Pulse Policy Secure Virtual Appliance on Amazon Web Services

Deployment Guide

Document Revision	1.2
Published Date	November 2020

Pulse Secure, LLC
2700 Zanker Road,
Suite 200 San Jose
CA 95134

https://www.pulsesecure.net

Pulse Policy Secure Virtual Appliance on Amazon Web Services - Deployment Guide

The information in this document is current as of the date on the title page.

END USER LICENSE AGREEMENT

Revision History

Revision and Date	Added/Updated/Removed	Remarks
1.2 November 2020	PPS in AWS Marketplace	
1.1 September 2020	Accessing the Pulse Policy Secure Virtual Appliance as an Administrator section is updated with PPS IP and credentials.	
1.0 March 2020	First version of the document.	
1.1 September 2020	Updated the "Accessing the Pulse Policy Secure Virtual Appliance as an Administrator" section with PPS url and default password.	

Table of Contents

Overview

About This Guide

This guide helps in deploying the Pulse Policy Secure Virtual Appliance on Amazon Web Services (AWS). A Pulse Policy Secure administrator can manually upload the Pulse Policy Secure Virtual Appliance image (AMI) into AWS storage account. Once the AMI package is available in the AWS storage account, the Pulse Policy Secure administrator can deploy Pulse Policy Secure on AWS in the cloud.

Assumptions

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

Pulse Policy Secure in AWS Marketplace

Pulse Policy Secure is made available in AWS Market Place. The CloudFormation templates are available at Amazon marketplace.

Prerequisites and System Requirements on AWS Marketplace

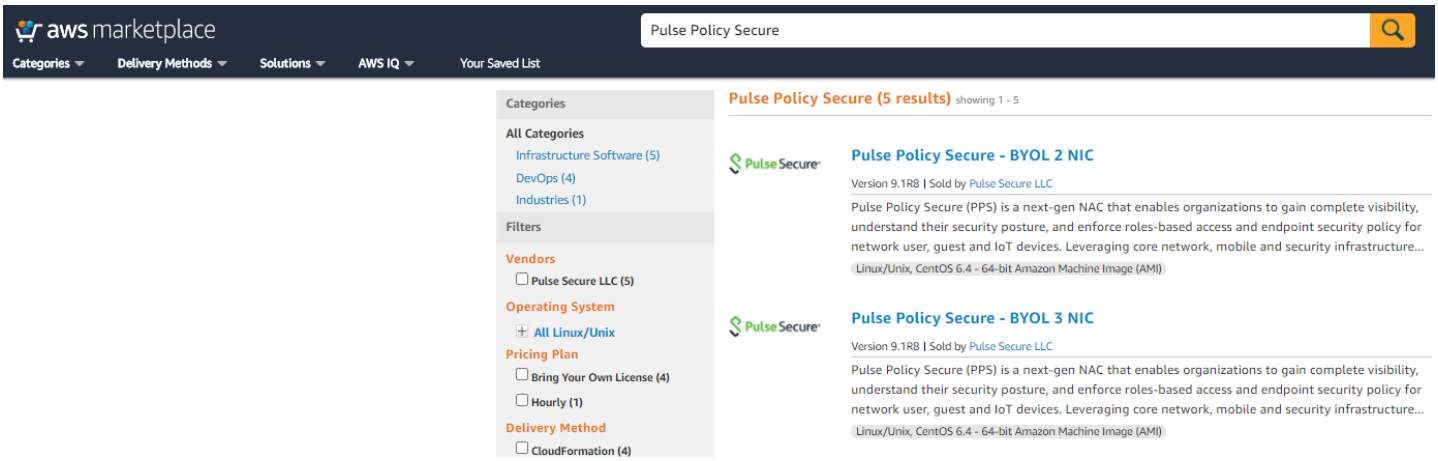
To deploy the Pulse Policy Secure Virtual Appliance on AWS Marketplace, you need the following:

- An AWS account
- Access to the AWS Marketplace (https://aws.amazon.com/marketplace)
- Pulse Policy Secure licenses *

Deploying Pulse Policy Secure on AWS Marketplace

1. Launch AWS Marketplace using the url: https://aws.amazon.com/marketplace and search with keyword Pulse Policy Secure.

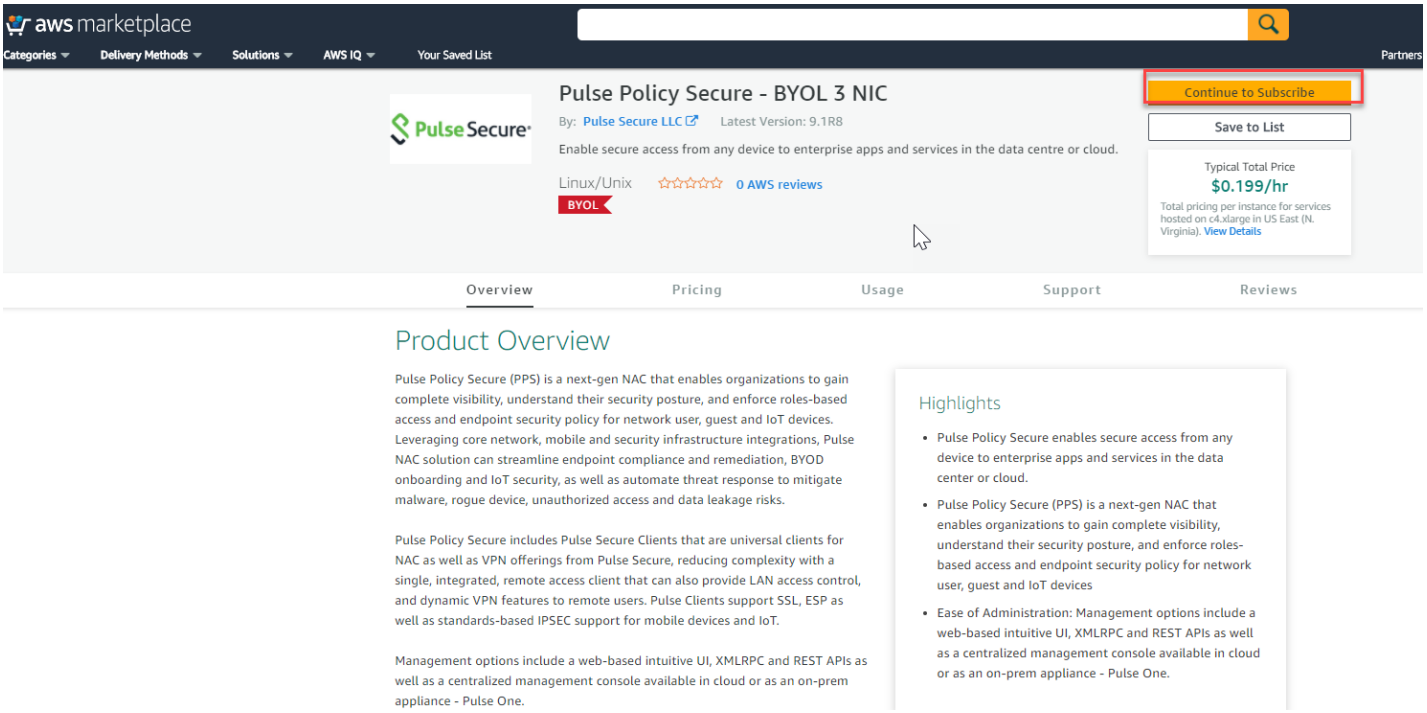
Figure 1: AWS Marketplace



AWS Marketplace contains the following two Pulse Policy Secure SKUs:

