Pulse Connect Secure Virtual Appliance on Microsoft Azure Deployment Guide

Release 8.3R3
Document Revision 1.0
Published Date October 2017
## Table of Contents

Overview ........................................................................................................................................... 7
  About This Guide .............................................................................................................................. 7
  Assumptions .................................................................................................................................. 7
Prerequisites and System Requirements .............................................................................................. 7
  Pulse Connect Secure on Microsoft Azure ...................................................................................... 8
Steps to Deploy Pulse Connect Secure on Azure .................................................................................. 8
  Upload Pulse Connect Secure Virtual Appliance Image to Azure Web Portal ................................. 8
  Upload Azure Resource Manager Template to Azure Account .......................................................... 10
Deploying Pulse Connect Secure on Azure using Azure Portal ........................................................... 12
  Deploying PCS on New Virtual Network ......................................................................................... 12
  Deploying PCS on an Existing Virtual Network ............................................................................. 15
Deploying Pulse Connect Secure on Azure using Azure CLI ................................................................. 19
Pulse Connect Secure Provisioning Parameters .................................................................................. 21
Provisioning Pulse Connect Secure with Predefined Configuration .................................................. 22
Configuring Licenses on the Pulse Connect Secure Appliance ............................................................. 23
  Pulse License Server in Corporate Network ..................................................................................... 23
  Pulse License Server in Cloud Network ............................................................................................ 23
    Adding Authentication Code in PCS Admin Console .................................................................... 24
    Including Authentication Code in ARM Template ......................................................................... 24
Managing the Pulse Connect Secure Virtual Appliance ........................................................................ 25
  Accessing the Pulse Connect Secure Virtual Appliance as an Administrator ................................. 25
  Accessing the Pulse Connect Secure Virtual Appliance as an End User ........................................ 26
  Accessing the Pulse Connect Secure Virtual Appliance using SSH Console ............................... 26
    On Linux and Mac OS X ................................................................................................................. 26
    On Windows ................................................................................................................................ 27
System Operations ............................................................................................................................... 28
Network Configuration ........................................................................................................................ 28
  IP Address Assignment for Internal, External and Management Interfaces .................................. 28
  IP Addressing Modes ...................................................................................................................... 28
  Modifying Network Parameters After Deployment ....................................................................... 29
  Controlling the Selection of Internal, External and Management Interfaces ............................... 29
Decommissioning Pulse Connect Secure ............................................................................................. 30
  Delete Entire Resource Group that the Pulse Connect Secure Is In .............................................. 30
  Delete Pulse Connect Secure and Resource It Uses, but not the Other Resources in Resource Group 31
Pricing ............................................................................................................................................... 32
Limitations ....................................................................................................................................... 32
Not Qualified ..................................................................................................................................... 33
Troubleshooting ................................................................................................................................. 33
Appendix A: Network Security Group (NSG) .................................................................................... 34
Appendix B: Pulse Connect Secure Resource Manager Template .................................................... 38

© 2017 Pulse Secure, LLC. All rights reserved
List of Figures

Figure 1: Pulse Connect Secure on Microsoft Azure .................................................................8
Figure 2: Storage Account - pcsgoldenstore ..........................................................................9
Figure 3: Container master ........................................................................................................9
Figure 4: Upload Pulse Connect Secure Virtual Appliance Image ...........................................10
Figure 5: Storage Accounts ...................................................................................................... Error!
Figure 6: Connect to Azure Storage ......................................................................................... Error!
Figure 7: Attach using SAS ...................................................................................................... Error!
Figure 8: Connection Summary ............................................................................................... Error!
Figure 9: Microsoft Azure Storage Explorer – Source Store ................................................ Error!
Figure 10: Microsoft Azure Storage Explorer – Destination Store ......................................... Error!
Figure 11: Microsoft Azure Storage Explorer – Copy Status ................................................ Error!
Figure 12: Templates ................................................................................................................ 10
Figure 13: Add Template ..........................................................................................................11
Figure 14: Template – General Information ..........................................................................11
Figure 15: Add ARM Template ..............................................................................................12
Figure 16: Custom Deployment – New Virtual Network .........................................................13
Figure 17: Deployment Succeeded ..........................................................................................14
Figure 18: PCS Management Public IP ..................................................................................15
Figure 19: PCS External Public IP ..........................................................................................15
Figure 20: Custom Deployment – Existing Virtual Network ..................................................16
Figure 21: Deployment Succeeded ..........................................................................................17
Figure 22: PCS Management Public IP ..................................................................................18
Figure 23: PCS External Public IP ..........................................................................................18
Figure 24: Pulse Configuration Server in Corporate Network ................................................22
Figure 25: Pulse License Server in a Corporate Network ........................................................23
Figure 26: Pulse License Server in Cloud Network .................................................................23
Figure 27: Enter Authentication Code ....................................................................................24
Figure 28: Pulse Management Interface ................................................................................25
Figure 29: Pulse External Interface ........................................................................................26
Figure 30: Resource in Corporate Network ..........................................................................26
Figure 31: Putty Configuration – Basic Options ....................................................................27
Figure 32: Putty Configuration – SSH Authentication .............................................................27
Figure 33: System Operations .................................................................................................28
Figure 34: Delete Resource Group ........................................................................................31
Overview

About This Guide

This guide helps in deploying the Pulse Connect Secure Virtual Appliance on Microsoft Azure. In this release, Pulse Connect Secure is not made available in Azure Market Place. Instead, a Pulse Connect Secure administrator needs to manually upload the Pulse Connect Secure Virtual Appliance image into Microsoft Azure storage account. Once the image is available in the Azure storage account, the Pulse Connect Secure administrator can deploy Pulse Connect Secure on Microsoft Azure.

Assumptions

The basic understanding of deployment models of Pulse Connect Secure on a data center and basic experience in using Microsoft Azure is needed for the better understanding of this guide.

Prerequisites and System Requirements

To deploy the Pulse Connect Secure Virtual Appliance on Azure, you need the following:

- A Microsoft Azure account
- Access to the Microsoft Azure portal (https://portal.azure.com)*
- Pulse Connect Secure Virtual Appliance Image (.vhd file)
- Azure Resource Manager template (ARM template)
- Pulse Connect Secure licenses **
- Site-to-Site VPN between Azure and the corporate network (optional)

⚠️ Note: This is needed only if the Pulse Connect Secure users need to access corporate resources.

- Pulse License Server (optional) **
  - Located at corporate network, accessible through site-to-site VPN
- Pulse Connect Secure configuration in XML format (optional)

⚠️ Note:

* Pulse Connect Secure Virtual Appliance can be deployed only through Azure Resource Manager (ARM) style. It does not support deployment in classic style.
** Pulse Connect Secure Virtual Appliance, by default, has two-users license. This release supports licensing with License server located at corporate network and licensing through Pulse Cloud Licensing Service (PCLS) server. For licensing through PCLS, administrator needs to obtain Authentication Code from Pulse Secure Support and apply it in the Pulse Connect Secure admin console.
Pulse Connect Secure on Microsoft Azure

As depicted in the below diagram, a remote user can use Pulse Connect Secure to securely access cloud resources as well as corporate resources. To access corporate resources, the Pulse Connect Secure administrator needs to ensure that site-to-site VPN is already established between Azure and corporate network.

Figure 1: Pulse Connect Secure on Microsoft Azure

Steps to Deploy Pulse Connect Secure on Azure

Below are the one-time activities to be followed to deploy Pulse Connect Secure on Azure.

- Upload Pulse Connect Secure Virtual Appliance Image to Azure Web Portal
- Upload Azure Resource Manager Template to Azure Account

Below are the steps to be followed for each deployment of Pulse Connect Secure.

- Deploying Pulse Connect Secure on Azure using Azure Portal
- Deploying Pulse Connect Secure on Azure using Azure CLI

Upload Pulse Connect Secure Virtual Appliance Image to Azure Web Portal

This section shows the steps to upload the Pulse Connect Secure Virtual Appliance image to Azure web portal. To upload Pulse Connect Secure Virtual Appliance image to Azure web portal, do the following:

1. Visit the Pulse Secure support site [www.pulsesecure.net](http://www.pulsesecure.net) and download the 8.3R3 Azure PCS image file which is in the zipped format.
2. Unzip the file and look for the Pulse Connect Secure Virtual Appliance image named **VA-SPE-SA-8.3R3-SERIAL-hyperv.vhd**.
3. Log in to the Azure portal.
4. Click **New** and create a storage account named ‘pcsgoldenstore’ under the resource group named ‘pcsgoldenstoreRG’.

