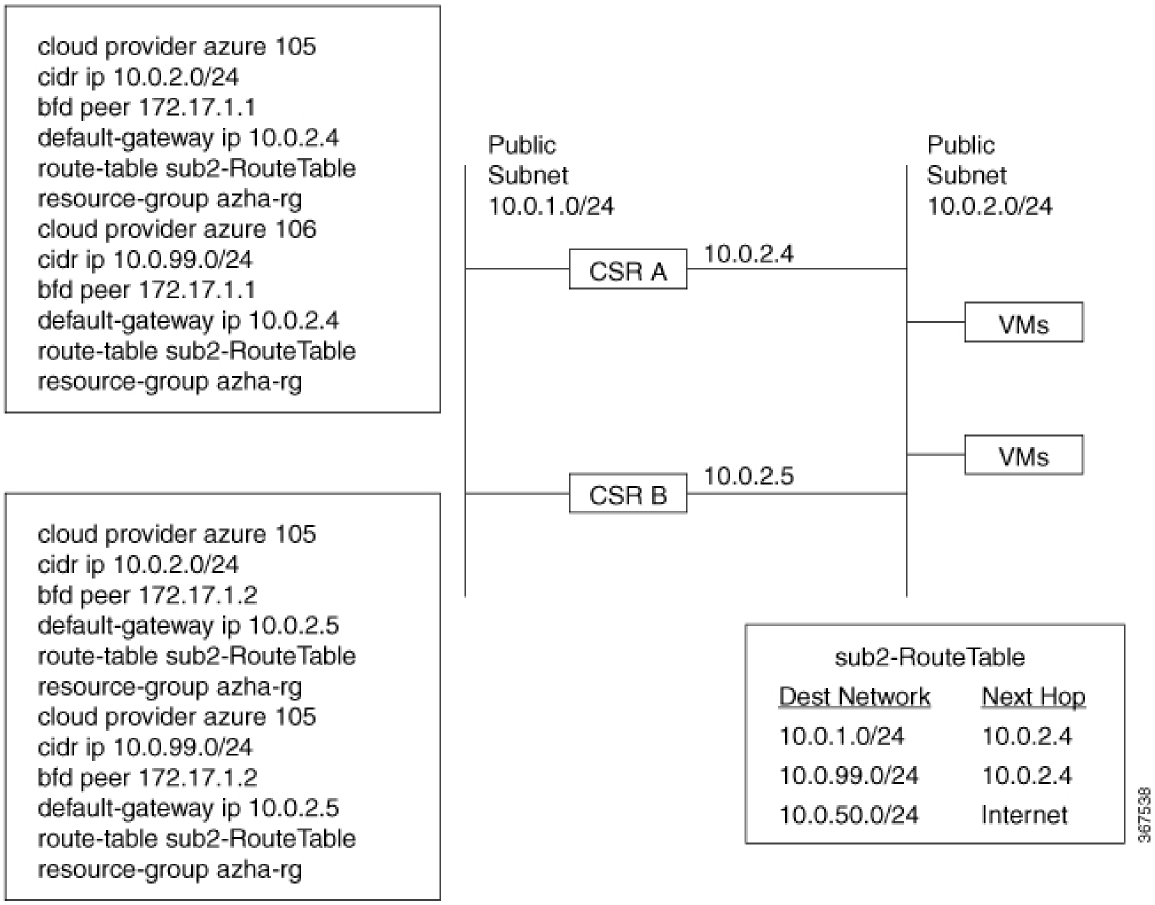
Parameter	Value
Node index	105
BFD Peer	172.17.1.2 (Address of CSR B on the tunnel)
Next Hop IP	10.0.2.4 (IP address of CSR A on the subnet)
Route Table	sub2-RouteTable
Resource Group	azha-rg
Mode	Primary

Table 11: CSR B Node Parameters

Parameter	Value
Node index	105
BFD Peer	172.17.7.1 (Address of CSR B on the tunnel)
Next Hop IP	10.0.2.5 (IP address of CSR A on the subnet)
Route Table	sub2-RouteTable
Resource Group	azha-rg
Mode	Secondary

Single Route Table, Nodes for Individual Routes, Both CSRs Secondary

In this example, the two peer routers are connected to the same subnet. The route table contains two routes which get split between the two CSR 1000vs. All nodes are configured in secondary mode.



```
cloud provider azure 105
cidr ip 10.0.2.0/24
bfd peer 172.17.1.1
default-gateway ip 10.0.2.4
route-table sub2-RouteTable
resource-group azha-rg
cloud provider azure 106
cidr ip 10.0.99.0/24
bfd peer 172.17.1.1
default-gateway ip 10.0.2.4
route-table sub2-RouteTable
resource-group azha-rg
```

```
cloud provider azure 105
cidr ip 10.0.2.0/24
bfd peer 172.17.1.2
default-gateway ip 10.0.2.5
route-table sub2-RouteTable
resource-group azha-rg
cloud provider azure 105
cidr ip 10.0.99.0/24
bfd peer 172.17.1.2
default-gateway ip 10.0.2.5
route-table sub2-RouteTable
resource-group azha-rg
```

sub2-RouteTable

Dest Network	Next Hop
10.0.1.0/24	10.0.2.4
10.0.99.0/24	10.0.2.4
10.0.50.0/24	Internet

Table 12: CSR A Node Parameters (First node)

Parameter	Value
Node index	105
BFD Peer	172.17.7.1 (Address of CSR B on the tunnel)
Next Hop IP	10.0.2.5 (IP address of CSR A on the subnet)
Route	15.0.0.0/8
Route Table	sub2-RouteTable
Resource Group	azha-rg
Mode	Primary

Table 13: CSR A Node Parameters (Second node)

Parameter	Value
Node index	106

Parameter	Value
BFD Peer	172.17.7.1 (Address of CSR B on the tunnel)
Next Hop IP	10.0.2.5 (IP address of CSR A on the subnet)
Route	16.0.0.0/8
Route Table	sub2-RouteTable
Resource Group	azha-rg
Mode	Primary

Table 14: CSR B Node Parameters (First node)

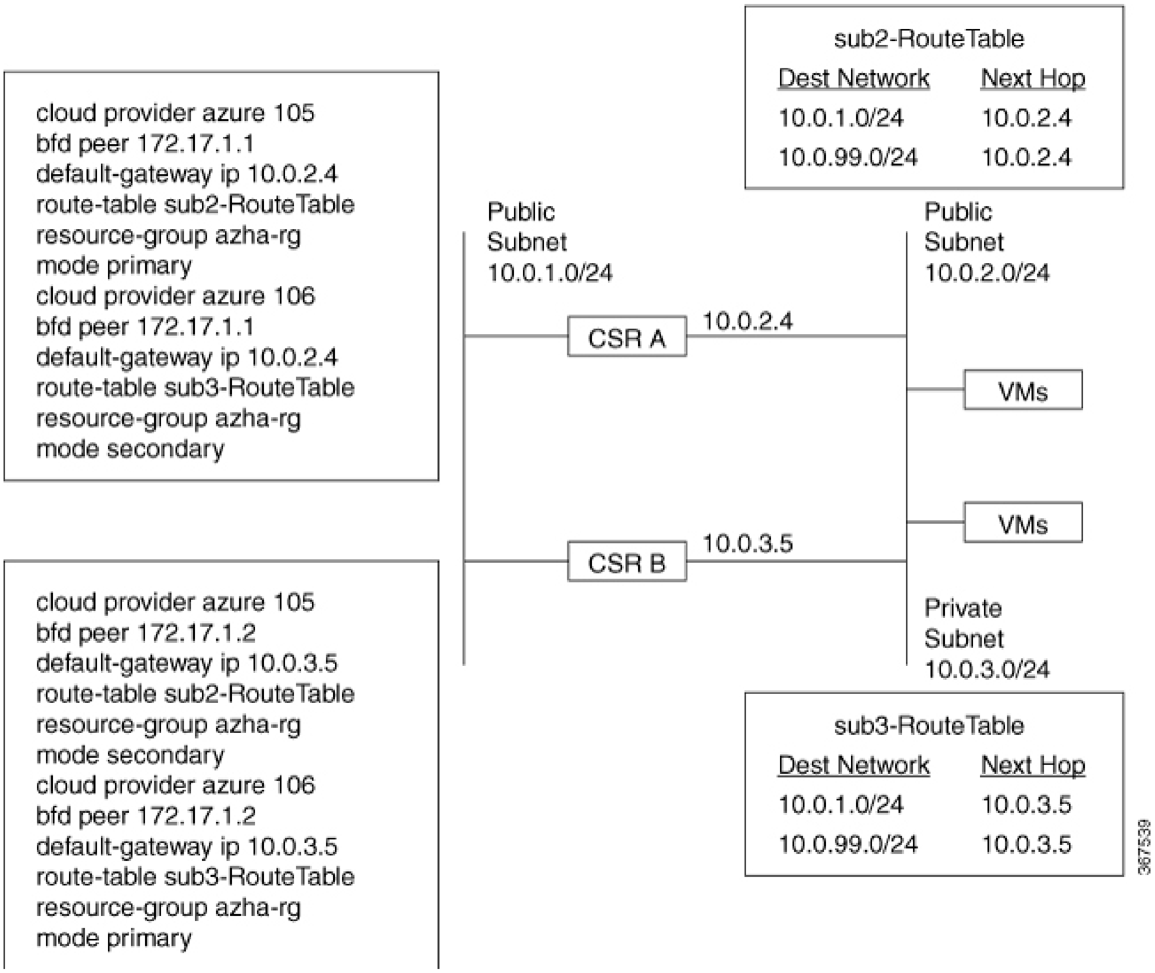
Parameter	Value
Node index	205
BFD Peer	172.17.7.1 (Address of CSR A on the tunnel)
Next Hop IP	10.0.2.5 (IP address of CSR B on the subnet)
Route	15.0.0.0/8
Route Table	sub2-RouteTable
Resource Group	azha-rg
Mode	Secondary

Table 15: CSR B Node Parameters (Second node)

Parameter	Value
Node index	206
BFD Peer	172.17.7.1 (Address of CSR A on the tunnel)
Next Hop IP	10.0.2.5 (IP address of CSR B on the subnet)
Route	15.0.0.0/8
Route Table	sub2-RouteTable
Resource Group	azha-rg
Mode	Secondary

Two Route Tables, One CSR Primary

In this example, the two peer routers are connected to two different subnets: CSR A on subnet A(10.0.2.0/24) and CSR B on subnet B (10.0.3.0/24). CSR A is the primary router on subnet A (with IP address 10.0.2.4). CSR B is the primary router on subnet B (with IP address 10.0.4.5).



```
cloud provider azure 105
bfd peer 172.17.1.1
default-gateway ip 10.0.2.4
route-table sub2-RouteTable
resource-group azha-rg
mode primary
cloud provider azure 106
bfd peer 172.17.1.1
default-gateway ip 10.0.2.4
route-table sub3-RouteTable
resource-group azha-rg
mode secondary
```

```
cloud provider azure 105
bfd peer 172.17.1.2
default-gateway ip 10.0.3.5
route-table sub2-RouteTable
resource-group azha-rg
mode secondary
cloud provider azure 106
bfd peer 172.17.1.2
default-gateway ip 10.0.3.5
route-table sub3-RouteTable
resource-group azha-rg
mode primary
```

sub2-RouteTable

Dest Network	Next Hop
10.0.1.0/24	10.0.2.4
10.0.99.0/24	10.0.2.4

sub3-RouteTable

Dest Network	Next Hop
10.0.1.0/24	10.0.3.5
10.0.99.0/24	10.0.3.5

Public Subnet 10.0.1.0/24
Public Subnet 10.0.2.0/24
CSR A
10.0.2.4
VMs
VMs
CSR B
10.0.3.5
Private Subnet 10.0.3.0/24
367539

Table 16: CSR A Node Parameters (First Node)

Parameter	Value
Node index	105
BFD Peer	172.17.7.2 (Address of CSR B on the tunnel)
Next Hop IP	10.0.2.4 (IP address of CSR A on the subnet)
Route	15.0.0.0/8
Route Table	subA-RouteTable
Resource Group	azha-rg

Parameter	Value
Mode	Primary

Table 17: CSR A Node Parameters (Second Node)

Parameter	Value
Node index	106
BFD Peer	172.17.7.2 (Address of CSR B on the tunnel)
Next Hop IP	10.0.2.4 (IP address of CSR A on the subnet)
Route	15.0.0.0/8
Route Table	subA-RouteTable
Resource Group	azha-rg
Mode	Primary

Table 18: CSR A Node Parameters (Third Node)

Parameter	Value
Node index	205
BFD Peer	172.17.7.2 (Address of CSR B on the tunnel)
Next Hop IP	10.0.2.4 (IP address of CSR A on the subnet)
Route	15.0.0.0/8
Route Table	subB-RouteTable
Resource Group	azha-rg
Mode	Secondary

Table 19: CSR A Node Parameters (Fourth Node)

Parameter	Value
Node index	206
BFD Peer	172.17.7.2 (Address of CSR B on the tunnel)
Next Hop IP	10.0.2.4 (IP address of CSR A on the subnet)
Route	16.0.0.0/8
Route Table	subB-RouteTable
Resource Group	azha-rg

Parameter	Value
Mode	Secondary

Table 20: CSR B Node Parameters (First Node)

Parameter	Value
Node index	105
BFD Peer	172.17.7.1 (Address of CSR A on the tunnel)
Next Hop IP	10.0.4.5 (IP address of CSR B on the subnet)
Route	15.0.0.0/8
Route Table	subB-RouteTable
Resource Group	azha-rg
Mode	Primary

Table 21: CSR B Node Parameters (Second Node)

Parameter	Value
Node index	106
BFD Peer	172.17.7.1 (Address of CSR A on the tunnel)
Next Hop IP	10.0.4.5 (IP address of CSR B on the subnet)
Route	16.0.0.0/8
Route Table	subB-RouteTable
Resource Group	azha-rg
Mode	Primary

Table 22: CSR B Node Parameters (Third Node)

Parameter	Value
Node index	205
BFD Peer	172.17.7.1 (Address of CSR A on the tunnel)
Next Hop IP	10.0.4.5 (IP address of CSR B on the subnet)
Route	15.0.0.0/8
Route Table	subA-RouteTable
Resource Group	azha-rg

Parameter	Value
Mode	Secondary

Table 23: CSR B Node Parameters (Fourth Node)

Parameter	Value
Node index	206
BFD Peer	172.17.1.1 (Address of CSR B on the tunnel)
Next Hop IP	10.0.4.5 (IP address of CSR A on the subnet)
Route	16.0.0.0/8
Route Table	subA-RouteTable
Resource Group	azha-rg
Mode	Secondary

Route Table Entry Types

The route tables in Microsoft Azure support different entry types. The entry type for a route can be one of the following: Virtual network gateway, Internet, or Virtual Appliance. The next hop address identifies a resource in the Azure network.

Routes with an entry type of Virtual network gateway or Internet do not have an explicit IP address for the next hop and are not supported by the High Availability feature.

(Cisco IOS XE Everest 16.6) When you configure High Availability on the Cisco CSR 1000v, all the routes within a route table must have an entry type of Virtual Appliance. These routes require an explicit IP address for the next hop.

(Cisco IOS XE Everest 16.7 or later) When you configure High Availability on the Cisco CSR 1000v, you can specify individual routes to be updated in the case of failure. Ensure that you configure each individual route as having an entry type of Virtual Appliance. If you configure a redundancy node that represents all of the entries in the route table, ensure that all of the routes have an entry type of Virtual Appliance.

Verifying High Availability for the CSR 1000v on Microsoft Azure using Cisco IOS XE Commands

Enter the following command to verify that the tunnel interface is configured and enabled:

```
# show ip interface brief
                       IP-Address      OK? Method Status                Protocol
GigabitEthernet1       192.168.35.20   YES DHCP   up                    up
GigabitEthernet2       192.168.36.12   YES DHCP   up                    up
Tunnel1                172.17.1.1      YES NVRAM  up                    up
VirtualPortGroup0      192.168.35.101  YES NVRAM  up                    up
```

Use the following command to verify that neighboring Cisco CSR 1000v routers have established a BFD session:

```
# show bfd neighbors

IPv4 Sessions
NeighAddr                            LD/RD           RH/RS     State     Int
172.17.1.2                           4097/0          Down      Down      Tu1
```

Use the following command to check the binding between a redundancy node and the BFD peer:

```
# show redundancy cloud provider azure 100
Cloud HA: work_in_progress=FALSE
Provider : AZURE node 100
 State : idle
 BFD peer     = 172.17.1.2
 BFD intf     = Tunnel1
```

Use the following command to verify that the high availability configuration commands that are entered in the preceding sections appear in the running configuration:

```
# show running-configuration int Tunnel1
Building configuration...

Current configuration : 225 bytes
!
interface Tunnel1
 ip address 172.17.1.1 255.255.255.252
 bfd interval 500 min_rx 500 multiplier 3
 tunnel source GigabitEthernet1
 tunnel mode vxlan-gpe ipv4
 tunnel destination 40.117.153.111
 tunnel vxlan vni 10000
```

Save the configuration for future use, with the command: **copy running-configuration startup-configuration**

High availability depends upon several services to be running in guestshell. Enter the following command to verify that IOX is configured and running:

```
# show iox

Virtual Service Global State and Virtualization Limits:
Infrastructure version : 1.7
Total virtual services installed : 0
Total virtual services activated : 0
Machine types supported   : LXC
Machine types disabled    : KVM
Maximum VCPUs per virtual service : 1
Resource virtualization limits:
Name                         Quota     Committed     Available
--------------------------------------------------------------
system CPU (%)                  75             0            75
memory (MB)                   3072             0          3072
bootflash (MB)               20000             0          5745

IOx Infrastructure Summary:
---------------------------
IOx service (CAF)    : Running
IOx service (HA)     : Not Running
IOx service (IOxman) : Running
Libvirtd             : Running
```

Enter the following command to verify that the guest application is defined and running:

```
# show app-hosting list
```

```
App id                           State
------------------------------------------------------
guestshell                       RUNNING
```

If the guestshell state shows as DEPLOYED in the output of the preceding command, then enable the guestshell using the following command:

```
# guestshell enable

Interface will be selected if configured in app-hosting
Please wait for completion
guestshell activated successfully
Current state is: ACTIVATED
guestshell started successfully
Current state is: RUNNING
Guestshell enabled successfully
```

Verifying High Availability for the CSR 1000v on Microsoft Azure using Guestshell Commands

Enter the following command in Cisco IOS XE to enter the guestshell:

```
guestshell
```

Navigate to the home directory and find a subdirectory named "azure":

```
cd ~
ls
```

Verify that the authentication token service is running:

```
[guestshell@guestshell azure]$ systemctl status auth-token
  auth-token.service - Authentication Token service
Loaded: loaded (/etc/systemd/user/auth-token.service; enabled; vendor preset: disabled)
Active: active (running) since Wed 2018-06-13 19:56:00 UTC; 5min ago
Main PID: 36 (python)
CGroup: /system.slice/libvirtd.service/system.slice/auth-token.service
└─36 python
/home/guestshell/.local/lib/python2.7/site-packages/csr_azure_guestshell/TokenMgr...
```

If the service is not running, start the service by executing the following command:

```
[guestshell@guestshell azure]$ sudo systemctl start auth-token
```

Verify the high availability service is running:

```
 [guestshell@guestshell azure]$ sudo systemctl start waagent

 [guestshell@guestshell azure]$ systemctl status azure-ha
azure-ha.service - Azure High Availability service
Loaded: loaded (/etc/systemd/user/azure-ha.service; enabled; vendor preset: disabled) Active:
 active (running) since Wed 2018-06-13 19:56:00 UTC; 7min ago
Main PID: 29 (python)
CGroup: /system.slice/libvirtd.service/system.slice/azure-ha.service
├─ 29 python
/home/guestshell/.local/lib/python2.7/site-packages/csr_azure_ha/server/ha_serve...
└─103 python
/home/guestshell/.local/lib/python2.7/site-packages/csr_azure_ha/server/ha_serve...
```

If the service is not running, start it with the command:

```
[guestshell@guestshell azure]$ sudo systemctl start azure-ha
```

Check if the Python path has been set

```
[guestshell@guestshell azure]$ which show_node.py
~/.local/lib/python2.7/site-packages/csr_azure_ha/client_api/show_node.py If this path is
not found, run this command in guestshell [guestshell@guestshell azure]$ source ~/.bashrc
```

Use the show_node script to display a node you have configured:

```
[guestshell@guestshell HA]$ show_node.py -i 1
Redundancy node configuration: index 1
routeTableName azha1-sub2-RouteTable

route  15.0.0.0/8 nextHop 192.168.35.102
resourceGroup azha-rg
subscriptionId b0b1a9e2-4444-4ca5-acd9-bebd1e6873eb cloud azure
```

Simulate a peer failure for this redundancy node. This is a diagnostic tool to verify that the Cisco CSR 1000v can read and write the route table specified by the redundancy node. It does not change the entry in the route table. It writes a verbose log file to the directory `~/azure/HA/events`. Examine this log file to verify the operation was successful.

```
[guestshell@guestshell events]$ node_event.py -i 1 -e verify
[guestshell@guestshell events]$ pwd
/home/guestshell/azure/HA/events
[guestshell@guestshell events]$ ls
event.2018-06-13 20:10:21.093942
```

Open the event file. It is a debug log of the attempt to read and update the route described by the redundancy node. If everything worked, the last line of the file will be Event handling completed.