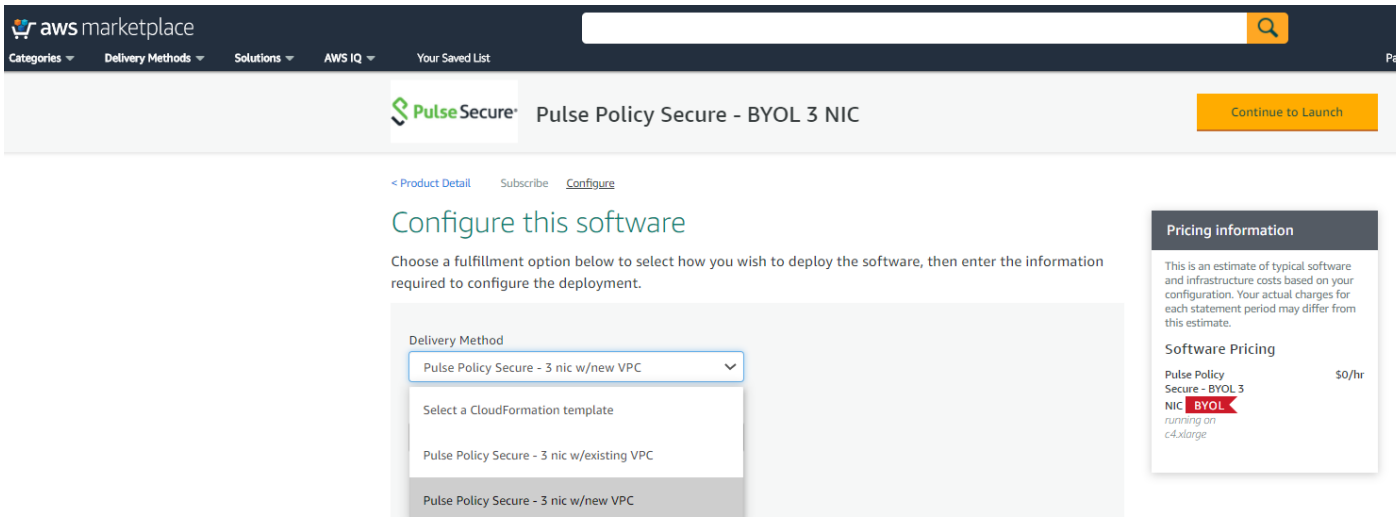
- Pulse Policy Secure - BYOL 2 NIC
- Pulse Policy Secure - BYOL 3 NIC

Figure 2: Subscribe to Pulse Policy Secure – BYOL 3 NIC

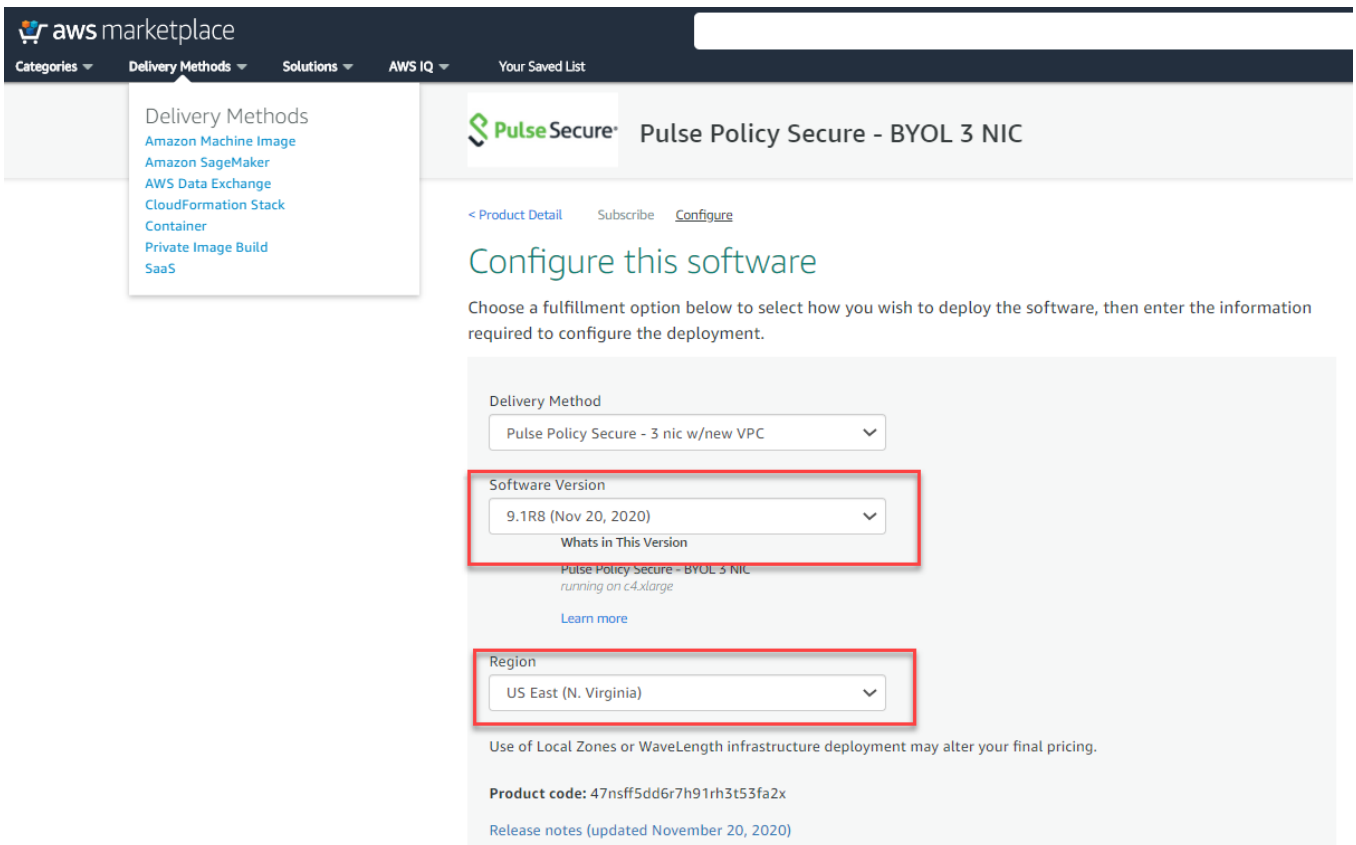


2. Select either 3-NIC model or 2-NIC model based on your requirement. In the Product Subscription page displayed, click **Continue to Subscribe**. In this section, 3-NIC model is chosen as example.
3. After subscribing, proceed to configuration by clicking **Continue to Configuration**.
4. Under Delivery Method, select either Existing VPC or New VPC that you want to deploy and click **Continue to Launch**.