**Figure 2: Storage Account - pcsgoldenstore**

5. Inside the pcsgoldenstore storage account, click on **Blobs** and create a container with access type as ‘Container’ named ‘master’.

**Figure 3: Container master**
6. Inside the ‘master’ blob, click on **upload** to upload the Pulse Connect Secure Virtual Appliance image. Inside the ‘Upload blob’, select the Blob type as Page blob and click on **Upload**.

**Figure 4: Upload Pulse Connect Secure Virtual Appliance Image**

---

**Upload Azure Resource Manager Template to Azure Account**

The Azure Resource Manager (ARM) template is a JSON-based file, which has instructions for Azure Fabric on all the resources that need to be created on Azure while running this script. More details on the ARM template can be found at [https://docs.microsoft.com/en-us/azure/azure-resource-manager/resource-manager-create-first-template](https://docs.microsoft.com/en-us/azure/azure-resource-manager/resource-manager-create-first-template).

Pulse Secure provides two sample Azure template files, namely “pulsesecure-pcs-3-nics.zip” and “pulsesecure-pcs-3-nics-existing-vnet.zip”. Users can modify the template to make it suitable for their need. Here are the steps to upload the template to Azure Portal:

1. Unzip the **pulsesecure-pcs-3-nics.zip** file to get **azuredeploy.json**.
2. Log in to the Azure portal.
3. Click on **More services** and select **Templates**.

**Figure 5: Templates**
4. In the Templates page, click **Add** to add template.

**Figure 6: Add Template**

![Add Template](image)

5. Provide a suitable name and description for the template.

**Figure 7: Template – General Information**

![General Information](image)
6. Copy the contents of `azuredeploy.json` and paste it in the template section.

Figure 8: Add ARM Template

Press the image to expand it.

Deploying Pulse Connect Secure on Azure using Azure Portal

Before proceeding with the deployment, refer the following sections:
- Upload Pulse Connect Secure Virtual Appliance Image to Azure
- Upload Azure Resource Manager Template to Azure Account

Pulse Connect Secure can be deployed on:
- a new Virtual network
- an already existing Virtual network

Deploying PCS on New Virtual Network

To deploy Pulse Connect Secure on Azure using the Azure portal, do the following:
1. Select the template file created in section ‘Upload Azure Resource Manager Template to Azure account’ and click Deploy.
2. Fill or modify the following parameters:
   - **Resource group**: Specify the resource group name in which Pulse Connect Secure needs to be deployed
   - **Location**: Region where resource group needs to be created
   - **PCS Storage Account Name**: Storage account name where the Pulse Connect Secure Virtual Appliance image is available
   - **PCS Storage Account Resource Group**: Resource group of where the Pulse Connect Secure Virtual Appliance image is copied
   - **PCS Image Location URI**: URI to Pulse Connect Secure Virtual Appliance Image
   - **PCSVM Name**: Name of the Pulse Connect Secure Virtual instance
   - **PCS Config**: Provisioning parameters in an XML format. Refer the section ‘Pulse Connect Secure Provisioning Parameters’
   - **SSH Public Key**: This key is used to access PCS via SSH. The SSH keys are generated using ssh-keygen on Linux and OS X, or PuTTYGen on Windows. For details about generating the SSH key pairs, refer: For Windows: https://docs.microsoft.com/en-us/azure/virtual-machines/linux/ssh-from-windows
   For MacOS and Linux: https://docs.microsoft.com/en-us/azure/virtual-machines/linux/mac-create-ssh-keys
   - **DNS Label Prefix Ext**: Prefix for the external interface DNS label
   - **DNS Label Prefix Mgmt**: Prefix for the management interface DNS label
- **Vnet Address Space**: Virtual network address space
- **Internal Subnet**: Subnet from which Pulse Connect Secure internal interface needs to lease IP
- **External subnet**: Subnet from which Pulse Connect Secure external interface needs to lease IP
- **Management subnet**: Subnet from which Pulse Connect Secure management interface needs to lease IP
- **Tunnel subnet**: Subnet which will be configured as tunnel IP pool in the Pulse Connect Secure VPN Profile

3. Agree to the Azure licensing terms and click **Purchase**.
4. Watch for the deployment succeeded message after 3 to 5 minutes.

**Figure 10: Deployment Succeeded**

5. Go to the resource group in which the Pulse Connect Secure Virtual Appliance was deployed to see the resources created.
6. Navigate to the resource group and click **PCS Management Public IP**. Make a note of the PCS Management Public IP and DNS name (FQDN) to access PCS for admin page.
Figure 11: PCS Management Public IP

7. Click **PCS External Public IP** and note down the **PCS External Public IP and DNS name (FQDN)** to access PCS for end user page.

**Figure 12: PCS External Public IP**

---

**Note:** Azure allows static as well as dynamic assignment of IP addresses to the network interfaces. The mode of IP assignment (static/dynamic) can be mentioned in the Azure Resource Manage template file. The current JSON template uses the dynamic method of allotting IP addresses to the network interfaces.

### Deploying PCS on an Existing Virtual Network

To deploy Pulse Connect Secure on Azure using the Azure portal, do the following:

1. Select the template file “pulsesecure-pcs-3-nics-existing-vnet” created in the section ‘[Upload Azure Resource Manager Template to Azure account](#)’ and click **Deploy**.
2. Fill or modify the following parameters:

- **Resource group:** Specify the resource group name in which Pulse Connect Secure needs to be deployed
- **Location:** Region where resource group needs to be created
- **PCS Storage Account Name:** Storage account name where the Pulse Connect Secure Virtual Appliance image is available
- **PCS Storage Account Resource Group:** Resource group of where the Pulse Connect Secure Virtual Appliance image is copied
- **PCS Image Location URI:** URI to Pulse Connect Secure Virtual Appliance Image
- **PCS VM Name:** Name of the Pulse Connect Secure Virtual instance
- **PCS Config:** Provisioning parameters in XML format. Refer ‘Pulse Connect Secure Provisioning Parameters’
- **SSH Public Key:** This key is used to access PCS via SSH. The SSH keys are generated using ssh-keygen on Linux and OS X, or PuTTYGen on Windows. For details about generating the SSH key pairs, refer: For Windows: https://docs.microsoft.com/en-us/azure/virtual-machines/linux/ssh-from-windows For MacOS and Linux: https://docs.microsoft.com/en-us/azure/virtual-machines/linux/mac-create-ssh-
keys

- **DNS Label Prefix Ext**: Prefix for the external interface DNS label
- **DNS Label Prefix Mgmt**: Prefix for the management interface DNS label
- **Resource Group Name of Existing Virtual Network**: Resource Group name of the Virtual network
- **Existing Vnet Name**: Virtual network name
- **Existing Internal Subnet**: Subnet from which the Pulse Connect Secure internal interface needs to lease IP
- **Existing External subnet**: Subnet from which the Pulse Connect Secure external interface needs to lease IP
- **Existing Management subnet**: Subnet from which the Pulse Connect Secure management interface needs to lease IP
- **Existing Tunnel subnet**: Subnet which will be configured as the tunnel IP pool in the Pulse Connect Secure VPN Profile

3. Agree to the Azure licensing terms and click **Purchase**.
4. Watch for the deployment succeeded message after 3 to 5 minutes.

**Figure 14: Deployment Succeeded**

5. Go to the resource group in which the Pulse Connect Secure Virtual appliance was deployed to see the resources created.
6. Navigate to the resource group and click **PCS Management Public IP**. Make a note of the PCS Management Public IP and DNS name (FQDN) to access PCS for admin page.
7. Click **PCS External Public IP** and note down the **PCS External Public IP and DNS name (FQDN)** to access PCS for end user page.

**Figure 16: PCS External Public IP**

---

**Note:** Azure allows static as well as dynamic assignment of the IP addresses to the network interfaces. The mode of IP assignment (static/dynamic) can be mentioned in the Azure Resource Manage template file. The current JSON template uses dynamic method of allotting IP addresses to the network interfaces.
Deploying Pulse Connect Secure on Azure using Azure CLI

Before proceeding with the deployment, refer [Upload Pulse Connect Secure Virtual Appliance Image to Azure](#).