Use the show_node script to display a node that you previously configured:

```
[guestshell@guestshell HA]$ show_node.py -i 1
Redundancy node configuration:
index  1
routeTableName  azha1-sub2-RouteTable
route  15.0.0.0/8
nextHop  192.168.35.102
resourceGroup  azha-rg
subscriptionId  b0b1a9e2-4444-4ca5-acd9-bebd1e6873eb
cloud   azure
```

If the service is not running, start it with the following command:

```
[guestshell@guestshell azure]$ sudo systemctl start auth-token
```

Use the following command to verify that the high availability service is running:

```
guestshell@guestshell azure]$ systemctl status azure-ha
● azure-ha.service - Azure High Availability service
   Loaded: loaded (/etc/systemd/user/azure-ha.service; enabled; vendor preset: disabled)
   Active: active (running) since Wed 2018-06-13 19:56:00 UTC; 7min ago
 Main PID: 29 (python)
   CGroup: /system.slice/libvirtd.service/system.slice/azure-ha.service
           ├─ 29 python
/home/guestshell/.local/lib/python2.7/site-packages/csr_azure_ha/server/ha_serve...
           └─103 python
/home/guestshell/.local/lib/python2.7/site-packages/csr_azure_ha/server/ha_serve...
```

If the service is not running, start it with the following command:

```
[guestshell@guestshell azure]$ sudo systemctl start azure-ha
```

Check if the Python path has been set using the following command:

```
[guestshell@guestshell azure]$ which show_node.py
~/.local/lib/python2.7/site-packages/csr_azure_ha/client_api/show_node.py
```

If this path is not found, run these commands in the guestshell:

```
[guestshell@guestshell azure]$ source ~/.bashrc
```

```
[guestshell@guestshell azure]$ sudo systemctl start waagent
```

Verify that the authentication token service is running:

```
[guestshell@guestshell azure]$ systemctl status auth-token
● auth-token.service - Authentication Token service
   Loaded: loaded (/etc/systemd/user/auth-token.service; enabled; vendor preset: disabled)
   Active: active (running) since Wed 2018-06-13 19:56:00 UTC; 5min ago
 Main PID: 36 (python)
   CGroup: /system.slice/libvirtd.service/system.slice/auth-token.service
           └─36 python
/home/guestshell/.local/lib/python2.7/site-packages/csr_azure_guestshell/TokenMgr...
```

If the service is not running, start it with the command

```
[guestshell@guestshell azure]$ sudo systemctl start auth-token
```

Verify the high availability service is running

```
[guestshell@guestshell azure]$ systemctl status azure-ha
 azure-ha.service - Azure High Availability service
  Loaded: loaded (/etc/systemd/user/azure-ha.service; enabled; vendor preset: disabled)
  Active: active (running) since Wed 2018-06-13 19:56:00 UTC; 7min ago
Main PID: 29 (python)
  CGroup: /system.slice/libvirtd.service/system.slice/azure-ha.service
          ├─ 29 python
/home/guestshell/.local/lib/python2.7/site-packages/csr_azure_ha/server/ha_serve...
          └─103 python
/home/guestshell/.local/lib/python2.7/site-packages/csr_azure_ha/server/ha_serve...
```

If the service is not running, start it with the command:

```
[guestshell@guestshell azure]$ sudo systemctl start azure-ha
```

Check if the Python path has been set

```
[guestshell@guestshell azure]$ which show_node.py
~/.local/lib/python2.7/site-packages/csr_azure_ha/client_api/show_node.py If this path is
not found, run this command in guestshell [guestshell@guestshell azure]$ source ~/.bashrc
```

Use the show_node script to display a node you have configured:

```
[guestshell@guestshell HA]$ show_node.py -i 1
Redundancy node configuration: index 1
routeTableName azha1-sub2-RouteTable

route  15.0.0.0/8 nextHop 192.168.35.102
resourceGroup azha-rg
subscriptionId b0b1a9e2-4444-4ca5-acd9-bebd1e6873eb cloud azure
```

Simulate a peer failure for this redundancy node. This is a diagnostic tool to verify the CSR is capable of reading and writing the route table specified by the redundancy node. It does not change the entry in the route table. It will write a verbose log file to the directory ~/azure/HA/events. Examine this log file to verify the operation was successful.

```
[guestshell@guestshell events]$ node_event.py -i 1 -e verify
[guestshell@guestshell events]$ pwd
/home/guestshell/azure/HA/events
```

```
[guestshell@guestshell events]$ ls
 event.2018-06-13 20:10:21.093942
```

Change into the azure directory and find subdirectories named HA, tools, and waagent.

```
cd azure
ls
```

Open the event file using **vi**, or view the event file using **cat**. The file is a debug log of the attempts made to read and update the route described in the redundancy node. If everything works, the last line of the file is Event handling completed.

Verify the Linux Azure agent is running:

```
[guestshell@guestshell azure]$ systemctl status waagent
waagent.service - Azure Linux Agent
   Loaded: loaded (/usr/lib/systemd/system/waagent.service; enabled; vendor preset: disabled)

   Active: active (running) since Wed 2018-06-13 19:56:00 UTC; 5min ago
 Main PID: 28 (python)
   CGroup: /system.slice/libvirtd.service/system.slice/waagent.service
           ├─28 /usr/bin/python -u /usr/sbin/waagent -daemon
           └─92 python -u /usr/sbin/waagent -run-exthandlers
```

If the service is not running, start it with the following command:

```
[guestshell@guestshell azure]$ sudo systemctl start atd
```

Check if the Python path has been set:

```
[guestshell@guestshell azure]$ which show_node.py
~/.local/lib/python2.7/site-packages/csr_azure_ha/client_api/show_node.py
```

If this path is not found, run the following command in the guestshell

```
[guestshell@guestshell azure]$ source ~/.bashrc
```

Use the show_node script to display a node you have previously configured:

```
[guestshell@guestshell HA]$ show_node.py -i 1
Redundancy node configuration:
index  1
routeTableName  azha1-sub2-RouteTable
route  15.0.0.0/8
nextHop  192.168.35.102
resourceGroup  azha-rg
subscriptionId  b0b1a9e2-4444-4ca5-acd9-bebd1e6873eb
cloud   azure
```

Simulate a peer failure for this redundancy node. This is a diagnostic tool that verifies that the Cisco CSR 1000v is capable of reading and writing the route table specified by the redundancy node. It does not change the entry in the route table. It writes a verbose log file to the directory `~/azure/HA/events`. Examine this log file to verify the operation was successful.

```
[guestshell@guestshell events]$ node_event.py -i 1 -e verify
[guestshell@guestshell events]$ pwd
/home/guestshell/azure/HA/events
[guestshell@guestshell events]$ ls
event.2018-06-13 20:10:21.093942
```

Open the event file. It is a debug log of the attempt to read and update the route described by the redundancy node. If everything worked, the last line of the file contain the message: Event handling completed.

Advanced Programming for High Availability on Microsoft Azure

Cisco CSR 1000v routers use the BFD protocol to detect failure of their peer router and trigger a peerFail event. However, the BFD protocol does not detect some failures. For example, if the Gigabit Ethernet interface on CSR A connected to the back-end network goes down, or stops forwarding traffic.

This section describes a few advanced programming techniques that you can use to customize your control of failover and reversion events.

Trigger Cisco CSR 1000v Failover Using EEM

Define an Event Manager applet to detect the transition of an interface state. The applet triggers a failover of the Cisco CSR 1000v.

Before You Begin

Enter configuration mode in the Cisco IOS XE CLI of the Cisco CSR 1000v.

Procedure

```
event manager applet Interface GigabitEthernet2
event syslog pattern Interface GigabitEthernet2, changed state to down
action 1 cli command enable
action 2 cli command guestshell run node_event.py -i 10 -e peerFail
exit
exit
```

Trigger Cisco CSR 1000v Reversion After Recovery

Define an Event Manager applet to detect when the router recovers and re-establishes a BFD session with its peer router. The applet reverts a route entry that previously changed, by pointing the route entry back to this router.

Before You Begin

Enter configuration mode in the Cisco IOS XE CLI of the Cisco CSR 1000v.

Procedure

```
event manager applet "bfd_session_up
event syslog pattern .*BFD_SESS_UP.*
action 1 cli command enable
action 2 cli command "guestshell run node_event.py -i 10 -e revert"
exit
```

User-Defined Triggers

You can write your own Python script to recognize an event or condition and call the node_event script. You can enter the command manually at the guestshell prompt.

Example

To process the redundancy node and update an associated route table entry, run the `node_event` Python script. `node_event.py -i` *`node_index`* `-e peerFail` processes the redundancy node and updates the associated route table entry.

Configuring High Availability Version 1 for the CSR 1000v on Microsoft Azure

Create an Application in a Microsoft Azure Active Directory

This section explains how to create an application in a Microsoft Azure Active Directory with permissions to access Microsoft Azure Resource Manager APIs. These configuration steps use the classic portal.

Note When entering the API Access Key using High Availability version 1, the key must be URL encoded.

When entering the API Access Key using High Availability version 2, the key must be URL unencoded.

Refer to Create an Authentication Key for the Application, on page 82.

Step 1 Go to the portal for Microsoft Azure: https://portal.azure.com.

Step 2 Choose your account name and sign in using your Microsoft Azure password.

Step 3 Click **Azure Active Directory** in the left navigation pane and select an active directory in the main pane. Click **Switch directory** at the top of the pane to select the active directory.

Step 4 Verify that you are authorized to create a new application. Refer to the following Microsoft Azure documentation for creating an application in the Azure Active Directory: Use portal to create an Azure Active Directory application and service principal that can access resources.

Step 5 To view applications, select **App registrations**.

Step 6 To create a new application, select **New application registration**.

Step 7 Specify the name of the application and ensure that "Web App / API" is selected as the Application type.

Step 8 Specify the Sign-on URL. Use a name for the sign-on URL which is in the URI format, but it does not have to be reachable. (Note that the APP-ID URI is not the App ID.) You can use a string in the following format: "http://<your_directory_domain_name>/<app_name>". For example, if your application name is "myapp" and the domain name of your directory is "\mydir.onmicrosoft.com", use the following example as the sign-on URL:

Example:

```
http://mydir.onmicrosoft.com/myapp
```

Step 9 Click **Create**.

Step 10 Click the checkmark symbol at the bottom right of the dialog box.

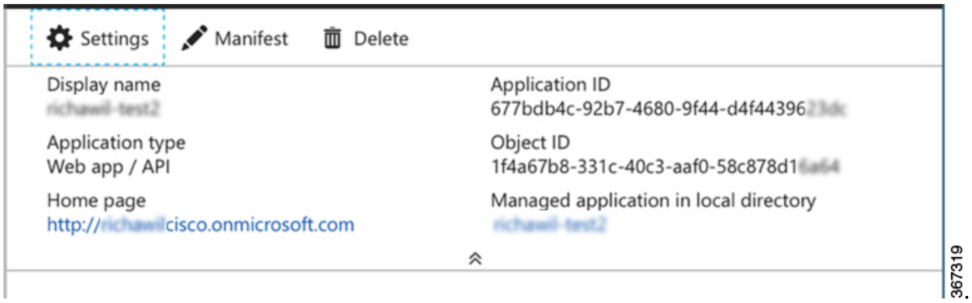
Step 11 Under the name of the application that you have added, click "CONFIGURE".

What to do next

Go to Obtain the Application ID and Tenant ID, on page 82.

Obtain the Application ID and Tenant ID

Step 1 After you create the application, the registered app should appear on the screen as shown below.



Display name
[redacted]

Application type
Web app / API

Home page
http://[redacted]cisco.onmicrosoft.com

Application ID
677bdb4c-92b7-4680-9f44-d4f44396[redacted]

Object ID
1f4a67b8-331c-40c3-aaf0-58c878d1[redacted]

Managed application in local directory
[redacted]

Also refer to step 2 in section "Get application ID and authentication key" in the Microsoft Documentation: Use portal to create an Azure Active Directory application and service principal that can access resources

Step 2 Take a note of the "Application ID".

Step 3 Select **Azure Active Directory**.

Step 4 Select **Properties**.

Step 5 Take a note of the value of the `Directory ID` field. This is your tenant ID.

What to do next

Go to Create an Authentication Key for the Application, on page 82.

Create an Authentication Key for the Application

Create an authentication key for the application by performing the following steps:

Step 1 Select **Azure Active Directory**.

Step 2 Select **App registrations**.

Step 3 Select the application that you previously created in Obtain the Application ID and Tenant ID, on page 82.

Step 4 Select **Settings**.

Step 5 To create a key for API access, select **Keys** and choose a value for **Duration**—the length of time until the key becomes invalid.

Step 6 Make a note of the API key from the Value field.

Step 7 For HA Version 1, convert the API key to URL encoded format. (To find a suitable conversion tool, enter URL encoder into an internet search engine.) Having a URL encoded API key prevents issues later;for example, when the API key is used in step 10 of Configure Failure Detection for the Cisco CSR 1000v on Microsoft Azure, on page 89.

Note When entering the API Access Key using High Availability version 1, the key must be URL encoded.

When entering the API Access Key using High Availability version 2, the key must be URL unencoded.

Note Store the API key carefully as it cannot be retrieved later.

Example:

API Key (unencoded)

```
5yOhH593dtD/O8gzAlWgulrkWz5dH02d2STk3LDbI4c=
```

API Key (URL encoded)

```
5yOhH593dtD%2FO8gzAlWgulrkWz5dH02d2STk3LDbI4c%3D
```

Example:

(Python 3.x example)

```
import urllib.parse
s = HOC6OhG5aJ4phxMSFVRtz59qWgIuD4C2zXd7Q7v5EFk=  # API Key (unencoded)
urllib.parse.quote_plus(s)
HOC6OhG5aJ4phxMSFVRtz59qWgIuD4C2zXd7Q7v5EFk%3D    # API Key (URL encoded)
```

What to do next

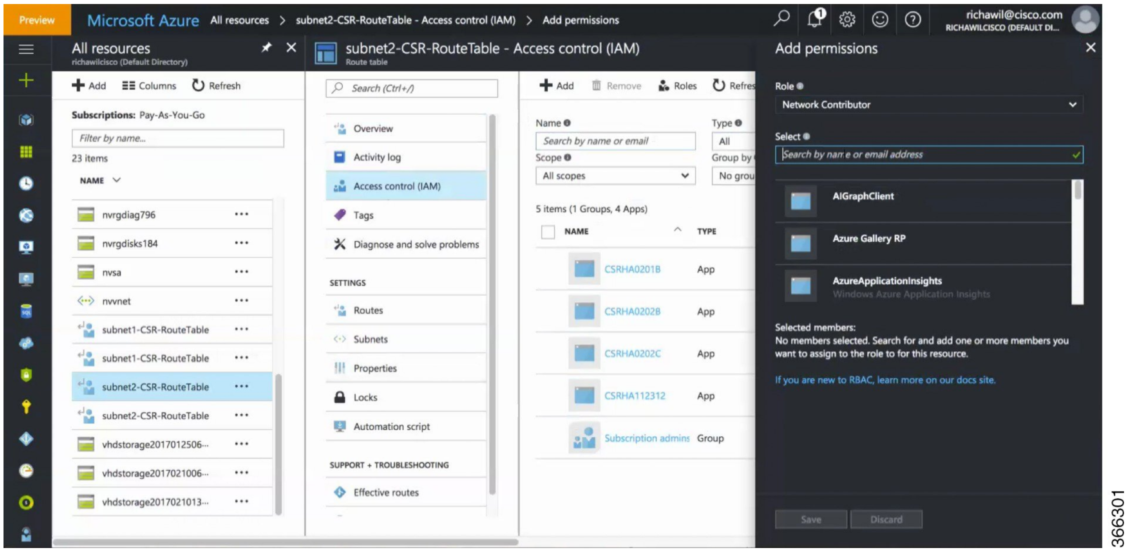
Go to Add an Application under Access Control to a Route Table , on page 83.

Add an Application under Access Control to a Route Table

This section explains how to configure the route table of a subnet to allow the application (for example, "CSRHA2") to modify the CSR 1000v route table.

Step 1 To add an application into an existing network, in the **All resources** pane, choose a private side subnet in the left pane;for example, "subnet2-CSR-RouteTable".

Example:



Role: Network Contributor

5 items (1 Groups, 4 Apps)

NAME	TYPE
CSRHA0201B	App
CSRHA0202B	App
CSRHA0202C	App
CSRHA112312	App
Subscription admins	Group

Selected members:
No members selected. Search for and add one or more members you want to assign to the role to for this resource.

If you are new to RBAC, learn more on our docs site.

Step 2 In the middle blade, select "Access control (IAM)" and in the right blade, click **+Add**.

Step 3 In the "Role" textbox, choose **Network contributor**and in the "Assign access to" checkbox, select "Azure AD user, group, or application".

Step 4 In the "Select" textbox, enter the name of the application.

Step 5 Click **Save**.

Step 6 After completing the procedures in this document up to this point, ensure that you have saved the values of the following IDs and keys:

- Tenant ID (For example: `227b0f8f-684d-48fa-9803-c08138b77ae9`)
- App ID (For example: `80848f32-8120-43fb-ba65-3d5aa596cd0c`).
- API key (For example: `5yOhH593dtD%2FO8gzAlWgulrkWz5dH02d2STk3LDbI4c%3D`).

The application is now authorized to update the route table.

What to do next

Next, go to either Configure a Trustpool, on page 86 or Configure a Trustpoint, on page 84 to establish a secure connection between the Microsoft Azure Management API and the Cisco CSR 1000v.

Configuring High Availability SSL

To establish a secure connection between the Microsoft Azure Management API and the Cisco CSR 1000v perform one of the following two procedures:

Configure a Trustpoint

To establish a secure connection between the Microsoft Azure Management API and Cisco CSR 1000v, follow this procedure on how to configure an individual trustpoint.

Perform the steps in this procedure or perform the steps in Configure a Trustpool, on page 86 to establish a secure connection between the Microsoft Azure Management API and the Cisco CSR 1000v.

Note Information in the steps below, for configuring a trustpoint, has changed compared to what was shown in previous versions of this document. The Microsoft certificates mentioned below need to be used from 7/21/2017 onwards to avoid interruption of the service. At the time of writing further information on certificate changes is found at: https://blogs.technet.microsoft.com/kv/2017/04/20/azure-tls-certificates-changes/?WT.mc_id=azurebg_email_Trans_33716_1407_SSL_Intermediate_Cert_Change. See also: https://www.microsoft.com/pki/mscorp/cps/default.htm.

Before you begin

Each Cisco CSR 1000v VM is assumed to be configured and running in Microsoft Azure.

Step 3 in the procedure below assumes that you have `openssl` available in the OS that you are using (e.g Linux). For a Windows operating system, which does not include OpenSSL tools, to generate a certificate you can choose from one of the following methods.

- Using `makecert.exe`

- Using IIS Manager - `inetmgr.exe`
- Installing and running OpenSSL tools. For example, see https://wiki.openssl.org/index.php/Binaries

---

Step 1 Go to the PKI Repository for Microsoft, which shows the active certificates used by Microsoft authentication servers.

Step 2 Choose and download the .crt file for a certificate; for example, Microsoft IT TLS CA 2.crt and save this file with a CA certificate name such as msit_tls_ca.crt.

Step 3 Convert the .crt file into a .pem file as follows:

openssl x509 **-in** *crtfile* **-inform der -outform pem -out** *pemfile*

Example:

```
openssl x509 -in msit_tls_ca.crt -inform der -outform pem -out msit_tls_ca.pem
```

Open the msit_tls_ca.pem file using a text editor and find the very long sequence of characters between the lines: -----BEGIN CERTIFICATE----- and -----END CERTIFICATE----- . Save this sequence of characters, which is the certificate, in a file; for example, cert.txt. (This is used later in step 11.)

Step 4 **configure terminal**

Enters configuration mode.

Step 5 **crypto pki trustpoint** *trustpoint-name*.