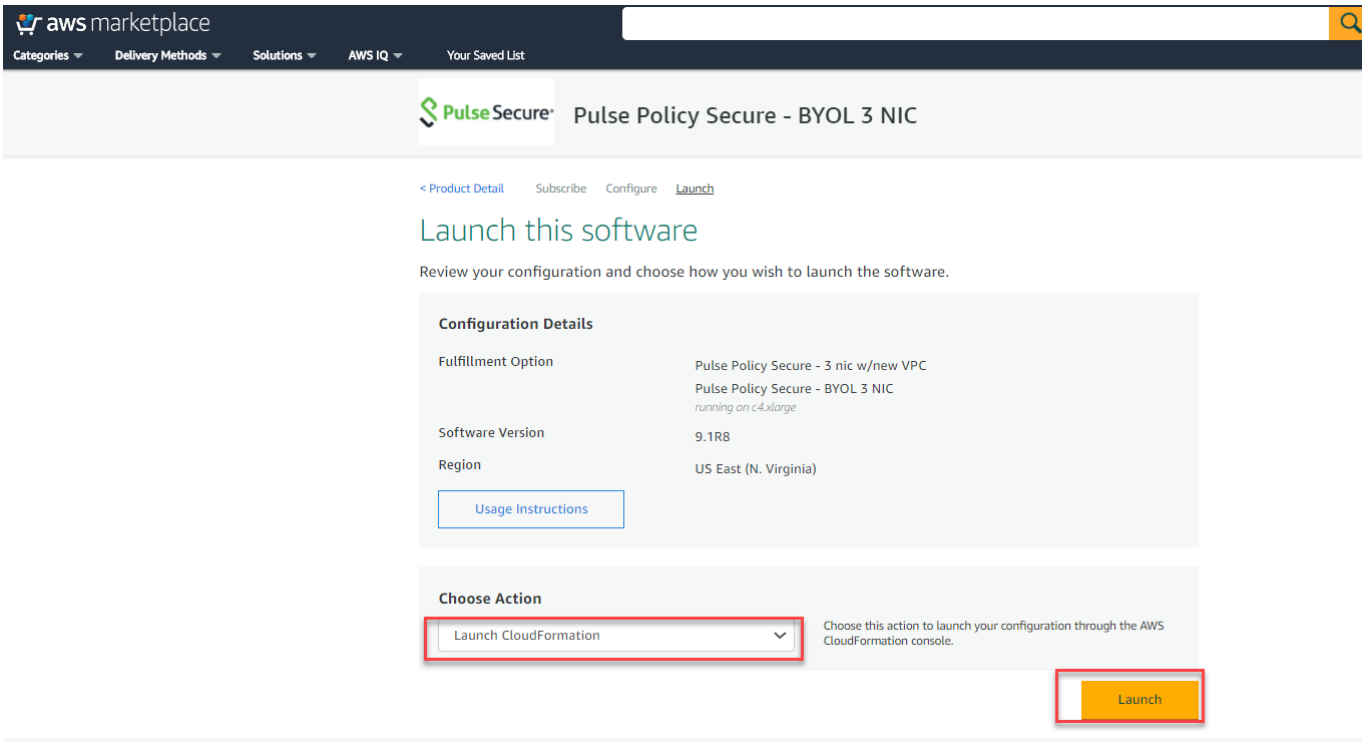
Figure 3: Launch CloudFormation



5. Select the Software Version and the Region to deploy the software.



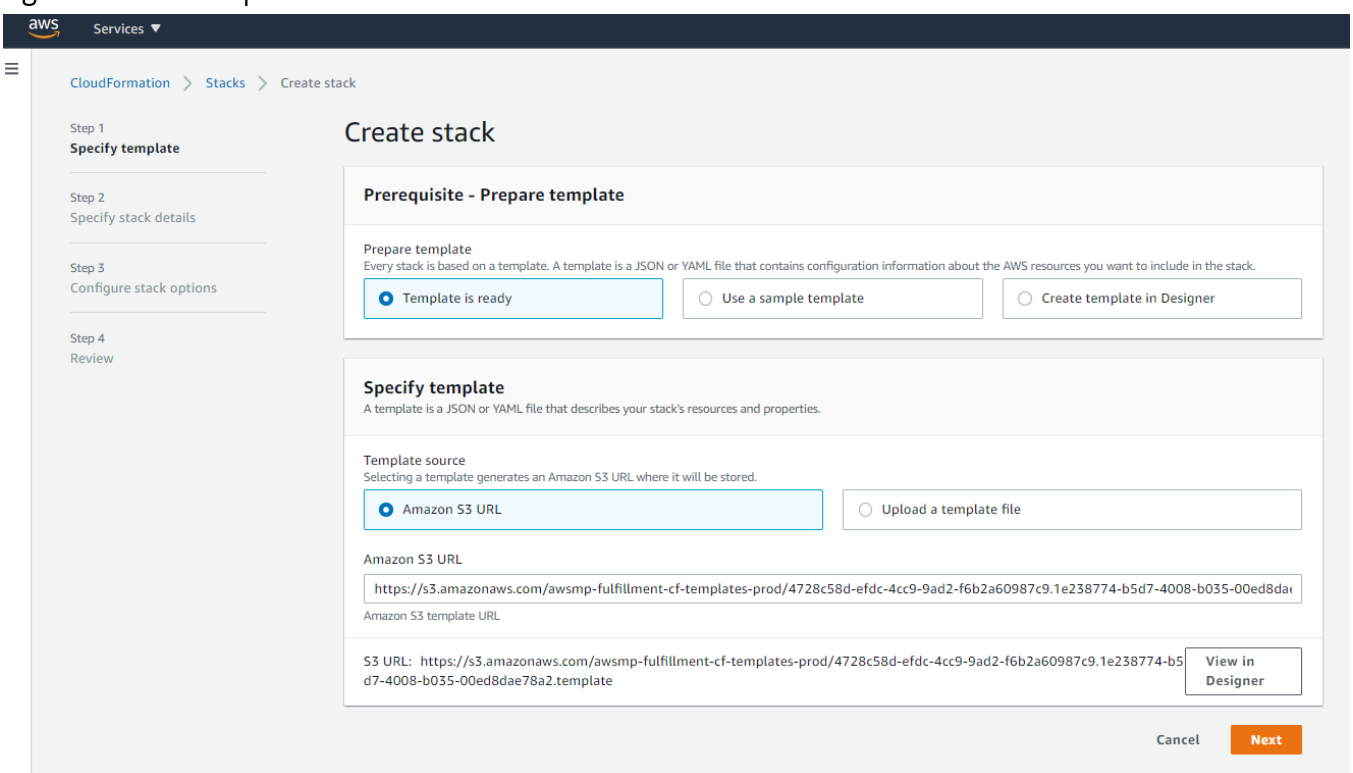
In the Launch page displayed, select **Launch CloudFormation** and click **Launch**.



Select Template

6. In the Create stack wizard, in the Select Template page choose the template that describes your stack's resources and their properties and, click **Next**.