3. Unzip the file and look for the pulsesecure-pcs-3-nics.zip file. Unzip the file to get azuredeploy.json
4. Ensure that parameters section has correct default values:
   - **PCS Storage Account Name**: Storage account name where the Pulse Connect Secure Virtual Appliance image is available
   - **PCS Storage Account Resource Group**: Resource group where the Pulse Connect Secure Virtual Appliance image is copied
   - **PCS Image Location URI**: URI to the Pulse Connect Secure Virtual Appliance Image
   - **PCS VM Name**: Name of the Pulse Connect Secure Virtual instance
   - **PCS Config**: Provisioning parameters in an XML format. Refer “Pulse Connect Secure Provisioning Parameters”
   - **DNS Label Prefix Ext**: Prefix for the external interface DNS label
   - **DNS Label Prefix Mgmt**: Prefix for the management interface DNS label
   - **Vnet Address Space**: Virtual network address space
   - **Internal Subnet**: Subnet from which the Pulse Connect Secure internal interface needs to lease IP
5. To deploy Pulse Connect Secure using Azure CLI, run the following commands

   $ az login
   $ az group create -l <location> -n <resource group name>
   $ az group deployment create -g <resource group name> --template-file <json file name>

   For example:

   $ az login

   ```
   Azure:
   To sign in, use a web browser to open the page https://aka.ms/devicelogin and enter the code 0027EE92 to authenticate.
   ```

   $ az group create -l southindia -n TestPCSDeploymentRG

© 2017 Pulse Secure, LLC. All rights reserved
TestPCS:Desktop TestPCS$ az group create -l southindia -n TestPCSDeploymentRG
{
  "id": "/subscriptions/TestPCS6-9175-9175-9175-southindiaae/resourceGroups/TestPCSDeploymentRG",
  "location": "southindia",
  "managedBy": null,
  "name": "TestPCSDeploymentRG",
  "properties": {
    "provisioningState": "Succeeded"
  },
  "tags": null
}
TestPCS:Desktop TestPCS$

$ az group deployment create -g TestPCSDeploymentRG --template-file azuredeploy.json

{  
  "id": null,
  "namespace": "Microsoft.Compute",
  "registrationState": null,
  "resourceTypes": [  
    {  
      "aliases": null,
      "apiVersions": null,
      "locations": [
        "southindia"
      ],
      "properties": null,
      "resourceType": "virtualMachines"
    }
  ],  
  "provisioningState": "Succeeded",
  "template": null,
  "templateLink": null,
  "timestamp": "2017-08-06T17:19:20.227838+00:00"
},  
"resourceGroup": "TestPCSDeploymentRG"
}
Pulse Connect Secure Provisioning Parameters

Provisioning parameters are those parameters which are required during the deployment of a virtual appliance. Pulse Connect Secure accepts the following parameters as provisioning parameters in the XML format.

```xml
<pulse-config>
  <primary-dns><value></primary-dns>
  <secondary-dns><value></secondary-dns>
  <wins-server><value></wins-server>
  <dns-domain><value></dns-domain>
  <admin-username><value></admin-username>
  <admin-password><value></admin-password>
  <cert-common-name><value></cert-common-name>
  <cert-random-text><value></cert-random-text>
  <cert-organisation><value></cert-organisation>
  <config-download-url><value></config-download-url>
  <config-data><value></config-data>
  <auth-code-license><value></auth-code-license>
</pulse-config>
```

The below table depicts the details of xml file.

<table>
<thead>
<tr>
<th>#</th>
<th>Parameter Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>primary-dns</td>
<td>IP address</td>
<td>Primary DNS for Pulse Connect Secure</td>
</tr>
<tr>
<td>2</td>
<td>secondary-dns</td>
<td>IP address</td>
<td>Secondary DNS for Pulse Connect Secure</td>
</tr>
<tr>
<td>3</td>
<td>wins-server</td>
<td>IP address</td>
<td>Wins server for Pulse Connect Secure</td>
</tr>
<tr>
<td>4</td>
<td>dns-domain</td>
<td>string</td>
<td>DNS domain of Pulse Connect Secure</td>
</tr>
<tr>
<td>5</td>
<td>admin-username</td>
<td>string</td>
<td>admin UI user name</td>
</tr>
<tr>
<td>6</td>
<td>admin-password</td>
<td>string</td>
<td>admin UI password</td>
</tr>
<tr>
<td>7</td>
<td>cert-common-name</td>
<td>string</td>
<td>Common name for the self-signed certificate generation</td>
</tr>
<tr>
<td>8</td>
<td>cert-random-text</td>
<td>string</td>
<td>Random text for the self-certificate generation</td>
</tr>
<tr>
<td>9</td>
<td>cert-organisation</td>
<td>string</td>
<td>Organization name for the self-signed certificate generation</td>
</tr>
<tr>
<td>10</td>
<td>config-download-url</td>
<td>String URL</td>
<td>Http based URL where XML based Pulse Connect Secure configuration can be found.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>During provisioning, Pulse Connect Secure fetches this file and comes up with preloaded configuration. XML based configuration can be present in another VM in Azure cloud or at corporate network which is accessible for Pulse Connect Secure through site to site VPN between Azure and corporate data center</td>
</tr>
<tr>
<td>11</td>
<td>config-data</td>
<td>string</td>
<td>base64 encoded XML based Pulse Connect Secure configuration</td>
</tr>
<tr>
<td>12</td>
<td>auth-code-license</td>
<td>string</td>
<td>Authentication code that needs to be obtained from Pulse Secure</td>
</tr>
</tbody>
</table>

**Note:** In the above list of parameters, primary dns, dns domain, admin username, admin password, cert-random name, cert-random text and cert-organization are mandatory parameters. The other parameters are optional parameters.
Provisioning Pulse Connect Secure with Predefined Configuration

The Pulse Connect Secure Virtual Appliance can be provisioned on Azure with a pre-defined Pulse Connect Secure configuration. The provisioning can be done in the following two ways:

- Pulse Connect Secure administrator needs to provide the location of the XML-based configuration as a provisioning parameter. Refer ‘Pulse Connect Secure Provisioning Parameters’ for details about the Pulse Connect Secure specific provisioning parameters.
  
Pulse Connect Secure configuration can be kept on Azure VM or on a machine located in the corporate network. If it is in the corporate network, the Pulse Connect Secure administrator needs to ensure that site-to-site VPN between Azure to corporate network is already established so that Pulse Connect Secure can access the machine located in the corporate network.

- Pulse Connect Secure administrator provides the configuration data encoded in the base64 encoded xml in the ARM template.

Figure 17: Pulse Configuration Server in Corporate Network
Configuring Licenses on the Pulse Connect Secure Appliance

By default, two-user licenses are provided. To add more licenses, the Pulse Connect Secure administrator needs to leverage the Pulse License server. The Pulse License server can be made available in:

- [corporate network](#)
- [cloud network](#)

**Pulse License Server in Corporate Network**

*Figure 18: Pulse License Server in a Corporate Network*  

**Pulse License Server in Cloud Network**

In 8.3R3, the Pulse Connect Secure virtual machines (VM) are enabled to provision licenses through the Pulse Cloud Licensing Service (PCLS). For this, the administrator needs to obtain an Authentication code from Pulse Secure Support and apply it in the Download Licenses page of PCS admin console. The PCS also periodically sends heartbeat messages to PCLS for auditing purposes. The Authentication code can also be specified in the ARM template. When PCS comes up, it automatically fetches the Authentication code.

- [Adding Authentication Code in PCS Admin Console](#)
- [Including Authentication Code in ARM Template](#)

*Figure 19: Pulse License Server in Cloud Network*
Adding Authentication Code in PCS Admin Console

To add Authentication code:

1. Go to **System > Configuration > Licensing > Download Licenses**.
2. Under On demand license downloads, enter the Authentication code in the text box.
3. Click on **Download and Install**.

Figure 20: Enter Authentication Code

Including Authentication Code in ARM Template

To include Authentication code in the ARM template:

1. In the ARM template, go to the PCSConfig section.
2. For the element `<auth-code-license>`, enter the Authentication code as the content.
3. Save the template.

For details about the license configuration, refer to [License Configuration Guide](#).
Accessing the Pulse Connect Secure Virtual Appliance

The Pulse Connect Secure appliance can be accessed:

- **as an administrator**
- **as an end user**
- **using SSH console**

**Accessing the Pulse Connect Secure Virtual Appliance as an Administrator**

To access the Pulse Connect Secure Virtual Appliance as an administrator, copy the IP address from the Pulse Management Interface resource.

*Figure 21: Pulse Management Interface*

Use the credentials provided in the provisioning parameters to log in as the administrator. The default PCS admin UI user configured in the azuredeploy.json config file is: user 'admin' and password 'password'.