Enters ca-trustpoint configuration mode. This declares the certificate authority for the Cisco CSR 1000v. *trustpoint-name*- the name of the certificate authority (CA).

Example:

```
(config)# crypto pki trustpoint MicrosoftSSL
```

Step 6 Specify manual cut-and-paste certificate enrollment using the following command: **enrollment terminal** .

Example:

```
(ca-trustpoint)# enrollment terminal
```

Step 7 Specify the subject name in the certificate request using the command: **subject-name cn=** *crtfile*.

crtfile- the CA Certificate Name from Step 3.

Example:

```
(ca-trustpoint)# subject-name cn=msit_tls_ca.crt
```

Step 8 **exit**

Exit ca-trustpoint configuration mode.

Step 9 **crypto pki authenticate** *trustpoint-name*

trustpoint-name- the name of the certificate authority (CA) in step 6.

Step 10 Paste in the certificate text that you previously extracted from the PEM file in step 4. Enter a blank line (Return key). Enter the word `quit` and press the Return key.

You are prompted to paste in the certificate. This certificate text is the text that you had previously saved;for example, in text file "cert.txt".

Step 11 Enter `yes` to accept the certificate.

Step 12 **exit**

Exit configuration mode.

Example:

(config)# exit

Configure a Trustpool

The following procedure provides instructions on configuring a trustpool to establish a secure connection between the Microsoft Azure Management API and the Cisco CSR 1000v. A trustpool is a list of certificate authorities (CAs) that has been approved by Cisco as being trustworthy. Perform the steps in this procedure or alternatively, go to Configure a Trustpoint, on page 84 to establish a secure connection between the Microsoft Azure Management API and the Cisco CSR 1000v.

Before you begin

Each Cisco CSR 1000v VM is assumed to be configured and running in Microsoft Azure.

Step 1 Use the `ssh` command to gain access to the Cisco IOS XE CLI on the Cisco CSR 1000v and enter commands in the following steps.

Step 2 **crypto pki trustpool import** http://www.cisco.com/security/pki/trs/ios.p7b

This command imports certificate authorities from the specified URL.

Example:

```
crypto pki trustpool import url http://www.cisco.com/security/pki/trs/ios.p7b

Reading file from http://www.cisco.com/security/pki/trs/ios.p7b Loading
http://www.cisco.com/security/pki/trs/ios.p7b !!!!
% PEM files import succeeded.
```

Step 3 **show crypto pki trustpool**

Shows the trustpool certificates in a verbose format.

Configure a Name Server

Configure a name server for the Cisco CSR 1000v. See the IP Addressing: DNS Configuration Guide.

Configure a Tunnel Between Cisco CSR 1000v Routers

This section describes how to configure a tunnel between Cisco CSR 1000v routers and enable Bi-directional Forwarding Detection (BFD) and a routing protocol (EIGRP or BGP) on the tunnel between the routers for peer failure detection. To authenticate and encrypt IP traffic as it traverses a network, choose between using either an IPSEC tunnel (step 1) or VxLAN GPE tunnel (step 2).

Step 1 To configure an IPSEC tunnel, enter the configuration mode commands to give the following configuration. (Use either an IPSEC tunnel (step1) or VxLAN tunnel (step 2)). The command **crypto isakmp policy** 1 defines an IKE policy, with a high priority (1), and enters config-isakmp configuration mode.

Example:

```
crypto isakmp policy 1
 encr aes 256
 authentication pre-share
crypto isakmp key cisco address 0.0.0.0
!
!
crypto ipsec transform-set uni-perf esp-aes 256 esp-sha-hmac
 mode tunnel
!
!
crypto ipsec profile vti-1
 set security-association lifetime kilobytes disable
 set security-association lifetime seconds 86400
 set transform-set uni-perf
 set pfs group2
!
!
interface Tunnel1
 ip address 192.168.101.1 255.255.255.252
 load-interval 30
 tunnel source GigabitEthernet1
 tunnel mode ipsec ipv4
 tunnel destination 23.96.91.169
 tunnel protection ipsec profile vti-1
 bfd interval 500 min_rx 500 multiplier 3
```

Note We recommend that you specify a BFD interval of 500 ms or more. You can increase the BFD timers to account for the varying latency in different regions.

Step 2 To create a VxLAN GPE tunnel, enter configuration mode commands to give the following configuration. (Use either an IPSEC tunnel (step1) or VxLAN (step 2)).

For further information on configuring a VxLAN GPE tunnel, see: http://www.cisco.com/c/en/us/td/docs/ios-xml/ios/cether/configuration/xe-16/ce-xe-16-book/vxlan-gpe-tunnel.html

The tunnel destination address must be the public IP address of the corresponding Cisco CSR 1000v. For the tunnel IP address, use any unique IP address. However, the tunnel end points of each redundant Cisco CSR 1000v must be in the same subnet.

Note To allow VxLAN to pass traffic through the tunnel, you must ensure that UDP ports 4789 and 4790 are allowed in a Microsoft Azure Network Security Group(NSG). For further information on NSGs, see: https://docs.microsoft.com/en-us/azure/virtual-network/virtual-networks-nsg.

Note We recommend that you specify a BFD interval of 500 ms or more. You can increase the BFD timers to account for the varying latency in different regions.

Example:

```
interface Tunnel100
 ip address 192.168.101.1 255.255.255.0
 shutdown
 bfd interval 500 min_rx 500 multiplier 3
 tunnel source GigabitEthernet1
 tunnel mode vxlan-gpe ipv4
 tunnel destination 40.114.93.164
 tunnel vxlan vni 10000
```

What to do next

After configuring either a VxLAN or IPSEC tunnel, you can configure either EIGRP, BGP or OSPF over the tunnel interface. The following section explains how to configure EIGRP: Configuring EIGRP over Virtual Tunnel Interfaces, on page 57.

Configuring EIGRP over Virtual Tunnel Interfaces

Configure EIGRP over the virtual tunnel interfaces using the following steps.

Note Other than using EIGRP, which is the protocol used in the following steps, you also have the option of using either BGP, or OSPF.

Before you begin

Configure either a VxLAN or IPsec tunnel between the Cisco CSR 1000v routers.

SUMMARY STEPS

1. **router eigrp** *as-number*
2. **network** *ip-address subnet-mask*
3. **bfd all-interfaces**
4. **end**
5. **show bfd neighbors**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**router eigrp** *as-number* **Example:** `Device(config)# router eigrp 1`	Enables the EIGRP routing process and enters router configuration mode.
Step 2	**network** *ip-address subnet-mask* **Example:** `network 192.168.101.0 0.0.0.255`	Share the network of the tunnel using EIGRP.
Step 3	**bfd all-interfaces** **Example:** `Device(config-router)# bfd all-interfaces`	Enables BFD globally on all interfaces that are associated with the EIGRP routing process.

	Command or Action	Purpose

Step 4	**end**
	Example:
	`Device(config-router)# end`
		Exits router configuration mode and returns the router to privileged EXEC mode.

Step 5	**show bfd neighbors**
	Example:
	`Device# show bfd neighbors`
	
	`IPv4 Sessions`
	`NeighAddr      LD/RD        RH/RS     State   Int`
	`192.168.101.2  4097/4097    Up        Up      Tu100`
		Verifies that the BFD neighbor is active and displays the routing protocols that BFD has registered.

Configure Failure Detection for the Cisco CSR 1000v on Microsoft Azure

Follow the steps in this procedure to configure failure detection for the Cisco CSR 1000v and to specify resource identifiers, such as "azure_subscription_id" to Microsoft Azure. Configure a Cisco CSR 1000v to monitor Bidirectional Forwarding Detection (BFD) events using the following steps:

Step 1 # **redundancy**

Enters redundancy mode. Enter commands in configuration mode, to give a configuration similar to the one shown in the above example.

Step 2 # **cloud provider azure** *node-id*

node-id is a numeric value (in the range 1-255) that identifies an instance of a routing table to be updated in case of a detected failure. A single Cisco CSR 1000v can be used to update multiple routing tables by creating multiple nodes.

Example:

```
# cloud provider azure 100
```

Step 3 # **bfd peer** *peer-ip-address*

peer-ip-address is the tunnel IP address of the neighboring Cisco CSR 1000v.

Example:

```
# bfd peer 192.168.101.2
```

Step 4 # **default-gateway ip addr** *ip-addr*

ip-addr is the IP address of the Cisco CSR 1000v on the private subnet.

Example:

```
# default-gateway ip addr 10.60.1.6
```

Step 5 # **route-table** *route-table-name*

route-table-name is the route table used in Add an Application under Access Control to a Route Table , on page 83.

Example:

```
# route-table HaEastRouteTable
```

Step 6 # **resource-group** *resource_group_name* *resource_group_name* is the name of the resource group containing the the subnet route table to be updated in the case of a CSR failure. For more information, see https://azure.microsoft.com/en-us/documentation/articles/resource-group-overview.

Note This resource group may not be the same resource group that contains other Microsoft Azure resources.

Example:

```
# resource group comapnynameusawest
```

Step 7 # **subscription-id** *azure_subscription_id*

azure_subscription_id is the Microsoft Azure subscription ID, which identifies the customer who is responsible for paying the cost of using these Microsoft Azure cloud services.

Example:

```
# subscription-id ab2fe6b2-c2bd-44
```

Step 8 # **tenant-id** *active_directory_tenant_id*

active_directory_tenant_id is the Tenant ID saved in Add an Application under Access Control to a Route Table , on page 83).

Example:

```
# tenant-id 227b0f8f-684d-48fa-9803-c08138b77ae9
```

Step 9 # **app-id** *application_id*

application_id is the App ID .

Example:

```
80848f32-8120-43fb-ba65-3d5aa596cd0c
```

Step 10 # **app-key** *api-key*

api-key is the (URL encoded) API key that you saved in Add an Application under Access Control to a Route Table , on page 83.

Example:

```
app-key 5yOhH593dtD%2FO8gzAlWgulrkWz5dH02d2STk3LDbI4c%3D
```

Step 11 **cidr ip** *ip_network_addr/mask*

ip_network_addr/mask identifies an individual route within the route table by its address prefix in CIDR format. This is an optional parameter available in Cisco IOS XE Fuji 16.7 or later. If this parameter is not specified, all the routes in the route table are updated. See the restriction on the use of an "all routes" configuration in Route Table Entry Types, on page 74.

Example:

```
cidr ip 15.0.0.0/8
```

Example

This is a summary showing the example configuration commands used in the steps above:

```
redundancy
 cloud provider azure 100
  bfd peer 192.168.101.2
  default-gateway ip 10.60.1.6
  route-table HaEastRouteTable
  cidr ip 15.0.0.0/8
```

```
resource-group companynameusawest
subscription-id ab2fe6b2-c2bd-44
tenant-id 227b0f8f-684d-48fa-9803-c08138b77ae9
app-id 80848f32-8120-43fb-ba65-3d5aa596cd0c
app-key 5yOhH593dtD%2FO8gzAlWgulrkWz5dH02d2STk3LDbI4c%3D
```

Route Table Entry Types

The route tables in Microsoft Azure support different entry types. The entry type for a route can be one of the following: Virtual network gateway, Internet, or Virtual Appliance. The next hop address identifies a resource in the Azure network.

Routes with an entry type of Virtual network gateway or Internet do not have an explicit IP address for the next hop and are not supported by the High Availability feature.

(Cisco IOS XE Everest 16.6) When you configure High Availability on the Cisco CSR 1000v, all the routes within a route table must have an entry type of Virtual Appliance. These routes require an explicit IP address for the next hop.

(Cisco IOS XE Everest 16.7 or later) When you configure High Availability on the Cisco CSR 1000v, you can specify individual routes to be updated in the case of failure. Ensure that you configure each individual route as having an entry type of Virtual Appliance. If you configure a redundancy node that represents all of the entries in the route table, ensure that all of the routes have an entry type of Virtual Appliance.

Verify the Configuration of CSR 1000v High Availability

Use the following EXEC mode commands to verify that the Cisco CSR 1000v has been successfully configured for High Availability.

Step 1 **show crypto pki trustpool**

You can use this command for verification if you imported the trustpool using the configuration command: **crypto pki trustpool import url** *URL* where *URL* is; for example, http://www.cisco.com/security/pki/trs/ios.p7b.

Step 2 **show crypto pki trustpoint**

You can use this command for verification if you installed an individual trustpoint using this configuration command: **crypto pki trustpoint** *name*.

Step 3 **show redundancy cloud provider azure** *node_id*

Use this command to check the redundancy configuration.

Step 4 **show bfd neighbors**

Use this command to verify that neighboring Cisco CSR 1000v routers have established a BFD session.

Step 5 **show running-configuration**

Use this command to verify that the high availability configuration commands entered in the preceding sections appear in the running configuration.

Example:

```
# show running-configuration
```

```
crypto isakmp policy 1
 encr aes 256
 authentication pre-share
crypto isakmp key cisco address 0.0.0.0
!
!
crypto ipsec transform-set uni-perf esp-aes 256 esp-sha-hmac
 mode tunnel
!
!
crypto ipsec profile vti-1
 set security-association lifetime kilobytes disable
 set security-association lifetime seconds 86400
 set transform-set uni-perf
 set pfs group2
!
!
interface Tunnel1
 ip address 192.168.101.1 255.255.255.252
 load-interval 30
 tunnel source GigabitEthernet1
 tunnel mode ipsec ipv4
 tunnel destination 23.96.39.216
tunnel protection ipsec profile vti-1
 bfd interval 500 min_rx 500 multiplier 3
interface GigabitEthernet2
 ip address 10.60.2.6 255.255.255.0
 negotiation auto
 no sh
 no mop enabled
 no mop sysid
```

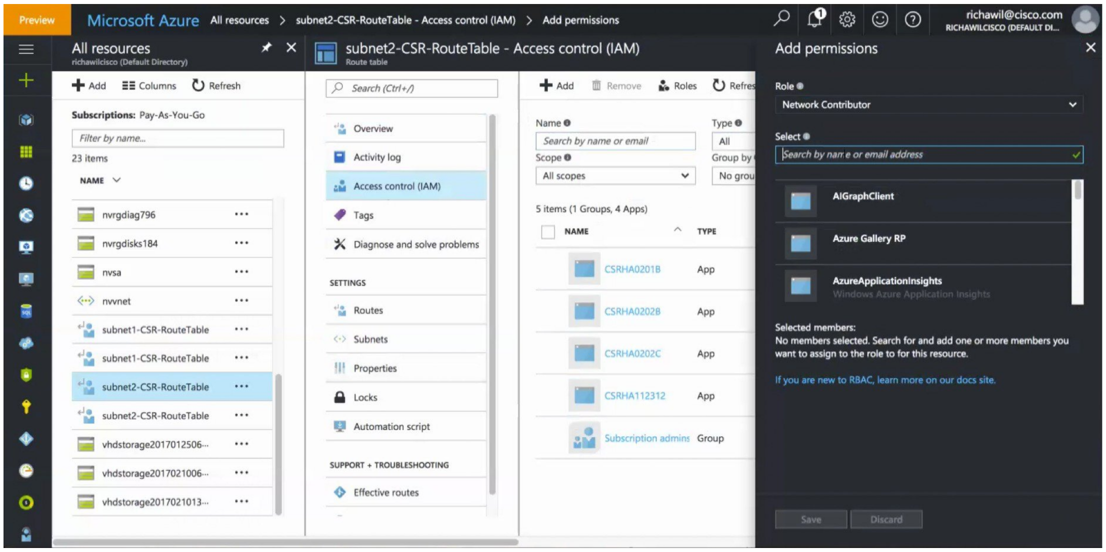
Save the configuration for future use, with the command: **copy running-configuration startup-configuration**.

Configuring IAM for the Route Table

This section explains how you can configure the route table of a subnet to allow the application (for example, "CSRHA2") to modify the CSR 1000v route table.

Step 1 To add an application into an existing network, in the **All resources** pane, choose a private side subnet in the left pane;for example, "subnet2-CSR-RouteTable".

Example:



Role: Network Contributor

Selected members:
No members selected. Search for and add one or more members you want to assign to the role to for this resource.

If you are new to RBAC, learn more on our docs site.

366301

Step 2 Select "Access control (IAM)" and click **+Add**.

Step 3 In the "Role" textbox, choose **Network contributor**and in the "Assign access to" checkbox, select "Azure AD user, group, or application".

Step 4 In the "Select" textbox, enter the name of the application.

Step 5 Click **Save**.

Step 6 Save the values of the following IDs and keys:

- Tenant ID (For example: `227b0f8f-684d-48fa-9803-c08138b77ae9`)
- App ID (For example: `80848f32-8120-43fb-ba65-3d5aa596cd0c`).
- API key (For example: `5yOhH593dtD%2FO8gzAlWgulrkWz5dH02d2STk3LDbI4c%3D`).

Troubleshooting High Availability Issues

If you face any issues when you configure High Availability for your CSR1000v instances, execute the debug_ha.sh script in the `~/azure/HA/` directory. This script gathers all the important logs regarding High Availability and compresses them into a single tar file named ha_debug.tar.gz on the bootflash. Send this .gz file to the Cisco TAC team to get help in resolving your issue.

CHAPTER 9

Deploying Azure Transit VNET DMVPN On Cisco Cloud Services Router 1000v Series

Information About Azure Transit VNET DMVPN Solution

Overview of Transit VNet

A transit VNet is a common strategy to connect multiple, geographically disperse VNet and remote networks. Enabling transit VNet simplifies network management and minimizes the number of connections required to connect multiple VNets and remote networks.

Microsoft Azure VNets leverage Virtual Network (VNet) peering to establish communication between VNETs. Microsoft Azure Transit VNet, also known as Gateway Transit, is a centralized vNET connecting multiple spoke VNets.

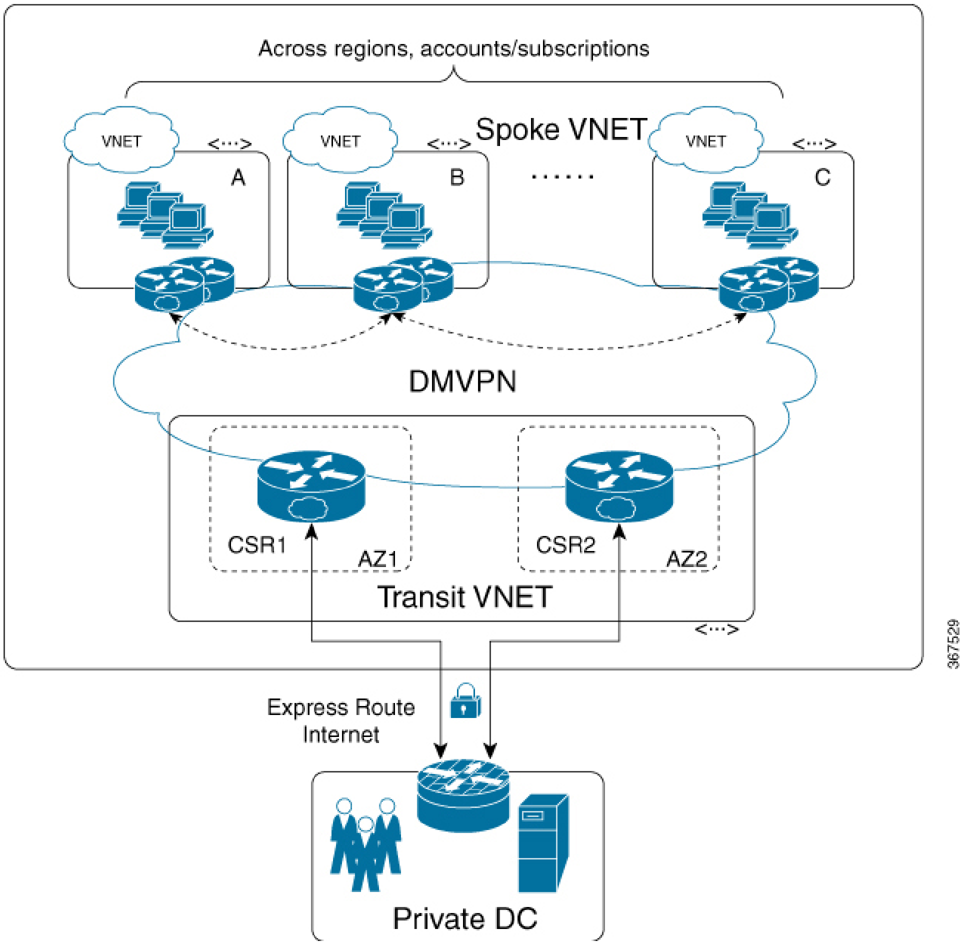
The Cisco Transit VNet solution on Azure uses two CSR 1000v routers that act as the DMVPN Hubs in the active/active mode. The spoke VNets also have a Cisco CSR 1000v acting as the DMVPN Spoke that connects to both the CSR 1000v devices in the transit VNet through EIGRP or BGP as the overlay routing. This solution does not require manual configuration and is completely automated. Once you deploy this solution and configure the essential parameters, the solution automatically creates dynamic spoke-to-spoke IPsec tunnels in an on-demand fashion.