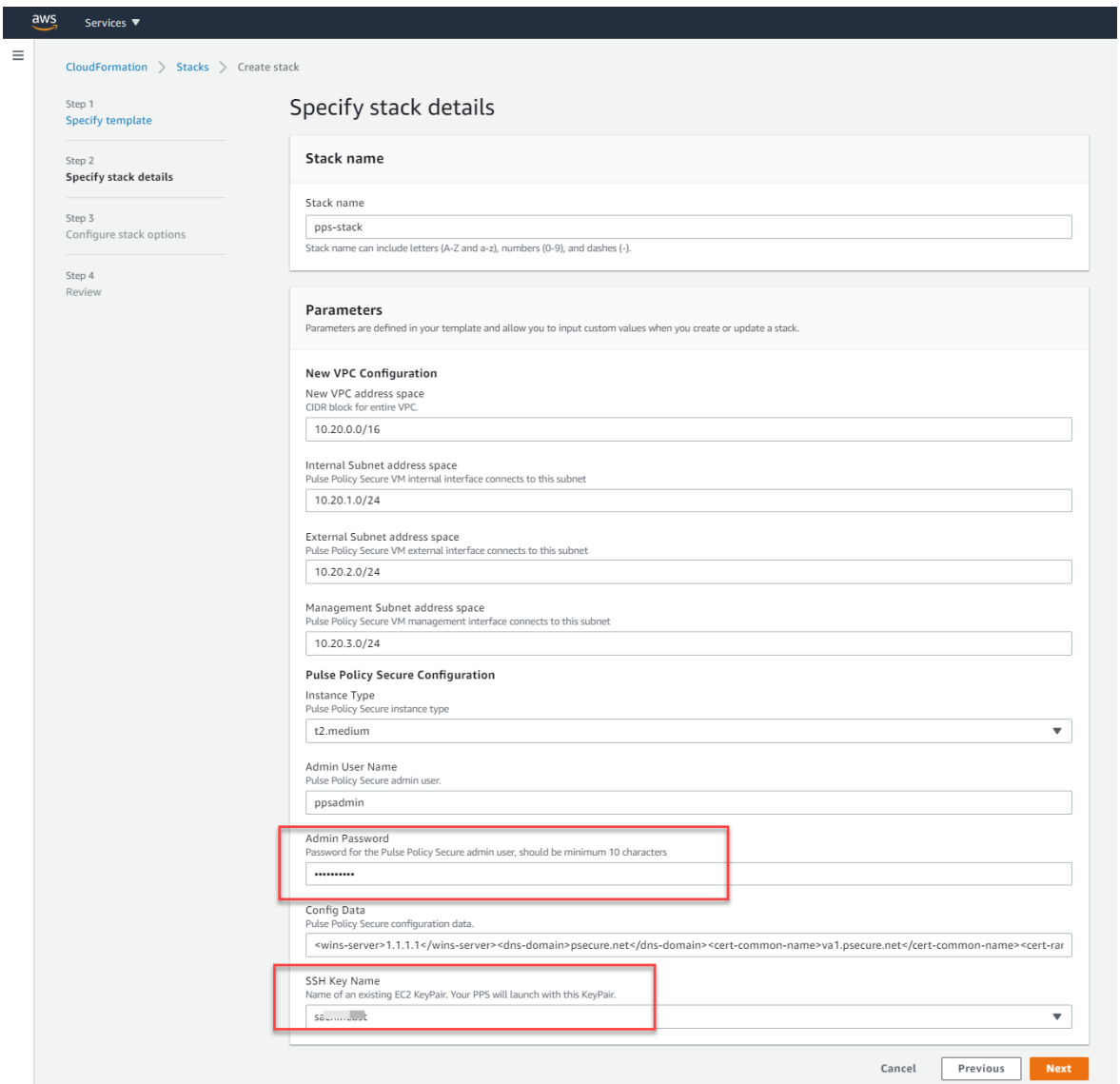
Figure 4: Select Template



Specify Details

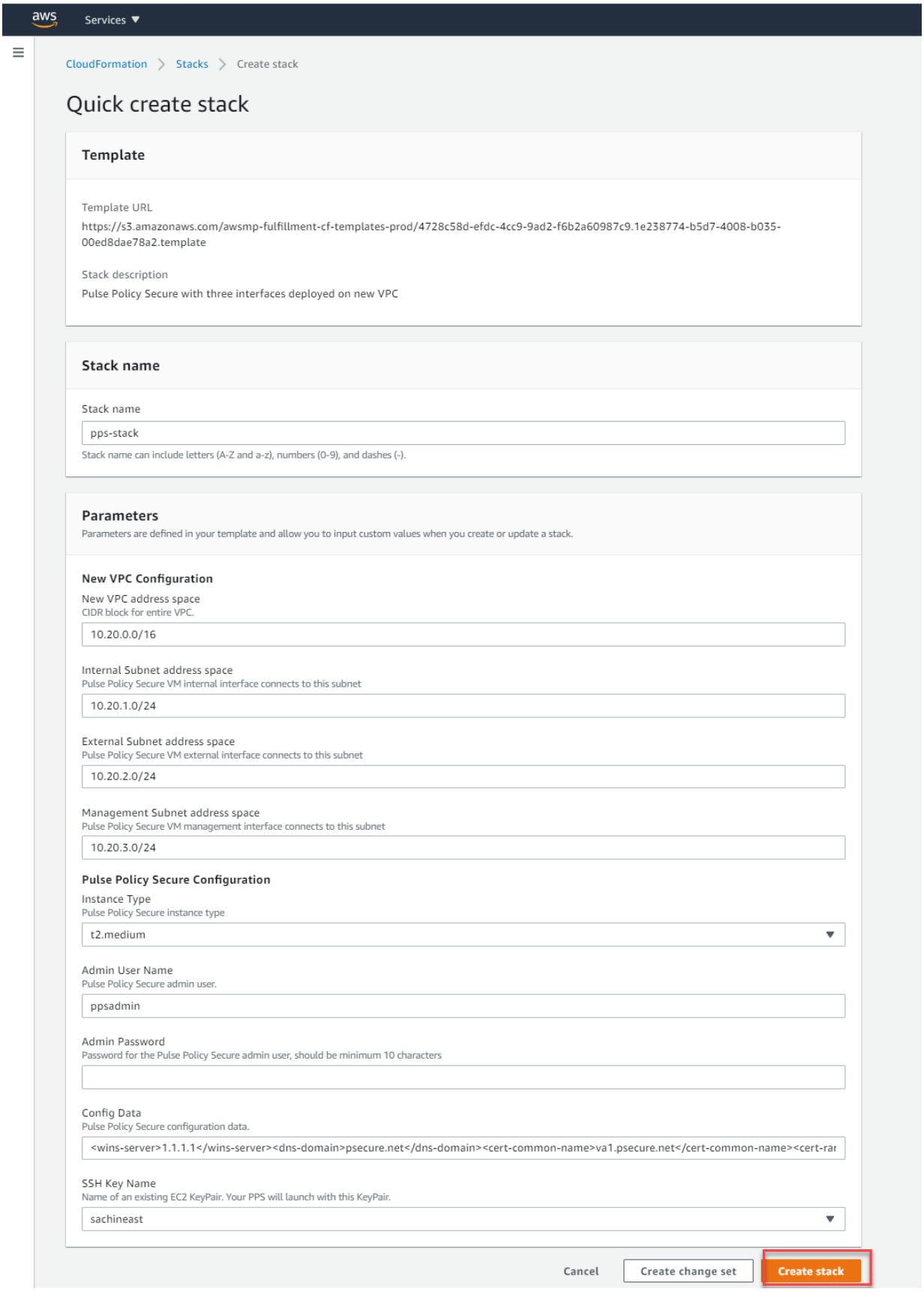
7. In the Specify Stack Details page, specify a name for the stack.
8. In the Parameters section, use the default parameter values. These are defined in the CloudFormation template.
9. In the Pulse Policy Secure Configuration section:
   - Select Pulse Policy Secure VM size. Default: t2.medium
   - PPS admin username is configured by default. Default: ppsadmin. You can give any other user name if you want to.
   - Enter the Admin user password.
   - Config Data: Provisioning parameters in an XML format. For details, see Pulse Policy Secure Provisioning Parameters.
   - Select SSH Key Name of EC2 key pair. This key is used to access PPS 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 http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ec2-key-pairs.html.

Figure 5: Specify Configuration Details

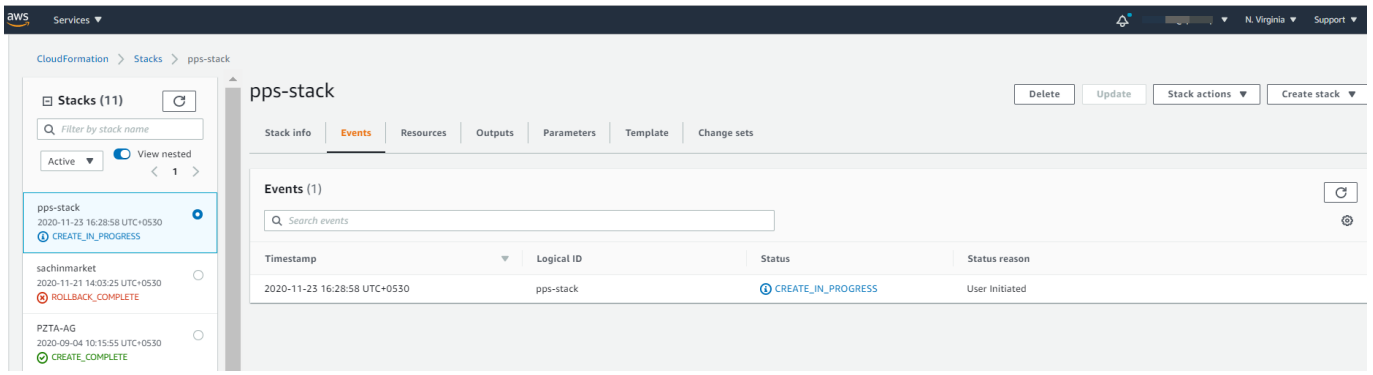


Review

10. In the Review page, verify the details and click **Create Stack**.



11. Wait for a few minutes while it creates all the resources. This completes deploying PPS on AWS Marketplace.



To access Pulse Policy Secure Virtual Appliance, see Accessing the Pulse Policy Secure Virtual Appliance

Pulse Policy Secure on Amazon Web Services

Prerequisites and System Requirements on AWS

To deploy the Pulse Policy Secure Virtual Appliance on AWS, you need the following:

- AWS account
- Access to the AWS portal (https://console.aws.amazon.com/)*
- Pulse Policy Secure Virtual Appliance Image (.ami file)
- AWS CloudFormation template
- Pulse Policy Secure licenses **
- Site-to-Site VPN between AWS and the corporate network (optional)