The administrator can configure Active Directory located in the corporate network for user authentication. The Pulse Connect Secure Virtual Appliance administrator can check troubleshooting tools provided in the Pulse Connect Secure admin UI (System->Maintenance->Troubleshooting), to verify whether Pulse Connect Secure is able to reach other cloud resources as well as corporate resources. For this, Azure network administrator needs to ensure that all other resources have Pulse Connect Secure Internal interface as its default gateway.
Accessing the Pulse Connect Secure Virtual Appliance as an End User

To access the Pulse Connect Secure Virtual Appliance as an end user, copy the IP address from Pulse External Interface resource.

Figure 22: Pulse External Interface

Accessing the Pulse Connect Secure Virtual Appliance using SSH Console

To access the Pulse Connect Secure Virtual Appliance using the SSH console, copy the Public IP address from the PCSManagementPublicIP resource.

On Linux and Mac OS X

Execute the following command:

```
ssh -i <rsa-private-key-file> <PCS-Management-Interface-PublicIP> -p 6667
```
On Windows

1. Launch the Putty terminal emulator.
2. In the Session category:
   - Enter the host name or IP address.
   - Enter the port number.
   - Select the connection type as SSH.

Figure 24: Putty Configuration – Basic Options

![Putty Configuration - Basic Options]

3. Select Connection > SSH > Auth. Click Browse and select the private key file for authentication.

Figure 25: Putty Configuration – SSH Authentication

![Putty Configuration - SSH Authentication]
System Operations

The Azure VA portal provides Start, Restart and Stop operations to control the Virtual Appliance connection.

Figure 26: System Operations

On the Azure portal top menu bar:

- Click Start to start a VM
- Click Stop to stop the VM
- Click Restart to restart the VM

The corresponding CLI commands are:

Start a VM

```bash
az vm start --resource-group myResourceGroup --name myVM
```

Stop a VM

```bash
az vm stop --resource-group myResourceGroup --name myVM
```

Restart a VM

```bash
az vm restart --resource-group myResourceGroup --name myVM
```

Network Configuration

IP Address Assignment for Internal, External and Management Interfaces

Each interface in Azure can have private and public IP addresses. Sample Azure Templates provided by Pulse Connect Secure creates the Pulse Connect Secure Virtual Appliance with public and private IP addresses for external and management interfaces and only private IP address for internal interface. More details about IP address types on Azure can be seen at: https://docs.microsoft.com/en-us/azure/virtual-network/virtual-network-ip-addresses-overview-arm

IP Addressing Modes

When Pulse Connect Secure gets deployed by using the sample templates provided by Pulse Secure, Pulse Connect Secure comes up with multiple interfaces. If you take an example of a template “pulsesecure-pcs-3-nics.zip” provided by Pulse Secure, you notice the following things.
PCS external interface and PCS management interface are having both Public and Private IP addresses. In the below code snippet, observe the network interface getting created with two IP addresses - private IP address and public IP address. Highlighted section points to private IP allocation method and Public IP address getting assigned to NIC.

```json
1. "type": "Microsoft.Network/networkInterfaces",
2. "name": "[variables('pcsExtNic')]",
3. "properties": {
4. "privateIPAllocationMethod": "Dynamic",
5. "privateIPAddressVersion": "IPv4",
6. "publicIPAddress": {
7. "id": "[resourceId('Microsoft.Network/publicIPAddresses',variables('publicIPAddr1'))]"
8. }
9. },
```

If you want to have control on the IP assigned to Network Interface, then you need to change the attribute “privateIPAllocationMethod” from “Dynamic” to “Static”. Also, you need to add an attribute called “privateIPAddress” which holds the static IP address. When you are assigning static IP address, make sure that it is not in the reserved IP category.

```json
1. "ipConfigurations": [{
2. "name": "ipconfig2",
3. "properties": {
4. "privateIPAllocationMethod": "Static",
5. "privateIPAddressVersion": "IPv4",
6. "privateIPAddress": "[variables('privateIPExternal')]"
7. }
8. }]
```

### Modifying Network Parameters After Deployment

Since Networking Infrastructure is provided by Azure, a PCS admin cannot change Networking configuration after deployment. Hence, both admin UI and ssh does not support changing network configuration.

### Controlling the Selection of Internal, External and Management Interfaces

Sample Azure Template, provided by Pulse Secure, requests Azure fabric to create three Network Interfaces. While running this template, Azure fabric creates interfaces named eth0, eth1 and eth2 and attaches them to PCS Virtual Interface.

So, the question is, among eth0, eth1 and eth2 which network interface will become external, internal or management interface? Below table answers this question.

<table>
<thead>
<tr>
<th>Interface Name</th>
<th>PCS Interface</th>
</tr>
</thead>
<tbody>
<tr>
<td>eth0</td>
<td>internal interface</td>
</tr>
<tr>
<td>eth1</td>
<td>external interface</td>
</tr>
<tr>
<td>eth2</td>
<td>management interface</td>
</tr>
</tbody>
</table>

Then, question is how you can control the order of network interfaces named eth0, eth1 and eth2 created through Azure Template?. Azure supports two types of interfaces: primary and secondary. Only one primary interface can be present on a VM.

The Pulse Connect Secure Virtual Appliance is qualified with internal interface as primary and other two are secondary. In the following code snippet, three network interfaces get assigned to VM. These three NICs with ID “nic1”, “nic2” and “nic3” are internally mapped to ‘eth0’, ‘eth1’, and ‘eth2’ respectively.

```
1. "networkProfile": {
2.   "networkInterfaces": [[]
3.   "id": "nic1",
4.   "properties": {
5.     "primary": true
6.   }
7. }, { 
8.   "id": "nic2",
9.   "properties": {
10.  "primary": false
11. }
12. }, { 
13.  "id": "nic3",
14.  "properties": {
15.   "primary": false
16. }
17. ]]
18. },
```

PCS converts eth0 to int0, eth1 to ext0 and eth2 to mgmt0. This means, the network interface with ID nic1 will be internal interface, nic2 will be external interface and nic3 will be management interface.

The below table depicts this scenario well:

<table>
<thead>
<tr>
<th>Interface Name</th>
<th>PCS Interface</th>
<th>Network ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>eth0</td>
<td>internal interface</td>
<td>nic1</td>
</tr>
<tr>
<td>eth1</td>
<td>external interface</td>
<td>nic2</td>
</tr>
<tr>
<td>eth2</td>
<td>management interface</td>
<td>nic3</td>
</tr>
</tbody>
</table>

Suppose if you make ‘nic2’ as primary, then the order may not be maintained and it is difficult to predict which interface will become internal interface of PCS. As a best practice, always assign ‘primary’ to the first network interface which will become internal interface of PCS.

**Decommissioning Pulse Connect Secure**

When deploying Pulse Connect Secure, if you have selected the option “Use existing resource group”, then follow the steps mentioned in the section **Delete Pulse Connect Secure and Resource It Uses, but not the Other Resources in Resource Group**. Else if you have selected the option “New resource group” then follow the steps mentioned in the section **Delete Entire Resource Group that the Pulse Connect Secure Is In**.

**Delete Entire Resource Group that the Pulse Connect Secure Is In**

1. Log into Azure portal.
3. Click on the resource group where Pulse Connect Secure is in.
4. Click on the **Delete resource group** button. In the confirmation page type in resource group name and click **Delete**.

**Figure 27: Delete Resource Group**

![Delete Resource Group](image)

5. Navigate to the storage account where the Pulse Connect Secure VHD image is stored.
6. In the storage account, click on **Blobs**. Find boot diagnostic folder and delete it. Boot diagnostic folder name will have the pattern “bootdiagnostics-<pcs-name>-<random-ascii-characters>”.
7. In the storage account, click on **Blobs**. Find and click on the **vhds** folder. Find and delete file size named “<pcs-name><13 digit unique string>pcsOSDisk.vhd”.

**Delete Pulse Connect Secure and Resource It Uses, but not the Other Resources in Resource Group**

1. Log into Azure portal.
3. Click on the resource group where Pulse Connect Secure is in.
4. Delete the following resources:
   - PCS Virtual Machine
   - Virtual Network named PCSVirtualNetwork
   - PCSInternalNIC, PCSExternalNIC and PCSManagementNIC
   - PCSExternalPublicIP and PCSManagementIP
   - Three Network Security Groups named NSGInternal, NSGExternal and NSGManagement
   - User-defined Routing table named Backend2PCSRoute
5. Navigate to the storage account where the Pulse Connect Secure VHD image is stored.
6. In the storage account, click on **Blobs**. Find boot diagnostic folder and delete it. Boot diagnostic folder name will have the pattern “bootdiagnostics-<pcs-name>-<random-ascii-characters>”. 
7. In the storage account, click on **Blobs**. Find and click on the **vhds** folder. Find and delete file size named “<pcs-name><13 digit unique string>pcsOSDisk.vhd”.