Once the CSR 1000v devices are created, the guestshell scripts are triggered, which run the configuration setup. The scripts configure the CSR Hub and then the necessary information that is associated with the storage account is saved. You should then deploy the Spokes with the necessary configuration to connect the Spokes to the Hub.

The following preconfigured deployments are available as a part of this solution:

- Transit VNet DMVPN all-CSR Hub Template
- Transit VNet DMVPN all-CSR Spoke with 2,4 and 8 NICs

Figure 4: DMVPN all-CSR based Transit VNet Supported on Azure



For more information, see the DMVPN Configuration Guide.

Benefits of using the transit VNet solution

- Higher IPsec throughput of transit-VNet (two Cisco CSR1000v devices in active/active state)
- Connects multiple VNETs spanning globally, across regions, subscriptions, etc.
- Dynamic Spoke-to-Spoke IPsec tunnel reduces billing charges, as the traffic can now flow directly between one spoke VNet to another without having to traverse the Transit-Hub VNet.
- Seamlessly connects to MultiCloud and Hybrid Cloud topologies with DMVPN as the Overlay.
- Support for up to 1000 IPsec tunnels
- End-to-End encryption is possible from spoke-VNet to another spoke-VNet or to remote branch or on-premise locations
- Enhances the cloud with Cisco IOS XE feature set that includes, QoS, ZBFW, NAT, AVC

Prerequisites for Deploying the Transit VNet Solution

- You must have an Azure account for your CSR 1000v devices.
- Ensure that your licenses are registered and valid.

- Ensure that the hub is up and running before you configure the spokes.

Restrictions for Deploying the Transit VNet Solution

- You cannot deploy a Spoke VNet in another Cloud Service Provider.
- You cannot configure the transit VNet solution for all locations. To view the list of locations that are supported, after you create an instance, see all the options in the **Location** field from the Configure Basic Settings page.

How to Deploy Azure Transit VNET DMVPN

Create a Transit VNet Hub

This procedure is the first step in configuring the transit VNet solution. This is a very important part of the deployment where you have to configure the Transit VNet settings. These settings correspond to the DMVPN IPsec parameters that are stored as metadata in the Transit-VNet storage account with an Access-Key. When configuring the spoke templates, you need to configure the TVNET Storage account and the Access-key only. The relevant DMVPN IPsec parameters required for spokes are automatically selected from the device.

Step 1 Sign in to the Microsoft Azure portal.

Step 2 Click **Create a Resource**, search for your Cisco CSR 1000v deployment, and press **Enter**. The system searches and displays the Transit VNET templates for DMVPN.

Step 3 Select **Transit VNET DMVPN** > **Create**.

Step 4 In the Basics screen, enter the name of the Virtual machine, the name for the Transit VNet hub, and your username.

Note Ensure that you use only lower case for **Transit VNet Name**.

Step 5 From the Authentication Type drop-down list, select the SSH Public Key option.

Step 6 Specify a password and reenter the password to confirm.

Step 7 Select the appropriate image version from the **SKU** drop-down list.

Step 8 From the **Location** drop-down list, select one of the regions where TVNET hub can be deployed.

Step 9 In the Cisco CSR Settings page, configure the settings. For more information on configuring CSR settings, see Deploying a Cisco CSR 1000v on Microsoft Azure.

Step 10 In the Transit VNet Settings, configure the following settings:

a) **TVNET Storage Account** – The storage account name that is derived from the Transit VNet name with the keyword ‘strg’ added to the name. You require this value while creating a spoke.The value in this field is auto-populated. However, you can edit the value in this field.
b) **Private TVNET Storage Account** – Select the storage account which is required for saving keys. This field is required for Autoscaler deployments.
c) **DMVPN Tunnel ID** - The Tunnel ID used for setting up tunnel in all the CSR 1000v devices – both hub and spoke.
d) **DMVPN Tunnel Key** - The Tunnel Key, which is a 6-8 digit numerical value.
e) **IPSEC Tunnel Authentication** -

f) **IPSEC Tunnel Cipher** -
g) **IPSEC Shared Key** – The keyword for the authenticating the tunnel.
h) **DMVPN Tunnel Network** – The tunnel network that is used for the DMVPN overlay.

Note The default option might clash with the VNet created for the Hub. Ensure that this value does not overlap with the exiting Virtual Networks (VNet).

At this point you do not need to configure subnets through the Configure Subnets section.

Step 11 Verify the parameters in the Summary screen, and click **OK**.

Step 12 From the **Buy** section, click **Create** to deploy the Transit VNet Hub solution. This step creates the following resources:

- 2 CSRs (CSR1 & CSR2) Virtual-machines deployed in a single Availability-Set
- 2 Storage disks (1 each for each CSR)
- 4 NICs (2 NICs for each CSR)
- 1 Security-Group for the entire Transit-VNET (which opens up only SSH for inbound)
- 2 Public-IP's (1 PIP for each CSR)
- 2 Route-Tables (1 RT for each subnet of the CSR)
- 2 Storage Accounts (1 Storage for the CSR Diagnostics and 1 Storage for Transit-VNET metadata)
- 1 VNET /16 CIDR
- All the above deployed using 1 Resource-Manager group (deleting this RG will delete all the above components)

It takes about 10 to 12 minutes for the deployment to be complete, and for the resources to be created. You can monitor the deployment by clicking **All Resources** and choosing the **Group By Type** option. After the deployment is complete, the notification panel displays the message *Deployment Succeeded*.

Create an Azure DMVPN Spoke VNET

Before you begin

Ensure that your Hub is created successfully before you create a Spoke for the transit VNet solution.

Step 1 From the Microsoft Azure Marketplace, search and select the appropriate **Cisco VPN DMVPN Template**.

Step 2 Click the template, and select the appropriate Spoke option that you want, from the drop-down list.

Step 3 Click **Create**.

Step 4 In the Basics settings screen, ensure that you specify the following configuration details:

a) Filename – Specify the name of the Transit VNet in this field.
b) Transit VNet Storage Name – This is the same as the TVNET Storage Account value from the Hub configuration. This name is derived from the Transit VNet name with 'strg' keywork added.
c) Storage Key – To access the Storage Key, search and click the public Hub and click the **Access Key** option.

Step 5 Configure the other values in the **BASICS** settings screen, and click **OK**.

Step 6 In the Cisco CSR Settings screen, you can choose to either configure the fields or leave them as is (default values).

For information about the parameters, see *How to Deploy a Cisco CSR 1000v on Microsoft Azure*.

Note Availability Zones are not yet fully supported with all the regions in Microsoft Azure. The solution template hence does not have an option for availability zones, but resiliency is taken care using "Availability-Sets". Please refer to the Microsoft Azures documentation here: https://docs.microsoft.com/en-us/azure/availability-zones/az-overview.

Step 7 Click the arrow next to Virtual Network to specify values for the virtual network and click **OK**.

- **Address Space**—Enter the address of the virtual network using Classless Inter-Domain Routing (CIDR) notation.

Note The VNET CIDR denotes the physical ip-address subnets that will be used for Cisco CSR1000v devices in the TVNET-HUB. The CIDR block is usually a /16 subnet which will be subnetted further into two /24 subnets. The first 3 IP addresses of each subnet will be reserved for Azure Route-Table and other services. The IP allocations begin from the 4th ip of the subnet and this will be automatically mapped to the "public ip" that is assigned dynamically. The "public ip" enables access to Internet, hence becomes the NBMA address in the DMVPN scenario.

Step 8 Click the arrow next to configure the subnets, and click **OK**.

Step 9 In the Summary screen, review the configured parameters. After you validate the template, click **OK**.

Step 10 Click **Create** to deploy the TVNet Spoke solution.

Note For every additional Spoke that you want to create, follow steps 1 through 10.

Verifying the Configuration

Verifying on the Transit VNET Hubs

The following commands show that the spokes have successfully established DMVPN tunnels to Transit VNet Hub1 and are able to exchange EIGRP routes with the Transit VNet Hub1. The solution enables DMVPN-Phase 3 feature—NHRP Shortcut Switching. When these commands are run on Transit VNet Hub2, the command outputs are similar to Transit VNet Hub1. This indicates that the spokes have successfully established DMVPN tunnels to both the Cisco CSR1000v in the Transit VNet hub and have successfully exchanged EIGRP routes with both hubs. The hubs are deployed in active-active mode for greater resiliency.

Step 1 Run the `show ip interface` brief command.

Example:

```
Transit-Hub# show ip interface brief
Interface              IP-Address      OK? Method Status                Protocol
GigabitEthernet1       10.1.0.4        YES DHCP   up                    up
GigabitEthernet2       10.1.1.5        YES DHCP   up                    up
Tunnel11               172.16.1.1      YES TFTP   up                    up
VirtualPortGroup0      192.168.35.1    YES TFTP   up                    up
p1-tvnet-csr-1#
```

Notice the higlighted portion in the configuration output. This indicates that the Tunnel is up. If the system does not display the Tunnel in this configuration output, you must go to the guestshell and look at the TVNet logs. Run the `show log` command to access the TVNet logs.

Step 2 Run the `show crypto isakmp sa` command to view the IKE sessions for the two DMVPN connections from the spokes.

Example:

```
Transit-Hub# show crypto isakmp sa
IPv4 Crypto ISAKMP SA
dst             src             state          conn-id status
10.1.0.4        168.62.164.228  QM_IDLE           1042 ACTIVE
10.1.0.4        40.114.69.24    QM_IDLE           1043 ACTIVE
IPv6 Crypto ISAKMP SA
```

Step 3 Run the `show crypto session` command to view the IPsec sessions for the two DMVPN connections from the spokes.

Example:

```
Transit-Hub# show crypto session detail
Crypto session current status
Code: C - IKE Configuration mode, D - Dead Peer Detection
K - Keepalives, N - NAT-traversal, T - cTCP encapsulation
X - IKE Extended Authentication, F - IKE Fragmentation
R - IKE Auto Reconnect, U - IKE Dynamic Route Update
Interface: Tunnel11
Uptime: 1w3d
Session status: UP-ACTIVE
Peer: 40.114.69.24 port 4500 fvrf: (none) ivrf: tvnet-Tun-11
      Phase1_id: 12.1.0.4
      Desc: (none)
  Session ID: 0
  IKEv1 SA: local 10.1.0.4/4500 remote 40.114.69.24/4500 Active
          Capabilities:DN connid:1043 lifetime:18:32:04
  IPSEC FLOW: permit 47 host 10.1.0.4 host 40.114.69.24
        Active SAs: 2, origin: crypto map
        Inbound:  #pkts dec'ed 32 drop 0 life (KB/Sec) 4607996/3474
        Outbound: #pkts enc'ed 32 drop 0 life (KB/Sec) 4607998/3474
Interface: Tunnel11
Uptime: 1w3d
Session status: UP-ACTIVE
Peer: 168.62.164.228 port 4500 fvrf: (none) ivrf: tvnet-Tun-11
      Phase1_id: 11.1.0.4
      Desc: (none)
  Session ID: 0
  IKEv1 SA: local 10.1.0.4/4500 remote 168.62.164.228/4500 Active
          Capabilities:DN connid:1042 lifetime:18:02:01
  IPSEC FLOW: permit 47 host 10.1.0.4 host 168.62.164.228
        Active SAs: 2, origin: crypto map
        Inbound:  #pkts dec'ed 32 drop 0 life (KB/Sec) 4607970/2427
        Outbound: #pkts enc'ed 32 drop 0 life (KB/Sec) 4607982/2427
```

Step 4 Run the `show dmvpn` command to view the status of the DMVPN on the device.

Example:

```
Transit-Hub# show dmvpn
Legend: Attrb --> S - Static, D - Dynamic, I - Incomplete
        N - NATed, L - Local, X - No Socket
        T1 - Route Installed, T2 - Nexthop-override
        C - CTS Capable, I2 - Temporary
        # Ent --> Number of NHRP entries with same NBMA peer
        NHS Status: E --> Expecting Replies, R --> Responding, W --> Waiting
        UpDn Time --> Up or Down Time for a Tunnel
==========================================================================
Interface: Tunnel11, IPv4 NHRP Details
Type:Hub, NHRP Peers:2,
 # Ent  Peer NBMA Addr Peer Tunnel Add State  UpDn Tm Attrb
 ----- --------------- --------------- ----- -------- -----
     1 40.114.69.24       172.16.1.137    UP     1w3d    DN
     1 168.62.164.228     172.16.1.147    UP     1w3d    DN
```

Step 5 Run the `show vrf` command to view the display routes from each of the spokes on the transit VNet.

Example:

```
Transit-Hub# show vrf
  Name                             Default RD            Protocols   Interfaces
  tvnet-Tun-11                     64512:11              ipv4        Tu11
```

Step 6 Run the `show ip eigrp vrf <vrf-name> neighbors` command to view the status of the EIGRP neighbors.

Example:

```
Transit-Hub# show ip eigrp vrf tvnet-Tun-11 neighbors
EIGRP-IPv4 Neighbors for AS(64512) VRF(tvnet-Tun-11)
H   Address                 Interface              Hold Uptime   SRTT   RTO  Q  Seq
                                                   (sec)         (ms)       Cnt Num
1   172.16.1.137            Tu11                     14 1w3d       13  1398  0  12
0   172.16.1.147            Tu11                     10 1w3d       12  1398  0  12
```

Step 7 Run the `show ip route vrf <vrf-name>` command to view the route specific to a VRF.

Example:

```
Transit-Hub# show ip route vrf tvnet-Tun-11
Routing Table: tvnet-Tun-11
Codes: L - local, C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2
       i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2
       ia - IS-IS inter area, * - candidate default, U - per-user static route
       o - ODR, P - periodic downloaded static route, H - NHRP, l - LISP
       a - application route
       + - replicated route, % - next hop override, p - overrides from PfR
Gateway of last resort is not set
      11.0.0.0/24 is subnetted, 2 subnets
D EX     11.1.0.0 [170/26880256] via 172.16.1.147, 1w1d, Tunnel11
D EX     11.1.1.0 [170/26880256] via 172.16.1.147, 1w1d, Tunnel11
      12.0.0.0/24 is subnetted, 2 subnets
D EX     12.1.0.0 [170/26880256] via 172.16.1.137, 1w1d, Tunnel11
D EX     12.1.1.0 [170/26880256] via 172.16.1.137, 1w1d, Tunnel11
      172.16.0.0/16 is variably subnetted, 2 subnets, 2 masks
C        172.16.1.0/24 is directly connected, Tunnel11
L        172.16.1.1/32 is directly connected, Tunnel11
D EX  192.168.35.0/24 [170/26905600] via 172.16.1.147, 1w1d, Tunnel11
                      [170/26905600] via 172.16.1.137, 1w1d, Tunnel11
```

Verifying the Connectivity Between the Spokes and the Hub

The following commands show that the spokes are connected to both the Cisco CSR 1000v TVNET HUB and have been able to exchange the EIGRP routes from both the hubs. As the DMVPN solution is deployed as DMVPN-Phase3 (NHRP shortcut-switching) and the hubs are deployed in the active-active mode, the EIGRP route towards SPOKE2 points to the tunnel-overlay ip-address of spoke2.

Step 1 Run the `show ip interface brief` command to view the interface ip addresses on the device.

Example:

```
Spoke# show ip interface brief
Interface              IP-Address      OK? Method Status                Protocol
GigabitEthernet1       11.1.0.4        YES DHCP   up                    up
GigabitEthernet2       11.1.1.4        YES DHCP   up                    up
Tunnel11               172.16.1.147    YES TFTP   up                    up
VirtualPortGroup0      192.168.35.1    YES TFTP   up                    up
```

Step 2 Run the `show dmvpn` command to check the status of the DMVPN on the device.

Example:

```
Spoke# show dmvpn
Legend: Attrb --> S - Static, D - Dynamic, I - Incomplete
        N - NATed, L - Local, X - No Socket
        T1 - Route Installed, T2 - Nexthop-override
        C - CTS Capable, I2 - Temporary
        # Ent --> Number of NHRP entries with same NBMA peer
        NHS Status: E --> Expecting Replies, R --> Responding, W --> Waiting
        UpDn Time --> Up or Down Time for a Tunnel
==========================================================================
Interface: Tunnel11, IPv4 NHRP Details
Type:Spoke, NHRP Peers:2,
 # Ent  Peer NBMA Addr Peer Tunnel Add State  UpDn Tm Attrb
 ----- --------------- --------------- ----- -------- -----
     1 40.117.131.133       172.16.1.1    UP     1w3d     S
     1 40.117.128.85        172.16.1.2    UP     1w3d     S
```

Notice the configuration output that is highlighted. This indicates that the spokes are up, and have established a connection with the hub.

Step 3 Run the `show crypto isakmp sa` command to view the IKE sessions for the two DMVPN connections from the spokes.

Example:

```
Spoke# show crypto isakmp sa
IPv4 Crypto ISAKMP SA
dst             src             state          conn-id status
40.117.131.133  11.1.0.4        QM_IDLE           1025 ACTIVE
40.117.128.85   11.1.0.4        QM_IDLE           1026 ACTIVE
IPv6 Crypto ISAKMP SA
```

Step 4 Run the `show crypto session` command to view the IPsec sessions for the two DMVPN connections from the spokes.

Example:

```
Spoke# show crypto session detail
Crypto session current status
Code: C - IKE Configuration mode, D - Dead Peer Detection
K - Keepalives, N - NAT-traversal, T - cTCP encapsulation
X - IKE Extended Authentication, F - IKE Fragmentation
R - IKE Auto Reconnect, U - IKE Dynamic Route Update
Interface: Tunnel11
Uptime: 1w3d
Session status: UP-ACTIVE
Peer: 40.117.131.133 port 4500 fvrf: (none) ivrf: (none)
      Phase1_id: 10.1.0.4
      Desc: (none)
  Session ID: 0
  IKEv1 SA: local 11.1.0.4/4500 remote 40.117.131.133/4500 Active
          Capabilities:DN connid:1025 lifetime:17:33:41
  IPSEC FLOW: permit 47 host 11.1.0.4 host 40.117.131.133
        Active SAs: 2, origin: crypto map
        Inbound:  #pkts dec'ed 2250 drop 0 life (KB/Sec) 4607927/726
        Outbound: #pkts enc'ed 2251 drop 0 life (KB/Sec) 4607957/726
Interface: Tunnel11
Uptime: 1w3d
Session status: UP-ACTIVE
Peer: 40.117.128.85 port 4500 fvrf: (none) ivrf: (none)
      Phase1_id: 10.1.0.5
      Desc: (none)
  Session ID: 0
  IKEv1 SA: local 11.1.0.4/4500 remote 40.117.128.85/4500 Active
          Capabilities:DN connid:1026 lifetime:17:33:44
  IPSEC FLOW: permit 47 host 11.1.0.4 host 40.117.128.85
        Active SAs: 2, origin: crypto map
        Inbound:  #pkts dec'ed 2252 drop 0 life (KB/Sec) 4607960/2046
        Outbound: #pkts enc'ed 2253 drop 0 life (KB/Sec) 4607976/2046
```

Step 5 Run the `show up eigrp neighbor` command to view the status of the EIGRP neighbors.

Example:

```
Spoke# show ip eigrp neighbor
EIGRP-IPv4 Neighbors for AS(64512)
H   Address                 Interface              Hold Uptime   SRTT   RTO  Q  Seq
                                                   (sec)         (ms)       Cnt Num
1   172.16.1.2              Tu11                     13 1w3d       24  1362  0  23
0   172.16.1.1              Tu11                     12 1w3d        8  1362  0  23
```

Step 6 Run the show ip route eigrp command to view the EIGRP route information.