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

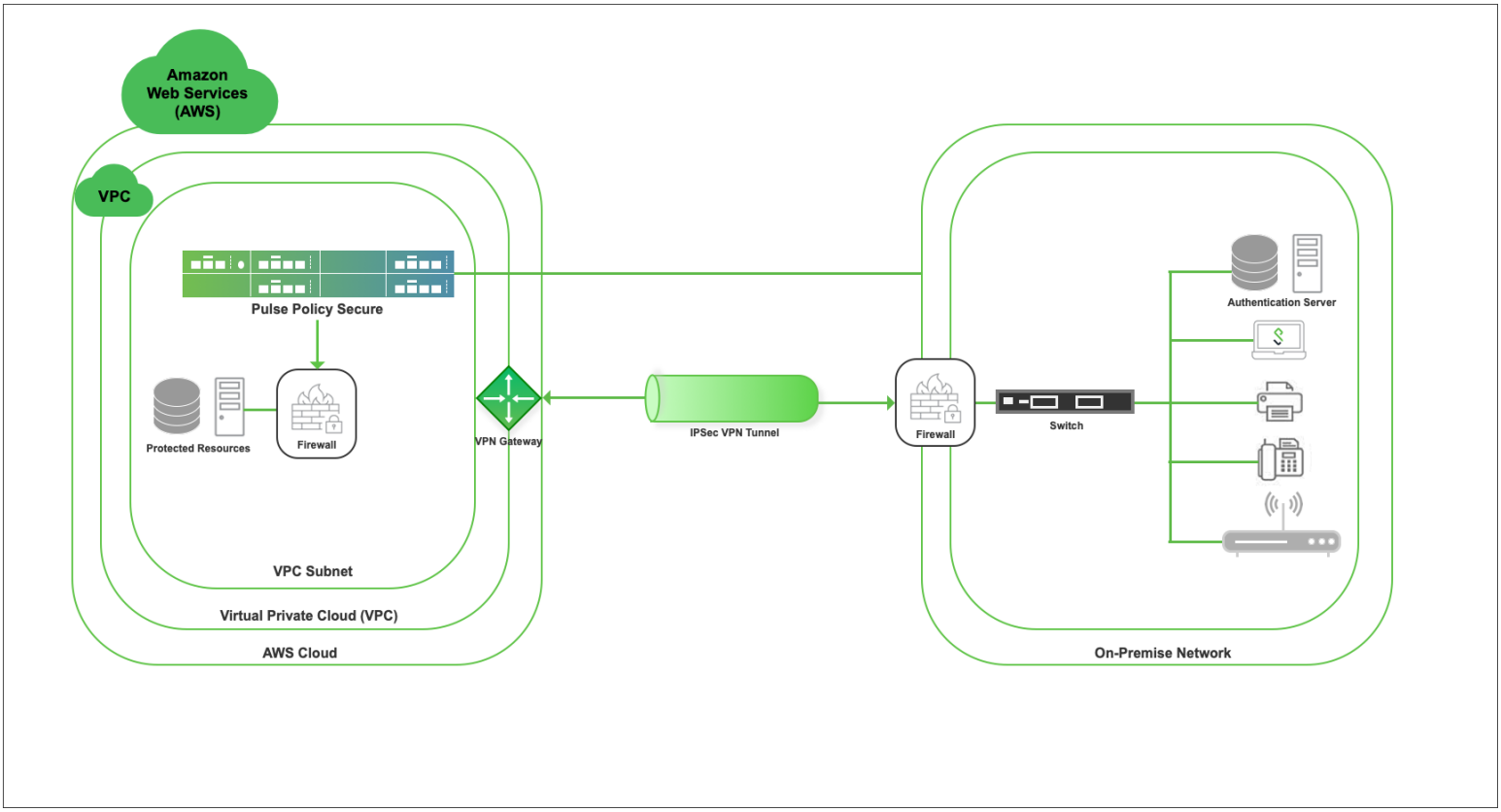
- Pulse License Server (optional)**
  - Located at corporate network, accessible through site-to-site VPN
- Pulse Policy Secure configuration in XML format (optional)

Note:
* Pulse Policy Secure Virtual Appliance can be deployed only through AWS CloudFormation style.
** From 9.0R3 release, Pulse Policy Secure Virtual Appliance, by default, has two evaluation licenses, and supports licensing with License server located at corporate network.

Deploying Pulse Policy Secure on Amazon Web Services

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

Figure 6: Pulse Policy Secure on AWS



Supported Platform Systems

This section helps you in choosing the instance types that should be deployed with Pulse Policy Secure for AWS.

- PSA3000v is equivalent to t2 medium
- PSA5000v is equivalent to t2.xlarge
- PSA7000v is equivalent to t2.2xlarge

Model	vCPU	CPU Credits / hour	Memory (GiB)	Storage
t2.medium	2	24	4	EBS-Only
t2.large	2	36	8	EBS-Only
t2.xlarge	4	54	16	EBS-Only

Steps to Deploy Pulse Policy Secure on AWS

Below is the one-time activity to be followed to deploy Pulse Policy Secure on AWS.

- Registering the AMI

Below is the step to be followed for each deployment of Pulse Policy Secure.

- Deploying Pulse Policy Secure on AWS using AWS Portal

Registering the AMI

This section describes the steps to register the AMI.

Prerequisites

- AWS command line should be configured on the host.
- the image should be available locally on the host.

To register AMI, do the following:

1. Download PPS Xen image which is in zip format from Pulse support site and unzip the file.
2. Install AWS CLI on the client machine. For the software and installation details, refer the link https://aws.amazon.com/cli/.
3. Copy PPS Xen image on the client machine.
4. Create Amazon S3 bucket and VM Import service role by following the procedures mentioned in https://docs.aws.amazon.com/vm-import/latest/userguide/vmimport-image-import.html#vmimport-iam-permissions
5. Upload the PPS Xen image to AWS S3 bucket by typing the following command:

   ```
   aws s3 cp <image> s3://<bucket>/<folder>/<imagename>
   ```

   where, bucket and folders are created in the desired S3 location.
6. Create a snapshot by doing the following:

   a. Prepare a container json file by entering the details:

   ```
   $ cat container.json
     {
      "Description": "fill-description",
      "Format": "raw",
      "UserBucket": {
         "S3Bucket": "bucket-name-where-image-is-uploaded",
         "S3Key": " path of image: <folder>/<imagename>"
       }
      }
   ```

   b. After preparing container.json appropriately, run the following command:

   ```
   aws ec2 import-snapshot --description “<description>" --disk-container file:container.json --region <your-ec2-region>
   ```

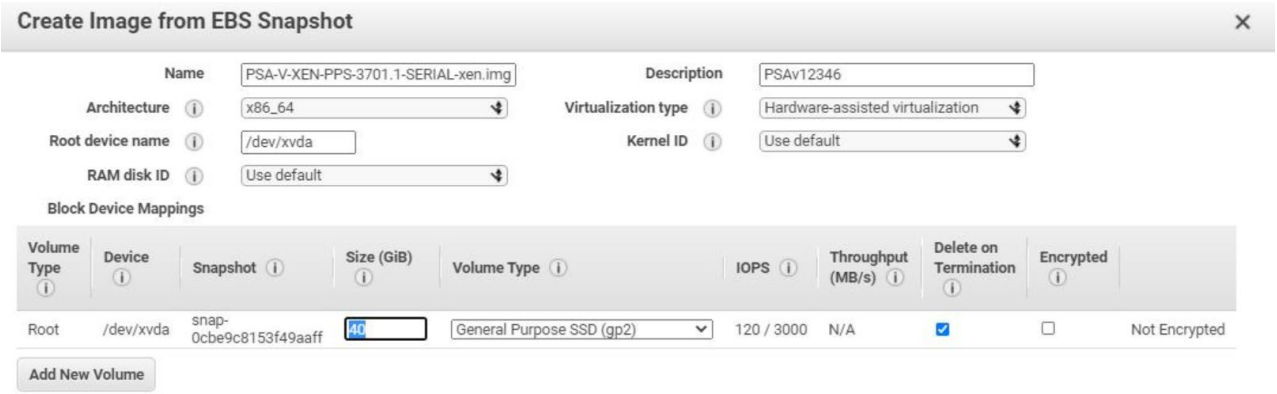
   This command will return a json file describing the status. Make a note of the “ImportTaskId” field from the json output.

   c. Monitor the progress by running the following command:

   ```
   aws ec2 describe-import-snapshot-tasks --region <your-ec2-region> --import-task-ids <import-task-id>
   ```

   Monitor the progress until the “status:Completed” message appears, and a snapshotId is added in the json output. Make note of the “SnapshotId”.
7. Register an AMI from the snapshot by running the following command:

   ```
   aws ec2 register-image --description “<description>" --architecture x86_64 --name <image-name> --block-device-mappings DeviceName="/dev/xvda",Ebs={SnapshotId=<snapshot-id>} --virtualization-type hvm --root-device-name "/dev/xvda" --region <your-ec2-region>
   ```
8. Once the snapshot is created, you can also copy the snapshot ID from the AWS console (Services > EC2 > Elastic Block Store > Snapshots). Select the snapshot and click **Actions** > **Create Image**. This completes AMI registration.



Deploying Pulse Policy Secure on AWS using AWS Portal

Once the access to the AMI file and CloudFormation template is obtained as mentioned in the above section, proceed with the Pulse Policy Secure deployment.

Pulse Policy Secure can be deployed:

- on a new Virtual Private Cloud or
- on an already existing Virtual Private Cloud
- as a license server

Deploying PPS on New Virtual Private Cloud

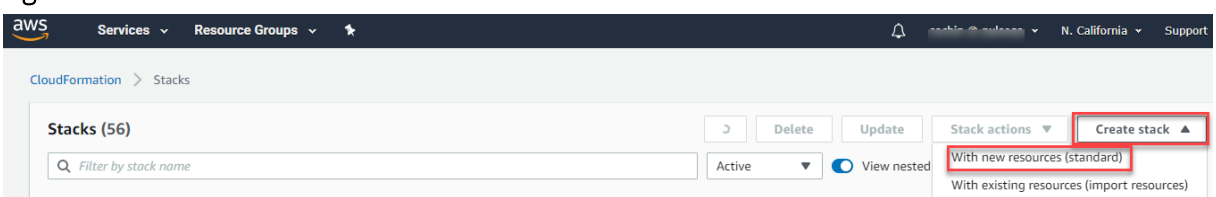
This section describes PPS deployment with three NIC cards and two NIC cards.