### Pricing

The cost of running this product is combination of License cost and Azure infrastructure cost. It will be very difficult to find out Azure infrastructure cost for this product as it may vary with Regions/Country/Time. Hence, it is recommended to use “Azure Calculator”, which is available online, to calculate the cost of running this product.

Here are resources that are created during deployment.

<table>
<thead>
<tr>
<th>Resources</th>
<th>Category</th>
<th>Chargeable</th>
</tr>
</thead>
<tbody>
<tr>
<td>PCS VM (Standard_DS3_V2)</td>
<td>Compute</td>
<td>Yes</td>
</tr>
<tr>
<td>Virtual Network with four subnets</td>
<td>Networking</td>
<td>No</td>
</tr>
<tr>
<td>Three NICs named PCSInternalNIC, PCSEnternalNIC and PCSManagementNIC</td>
<td>Networking</td>
<td>No</td>
</tr>
<tr>
<td>Two static Public IPs name PCSEnternalPublicIP and PCSManagementIP</td>
<td>Networking</td>
<td>Yes</td>
</tr>
<tr>
<td>Three Network Security Groups named NSGInternal, NSGExternal and NSGManagement</td>
<td>Networking</td>
<td>No</td>
</tr>
<tr>
<td>User Defined Routing table named Backend2PCSRoute</td>
<td>Networking</td>
<td>No</td>
</tr>
<tr>
<td>Boot diagnostic file under existing storage account (Less than 5MB)</td>
<td>Storage</td>
<td>Yes</td>
</tr>
<tr>
<td>File size of 40GB in the existing storage account under Blobs and container VHDs named “&lt;pcs-name&gt;&lt;13 digit unique string&gt;pcsOSDisk.vhd”</td>
<td>Storage</td>
<td>Yes</td>
</tr>
</tbody>
</table>

### Limitations

The following list of Pulse Connect Secure features are not supported in this release:

- VLAN tagging
- IPv6 capabilities
- Layer 3 Tunnel IP pool assignment via DHCP
- Layer 2 functionality like ARP Cache and ND Cache
- Pulse Connect Secure in Azure marketplace
- Deployment of Pulse Connect Secure as license server on Azure
- Virtual Ports
- Multicast capabilities
- Bandwidth management
**Not Qualified**

The following list of Pulse Connect Secure features are not qualified in this release:

- Clustering capabilities
- Pulse Connect Secure and Pulse One interaction
- Pulse Connect Secure and PWS interaction
- IF-MAP support

**Troubleshooting**

Pulse Connect Secure emits booting logs at a specified storage. You can check the storage details of the boot diagnostic logs as shown below:

*Figure 28: Boot Diagnostics*
Appendix A: Network Security Group (NSG)

Microsoft Azure has a limitation where virtual machine with multiple network interfaces cannot connect to different Virtual Networks (VNETs). For example, a VM with two NICs, NIC1 and NIC2, will not be able to connect to Vnet1 and Vnet2 respectively.

Figure 29: Virtual Machine with two NICs Connecting to VNet1 and Vnet2

Microsoft Azure supports a virtual machine with multiple NICs to connect to different Subnets under a same Virtual Network. For example, a VM with two NICs, NIC1 and NIC2, can connect to ‘Subnet1’ and ‘Subnet2’ where these subnets exist under a same Virtual Network respectively.

Figure 30: Virtual Machine with two NICs Connecting to Subnet1 and Subnet2

Azure provides isolation between different Vnets. But it does not provide the same kind of isolation when it comes to subnets in the same Vnet. For example, consider a Vnet has two subnets, Subnet1 and Subnet2. And consider two VMs, VM-1 and VM-2, which are connected to Subnet1 and Subnet2 respectively. In this scenario VM-1 can access the resources from VM-2 and vice versa.

Figure 31: Virtual Machine VM-1 can Access Resources in VM-2 and Vice Versa
Application isolation is an important concern in enterprise environments, as enterprise customers seek to protect various environments from unauthorized or unwanted access. To achieve the traffic isolation between subnets, go for an option of filtering traffic using “Network Security Group” provided by Azure.

**Figure 32: Traffic Filtering by MS Azure Network Support Group**

Pulse Connect Secure, when provisioned through the ARM template provided by Pulse Secure, creates four subnets under a virtual network named “PCSVirtualNetwork”. The four Subnets are:

1. PCSInternalSubnet
2. PCSExternalSubnet
3. PCSManagementSubnet
4. PCSTunnelVPNPoolSubnet

Along with above mentioned subnets, create the following three Network Security Groups (NSG) policies:

1. NSGExternalSubnet
2. NSGInternalSubnet
3. NSGManagementSubnet

**Figure 33: NSG External, Internal and Management Subnets**

In Network Security Group (NSG) we need to create policies for Inbound and outbound traffic.

1. The list of NSG Inbound/Outbound rules created “NSGExternalSubnet” are:
2. The list of NSG Inbound/Outbound rules created “NSGInternalSubnet” are:

**Figure 36: NSG Internal - Inbound Rules**
3. The list of NSG Inbound/Outbound rules created “NSGManagementSubnet” are:

**Figure 37: NSG Internal - Outbound Rules**

<table>
<thead>
<tr>
<th>Priority</th>
<th>Name</th>
<th>Port</th>
<th>Protocol</th>
<th>Source</th>
<th>Destination</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>65000</td>
<td>AllowInternalOutbound</td>
<td>Any</td>
<td>Any</td>
<td>VirtualNetwork</td>
<td>VirtualNetwork</td>
<td>Allow</td>
</tr>
<tr>
<td>65001</td>
<td>AllowExternalOutbound</td>
<td>Any</td>
<td>Any</td>
<td>Any</td>
<td>Internet</td>
<td>Allow</td>
</tr>
<tr>
<td>65500</td>
<td>DenyAllOutbound</td>
<td>Any</td>
<td>Any</td>
<td>Any</td>
<td>Any</td>
<td>Deny</td>
</tr>
</tbody>
</table>

**Figure 38: NSG Management - Inbound Rules**

<table>
<thead>
<tr>
<th>Priority</th>
<th>Name</th>
<th>Port</th>
<th>Protocol</th>
<th>Source</th>
<th>Destination</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>allowHTTP</td>
<td>80</td>
<td>Any</td>
<td>Any</td>
<td>Any</td>
<td>Allow</td>
</tr>
<tr>
<td>200</td>
<td>allowHTTPS</td>
<td>443</td>
<td>Any</td>
<td>Any</td>
<td>Any</td>
<td>Allow</td>
</tr>
<tr>
<td>300</td>
<td>allowCustomSSH</td>
<td>6667</td>
<td>Any</td>
<td>Any</td>
<td>Any</td>
<td>Allow</td>
</tr>
<tr>
<td>400</td>
<td>allowDM</td>
<td>830</td>
<td>Any</td>
<td>Any</td>
<td>Any</td>
<td>Allow</td>
</tr>
<tr>
<td>4000</td>
<td>denyAll</td>
<td>Any</td>
<td>Any</td>
<td>Any</td>
<td>Any</td>
<td>Deny</td>
</tr>
<tr>
<td>65000</td>
<td>AllowInternalBound</td>
<td>Any</td>
<td>Any</td>
<td>VirtualNetwork</td>
<td>VirtualNetwork</td>
<td>Allow</td>
</tr>
<tr>
<td>65001</td>
<td>AllowInternaloExternalBalance</td>
<td>Any</td>
<td>Any</td>
<td>AzureLoadBalancers</td>
<td>Any</td>
<td>Allow</td>
</tr>
<tr>
<td>65500</td>
<td>DenyAllInbound</td>
<td>Any</td>
<td>Any</td>
<td>Any</td>
<td>Any</td>
<td>Deny</td>
</tr>
</tbody>
</table>

**Figure 39: NSG Management - Outbound Rules**

<table>
<thead>
<tr>
<th>Priority</th>
<th>Name</th>
<th>Port</th>
<th>Protocol</th>
<th>Source</th>
<th>Destination</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>denyInternalSubnet</td>
<td>Any</td>
<td>Any</td>
<td>Any</td>
<td>10.20.10.24</td>
<td>Deny</td>
</tr>
<tr>
<td>200</td>
<td>denyExternalSubnet</td>
<td>Any</td>
<td>Any</td>
<td>Any</td>
<td>10.20.20.24</td>
<td>Deny</td>
</tr>
<tr>
<td>65000</td>
<td>AllowInternalOutbound</td>
<td>Any</td>
<td>Any</td>
<td>VirtualNetwork</td>
<td>VirtualNetwork</td>
<td>Allow</td>
</tr>
<tr>
<td>6501</td>
<td>AllowInternetOutbound</td>
<td>Any</td>
<td>Any</td>
<td>Internet</td>
<td>Any</td>
<td>Allow</td>
</tr>
<tr>
<td>65500</td>
<td>DenyAllOutbound</td>
<td>Any</td>
<td>Any</td>
<td>Any</td>
<td>Any</td>
<td>Deny</td>
</tr>
</tbody>
</table>
Appendix B: Pulse Connect Secure Resource Manager Template

Pulse Secure provides sample Azure template files to deploy the Pulse Connect Secure Virtual Appliance on Azure. Users can modify this to make it suitable for their need. Visit https://www.pulsesecure.net and download the pulsesecure-pcs-3-nics.zip file, and unzip it to get azuredeploy.json.