Example:

```
Spoke# show ip route eigrp
Codes: L - local, C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2
       i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2
       ia - IS-IS inter area, * - candidate default, U - per-user static route
       o - ODR, P - periodic downloaded static route, H - NHRP, l - LISP
       a - application route
       + - replicated route, % - next hop override, p - overrides from PfR
Gateway of last resort is 11.1.0.1 to network 0.0.0.0
      12.0.0.0/24 is subnetted, 2 subnets
D EX     12.1.0.0 [170/28160256] via 172.16.1.137, 1w3d, Tunnel11
                  [170/28160256] via 172.16.1.137, 1w3d, Tunnel11
D EX     12.1.1.0 [170/28160256] via 172.16.1.137, 1w3d, Tunnel11
                  [170/28160256] via 172.16.1.137, 1w3d, Tunnel11
```

Verifying Spoke to Spoke Connectivity

The following commands help in testing connection between two spokes. As the feature supported is DMVPN-Phase 3, the **traceroute** command displays the packets sent from spoke 1 to spoke 2. However, the first packet is lost due to NHRP resolution as Spoke 1 sends the packet to the hub to obtain the address of

Spoke 2. When Spoke 1 recieves the address, a dynamic IPsec tunnel is established between Spoke 1 and Spoke 2.

```
Spoke1# clear crypto sa counters
Spoke1# ping 12.1.1.4 source gigabitEthernet 2 repeat 100
Type escape sequence to abort.
Sending 100, 100-byte ICMP Echos to 12.1.1.4, timeout is 2 seconds:
Packet sent with a source address of 11.1.1.4
.!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
Success rate is 99 percent (99/100), round-trip min/avg/max = 1/1/6 ms
Spoke# show dmvpn
Legend: Attrb --> S - Static, D - Dynamic, I - Incomplete
        N - NATed, L - Local, X - No Socket
        T1 - Route Installed, T2 - Nexthop-override
        C - CTS Capable, I2 - Temporary
        # Ent --> Number of NHRP entries with same NBMA peer
        NHS Status: E --> Expecting Replies, R --> Responding, W --> Waiting
        UpDn Time --> Up or Down Time for a Tunnel
==========================================================================
Interface: Tunnel11, IPv4 NHRP Details
Type:Spoke, NHRP Peers:3,
 # Ent  Peer NBMA Addr Peer Tunnel Add State  UpDn Tm Attrb
 ----- --------------- --------------- ----- -------- -----
     1 40.117.131.133       172.16.1.1    UP     1w3d     S
     1 40.117.128.85        172.16.1.2    UP     1w3d     S
     1 40.114.69.24       172.16.1.137    UP 00:00:07    DN
Spoke# traceroute 12.1.1.4 source gigabitEthernet 2
Type escape sequence to abort.
Tracing the route to 12.1.1.4
VRF info: (vrf in name/id, vrf out name/id)
  1 172.16.1.137 2 msec *  3 msec
p1spoke1#
p1spoke1#
p1spoke1#sh crypto sess detail | i pkts
        Inbound:  #pkts dec'ed 101 drop 0 life (KB/Sec) 4607985/3581
        Outbound: #pkts enc'ed 100 drop 0 life (KB/Sec) 4607989/3581
        Inbound:  #pkts dec'ed 12 drop 0 life (KB/Sec) 4607924/621
        Outbound: #pkts enc'ed 14 drop 0 life (KB/Sec) 4607955/621
        Inbound:  #pkts dec'ed 13 drop 0 life (KB/Sec) 4607957/1941
        Outbound: #pkts enc'ed 13 drop 0 life (KB/Sec) 4607975/1941
Spoke# show crypto session detail
Crypto session current status
Code: C - IKE Configuration mode, D - Dead Peer Detection
K - Keepalives, N - NAT-traversal, T - cTCP encapsulation
X - IKE Extended Authentication, F - IKE Fragmentation
R - IKE Auto Reconnect, U - IKE Dynamic Route Update
Interface: Tunnel11
Uptime: 00:00:36
Session status: UP-ACTIVE
Peer: 40.114.69.24 port 4500 fvrf: (none) ivrf: (none)
      Phase1_id: 12.1.0.4
      Desc: (none)
  Session ID: 0
  IKEv1 SA: local 11.1.0.4/4500 remote 40.114.69.24/4500 Active
          Capabilities:DN connid:1027 lifetime:23:59:23
  IPSEC FLOW: permit 47 host 11.1.0.4 host 40.114.69.24
        Active SAs: 4, origin: crypto map
        Inbound:  #pkts dec'ed 101 drop 0 life (KB/Sec) 4607985/3563
        Outbound: #pkts enc'ed 100 drop 0 life (KB/Sec) 4607989/3563
Interface: Tunnel11
Uptime: 1w3d
Session status: UP-ACTIVE
Peer: 40.117.131.133 port 4500 fvrf: (none) ivrf: (none)
```

```
      Phase1_id: 10.1.0.4
      Desc: (none)
  Session ID: 0
  IKEv1 SA: local 11.1.0.4/4500 remote 40.117.131.133/4500 Active
          Capabilities:DN connid:1025 lifetime:17:31:38
  IPSEC FLOW: permit 47 host 11.1.0.4 host 40.117.131.133
        Active SAs: 2, origin: crypto map
        Inbound:  #pkts dec'ed 16 drop 0 life (KB/Sec) 4607923/603
        Outbound: #pkts enc'ed 18 drop 0 life (KB/Sec) 4607955/603
Interface: Tunnel11
Uptime: 1w3d
Session status: UP-ACTIVE
Peer: 40.117.128.85 port 4500 fvrf: (none) ivrf: (none)
      Phase1_id: 10.1.0.5
      Desc: (none)
  Session ID: 0
  IKEv1 SA: local 11.1.0.4/4500 remote 40.117.128.85/4500 Active
          Capabilities:DN connid:1026 lifetime:17:31:41
  IPSEC FLOW: permit 47 host 11.1.0.4 host 40.117.128.85
        Active SAs: 2, origin: crypto map
        Inbound:  #pkts dec'ed 17 drop 0 life (KB/Sec) 4607957/1923
        Outbound: #pkts enc'ed 17 drop 0 life (KB/Sec) 4607975/1923
```

Troubleshooting

To view the status of your deployment, log in to your CSR 1000v instance and run the `show log` command. If your depolyment is successful you should see the *[AzureTransitVNET] Success. Configured all the required IOS configs* message.

If you do not see this message, and experience any errors while configuring the Transit VNet solution, check whether:

- The DMVPN tunnel is established between the hub and the spoke. In most cases, there might be a problem with the following values: TransitVNETname, TransitVNETStoragename or TransitVNETStoragekey.
- The Guestshell is up and running for the TVNet packages that are to be installed.

CHAPTER 10

Deploying Transit VNet Solution with Autoscaling

This chapter describes how you can deploy the autoscaling functionality (Autoscaler) on Transit VNet for CSR 1000v instances running on Microsoft Azure to monitor and perform scale-out and scale-in operations.

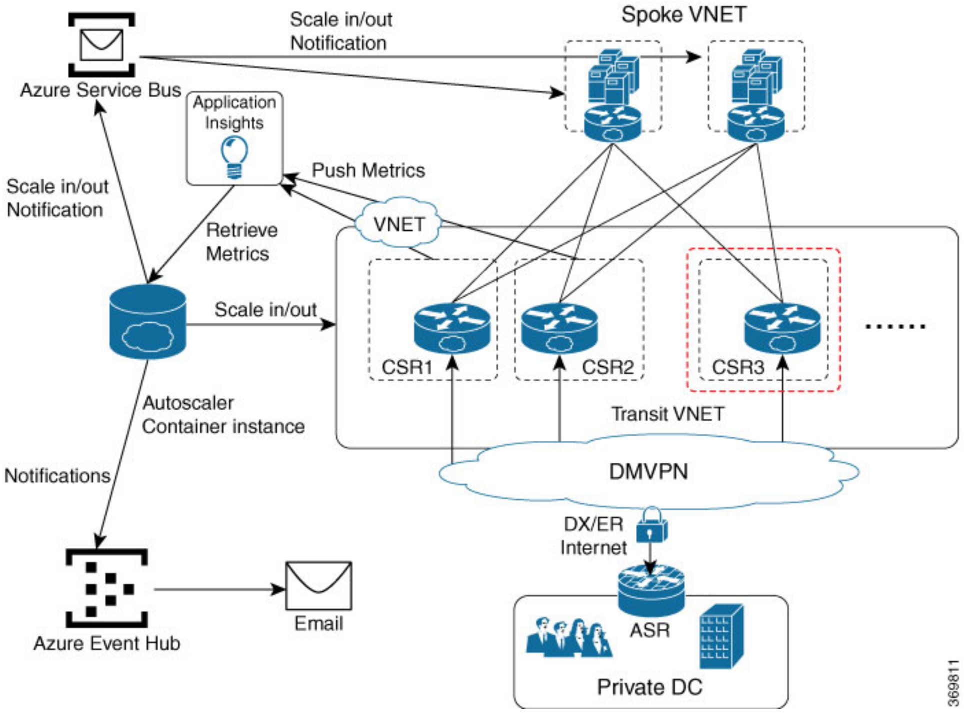
Overview of Autoscaler

Autoscaling is a functionality that is available through the **Cisco CSR 1000v DMVPN Transit VNET** template on Microsoft Azure. When you deploy this template, resources such as the Transit VNet solution, an Autoscaler container, an Azure Service bus, and the Application Insights are created. The Autoscaler functionality is enabled, which automatically performs scale-in and scale-out operations by adding and removing CSR 1000v instances depending on the volume of traffic.

In an Autoscaler solution, there are a minimum of two CSR 1000v instances that act as the Hub. These instances connect to the Spoke VNets. The Hub also continuously sends metrics to the Application Insights on a periodic basis, and the Autoscaler container retrieves the metrics periodically.

Whenever the traffic load increases or decreases, it triggers a scale-out or a scale-in action. In turn, Autoscaler adds a new CSR1000v instance to the Transit VNet Hub and delivers a message to the Topic in the Service bus namespace of the Autoscaler resource group. This message contains the configuration information about the new CSR 1000v instance in the Hub. Since the Spoke VNet is subscribed to this Topic, it receives the message and self-configures itself to use the new CSR 1000v instance to forward traffic.

The following image represents a sample Transit VNet solution with the Autoscaling functionality:



Autoscaler Functionalists

Before you deploy the Autoscaler solution, you might need to be familiar with the following actions that an Autoscaler solution performs:

Scale-out: Scaling out refers to adding a new CSR 1000v instance in the Transit VNet solution to increase the capacity of the Transit VNet solution. When the Autoscaler detects an increased load for a sustained period of time, it performs a scale-out by adding a new CSR instance to the Transit VNet capacity. While performing scale-out, the Autoscaler configures the CSR 1000v instance for all the existing on-premise VPN networks and the spoke VNets.

Scale-in: Scaling in refers to deleting or removing CSR 1000v instances to reduce the extended capacity. When the Autoscaler detects a decrease in traffic for a sustained period of time, it performs scale-in action by terminating one or more CSR1000v instances from the Transit Vnet.

Note Autoscaler performs scale-out and scale-in operations based on the metrics that are published on Azure Application Insights. When Autoscaler detects that the metrics meet the per-defined conditions, it takes appropriate action. To know about how metrics are published, see Cloud Service Monitoring in Azure.

To learn more about the configuration conditions, see the contents of the `autoscale_config.json` file that is available in File Share under your private Transit VNet account.

Rotate: When Autoscaler monitors the solution, if the Image ID of the CSR 1000v instances do not match the Image ID in the autoscaler configuration file, Autoscaler spins up a new CSR 1000v instance with a new Image ID that is mentioned in the autoscale_config.json file. When the Autoscaler monitors the solution, if the image ID does not match the image ID mentioned in the config file, Autoscaler spins a new instance with a new or modified image ID that is in the config file for the same license type for which the current image ID fits in.

Replace: When Autoscaler detects a CSR1000v instance failure, Autoscaler replaces the faulty instance with a new CSR 1000v instance. For example, when the CSR Gi1 interface which the Transit VNet solution uses to pass the IPsec tunnel traffic goes down, Autoscaler performs the Replace action. However, the instance configuration such as the instance number, ios configuration etc. is retained by the Autoscaler solution.

Benefits of Autoscaler

- Automatically scales a Transit VNet by adding or removing CSR 1000v instances to meet the changing network bandwidth requirements. There is no need for any manual intervention, including manually adding or removing additional CSR 1000v instances.
- Performs automatic license activation for CSR 1000v instances for Bring Your Own License (BYOL) type.
- Effectively utilizes the CSR 1000v instances in Transit VNet to save cost.
- Monitors the health of the CSR 1000v instances and automatically helps to replace an instance that is either faulty or failing.

Prerequisites for Autoscaler

- You must deploy a Transit VNet solution before you can configure the Autoscaler functionality.
- You must create an Azure Service principal Application to enable Autoscaler to automatically create the required resources to scale-in or scale-out resource groups. To create a Service Principal Application, execute the `az ad sp create-for-rbac --role="Contributor" --scopes="/subscriptions/<subscription_id>" -p <password>` script.

  For more information, see Create an Azure service principal with Azure CLI.

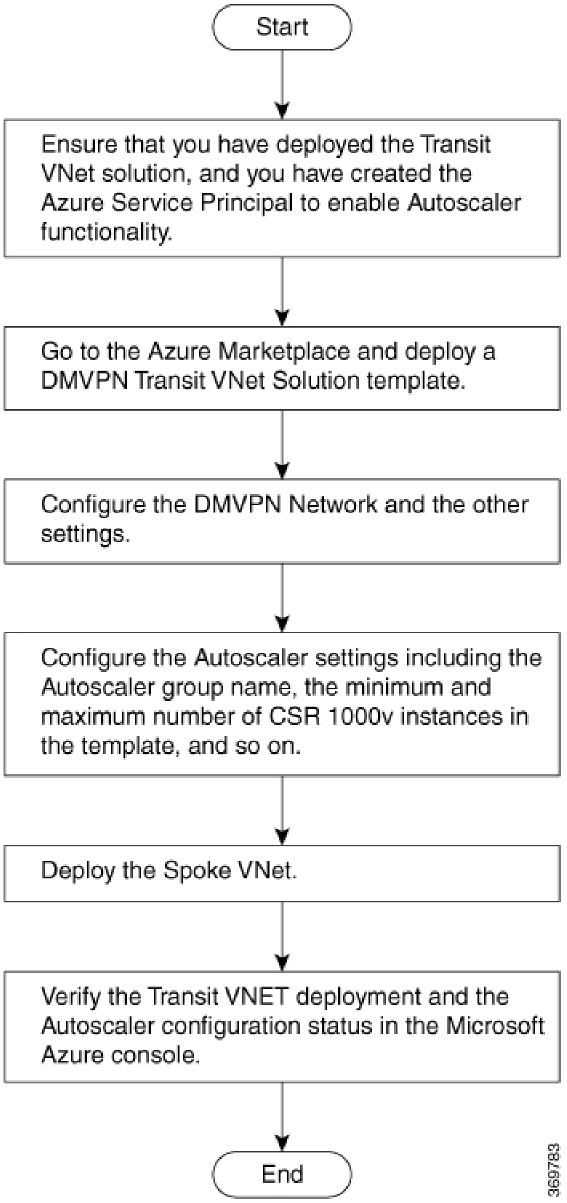
Restrictions for Deploying the Autoscaler Solution

- In an Autoscaler solution, only one CSR 1000v instance is permitted to be Out of Compliance (OOC) at any given time. Note that the PAYG licenses are automatically considered In-compliance.
- If a scale-out is requested, but even if there is one CSR 1000v instance that is Out of Compliance, the scale-out operation is not successful.
- If there are no OOC instances and a scale-out operation happens with a BYOL license, and no license is available on the smart Cisco server, a 30-day grace period is started. The throughput of the CSR 1000v instance matches that of the configured rate of the other CSR 1000v instances. The 30-day time period is tracked per the CSR 1000v instance and once the 30-day time period is done, the data rate reverts to 1Mb/s until you install a license.

Deploy the Transit VNet Solution with Autoscaler

The following image displays the Autoscaler deployment workflow for CSR 1000v instances running on Microsoft Azure:

Figure 5: Autoscaler Deployment Workflow for Azure



Ensure that you have deployed the Transit VNet solution, and you have created the Azure Service Principal to enable Autoscaler functionality.

Go to the Azure Marketplace and deploy a DMVPN Transit VNet Solution template.

Configure the DMVPN Network and the other settings.

Configure the Autoscaler settings including the Autoscaler group name, the minimum and maximum number of CSR 1000v instances in the template, and so on.

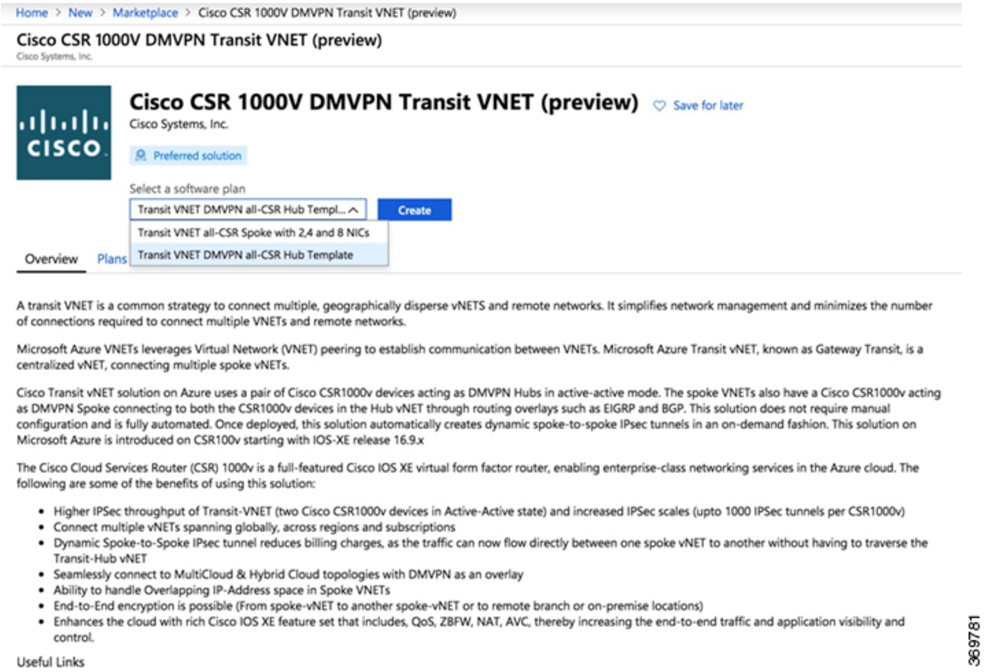
Deploy the Spoke VNet.

Verify the Transit VNET deployment and the Autoscaler configuration status in the Microsoft Azure console.

To deploy the Transit VNet solution with Autoscaler, perform the following procedures:

Deploying the Transit VNet Solution With Autoscaling

Step 1 Go to the **Azure Marketplace**, search and lauch a **Cisco CSR 1000v DMVPN Transit VNET** solution.


Home > New > Marketplace > Cisco CSR 1000V DMVPN Transit VNET (preview)
Cisco CSR 1000V DMVPN Transit VNET (preview)
Cisco Systems, Inc.
Cisco CSR 1000V DMVPN Transit VNET (preview)
Save for later
Cisco Systems, Inc.
Preferred solution
Select a software plan
Transit VNET DMVPN all-CSR Hub Templ...
Create
Transit VNET all-CSR Spoke with 2,4 and 8 NICs
Transit VNET DMVPN all-CSR Hub Template
Overview
Plans
Useful Links
369781

A transit VNET is a common strategy to connect multiple, geographically disperse vNETS and remote networks. It simplifies network management and minimizes the number of connections required to connect multiple VNETs and remote networks.

Microsoft Azure VNETs leverages Virtual Network (VNET) peering to establish communication between VNETs. Microsoft Azure Transit vNET, known as Gateway Transit, is a centralized vNET, connecting multiple spoke vNETs.

Cisco Transit vNET solution on Azure uses a pair of Cisco CSR1000v devices acting as DMVPN Hubs in active-active mode. The spoke VNETs also have a Cisco CSR1000v acting as DMVPN Spoke connecting to both the CSR1000v devices in the Hub vNET through routing overlays such as EIGRP and BGP. This solution does not require manual configuration and is fully automated. Once deployed, this solution automatically creates dynamic spoke-to-spoke IPsec tunnels in an on-demand fashion. This solution on Microsoft Azure is introduced on CSR100v starting with IOS-XE release 16.9.x

The Cisco Cloud Services Router (CSR) 1000v is a full-featured Cisco IOS XE virtual form factor router, enabling enterprise-class networking services in the Azure cloud. The following are some of the benefits of using this solution:

- Higher IPSec throughput of Transit-VNET (two Cisco CSR1000v devices in Active-Active state) and increased IPSec scales (upto 1000 IPSec tunnels per CSR1000v)
- Connect multiple vNETs spanning globally, across regions and subscriptions
- Dynamic Spoke-to-Spoke IPsec tunnel reduces billing charges, as the traffic can now flow directly between one spoke vNET to another without having to traverse the Transit-Hub vNET
- Seamlessly connect to MultiCloud & Hybrid Cloud topologies with DMVPN as an overlay
- Ability to handle Overlapping IP-Address space in Spoke VNETs
- End-to-End encryption is possible (From spoke-vNET to another spoke-vNET or to remote branch or on-premise locations)
- Enhances the cloud with rich Cisco IOS XE feature set that includes, QoS, ZBFW, NAT, AVC, thereby increasing the end-to-end traffic and application visibility and control.