Deployment on VM with Three NIC Cards

To deploy Pulse Policy Secure on AWS using the AWS portal, do the following:

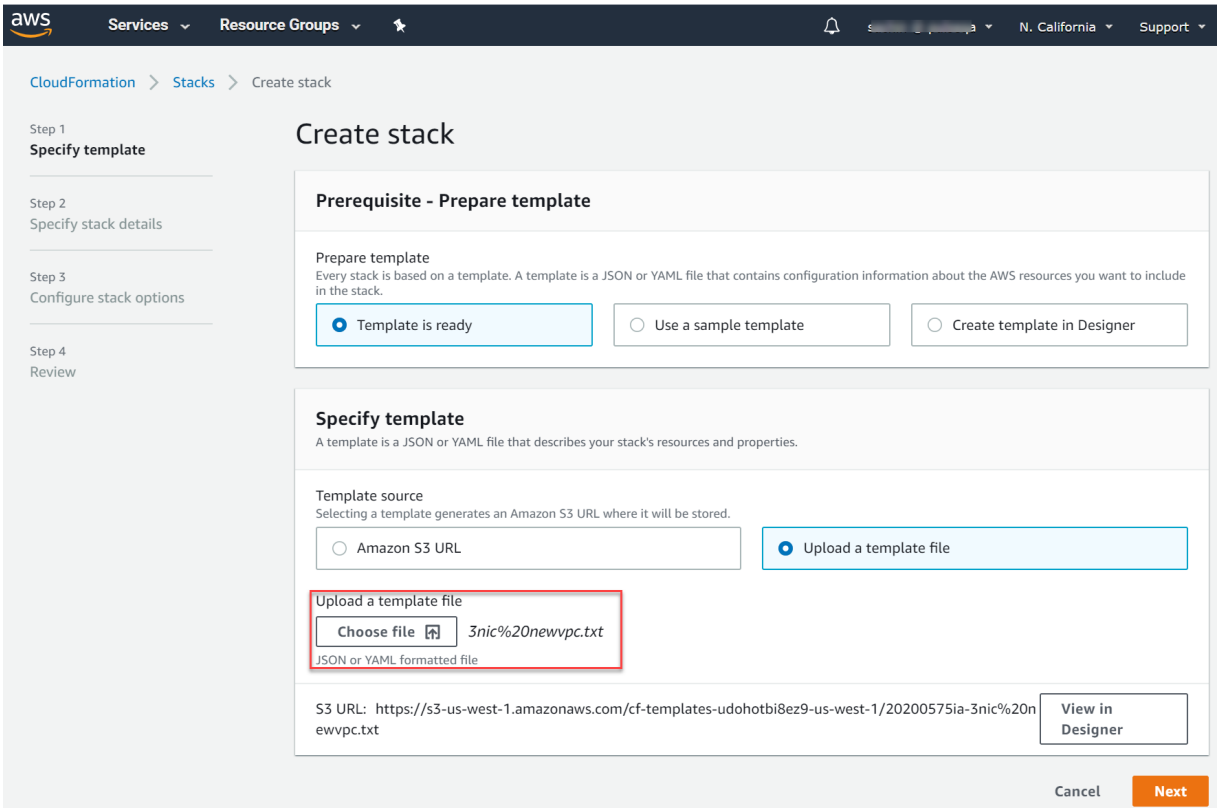
1. Select **AWS Services > CloudFormation** and click **Create stack** > **With new resources (standard)**.

Figure 7: Create New Stack



2. Select **Upload a template file**. Click **Choose File** and select "pulsesecure-pps-3-nics-new-network.json" template file for the new VPC. Click **Next**.

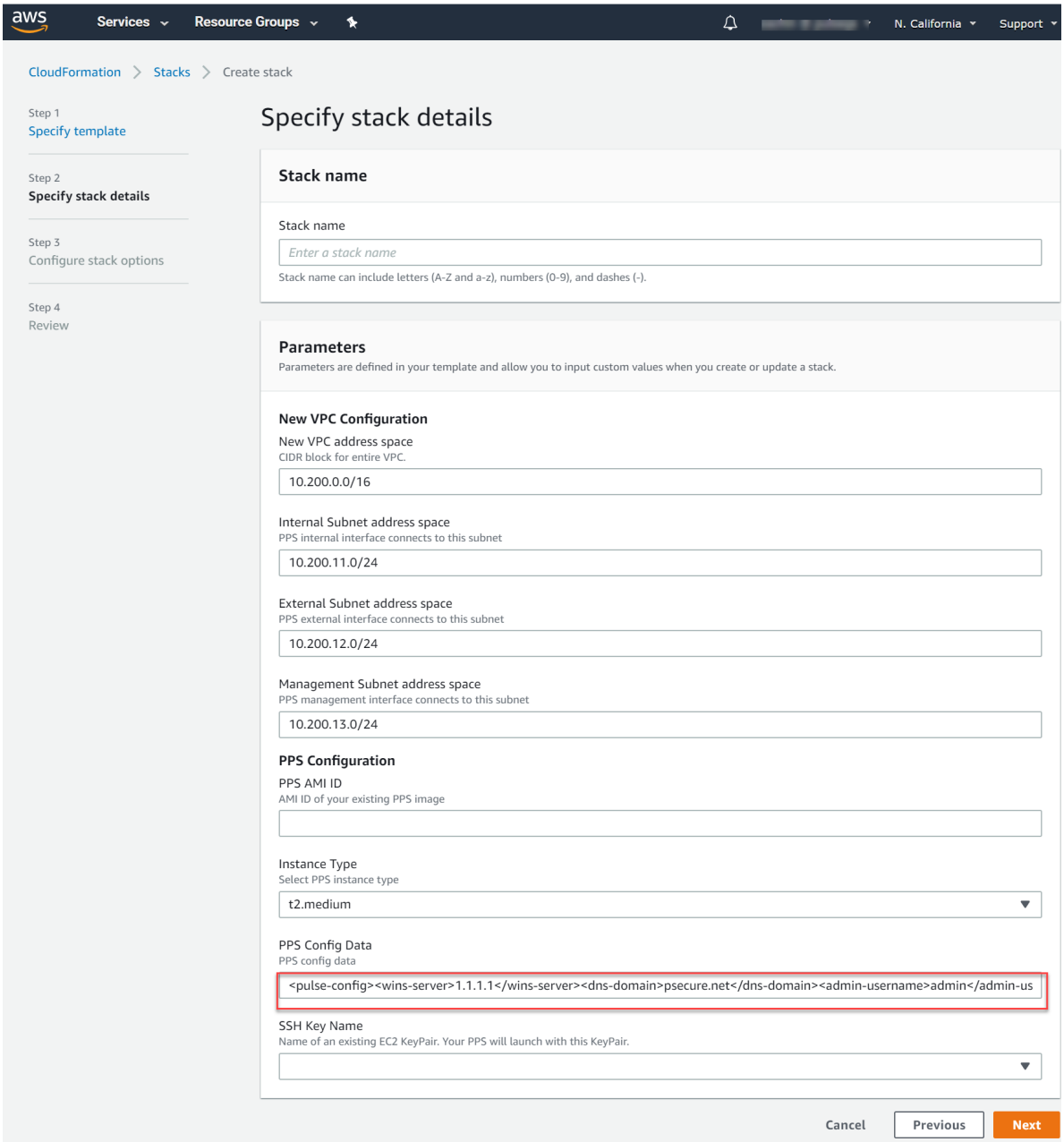
Figure 8: Upload Template



3. In the Specify stack details page, fill or modify the following parameters.

Note: Before proceeding with deployment, ensure that the attribute "**accept-license-agreement"** in PPSConfigData is set to "**y**".