This template creates a new PCS with 3 NICs, Vnet, four subnets, NSG policies attached to PCS internal, external and management subnets and user-defined routes on the PCS internal subnet to ensure PCS is used as default gateway for L3 tunnel. All 3 NICs of PCS are configured with dynamic IP configuration and enabled IP forwarding. Public IPs are attached to the PCS external and management NIC.

The template has following sections:

<table>
<thead>
<tr>
<th>Section</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>parameters</td>
<td>This section defines the parameters used for deploying PCS on Azure. It contains parameter name, its default value and the mouse-over help text that is displayed when mouse is placed over the parameter in Azure Web portal. The parameters defined here are displayed in the Custom Deployment page of Azure portal.</td>
</tr>
<tr>
<td>variables</td>
<td>This section defines variables that will be used in the functions defined in the resources section.</td>
</tr>
<tr>
<td>resources</td>
<td>This section defines resource types that are deployed or updated in a resource group.</td>
</tr>
<tr>
<td>outputs</td>
<td>This section defines the public IP address and FQDN returned after successful deployment of PCS on Azure.</td>
</tr>
</tbody>
</table>

parameters

Figure 40: Custom Deployment
PCS Storage Account Name: This is the name of the PCS Storage Account where the PCS Azure vhd image is stored.

```json
"parameters": {
  "PCSTorageAccountName": {
    "type": "string",
    "defaultValue": "pcsgoldenstorage",
    "metadata": {
      "description": "Storage account name where PCS image is uploaded"
    }
  }
}
```

PCS Storage Account Resource Group Name: The is the name of the PCS Storage Account Resource Group where the PCS Azure vhd image is stored.

```json
"PCSTorageAccountResourceGroupName": {
  "type": "string",
  "defaultValue": "GoldenImageRG",
  "metadata": {
    "description": "Resource group of the existing storage account where PCS image is uploaded"
  }
}
```

PCS Image Location URI: This is the URL to the location where PCS Azure vhd image is stored.

```json
"PCSTrustedVMImageLocationURI": {
  "type": "string",
  "defaultValue": "https://pcsgoldenstorage.blob.core.windows.net/master/pcs-azure-drop5-upgrade.vhd",
  "metadata": {
    "description": "URL of PCS vhd image"
  }
}
```

PCS VM Name: This is the name given to PCS Virtual Appliance.

```json
"PCSTrustedVMName": {
  "type": "string",
  "defaultValue": "PCS Azure VA",
  "metadata": {
    "description": "PCS VA Name"
  }
}
```

SSH Public Key: This is an RSA public key that is used to access Pulse Connect Secure via SSH.

```json
"SSHPublicKey": {
  "type": "string",
  "metadata": {
    "description": "Provide a RSA public key. This key is used to access PCS via SSH. You can generate SSH keys using ssh-keygen on Linux or OS X, or PuTTYGen on Windows."
  }
}
```

PCS Config: This section contains provisioning parameters that are required during the deployment of a Virtual Appliance. An XML-based configuration file can be present in another Virtual Machine in Azure cloud or in the corporate network which is accessible for Pulse Connect Secure through site-to-site VPN between Azure and the corporate data center.

Pulse Connect Secure accepts the following parameters as provisioning parameters:

- primary-dns
- secondary-dns
- wins-server
- dns-domain
- username
- ssh-publickey
- cert-common-name
- cert-random-text
For details about these parameters, see Pulse Connect Secure Provisioning Parameters.

**DNS Label Prefix Ext:** This is the prefix for External Interface DNS label.

```json
  "dnxlabelPrefixExt": {
    "type": "string",
    "defaultVal": "mycloudpcesxt",
    "metadata": {
      "description": "Unique DNS Name for the Public IP used to access FCS"
    }
  }
```

**DNS Label Prefix Mgmt:** This is the prefix for Management Interface DNS label.

```json
  "dnxlabelPrefixMgmt": {
    "type": "string",
    "defaultVal": "mycloudpcesmgmt",
    "metadata": {
      "description": "Unique DNS Name for the Public IP used to access FCS"
    }
  }
```

**VNet Address Space:** This is a Virtual Network address space.

```json
  "VnetAddressSpace": {
    "type": "string",
    "defaultVal": "10.20.0.0/16",
    "metadata": {
      "description": "Virtual Network Address Space"
    }
  }
```

**Internal Subnet:** Subnet from which Pulse Connect Secure Internal Interface needs to lease IP.

```json
  "InternalSubnet": {
    "type": "string",
    "defaultVal": "10.20.1.0/24",
    "metadata": {
      "description": "FCS internal interface connects to this subnet"
    }
  }
```

**External Subnet:** Subnet from which Pulse Connect Secure External Interface needs to lease IP.

```json
  "ExternalSubnet": {
    "type": "string",
    "defaultVal": "10.20.2.0/24",
    "metadata": {
      "description": "FCS external interface connects to this subnet"
    }
  }
```
Management Subnet: Subnet from which Pulse Connect Secure Management Interface needs to lease IP.

```json
"ManagementSubnet": {
  "type": "string",
  "defaultValue": "10.20.3.0/24",
  "metadata": {
    "description": "PCS management interface connects to this subnet"
  }
},
```

Tunnel Subnet: Subnet which will be configured as Tunnel IP pool in Pulse Connect Secure VPN profile.

```json
"TunnelSubnet": {
  "type": "string",
  "defaultValue": "10.20.4.0/24",
  "metadata": {
    "description": "Subnet used for VPN Pools"
  }
},
```

variables

PCS Virtual Network: This is the variable associated with the PCS Virtual Network.

```json
"pcsvnetname" : "PCSVirtualNetwork",
```

PCS Internal Subnet: This is the variable associated with the Subnet from which Pulse Connect Secure Internal Interface needs to lease IP.

```json
"pcsVnetIntSubnet" : "PCSInternalSubnet",
```

PCS External Subnet: This is the variable associated with the Subnet from which Pulse Connect Secure External Interface needs to lease IP

```json
"pcsVnetExtSubnet" : "PCSExternalSubnet",
```

PCS Management Subnet: This is the variable associated with the Subnet from which Pulse Connect Secure Management Interface needs to lease IP.

```json
"pcsVnetMgmtSubnet" : "PCSMgmtSubnet",
```

PCS Tunnel VPN Pools Subnet: This is the variable associated with the Subnet which will be configured as Tunnel IP pool in Pulse Connect Secure VPN Profile.

```json
"pcsVnetTunnelPool" : "PCSTunnelVPNPoolSubnet",
```

Backend to PCS Route: This creates route table for accessing the backend resources in Pulse Connect Secure Internal Interface.

```json
"routeTableName" : "Backend2PCSRoute",
```

PCS Internal Private IP: This is the private IP address of the Internal IP.

```json
"pcsIntPrivateIP" : "10.20.1.4",
```

PCS Internal NIC: This is network interface card of PCS Internal network.

```json
"pcsIntNic" : "PCSIInternalNIC",
```

PCS External NIC: This is network interface card of PCS External network.

```json
"pcsExtNic" : "PCSExternalNIC",
```

PCS Management NIC: This is network interface card of PCS Management network.

```json
"pcsMgmtNic" : "PCSMgmtNIC",
```
PCS External Public IP: This is public IP address assigned to PCS External Subnet.

```
"publicIPAddr1" : "PCSEmternalPublicIP"
```