Step 2 On the Deployment screen, configure the **Basic Settings** and the **CSR Settings**. To know how to configure these settings, see Deploy a CSR1000v Instance on Microsoft Azure.

Step 3 Configure the Transit VNet Settings. For more details, see Configuring Transit VNet Solution.

Step 4 From the Autoscaler Settings page, configure the following settings:

a) **Enable Autoscaling for Transit Hub CSRs**: Select Enable from the drop-down list.

b) **Autoscaling Group Name**: Specify a name for the Autoscaler.

c) **Autoscaling Group Min**: The minimum number of CSR1000v instances you want for the solution. This value should be a minimum of 2.

d) **Autoscaling Group Max**: The maximum number of CSR1000v instances you want in a scale-out operation. This value should not exceed 8.

e) **Autoscaling Group License Model**: Select the appropriate license type from this drop-down list. You can either choose BYOL or Pas As You Go.

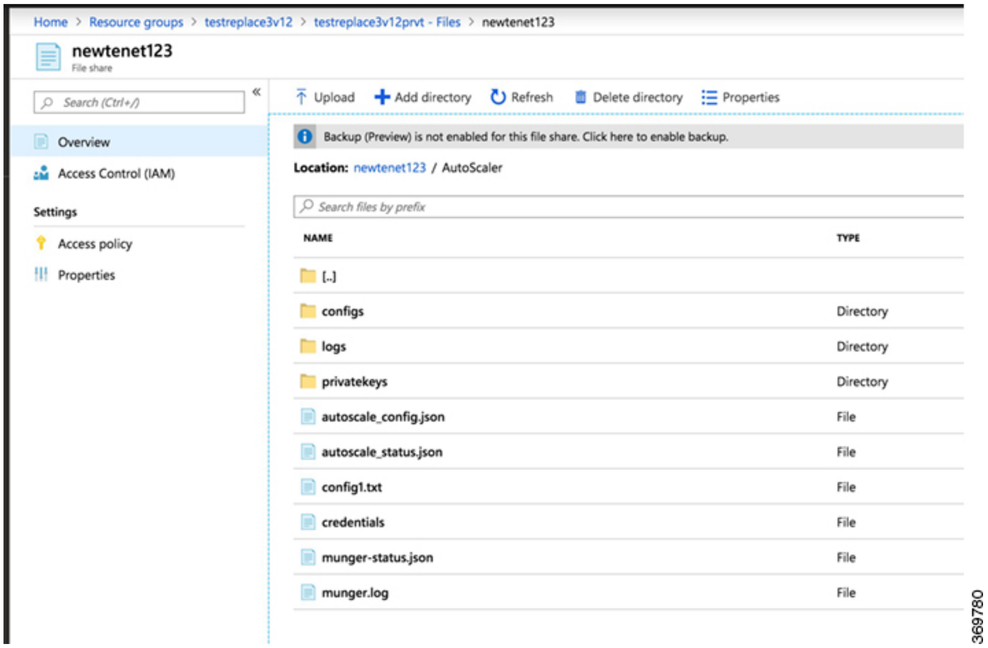
f) **Enable Scale-In**: Select the Enable option from the drop-down list if you want to enale the scale-in operation.

g) **License ID Token For BYOL CSR**: From the Smart License configration, specify the License token that you use for registering your smart licenses. If you do not specify the token here, the CSR 1000v instances go Out Of Complaiance.

h) **License Throughput Level for BYOL CSR**: Select the Throughput Level that you want to configure from the drop-down list.

i) **Email Address for Licensing Purpose**: Specify the registered email address that is associated with the CSR 1000v licenses.

j) **Autoscaling Custom Config File Share**: If you need a custom configuration, specify the custom configuration in the form of a file. For example, to connect to the on-premise devices, provide the on-premise configuration in the form of a file via File Share. Specify the location of the File Share location.



Home > Resource groups > testreplace3v12 > testreplace3v12prvt - Files > newtenet123

newtenet123
File share

Backup (Preview) is not enabled for this file share. Click here to enable backup.

Location: newtenet123 / AutoScaler

Search files by prefix

NAME	TYPE
[..]	
configs	Directory
logs	Directory
privatekeys	Directory
autoscale_config.json	File
autoscale_status.json	File
config1.txt	File
credentials	File
munger-status.json	File
munger.log	File

k) **Autoscaling Custom Storage Name**: The Storage Name where the File Share resides.

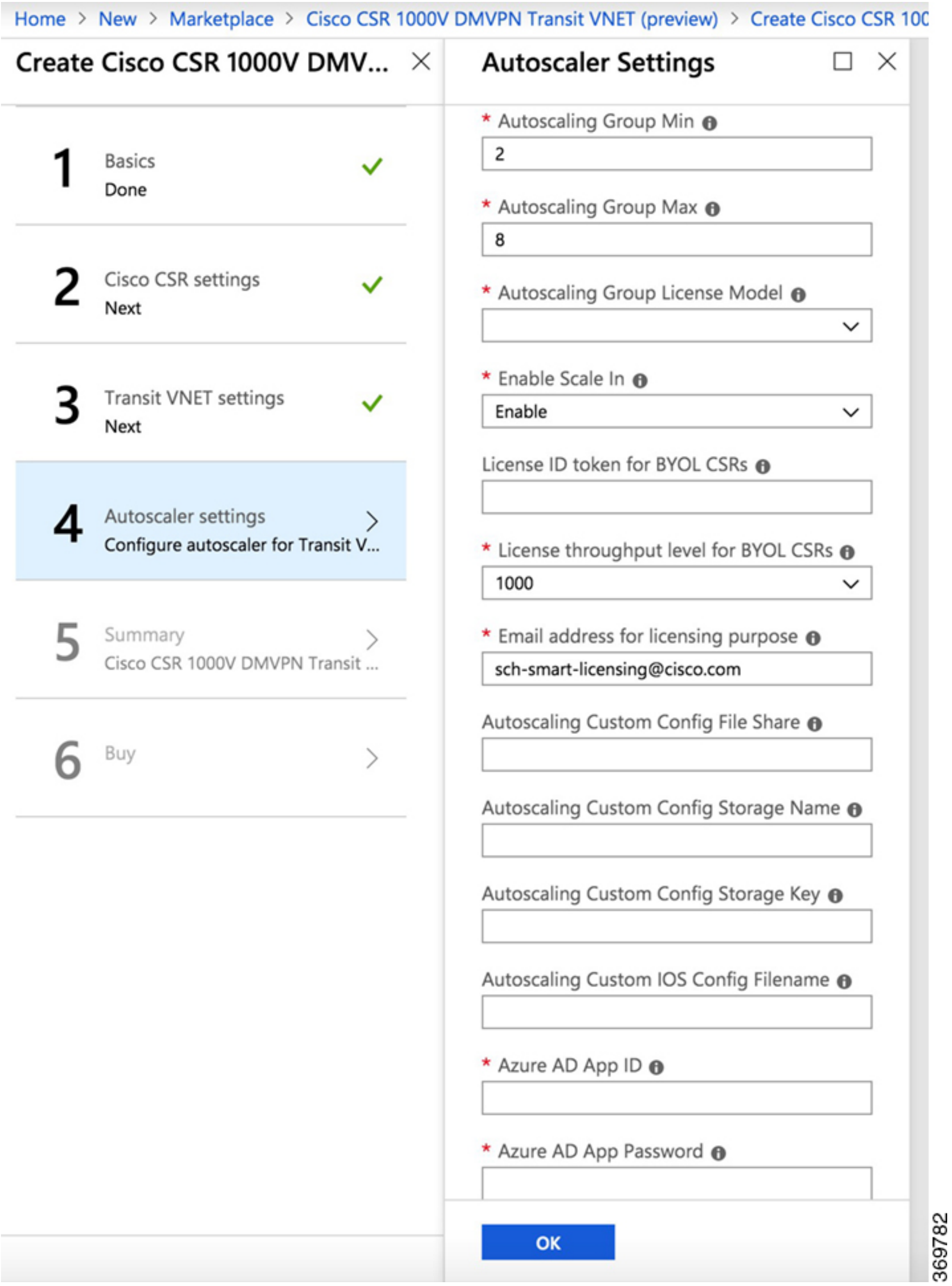
l) Autoscaler Storage Key:

m) **IOS Config Filename**: This is the name of the .txt configuration file that you provide via File share.

n) **Azure AD App ID**: The client ID or the Image ID that is displayed on the screen when you create the service application through the CLI.

o) **Azure AD App Password**: Enter the password that you used while configuring the service application.

Figure 6:



Home > New > Marketplace > Cisco CSR 1000V DMVPN Transit VNET (preview) > Create Cisco CSR 100

Create Cisco CSR 1000V DMV...

1 Basics
Done ✓

2 Cisco CSR settings
Next ✓

3 Transit VNET settings
Next ✓

4 Autoscaler settings
Configure autoscaler for Transit V...

5 Summary
Cisco CSR 1000V DMVPN Transit ...

6 Buy

Autoscaler Settings

* Autoscaling Group Min: 2

* Autoscaling Group Max: 8

* Autoscaling Group License Model:

* Enable Scale In: Enable

License ID token for BYOL CSRs:

* License throughput level for BYOL CSRs: 1000

* Email address for licensing purpose: sch-smart-licensing@cisco.com

Autoscaling Custom Config File Share:

Autoscaling Custom Config Storage Name:

Autoscaling Custom Config Storage Key:

Autoscaling Custom IOS Config Filename:

* Azure AD App ID:

* Azure AD App Password:

Step 5 Click **OK**. The system displays the Summary page with all your configuration options.

Step 6 Click **OK** to deploy the Autoscaler solution.

After the solution is deployed, the template creates the network, storage, and compute infrastructure including two CSR 1000v instances in a Transit VNet solution.

Verifying on the Transit VNET Hubs

The following commands show that the spokes have successfully established DMVPN tunnels to Transit VNet Hub1 and are able to exchange EIGRP routes with the Transit VNet Hub1. The solution enables DMVPN-Phase 3 feature - NHRP Shortcut Switching. When these commands are run on Transit VNet Hub2, the command outputs are similar to Transit VNet Hub1. This indicates that the spokes have successfully established DMVPN tunnels to both the Cisco CSR1000v in the Transit VNet hub and have successfully exchanged EIGRP routes with both hubs. The hubs are deployed in active-active mode for greater resiliency.

Step 1 Run the `show ip interface` brief command.

Example:

```
Transit-Hub# show ip interface brief
Interface              IP-Address      OK? Method Status                Protocol
GigabitEthernet1       10.1.0.4        YES DHCP   up                    up
GigabitEthernet2       10.1.1.5        YES DHCP   up                    up
Tunnel11               172.16.1.1      YES TFTP   up                    up
VirtualPortGroup0      192.168.35.1    YES TFTP   up                    up
p1-tvnet-csr-1#
```

Notice the higlighted portion in the configuration output. This indicates that the Tunnel is up. If the system does not display the Tunnel in this configuration output, you must go to the guestshell and look at the TVNet logs. Run the `show log` command to access the TVNet logs.

Step 2 Run the `show crypto isakmp sa` command to view the IKE sessions for the two DMVPN connections from the spokes.

Example:

```
Transit-Hub# show crypto isakmp sa
IPv4 Crypto ISAKMP SA
dst             src             state          conn-id status
10.1.0.4        168.62.164.228  QM_IDLE           1042 ACTIVE
10.1.0.4        40.114.69.24    QM_IDLE           1043 ACTIVE
IPv6 Crypto ISAKMP SA
```

Step 3 Run the `show crypto session` command to view the IPsec sessions for the two DMVPN connections from the spokes.

Example:

```
Transit-Hub# show crypto session detail
Crypto session current status
Code: C - IKE Configuration mode, D - Dead Peer Detection
K - Keepalives, N - NAT-traversal, T - cTCP encapsulation
X - IKE Extended Authentication, F - IKE Fragmentation
R - IKE Auto Reconnect, U - IKE Dynamic Route Update
Interface: Tunnel11
Uptime: 1w3d
Session status: UP-ACTIVE
Peer: 40.114.69.24 port 4500 fvrf: (none) ivrf: tvnet-Tun-11
      Phase1_id: 12.1.0.4
      Desc: (none)
  Session ID: 0
  IKEv1 SA: local 10.1.0.4/4500 remote 40.114.69.24/4500 Active
          Capabilities:DN connid:1043 lifetime:18:32:04
  IPSEC FLOW: permit 47 host 10.1.0.4 host 40.114.69.24
        Active SAs: 2, origin: crypto map
        Inbound:  #pkts dec'ed 32 drop 0 life (KB/Sec) 4607996/3474
        Outbound: #pkts enc'ed 32 drop 0 life (KB/Sec) 4607998/3474
```

```
Interface: Tunnel11
Uptime: 1w3d
Session status: UP-ACTIVE
Peer: 168.62.164.228 port 4500 fvrf: (none) ivrf: tvnet-Tun-11
      Phase1_id: 11.1.0.4
      Desc: (none)
  Session ID: 0
  IKEv1 SA: local 10.1.0.4/4500 remote 168.62.164.228/4500 Active
          Capabilities:DN connid:1042 lifetime:18:02:01
  IPSEC FLOW: permit 47 host 10.1.0.4 host 168.62.164.228
        Active SAs: 2, origin: crypto map
        Inbound:  #pkts dec'ed 32 drop 0 life (KB/Sec) 4607970/2427
        Outbound: #pkts enc'ed 32 drop 0 life (KB/Sec) 4607982/2427
```

Step 4 Run the `show dmvpn` command to view the status of the DMVPN on the device.

Example:

```
Transit-Hub# show dmvpn
Legend: Attrb --> S - Static, D - Dynamic, I - Incomplete
        N - NATed, L - Local, X - No Socket
        T1 - Route Installed, T2 - Nexthop-override
        C - CTS Capable, I2 - Temporary
        # Ent --> Number of NHRP entries with same NBMA peer
        NHS Status: E --> Expecting Replies, R --> Responding, W --> Waiting
        UpDn Time --> Up or Down Time for a Tunnel
==========================================================================
Interface: Tunnel11, IPv4 NHRP Details
Type:Hub, NHRP Peers:2,
 # Ent  Peer NBMA Addr Peer Tunnel Add State  UpDn Tm Attrb
 ----- --------------- --------------- ----- -------- -----
     1 40.114.69.24       172.16.1.137    UP     1w3d    DN
     1 168.62.164.228     172.16.1.147    UP     1w3d    DN
```

Step 5 Run the `show vrf` command to view the display routes from each of the spokes on the transit VNet.

Example:

```
Transit-Hub# show vrf
  Name                             Default RD            Protocols   Interfaces
  tvnet-Tun-11                     64512:11              ipv4        Tu11
```

Step 6 Run the `show ip eigrp vrf <vrf-name> neighbors` command to view the status of the EIGRP neighbors.

Example:

```
Transit-Hub# show ip eigrp vrf tvnet-Tun-11 neighbors
EIGRP-IPv4 Neighbors for AS(64512) VRF(tvnet-Tun-11)
H   Address                 Interface              Hold Uptime   SRTT   RTO  Q  Seq
                                                   (sec)         (ms)       Cnt Num
1   172.16.1.137            Tu11                     14 1w3d       13  1398  0  12
0   172.16.1.147            Tu11                     10 1w3d       12  1398  0  12
```

Step 7 Run the `show ip route vrf <vrf-name>` command to view the route specific to a VRF.

Example:

```
Transit-Hub# show ip route vrf tvnet-Tun-11
Routing Table: tvnet-Tun-11
Codes: L - local, C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2
       i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2
       ia - IS-IS inter area, * - candidate default, U - per-user static route
       o - ODR, P - periodic downloaded static route, H - NHRP, l - LISP
       a - application route
       + - replicated route, % - next hop override, p - overrides from PfR
Gateway of last resort is not set
      11.0.0.0/24 is subnetted, 2 subnets
D EX     11.1.0.0 [170/26880256] via 172.16.1.147, 1w1d, Tunnel11
D EX     11.1.1.0 [170/26880256] via 172.16.1.147, 1w1d, Tunnel11
      12.0.0.0/24 is subnetted, 2 subnets
D EX     12.1.0.0 [170/26880256] via 172.16.1.137, 1w1d, Tunnel11
D EX     12.1.1.0 [170/26880256] via 172.16.1.137, 1w1d, Tunnel11
      172.16.0.0/16 is variably subnetted, 2 subnets, 2 masks
C        172.16.1.0/24 is directly connected, Tunnel11
L        172.16.1.1/32 is directly connected, Tunnel11
D EX  192.168.35.0/24 [170/26905600] via 172.16.1.147, 1w1d, Tunnel11
                      [170/26905600] via 172.16.1.137, 1w1d, Tunnel11
```

Note To access the debug log files, go to the private storage account, and go to **Files** > **Transit VNet FileShare** > **Autoscaler dir** > **logs dir**.

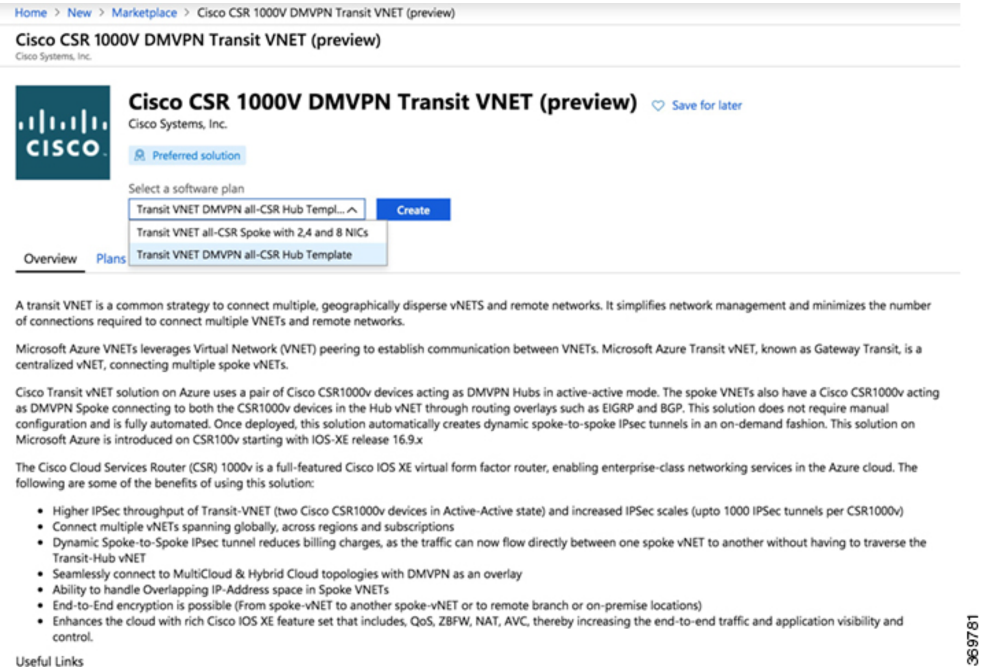
Deploy a Azure DMVPN Spoke VNet

Before you begin

Ensure that your Hub is created successfully before you create a Spoke for the transit VNet solution with Autoscaler.

Step 1 From the Microsoft Azure Marketplace, search and select the appropriate **Cisco CSR 1000v DMVPN Transit VNet**.

Step 2 Click the template, and select the **Transit VNet all-CSR Spoke with 2,4 and 8 NICs** Spoke option from the drop-down list.


Home > New > Marketplace > Cisco CSR 1000V DMVPN Transit VNET (preview)
Cisco CSR 1000V DMVPN Transit VNET (preview)
Cisco Systems, Inc.
Cisco CSR 1000V DMVPN Transit VNET (preview)
Save for later
Cisco Systems, Inc.
Preferred solution
Select a software plan
Transit VNET DMVPN all-CSR Hub Templ...
Create
Transit VNET all-CSR Spoke with 2,4 and 8 NICs
Transit VNET DMVPN all-CSR Hub Template
Overview
Plans
369781

A transit VNET is a common strategy to connect multiple, geographically disperse vNETS and remote networks. It simplifies network management and minimizes the number of connections required to connect multiple VNETs and remote networks.

Microsoft Azure VNETs leverages Virtual Network (VNET) peering to establish communication between VNETs. Microsoft Azure Transit vNET, known as Gateway Transit, is a centralized vNET, connecting multiple spoke vNETs.