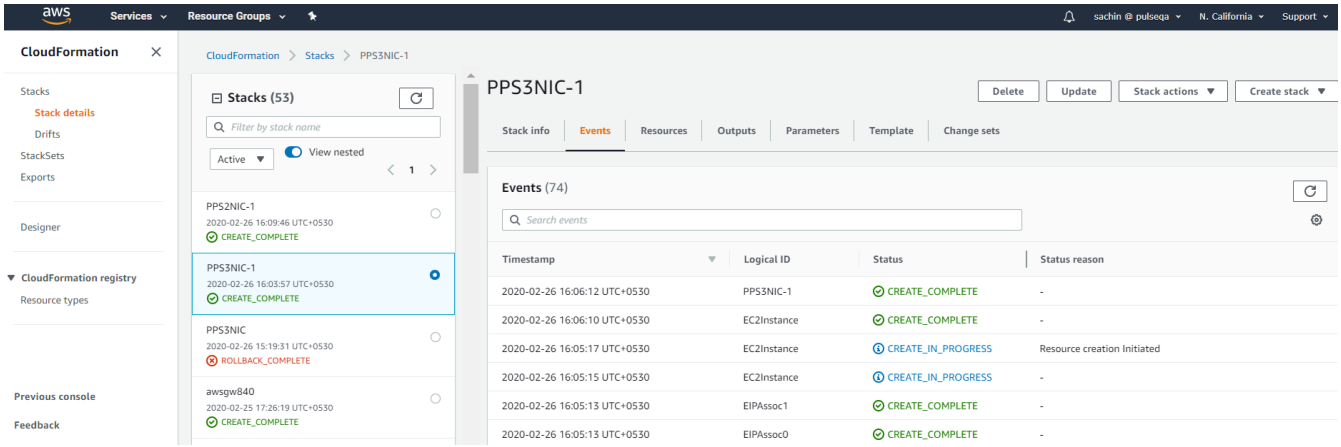
Figure 9: Specify Details for New Virtual Private Cloud



- **Stack name**: Specify the stack name in which Pulse Policy Secure needs to be deployed
- **New VPC address space**: Virtual private cloud address space
- **Internal Subnet address space**: Subnet from which Pulse Policy Secure internal interface needs to lease IP
- **External Subnet address space**: Subnet from which Pulse Policy Secure external interface needs to lease IP
- **Management Subnet address space**: Subnet from which Pulse Policy Secure management interface needs to lease IP
- **PPS AMI ID**: ID of the uploaded AMI file
- **Instance Type**: Size of the instance – t2.medium or t2.xlarge or t2.2xlarge.
- **PPS Config Data**: Provisioning parameters in an XML format. For details, see Pulse Policy Secure Provisioning Parameters.
- **SSH Key Name**: This key is used to access PPS 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 http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ec2-key-pairs.html.

4. Click Next. Review the specified details and click **Create Stack stack**. Observe the deployed PPS in a few minutes.

Figure 10: New VPC

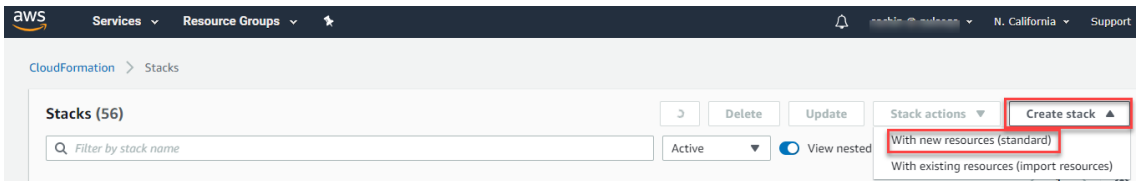


Deployment on VM with Two NIC Cards

To deploy Pulse Policy Secure on AWS using the AWS portal, do the following:

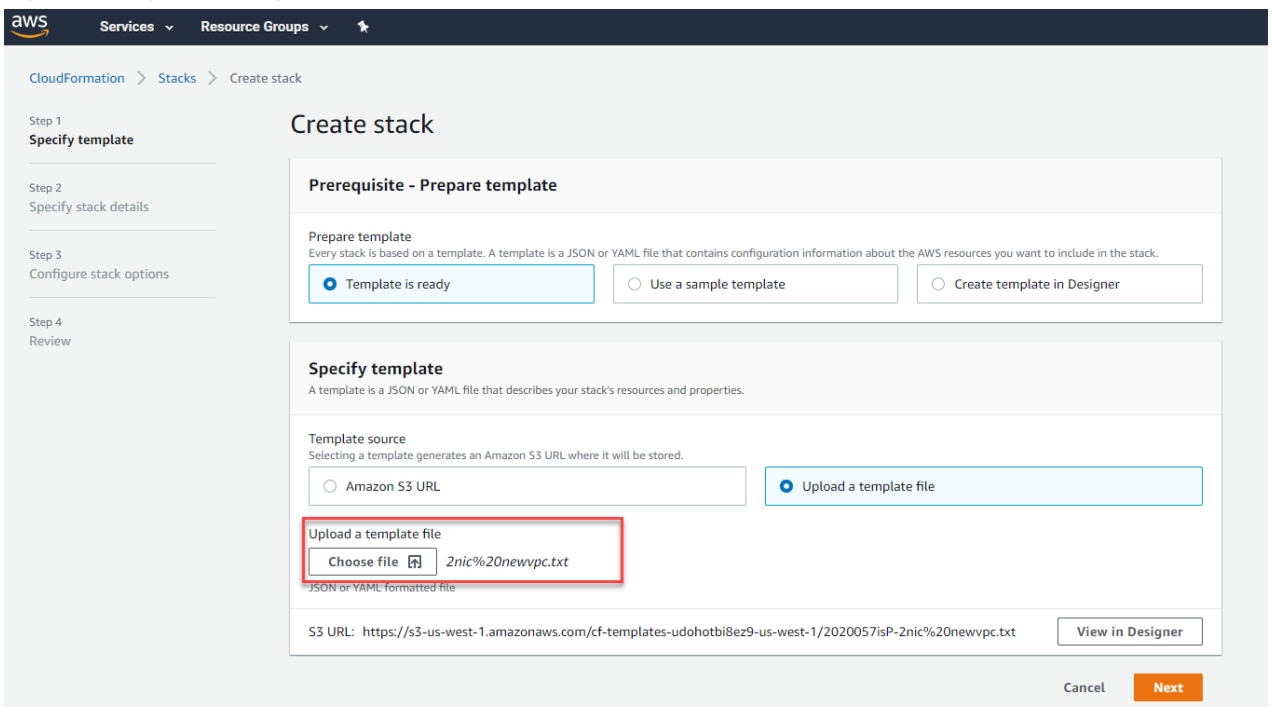
1. Select **AWS Services > CloudFormation** and click **Create stack** > **With new resources (standard)**.

Figure 11: Create New Stack



2. Select **Upload a template file**. Click **Choose file** and select "pulsesecure-pps-2-nics-new-network.json" template file for the new VPC. Click **Next**.