PCS Management Public IP: This is public IP address assigned to PCS Management Subnet.

```
"publicIPAddr2" : "PCSMamagementPublicIP"
```

Public IP Address Type: This variable is defined as static IP.

```
"publicIPAddressType" : "Static"
```

NSG Internal Subnet: This variable defines Network Security Group's Internal Subnet policy.

```
"nsgInt" : "NSGIInternalSportnet"
```

NSG External Subnet: This variable defines Network Security Group's External Subnet policy.

```
"nsgExt" : "NSGIExternalSubnet"
```

NSG Management Subnet: This variable defines Network Security Group's Management Subnet policy.

```
"nsgMgmt" : "NSGIManagementSubnet"
```

VM Name: This variable defines PCS Virtual Machine name.

```
"vmName" : "MyPCSVM"
```

VM Size: This variable defines PCS Virtual Machine size. It is 4 cores, 144MB memory.

```
"vmSize" : "Standard_DS_v2"
```

Virtual Network ID: This variable defines PCS Virtual Network name.

```
"vnetID" : "[resourceId('Microsoft.Network/virtualNetworks',variables('pcsvnetname'))]
```

```
"subnetRefInt" : "[concat(variables('vnetID'),'/subnets/',variables('pcsvnetIntSubnet'))]
```

```
"subnetRefExt" : "[concat(variables('vnetID'),'/subnets/',variables('pcsvnetExtSubnet'))]
```

```
"subnetRefMgmt" : "[concat(variables('vnetID'),'/subnets/',variables('pcsvnetMgmtSubnet'))]
```

API Version

```
"apiVersion" : "2015-06-15"
```

resources

publicIPAddresses/publicIPAddr1: This block is responsible for creating public IP address which is static in nature. This is used for external interface IP address of PCS.

```
"type": "Microsoft.Network/publicIPAddresses",
"name": "[variables('publicIPAddr1')]"
```

publicIPAddresses/publicIPAddr2: This block is responsible for creating public IP address which is static in nature. This is used for management interface IP address of PCS.

```
"type": "Microsoft.Network/publicIPAddresses",
"name": "[variables('publicIPAddr2')]"
```

virtualNetworks/pcvnetname: This block is responsible for creating PCS Virtual Network name. The creation of
PCS Virtual Network name depends on:

- Backend to PCS route
- NSG Internal Subnet
- NSG External Subnet
- NSG Management Subnet

```json
"type": "Microsoft.Network/virtualNetworks",
"name": "[variables('pcsVnetname')]",
```

**virtualNetworks/pcsVnetIntSubnet**: This block is responsible for creating subnet. The created subnet is applied to PCS Internal interface.

```json
"name": "[variables('pcsVnetIntSubnet')]",
```

**virtualNetworks/pcsVnetExtSubnet**: This block is responsible for creating subnet. The created subnet is applied to PCS External interface.

```json
"name": "[variables('pcsVnetExtSubnet')]",
```

**virtualNetworks/pcsVnetMgmtSubnet**: This block is responsible for creating subnet. The created subnet is applied to PCS Management interface.

```json
"name": "[variables('pcsVnetMgmtSubnet')]",
```

**virtualNetworks/pcsVnetTunnelPool**: This block is responsible for creating tunnel pool. The created tunnel pool is applied to PCS Tunnel Pool.

```json
"name": "[variables('pcsVnetTunnelPool')]",
```

**routeTables/routeTableName**: This block is responsible for creating route table. The created route table is used for accessing the backend resources in PCS Internal interface.

```json
"type": "Microsoft.Network/routeTables",
"name": "[variables('routeTableName')]",
```

**networkInterfaces/pcsExtNic**: This block is responsible for creating network interface. The created network interface is applied to network interface card of PCS External interface. The creation of this network interface depends on:

- PCS Virtual Network name
- Public IP address of External Subnet

```json
"type": "Microsoft.Network/networkInterfaces",
"name": "[variables('pcsExtNic')]",
```

**networkInterfaces/pcsMgmtNic**: This block is responsible for creating network interface. The created network interface is applied to network interface card of PCS Management interface. The creation of this network interface depends on:

- PCS Virtual Network name
- Public IP address of Management Subnet

```json
"type": "Microsoft.Network/networkInterfaces",
"name": "[variables('pcsMgmtNic')]",
```

**networkInterfaces/pcsIntNic**: This block is responsible for creating network interface. The created network interface is applied to network interface card of PCS Internal interface. The creation of this network interface depends on:
virtualMachines/PCSVmName: This block is responsible for creating Virtual Machine name. The created Virtual machine name is applied to PCS Virtual Machine. The creation of PCS Virtual Machine name depends on:

- Network Interface Card of PCS Internal interface
- Network Interface Card of PCS External interface
- Network Interface Card of PCS Management interface

networkSecurityGroups/nsgExt: This block is responsible for creating policy. The created policy is applied to Network Security Group's External interface.

The following security rules can be defined:

- allowHTTPS
- allowHTTP
- allowPTP
- allowESP
- allowIKEv2
- denyAll
- denyInternalSubnet
- denyManagementSubnet
- denyPoolRange

networkSecurityGroups/nsgMgmt: This block is responsible for creating policy. The created policy is applied to Network Security Group's Management interface.

The following security rules can be defined:

- allowHTTPS
- allowHTTP
- allowCustomSSH
- allowDMI
- denyAll
- denyInternalSubnet
- denyExternalSubnet

networkSecurityGroups/nsgInt: This block is responsible for creating policy. The created policy is applied to Network Security Group's Internal interface.
The following security rules can be defined:

- denyExternalSubnet
- allow-custom-ssh

## outputs

The outputs section defines the public IP address and FQDN that is displayed on successful deployment of PCS on Azure.

```json
"outputs": {
  "hostname": {
    "type": "string",
    "value": "[reference(variables('publicIPAddr1')).dnsSettings.fqdn]"
  }
}
```
Appendix C: Pulse Connect Secure Resource Manager Template for an Existing Virtual Network

Pulse Secure provides sample Azure template files to deploy Pulse Connect Secure Virtual Appliance on Azure. Users can modify this to make it suitable for their need. Visit https://www.pulsesecure.net and download the pulsesecure-pcs-3-nics.zip file, and unzip it to get azuredeploy.json.

This template creates a new PCS with 3 NICs, Vnet, four subnets, NSG policies attached to PCS internal, external and management subnets and user-defined routes on the PCS internal subnet to ensure PCS is used as default gateway for L3 tunnel. All 3 NICs of PCS are configured with dynamic IP configuration and enabled IP forwarding. Public IPs are attached to the PCS external and management NIC.

The template has following sections:

<table>
<thead>
<tr>
<th>Section</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>parameters</td>
<td>This section defines the parameters used for deploying PCS on Azure. It contains parameter name, its default value and the mouse-over help text that is displayed when mouse is placed over the parameter in Azure Web portal. The parameters defined here are displayed in the Custom Deployment page of Azure portal.</td>
</tr>
<tr>
<td>variables</td>
<td>This section defines variables that will be used in the functions defined in the resources section.</td>
</tr>
<tr>
<td>resources</td>
<td>This section defines resource types that are deployed or updated in a resource group.</td>
</tr>
<tr>
<td>outputs</td>
<td>This section defines the public IP address and FQDN returned after successful deployment of PCS on Azure.</td>
</tr>
</tbody>
</table>

parameters

Figure 41: Custom Deployment
PCS Storage Account Name: This is the name of the PCS Storage Account where the PCS Azure vhd image is stored.

```json
"parameters": {
  "PCSStorageAccountName": {
    "type": "string",
    "defaultValue": "pogoldenstorage",
    "metadata": {
      "description": "Storage account name where PCS image is uploaded"
    }
  }
},
```

PCS Storage Account Resource Group Name: The is the name of the PCS Storage Account Resource Group where the PCS Azure vhd image is stored.

```json
"PCSStorageAccountResourceGroupName": {
  "type": "string",
  "defaultValue": "GoldenImageRG",
  "metadata": {
    "description": "Resource group of the existing storage account where PCS image is uploaded"
  }
},
```

PCS Image Location URI: The is the URL to the location where PCS Azure vhd image is stored.

```json
"PCSImageLocationURI": {
  "type": "string",
  "defaultValue": "https://pogoldenstorage.blob.core.windows.net/master/pog-azure.vhd",
  "metadata": {
    "description": "URL of PCS vhd image"
  }
},
```

PCS VM Name: This is the name given to Pulse Connect Secure Virtual Appliance.

```json
"PCSVMName": {
  "type": "string",
  "defaultValue": "PCSAzureVA",
  "metadata": {
    "description": "Pulse Connect Secure Name"
  }
},
```

PCS Config: This section contains provisioning parameters that are required during the deployment of a Virtual Appliance. An XML-based configuration file can be present in another Virtual Machine in Azure cloud or in the corporate network which is accessible for Pulse Connect Secure through site-to-site VPN between Azure and the corporate data center.