Cisco Transit vNET solution on Azure uses a pair of Cisco CSR1000v devices acting as DMVPN Hubs in active-active mode. The spoke VNETs also have a Cisco CSR1000v acting as DMVPN Spoke connecting to both the CSR1000v devices in the Hub vNET through routing overlays such as EIGRP and BGP. This solution does not require manual configuration and is fully automated. Once deployed, this solution automatically creates dynamic spoke-to-spoke IPsec tunnels in an on-demand fashion. This solution on Microsoft Azure is introduced on CSR100v starting with IOS-XE release 16.9.x

The Cisco Cloud Services Router (CSR) 1000v is a full-featured Cisco IOS XE virtual form factor router, enabling enterprise-class networking services in the Azure cloud. The following are some of the benefits of using this solution:

- Higher IPSec throughput of Transit-VNET (two Cisco CSR1000v devices in Active-Active state) and increased IPSec scales (upto 1000 IPSec tunnels per CSR1000v)
- Connect multiple vNETs spanning globally, across regions and subscriptions
- Dynamic Spoke-to-Spoke IPsec tunnel reduces billing charges, as the traffic can now flow directly between one spoke vNET to another without having to traverse the Transit-Hub vNET
- Seamlessly connect to MultiCloud & Hybrid Cloud topologies with DMVPN as an overlay
- Ability to handle Overlapping IP-Address space in Spoke VNETs
- End-to-End encryption is possible (From spoke-vNET to another spoke-vNET or to remote branch or on-premise locations)
- Enhances the cloud with rich Cisco IOS XE feature set that includes, QoS, ZBFW, NAT, AVC, thereby increasing the end-to-end traffic and application visibility and control.

Useful Links

Step 3 Click **Create**.

Step 4 In the **Basics settings** screen, ensure that you specify the following configuration details:

a) **Filename** – Specify the name of the Transit VNet in this field.

b) **Transit VNet Storage Name** – This is the same as the TVNET Storage Account value from the Hub configuration. This name is derived from the Transit VNet name with ‘strg’ keyword added.

c) **Storage Key** – To access the Storage Key, search and click the public Hub and click the **Access Key** option.

Step 5 Configure the other values in the **BASICS** settings screen, and click **OK**.

Step 6 In the **Cisco CSR Settings** screen, configure the following Autoscaler-specific paramteres:

a) **Is Spoke Deployment Part of Autoscaler Enabled Transit VNet Hub**: Select Yes from the drop-down list to enable Spoke deployment.

b) **Spoke Storage Account**: Enter the name of the Transit VNet Storage Name that you have entered at the beginning of this screen.

c) **Azure ID App ID**: The client ID or the Image ID that is displayed on the screen when you create the service applicationthrough the CLI.

d) **Enter Azure ID App Password**: Enter the password for your App ID here.

a) **Public IP address**:

Note To configure the other the parameters, see *How to Deploy a Cisco CSR 1000v on Microsoft Azure*.

Availability Zones are not yet fully supported with all the regions in Microsoft Azure. The solution template hence does not have an option for availability zones, but resiliency is taken care using *Availability-Sets*. See the Microsoft Azures documentation here: https://docs.microsoft.com/en-us/azure/availability-zones/az-overview.

Step 7 Click the arrow next to Virtual Network to specify values for the virtual network and click **OK**.

- **Address Space** - Enter the address of the virtual network using Classless Inter-Domain Routing (CIDR) notation.

Note The VNET CIDR denotes the physical ip-address subnets that will be used for Cisco CSR1000v devices in the TVNET-HUB. The CIDR block is usually a /16 subnet which will be subnetted further into two /24 subnets. The first 3 IP addresses of each subnet will be reserved for Azure Route-Table and other services. The IP allocations begin from the 4th ip of the subnet and this will be automatically mapped to the “public ip” that is assigned dynamically. The “public ip” enables access to Internet, hence becomes the NBMA address in the DMVPN scenario.

Step 8 Click the arrow next to configure the subnets, and click **OK**.

Step 9 In the **Summary** screen, review the configured parameters. After you validate the template, click **OK**.

Step 10 Click **Create** to deploy the TVNet Spoke solution.

Note For every additional Spoke that you want to create, follow steps 1 through 10.

CHAPTER 11

Configure L2 Extension for Public Cloud

This chapter describes how to enable enterprise and cloud providers to configure an L2 extension for public clouds with CSR 1000V instances using LISP. Use the command-line interface to extend a layer 2 domain between your public cloud network and the enterprise network.

The following are some of the terminologies and concepts that you should be aware before you configure the LISP Layer 2 Extension:

- **Locator/ID Separation Protocol (LISP)**: LISP is a network architecture and protocol that implements the use of two namespaces instead of a single IP address:
    - Endpoint identifiers (EIDs) - assigned to end hosts.
    - Routing locators (RLOCs) - assigned to devices (primarily routers) that make up the global routing system.
- **LISP-enabled virtualized router**: A virtual machine or appliance that supports routing and LISP functions, including host mobility.
- **Endpoint ID (EID)**: An EID is an IPv4 or IPv6 address used in the source and destination address fields of the first (most inner) LISP header of a packet.
- **Routing locator (RLOC)**: The IPv4 or IPv6 addresses that are used to encapsulate and transport the flow between the LISP nodes. An RLOC is the output of an EID-to-RLOC mapping lookup.
- **Egress Tunnel Router (ETR)**: An ETR is a device that is the tunnel endpoint and connects a site to the LISP-capable part of a core network (such as the Internet), publishes EID-to-RLOC mappings for the site, responds to Map-Request messages, and decapsulates and delivers LISP-encapsulated user data to the end systems at the site. During operation, an ETR sends periodic Map-Register messages to all its configured map servers. These Map-Register messages contain all the EID-to-RLOC entries for the EID-numbered networks that are connected to the ETR’s site.
- **Ingress Tunnel Router (ITR)**: An ITR is a device that is the tunnel start point. An ITR is responsible for finding EID-to-RLOC mappings for all traffic destined for LISP-capable sites. When the ITR receives a packet destined for an EID, it first looks for the EID in its mapping cache. If the ITR finds a match, it encapsulates the packet inside a LISP header with one of its RLOCs as the IP source address and one of the RLOCs from the mapping cache entry as the IP destination. The ITR then routes the packet normally.
- **xTR**: A generic name for a device performing both Ingress Tunnel Router (ITR) and Egress Tunnel Router (ETR) functions.

- **PxTR**: The point of interconnection between an IP network and a LISP network, playing the role of ITR and ETR at this peering point.
- **Map-Server (MS)**: An MS is a LISP Infrastructure device that LISP site ETRs register to with their EID prefixes. An MS implements part of the distributed LISP mapping database by accepting registration requests from its client egress tunnel routers (ETRs), aggregating the successfully registered EID prefixes of those ETRs, and advertising the aggregated prefixes into the alternative logical topology (ALT) with border gateway protocol (BGP).

  In a small private mapping system deployment, an MS may be configured to stand alone (or there may be several MSs) with all ETRs configured to register to each MS. If more than one, all MSs have full knowledge of the mapping system in a private deployment.

  In a larger or public mapping system deployment, an MS is configured with a partial mesh of generic routing encapsulation (GRE) tunnels and BGP sessions to other map server systems
- **Map-Resolver (MR)**: An MR is a LISP Infrastructure device to which the ITRs send LISP Map-Request queries when resolving EID-to-RLOC mappings. MRs receive the request and select the appropriate map server

For detailed overview information on LISP and the terminologies, see Locator ID Separation Protocol Overview.

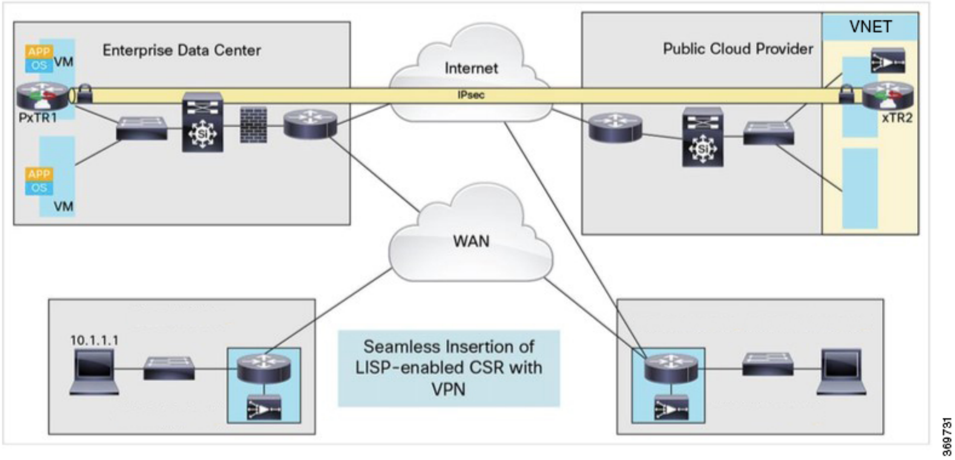
Information about Configuring LISP Layer 2 Extension

You can deploy Cisco CSR 1000v instances on public, private, and hybrid clouds. When enterprises move to a hybrid cloud, they need to migrate the servers to the cloud without making any changes to the servers. Enterprises may want to use the same server IP address, subnet mask, default gateway configurations, and their own IP addressing scheme in the cloud, and not be limited by the addressing scheme of the cloud provider infrastructure.

To fulfill this requirement, you can use LISP, which is an architecture that allows you to separate the location (enterprise data center or public cloud) and the identity (server IP address) so that you can create new servers on the cloud with the same IP address. In the LISP architecture, the endpoint ID-to-router locator (EID-to-RLOC) mapping of the server is updated to reflect the new location that is moved to the cloud. Further, no changes are required to the end systems, users, or servers because LISP handles the mapping between the identity and the location.

LISP operates as an overlay, encapsulating the original packet from the server into a User Datagram Protocol (UDP) packet along with an additional outer IPv4 or IPv6 header. This encapsulation holds the source and destination router locators and allows the server administrators to address the server in the cloud according to their own IP addressing scheme, independent of the cloud provider's addressing structure.

From the 16.12.1 release, you can configure Layer 2 Extension on CSR 1000v instances running on Microsoft Azure, where the CSR 1000v instance acts as the bridge between the enterprise data center and the public cloud. By configuring the Layer 2 Extension, you can extend your Layer 2 networks in the private data center to a public cloud to achieve host reachability between your site and the public cloud. You can also enable the migration of your application workload between the data center and the public cloud.



Benefits

- Move the Public IP addresses between different geographic locations or split them between different public clouds. In either case, the LISP IP-Mobility solution provides optimal routing between clients on the Internet and the public IP address that has moved, regardless of the location. To know more about achieving IP mobility for the Azure cloud, see Achieving IP Mobility.
- Carry out data migration with ease and optimize the workload IP address in your network. Usually, IP address changes cause complexity and additional delays in a solution. By using L2 extension for cloud, you can migrate workloads while retaining the original IP address without any network constraints. To learn more about this use case, see Data Migration Use Case.
- Virtually add a VM that is on the provider site to facilitate cloud bursting to virtually insert a VM in the Enterprise server while the VM runs on the provider site.
- Provide backup services for partial disaster recovery and disaster avoidance.

Prerequisites for configuring LISP Layer 2 Extension

- Ensure that the underlay for your solution is ready before you configure the L2 Extension.
- Since clouds do not support Address Resolution Protocol (ARP), and the cloud infrastructure is not aware of the hosts in the remote site, you must add a virtual IP to help the cloud route the packets appropriately to the edge router. To add a virtual or alias IP, see Add an IP address for an Azure interface.
- Each CSR 1000V router must be configured with one external IP address. In this case, an IPsec tunnel is built either between the IP addresses of the two CSR 1000v instances, or between the CSR 1000v instance and the ASR1k device. Ensure that the IPsec tunnel has a private address.
- Ensure that the IPsec tunnel is working between the IP address of the two CSR 1000v instances or between the CSR 1000v instance and the ASR1k device.
- Depending on your solution, ensure that a ping is successful between: the two CSR 1000v instances, between a CSR 1000v and an ASR1k, and between the VMs and the hosts.

Restrictions for configuring LISP Layer 2 Extension

- If you move a host from the data center to the cloud or vice-versa, you must first add or remove the secondary address from the virtual IP table in the cloud.
- If you move a VM to the cloud, you must initiate packets to the CSR 1000v instance so that the CSR 1000v device realizes that the VM is now added from the data center to the cloud.
- For the 16.12 release, High Availability does not work with the L2 Extension functionality.
- Azure supports a maximum of 256 IPs. The maximum number of hosts on the remote site or the data center is thus 256.

How to configure LISP Layer 2 Extension

To configure the L2 extension functionality, you must first deploy the CSR 1000v instance on Microsoft Azure and configure the instance as an xTR. You must then configure the mapping system to complete the deployment.

The LISP site uses the CSR 1000v instance configured as both an ITR and an ETR (also known as an xTR) with two connections to upstream providers. The LISP site then registers to the standalone device that you have configured as the map resolver/map server (MR/MS) in the network core. The mapping system performs LISP encapsulation and de-encapsulation of the packets that are going to the migrated public IPs within Azure. Optionally, for traffic that is leaving Azure, whenever a route to the destination is not found on the CSR routing table, the CSR 1000v instance routes that traffic through the PxTR at the enterprise data center.

Perform the following steps to enable and configure the LISP xTR functionality when using a LISP map server and map resolver for mapping services:

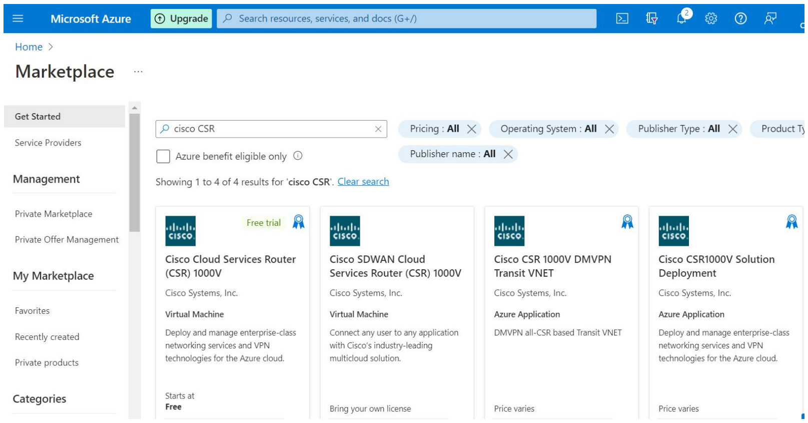
Deploy a CSR 1000v with Multiple Interfaces on Microsoft Azure

Perform the following steps to deploy a Cisco CSR1000V instance with multiple interfaces on Microsoft Azure.

Step 1 Log in to the Microsoft Azure Marketplace.

Step 2 On the Search bar, search for `Cisco CSR`.

Step 3 The system displays the various offerings under Cisco CSR1000V. Select **Cisco CSR1000V Solution Deployment**, and click **Create**.



cisco CSR

Showing 1 to 4 of 4 results for 'cisco CSR'.

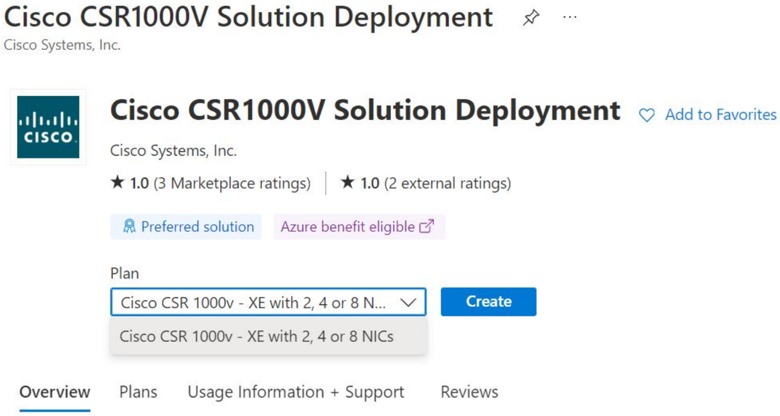
Cisco Cloud Services Router (CSR) 1000V
Cisco Systems, Inc.
Virtual Machine
Deploy and manage enterprise-class networking services and VPN technologies for the Azure cloud.

Cisco SDWAN Cloud Services Router (CSR) 1000V
Cisco Systems, Inc.
Virtual Machine
Connect any user to any application with Cisco's industry-leading multicloud solution.

Cisco CSR 1000V DMVPN Transit VNET
Cisco Systems, Inc.
Azure Application
DMVPN all-CSR based Transit VNET

Cisco CSR1000V Solution Deployment
Cisco Systems, Inc.
Azure Application
Deploy and manage enterprise-class networking services and VPN technologies for the Azure cloud.

Step 4 In the Cisco CSR1000V Solution Deployment page, the **Cisco CSR 1000v - XE with 2, 4 or 8 NICs** solution is available in the **Plan** drop-down field. Click **Create**.


Cisco CSR1000V Solution Deployment
Cisco Systems, Inc.
Cisco CSR1000V Solution Deployment
Add to Favorites
Cisco Systems, Inc.
★ 1.0 (3 Marketplace ratings)
★ 1.0 (2 external ratings)
Preferred solution
Azure benefit eligible
Plan
Create
Overview
Plans
Usage Information + Support
Reviews

Plan: Cisco CSR 1000v - XE with 2, 4 or 8 N...

Cisco CSR 1000v - XE with 2, 4 or 8 NICs

Step 5 In the Basics page, enter the following details:

a) **Subscription Name**: The name of your subscription. A default subscription name is available. You can modify the subscription name, if required.

b) **Resource Group**: A container that holds the resources for your solution. From this drop-down field, choose either **Create New** or **Select Existing**. You can create a Cisco CSR 1000V instance only in a new Resource Group or in a completely empty existing resource group. To remove a Resource Group, first delete the Cisco CSR 1000V VM and then delete the Resource Group.

c) **Region**: The region or location where you are performing this deployment. From this drop-down field choose your region.

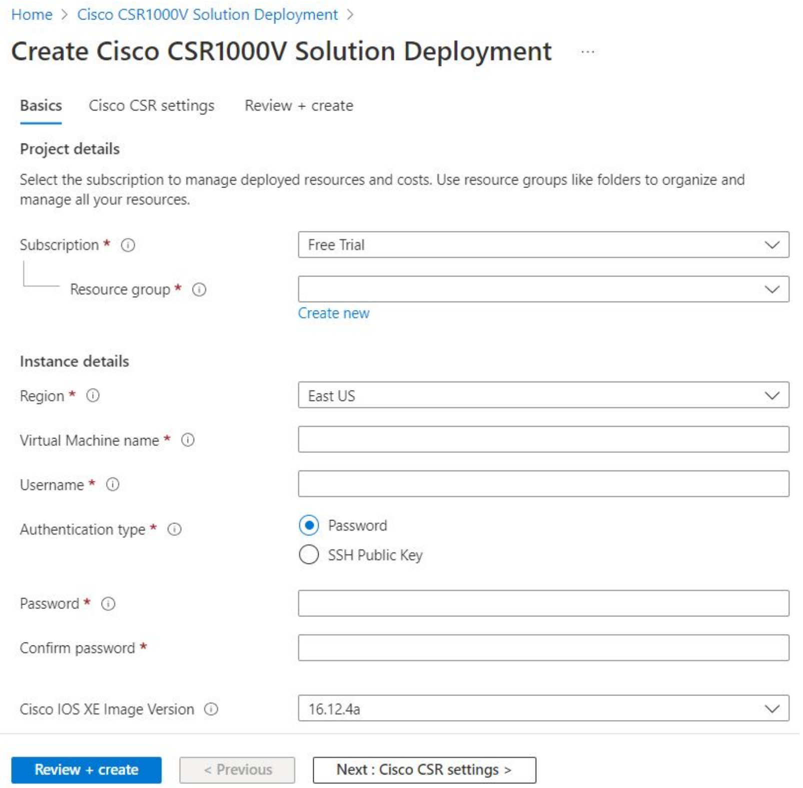
d) **Virtual Machine Name**: The name of the cloud-based network used by Microsoft Azure to represent a private network. Enter a name for the virtual machine.

e) **Username**: The username for your VM using which you can log in to the Cisco CSR 1000V instance. Enter a user name for your VM.

Note For Cisco IOS XE versions 3.16 and 16.4, if you're planning to choose SSH Key as an authentication type, enter the **Username** as `azureuser`.

f) **Authentication type**: The authentication type for the administrator account. You can use a username and password or an SSH key for authentication. If you select the **SSH Key** option, select the **SSH Public Key Source** and provide the **Key Pair Name**. If you select the **Password** opition, enter a password for authentication.

g) **Cisco IOS XE Image Version**: The version of the Cisco IOS XE software. From this drop-down field, choose your Cisco IOS XE version.



Home > Cisco CSR1000V Solution Deployment >

Create Cisco CSR1000V Solution Deployment ...

Basics | Cisco CSR settings | Review + create

Project details

Select the subscription to manage deployed resources and costs. Use resource groups like folders to organize and manage all your resources.