Pulse Connect Secure accepts the following parameters as provisioning parameters:

- primary-dns
- secondary-dns
- wins-server
- dns-domain
- username
- ssh-publickey
- cert-common-name
- cert-random-text
- cert-organization
- config-download-url
- config-data
- auth-code-license

For details about these parameters, see Pulse Connect Secure Provisioning Parameters.
SSH Public Key: This is an RSA public key that is used to access Pulse Connect Secure via SSH.

DNS Label Prefix Ext: This is the prefix for External Interface DNS label.

DNS Label Prefix Mgmt: This is the prefix for Management Interface DNS label.

Resource Group Name of Exiting Virtual Network: Name of the Resource Group that contains the existing Virtual network.

Existing Virtual Network Name: Name of the existing Virtual network.

Existing Internal Subnet: Subnet from which Pulse Connect Secure Internal Interface needs to lease IP.
**Existing External Subnet:** Subnet from which Pulse Connect Secure External Interface needs to lease IP.

```json
"existingExternalSubnet": {
  "type": "string",
  "defaultValue": "subnet1",
  "metadata": {
    "description": "PCS external interface connects to this subnet"
  }
}
```

**Existing Management Subnet:** Subnet from which Pulse Connect Secure Management Interface needs to lease IP.

```json
"existingManagementSubnet": {
  "type": "string",
  "defaultValue": "subnet2",
  "metadata": {
    "description": "PCS management interface connects to this subnet"
  }
}
```

**Existing Tunnel Subnet:** Subnet configured as Tunnel IP pool in Pulse Connect Secure VPN profile.

```json
"existingTunnelSubnet": {
  "type": "string",
  "defaultValue": "subnet3",
  "metadata": {
    "description": "Subnet used for VPN Pools"
  }
}
```

**variables**

**Backend to PCS Route:** This creates route table for accessing the backend resources in Pulse Connect Secure Internal Interface.

`"routeTableName" : "Backend2PCSRout"`

**PCS Internal NIC:** This is network interface card of PCS Internal network.

`"pcsIntNic" : "PCSInternalNIC"`

**PCS External NIC:** This is network interface card of PCS External network.

`"pcsExtNic" : "PCSEnternalNIC"`

**PCS Management NIC:** This is network interface card of PCS Management network.

`"pcsMgmtNic" : "PCSMangementNIC"`

**PCS External Public IP:** This is public IP address assigned to PCS External Subnet.

`"publicIPAddr1" : "PCSEnternalPublicIP"`

**PCS Management Public IP:** This is public IP address assigned to PCS Management Subnet.

`"publicIPAddr2" : "PCSMangementPublicIP"`

**Public IP Address Type:** This variable is defined as static IP.

`"publicIPAddressType" : "Static"`

**NSG Internal Subnet:** This variable defines Network Security Group's Internal Subnet policy.

`"nagint" : "NSGInternalSubnet"`
**NSG External Subnet:** This variable defines Network Security Group's External Subnet policy.

```
"nsgExt": "NSGExternalSubnet",
```

**NSG Management Subnet:** This variable defines Network Security Group's Management Subnet policy.

```
"nsgMgmt": "NSGManagementSubnet",
```

**VM Size:** This variable defines PCS Virtual Machine size. It is 4 cores, 144MB memory.

```
"vmSize": "Standard_D3_v2",
```

**Virtual Network ID:** This variable defines PCS Virtual Network name.

```
"vnetID": 
  
  [resourceId('Microsoft.Network/VirtualNetworks',variables('pcsNetname'))]

subnetRefInt: 
  
  [concat(variables('vnetID'),'/subnets/',variables('pcsVnetIntSubnet'))]

subnetRefExt: 
  
  [concat(variables('vnetID'),'/subnets/',variables('pcsVnetExtSubnet'))]

subnetRefMgmt: 
  
  [concat(variables('vnetID'),'/subnets/',variables('pcsVnetMgmtSubnet'))]

subnetRefTunnel: 
  
  [concat(variables('vnetID'),'/subnets/parameters(existingTunnelSubnet))]
```

**API Version**

```
"apiVersion": "2015-06-15"
```

**resources**

**publicIPAddresses/publicIPAddr1:** This block is responsible for creating public IP address which is static in nature. This is used for external interface IP address of PCS.

```
"type": "Microsoft.Network/publicIPAddresses",
"name": "[variables('publicIPAddr1')]",
```

**publicIPAddresses/publicIPAddr2:** This block is responsible for creating public IP address which is static in nature. This is used for management interface IP address of PCS.

```
"type": "Microsoft.Network/publicIPAddresses",
"name": "[variables('publicIPAddr2')]",
```

**networkSecurityGroups/nsgExt:** This block is responsible for creating policy. The created policy is applied to Network Security Group's External interface.

```
"type": "Microsoft.Network/networkSecurityGroups",
"name": "[variables('nsgExt')]",
```

The following security rules can be defined:

- allowHTTPS
- allowHTTP
- allowPTP
- allowESP
- allowIKEv2
- denyAll
- denyInternalSubnet
- denyManagementSubnet
• denyPoolRange

**networkSecurityGroups/nsgMgmt**: This block is responsible for creating policy. The created policy is applied to Network Security Group's Management interface.

```json
"type": "Microsoft.Network/networkSecurityGroups",
"name": ".variables\[nsgMgmt\]"
```

The following security rules can be defined:

- allowHTTPS
- allowHTTP
- allowCustomSSH
- allowDMI
- denyAll
- denyInternalSubnet
- denyExternalSubnet

**networkSecurityGroups/nsgInt**: This block is responsible for creating policy. The created policy is applied to Network Security Group's Internal interface.

```json
"type": "Microsoft.Network/networkSecurityGroups",
"name": ".variables\[nsgInt\]"
```

The following security rules can be defined:

- denyExternalSubnet
- allow-custom-ssh

**routeTables/routeTableName**: This block is responsible for creating route table. The created route table is used for accessing the backend resources in PCS Internal interface.

```json
"type": "Microsoft.Network/routeTables",
"name": ".variables\[routeTableName\]"
```

**networkInterfaces/pcsExtNic**: This block is responsible for creating network interface. The created network interface is applied to network interface card of PCS External interface. The creation of this network interface depends on:

- PCS Virtual Network name
- Public IP address of External Subnet

```json
"type": "Microsoft.Network/networkInterfaces",
"name": ".variables\[pcsExtNic\]"
```

**networkInterfaces/pcsMgmtNic**: This block is responsible for creating network interface. The created network interface is applied to network interface card of PCS Management interface. The creation of this network interface depends on:

- PCS Virtual Network name
- Public IP address of Management Subnet

```json
"type": "Microsoft.Network/networkInterfaces",
"name": ".variables\[pcsMgmtNic\]"
```

**networkInterfaces/pcsIntNic**: This block is responsible for creating network interface. The created network interface is applied to network interface card of PCS Internal interface. The creation of this network interface
depends on:

- PCS Virtual Network name

```json
"type": "Microsoft.Network/networkInterfaces",
"name": "[variables('pcsIntNic')]",
```

**virtualMachines/PCSVmName**: This block is responsible for creating Virtual Machine name. The created Virtual machine name is applied to PCS Virtual Machine. The creation of PCS Virtual Machine name depends on:

- Network Interface Card of PCS Internal interface
- Network Interface Card of PCS External interface
- Network Interface Card of PCS Management interface

```json
"type": "Microsoft.Compute/virtualMachines",
"name": "[parameters('PCSVmName')]",
```

**outputs**

The outputs section defines the public IP address and FQDN that is displayed on successful deployment of PCS on Azure.

```json
"outputs": {
  "hostname": {
    "type": "string",
    "value": "[reference(variables('publicIPAddr1')).dnsSettings.fqdn]"
  }
}
```
References

Microsoft Azure documentation: https://docs.microsoft.com/en-us/azure/

Requesting Technical Support

Technical product support is available through the Pulse Secure Global Support Center (PSGSC). If you have a support contract, then file a ticket with PSGSC.

- Product warranties—for product warranty information, visit https://www.pulsesecure.net.