Subscription * ⓘ: Free Trial

Resource group * ⓘ:

Create new

Instance details

Region * ⓘ: East US

Virtual Machine name * ⓘ:

Username * ⓘ:

Authentication type * ⓘ: ● Password ○ SSH Public Key

Password * ⓘ:

Confirm password *:

Cisco IOS XE Image Version ⓘ: 16.12.4a

Step 6 Click **Next** and proceed to the Cisco CSR Settings page.

Step 7 In the **Cisco CSR Settings** page, enter the following details:

a) **Number of Network Interfaces in CSR**: The number of network interfaces you want to attach to the VM. From the drop-down list, choose the number of interfaces: 2, 4, or 8.

b) **License Type**: The license type. From this drop-down field, choose either **BYOL** or **PAYG** as the license type.

c) **Managed Disk**: The option that allows you to specify whether you want Azure to manage the disk for you. Select **Enabled**.

d) **Virtual machine size**: The size of the VM to provision. Select the appropriate virtual machine size. Based on the number of interfaces that you are using, select the appropriate virtual machine size. Microsoft Azure supports different image types with different performance expectations.

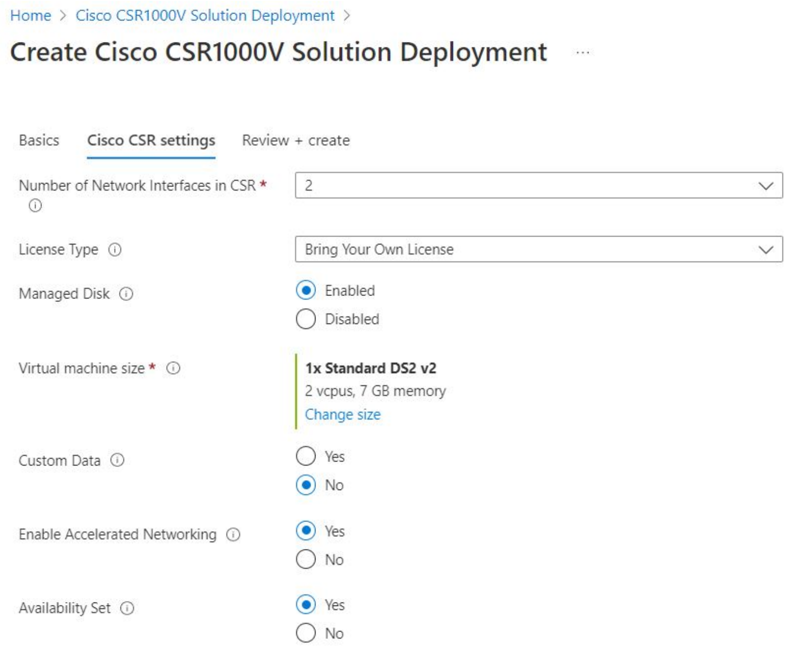
To view the supported instance types and the virtual machine sizes, see the following links:

- Dv2 and DSv2 series
- Fsv2 series

e) **Custom Data**: The provisioning configuration information for your VM. Select **Yes** if you want to provide a bootstrap configuration file for your Cisco CSR 1000V instance. For further information about providing a bootstrap configuration file for the Cisco CSR 1000V instance, see: Deploying a Cisco CSR 1000v VM on Microsoft Azure using a Day 0 Bootstrap File and customdata-examples.

f) **Enable Accelerated Networking**: The option to enable single root I/O virtualization (SR-IOV) to your VM. Select **Yes** to enable the accelerated networking feature.

g) **Availability Set**: The logical grouping of resources to create an availability set. Select **Yes** to create a new availability set.



Home > Cisco CSR1000V Solution Deployment >

Create Cisco CSR1000V Solution Deployment ...

Number of Network Interfaces in CSR * ⓘ: 2

License Type ⓘ: Bring Your Own License

Managed Disk ⓘ: ● Enabled ○ Disabled

Virtual machine size * ⓘ: 1x Standard DS2 v2, 2 vcpus, 7 GB memory, Change size

Custom Data ⓘ: ○ Yes ● No

Enable Accelerated Networking ⓘ: ● Yes ○ No

Availability Set ⓘ: ● Yes ○ No

h) **Availability Set name**: The name of your availability set. Enter a name for your availability set.

i) **Availability Set Fault Domain Count**: The group of VMs that share a common power source and network switch. Availability sets arrange virtual machines across fault domains. In the field, enter the availability set fault domain count.

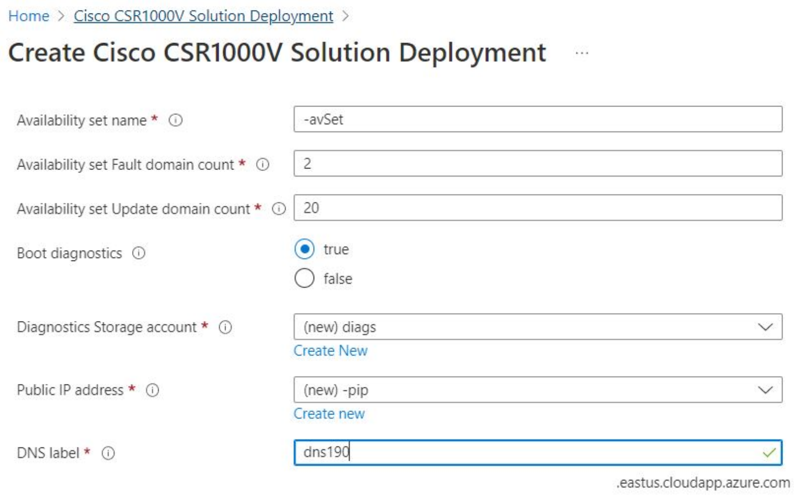
j) **Availability Set Update Domain Count**: A group of VMs and underlying physical hardware that can be rebooted at the same time. Enter the availability set update domain count.

k) **Boot diagnostics**: The option that enables you to capture the boot logs and screenshots of the virtual machine. Select **True** to enable boot diagnostics. For more information on boot diagnostics, see Microsoft Azure Resources, on page 2.

l) **Diagnostics Storage account**: The storage account for the boot diagnostics. Enter the storage account name. For more information on storage accounts, see Microsoft Azure Resources, on page 2.

m) **Public IP Address**: The public IP address name. For more information on the public IP address, see Microsoft Azure Resources, on page 2.

n) **DNS label**: The name of the public IP address to be assigned to the Cisco CSR 1000V instance. A default value for the DNS label is shown in the text box which is the VM name followed by "`-dns`". Change the name of the DNS label, if required.



Home > Cisco CSR1000V Solution Deployment >

Create Cisco CSR1000V Solution Deployment ...

Availability set name * ⓘ: -avSet

Availability set Fault domain count * ⓘ: 2

Availability set Update domain count * ⓘ: 20

Boot diagnostics ⓘ: ● true ○ false

Diagnostics Storage account * ⓘ: (new) diags
Create New

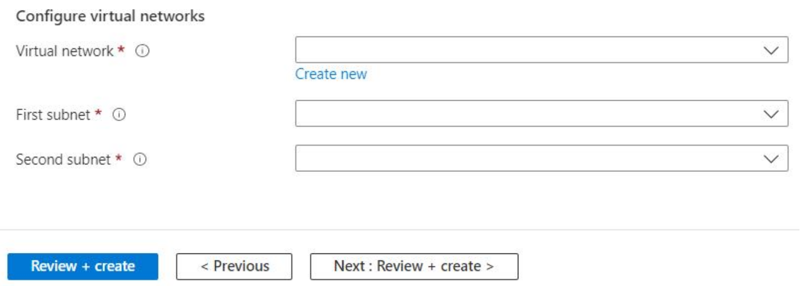
Public IP address * ⓘ: (new) -pip
Create new

DNS label * ⓘ: dns190 ✓
.eastus.cloudapp.azure.com

Step 8 In the Configure Virtual Networks section, specify the following details:

a) **Virtual network**: From the drop-down field, choose either **Create New** or **Use existing**. For a new virtual network, enter the name and IP address.

b) **First Subnet**/**Second Subnet**: The name and the IP address for your subnets. For more information on subnets, see "Interfaces" in Microsoft Azure Resources, on page 2.



Configure virtual networks

Virtual network * ⓘ:
Create new

First subnet * ⓘ:

Second subnet * ⓘ:

Step 9 Click **Next: Review + Create**.

Step 10 The system displays the summary of all your settings. Review your settings and then click **Next**.

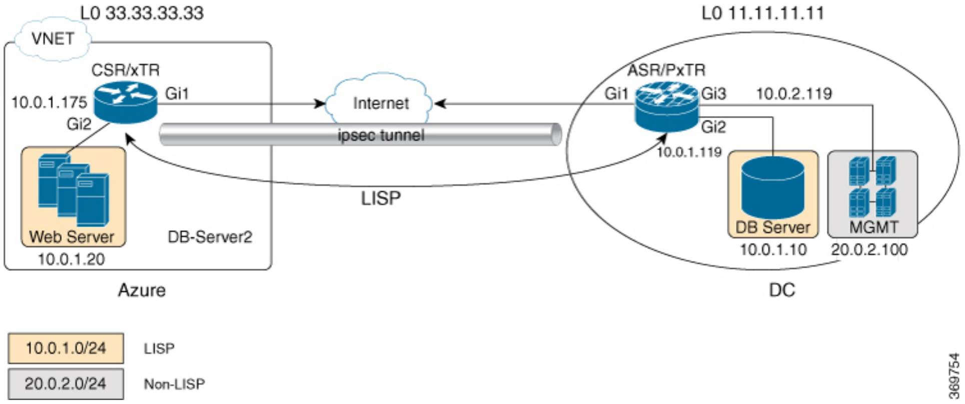
Step 11 Click **Create**

The VM is created and the purchase is confirmed.

Step 12 To verify the successful creation of your VM, click **Virtual machines** in the left hand panel. After a few minutes, the status of the recently created VM changes from “Creating” to “Running”. Make a note of the Public IP address name.

Configure a tunnel between CSR 1000v on Azure and CSR 1000v on the enterprise system

The communication between the CSR 1000v instance deployed within the enterprise data center and the CSR 1000v instance deployed within the public cloud is secured by an IP Security (IPsec) tunnel established between them. The LISP-encapsulated traffic is protected with the IPsec tunnel that provides data origin authentication, integrity protection, anti-reply protection, and confidentiality between the public cloud and the enterprise.



Step 1 Configure a CSR 1000v instance on Microsoft Azure.

Run the **interface Loopback** command. Loopback is used as the LISP RLOC, which identifies where the migrated customer IP space is located.

Run the **interface Tunnel** command to connect to the CSR 1000v instance on the cloud.

```
interface Loopback1
 ip address 33.33.33.33 255.255.255.255
!
interface Tunnel2
 ip address 30.0.0.2 255.255.255.0
 tunnel source GigabitEthernet1
 tunnel mode ipsec ipv4
 tunnel destination 173.39.145.79
 tunnel protection ipsec profile p2p_pf1
!
interface GigabitEthernet2
 ip address 10.10.10.140 255.255.255.0
 negotiation auto
 lisp mobility subnet1 nbr-proxy-reply requests 3
 no mop enabled
 no mop sysid
!
```

Step 2 Configure a second CSR 1000v instance on the enterprise site.

```
interface Loopback1
 ip address 11.11.11.11 255.255.255.255

interface Tunnel2
 ip address 30.0.0.1 255.255.255.0
 tunnel source GigabitEthernet2
 tunnel mode ipsec ipv4
 tunnel destination 52.14.116.161
 tunnel protection ipsec profile p2p_pf1
!
!
interface GigabitEthernet3
 ip address 10.10.10.2 255.255.255.0
 negotiation auto
 lisp mobility subnet1 nbr-proxy-reply requests 3
 no mop enabled
 no mop sysid
!
```

Configure LISP xTR on the CSR1000v instance running on Azure

To configure LISP xTR on the CSR instance running on the service provider, follow the configuration steps in the Configuring LISP (Location ID Separation Protocol) section.

The CSR 1000v instance on Azure uses the enterprise LISP router as the proxy ETR. Whenever the routing table points to the default route, it sends the traffic to the PETR.

Run the router **lisp command** to enable LISP. Execute the **itr map resolver** and the **itr map server** commands, to configure the CSR 1000v instance on the enterprise as the map server/map resolver.

Example:

```
router lisp
 locator-set azure
  33.33.33.33 priority 1 weight 100
  exit-locator-set
 !
 service ipv4
  itr map-resolver 11.11.11.11
  itr
  etr map-server 11.11.11.11 key cisco
  etr
  use-petr 11.11.11.11
  exit-service-ipv4
 !
 instance-id 0
  dynamic-eid subnet1
   database-mapping 10.10.10.0/24 locator-set azure
   map-notify-group 239.0.0.1
   exit-dynamic-eid
  !
  service ipv4
   eid-table default
   exit-service-ipv4
  !
  exit-instance-id
 !
 exit-router-lisp
!
router ospf 11
 network 30.0.0.2 0.0.0.0 area 11
 network 33.33.33.33 0.0.0.0 area 11
!

router lisp
 locator-set dmz
  11.11.11.11 priority 1 weight 100
  exit-locator-set
 !
 service ipv4
  itr map-resolver 11.11.11.11
  etr map-server 11.11.11.11 key cisco
  etr
  proxy-etr
  proxy-itr 11.11.11.11
  map-server
  map-resolver
  exit-service-ipv4
 !
 instance-id 0
  dynamic-eid subnet1
   database-mapping 10.10.10.0/24 locator-set dmz
   map-notify-group 239.0.0.1
   exit-dynamic-eid
  !
  service ipv4
   eid-table default
   exit-service-ipv4
  !
  exit-instance-id
 !
 site DATA_CENTER
  authentication-key cisco
  eid-record 10.10.10.0/24 accept-more-specifics
  exit-site
 !
 exit-router-lisp
!
router ospf 11
 network 11.11.11.11 0.0.0.0 area 11
 network 30.0.0.1 0.0.0.0 area 11
!

!
!
```

Verify the LISP Layer 2 Traffic between CSR 1000v on Azure and CSR 1000v on the enterprise system

Run the following show lisp commands to verify the LISP Layer 2 traffic:

Example:

```
csr-azure#show ip lisp database
LISP ETR IPv4 Mapping Database for EID-table default (IID 0), LSBs: 0x1
Entries total 2, no-route 0, inactive 0

10.0.1.1/32, dynamic-eid subnet1, inherited from default locator-set dc
  Locator  Pri/Wgt  Source     State
33.33.33.33    1/100  cfg-addr   site-self, reachable
10.0.1.20/32, dynamic-eid subnet1, inherited from default locator-set dc
  Locator  Pri/Wgt  Source     State
33.33.33.33    1/100  cfg-addr   site-self, reachable
csr-azure#show ip lisp map-cache
LISP IPv4 Mapping Cache for EID-table default (IID 0), 4 entries

0.0.0.0/0, uptime: 00:09:49, expires: never, via static-send-map-request
  Negative cache entry, action: send-map-request
10.0.1.0/24, uptime: 00:09:49, expires: never, via dynamic-EID, send-map-request
  Negative cache entry, action: send-map-request
10.0.1.4/30, uptime: 00:00:55, expires: 00:00:57, via map-reply, forward-native
```

```
  Encapsulating to proxy ETR
10.0.1.100/32, uptime: 00:01:34, expires: 23:58:26, via map-reply, complete
  Locator  Uptime    State      Pri/Wgt     Encap-IID
11.11.11.11  00:01:34  up           1/100       -
csr-azure#show lisp dynamic-eid detail
% Command accepted but obsolete, unreleased or unsupported; see documentation.

LISP Dynamic EID Information for VRF "default"

Dynamic-EID name: subnet1
  Database-mapping EID-prefix: 10.0.1.0/24, locator-set dc
  Registering more-specific dynamic-EIDs
  Map-Server(s): none configured, use global Map-Server
  Site-based multicast Map-Notify group: 239.0.0.1
  Number of roaming dynamic-EIDs discovered: 2
  Last dynamic-EID discovered: 10.0.1.20, 00:01:37 ago
    10.0.1.1, GigabitEthernet2, uptime: 00:09:23
      last activity: 00:00:42, discovered by: Packet Reception
    10.0.1.20, GigabitEthernet2, uptime: 00:01:37
      last activity: 00:00:40, discovered by: Packet Reception

CSR-DC#show ip lisp
CSR-DC#show ip lisp data
CSR-DC#show ip lisp database
LISP ETR IPv4 Mapping Database for EID-table default (IID 0), LSBs: 0x1
Entries total 1, no-route 0, inactive 0

10.0.1.100/32, dynamic-eid subnet1, inherited from default locator-set dc
  Locator  Pri/Wgt  Source     State
11.11.11.11    1/100  cfg-addr   site-self, reachable
CSR-DC#show ip lisp
CSR-DC#show ip lisp map
CSR-DC#show ip lisp map-cache
LISP IPv4 Mapping Cache for EID-table default (IID 0), 2 entries

10.0.1.0/24, uptime: 1d08h, expires: never, via dynamic-EID, send-map-request
  Negative cache entry, action: send-map-request
10.0.1.20/32, uptime: 00:00:35, expires: 23:59:24, via map-reply, complete
  Locator  Uptime    State      Pri/Wgt     Encap-IID
33.33.33.33  00:00:35  up           1/100

CSR-DC#show lisp dynamic-eid detail
% Command accepted but obsolete, unreleased or unsupported; see documentation.

LISP Dynamic EID Information for VRF "default"

Dynamic-EID name: subnet1
  Database-mapping EID-prefix: 10.0.1.0/24, locator-set dc
  Registering more-specific dynamic-EIDs
  Map-Server(s): none configured, use global Map-Server
  Site-based multicast Map-Notify group: 239.0.0.1
  Number of roaming dynamic-EIDs discovered: 1
  Last dynamic-EID discovered: 10.0.1.100, 1d08h ago
    10.0.1.100, GigabitEthernet2, uptime: 1d08h
      last activity: 00:00:47, discovered by: Packet Reception

CSR-DC#show lisp site
LISP Site Registration Information
* = Some locators are down or unreachable
# = Some registrations are sourced by reliable transport

Site Name      Last      Up     Who Last             Inst     EID Prefix
               Register         Registered           ID
dc             never     no     --                            10.0.1.0/24
               00:08:41  yes#   33.33.33.33                   10.0.1.1/32
               00:01:00  yes#   33.33.33.33                   10.0.1.20/32
               1d08h     yes#   11.11.11.11                   10.0.1.100/32
CSR-DC#show ip cef 10.0.1.20
10.0.1.20/32
  nexthop 33.33.33.33 LISP0
CSR-DC#
```

```
newlispcsr#show lisp instance-id 0 ipv4 database
LISP ETR IPv4 Mapping Database for EID-table default (IID 0), LSBs: 0x1
Entries total 7, no-route 0, inactive 4

10.20.20.1/32, locator-set dc
Locator Pri/Wgt Source State
3.3.3.3 1/100 cfg-addr site-self, reachable
10.230.1.5/32, dynamic-eid subnet1, inherited from default locator-set dc
Locator Pri/Wgt Source State
3.3.3.3 1/100 cfg-addr site-self, reachable
10.230.1.6/32, Inactive, expires: 01:20:16
10.230.1.7/32, Inactive, expires: 01:20:16
10.230.1.8/32, dynamic-eid subnet1, inherited from default locator-set dc
Locator Pri/Wgt Source State
3.3.3.3 1/100 cfg-addr site-self, reachable
10.230.1.31/32, Inactive, expires: 01:21:52
10.230.1.32/32, Inactive, expires: 01:20:16
newlispcsronprem#show lisp instance-id 0 ipv4 map
newlispcsr#show lisp instance-id 0 ipv4 map-cache
LISP IPv4 Mapping Cache for EID-table default (IID 0), 6 entries

10.20.0.0/16, uptime: 22:39:53, expires: never, via static-send-map-request
Negative cache entry, action: send-map-request
10.230.1.0/24, uptime: 22:39:53, expires: never, via dynamic-EID, send-map-request
Negative cache entry, action: send-map-request
10.230.1.6/32, uptime: 22:37:05, expires: never, via away, send-map-request
Negative cache entry, action: send-map-request
10.230.1.7/32, uptime: 22:37:05, expires: never, via away, send-map-request
Negative cache entry, action: send-map-request
10.230.1.31/32, uptime: 22:38:14, expires: 01:21:45, via map-reply, complete
Locator Uptime State Pri/Wgt Encap-IID
11.11.11.11 22:38:14 up 1/100 -
10.230.1.32/32, uptime: 22:37:05, expires: never, via away, send-map-request
Negative cache entry, action: send-map-request
```