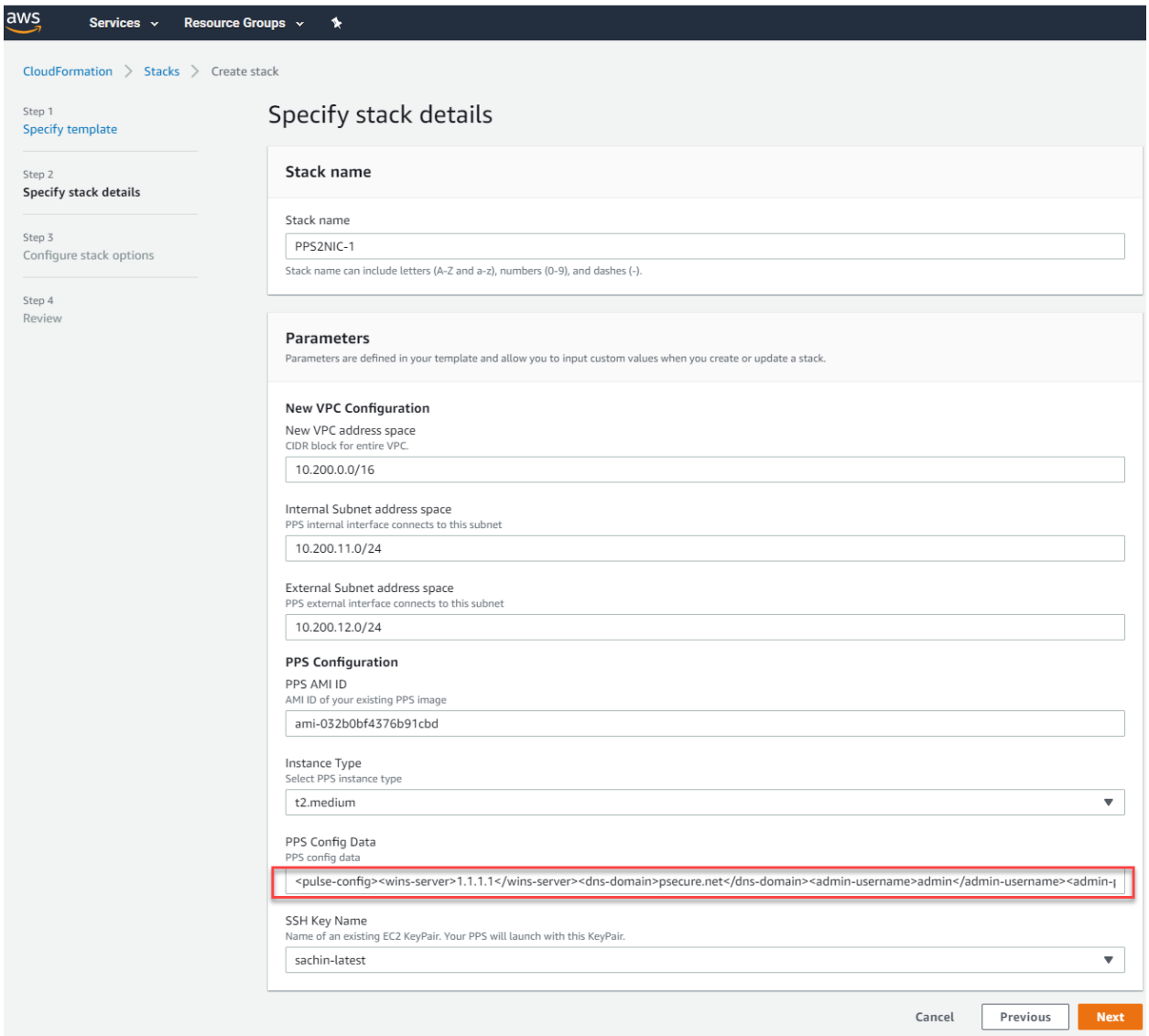
Figure 12: Upload Template



3. In the Specify Details page, fill or modify the following parameters.

Note: Before proceeding with deployment, ensure that the attribute "**accept-license-agreement"** in PPSConfigData is set to "**y**".

Figure 13: Specify Details for New Virtual Private Cloud

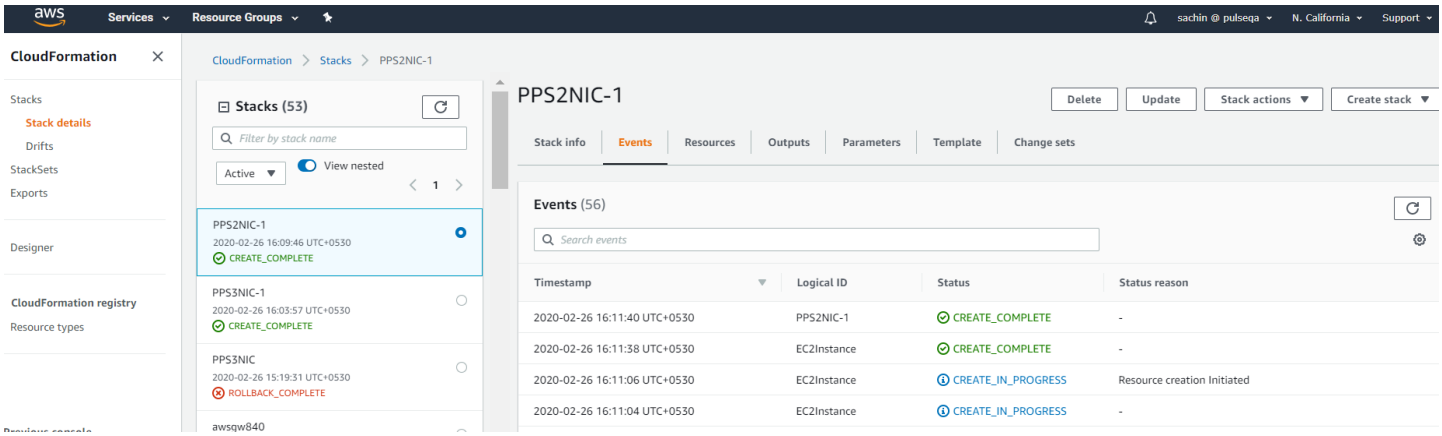


- **Stack name**: Specify the stack name in which Pulse Policy Secure needs to be deployed
- **New VPC address space**: Virtual private cloud address space
- **Internal Subnet address space**: Subnet from which Pulse Policy Secure internal interface needs to lease IP
- **External Subnet address space**: Subnet from which Pulse Policy Secure external interface needs to lease IP
- **PPS AMI ID**: ID of the uploaded AMI file
- **Instance Type**: Size of the instance – t2.medium or t2.large
- **PPS Config Data**: Provisioning parameters in an XML format. For details, see Pulse Policy Secure Provisioning Parameters.

- **SSH Key Name**: This key is used to access PPS 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 http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ec2-key-pairs.html.

4. Review the specified details and click **Create stack**. Observe the deployed PPS in a few minutes.

Figure 14: New VPC



Deploying PPS on an Existing Virtual Private Cloud

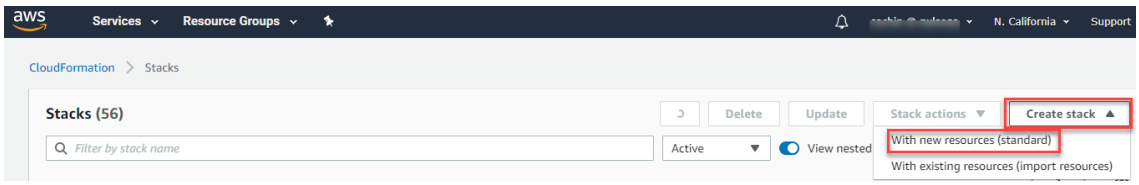
This section describes PPS deployment with three NIC cards and two NIC cards.

Deployment on VM with Three NIC Cards

To deploy Pulse Policy Secure on AWS using the AWS portal, do the following:

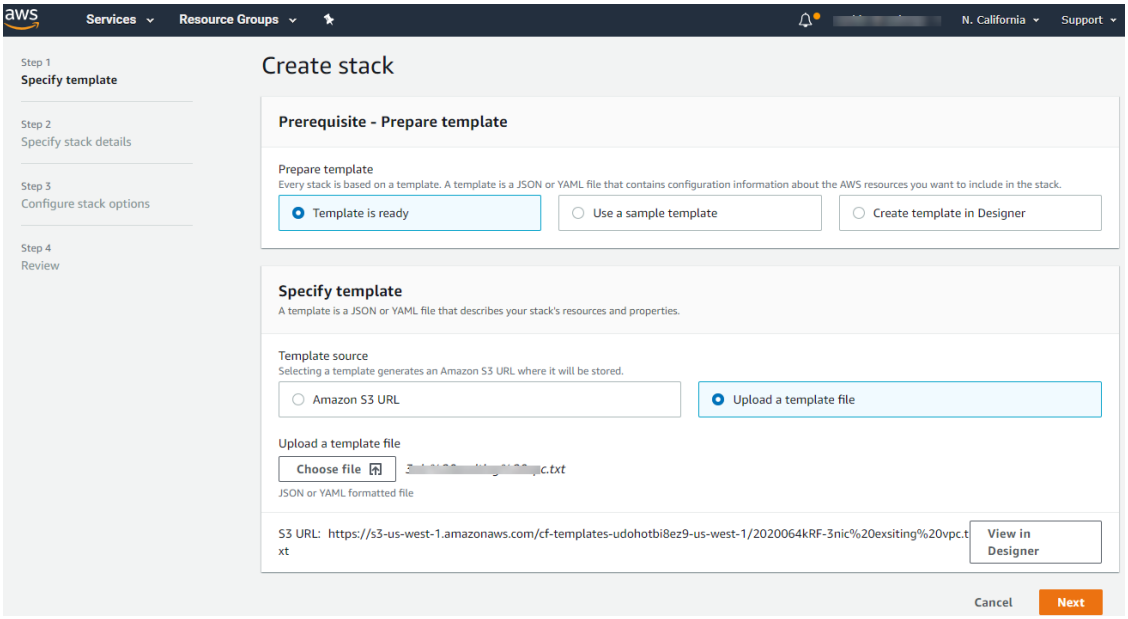
1. Select **AWS Services > CloudFormation** and click **Create stack** > **With new resources (standard)**.

Figure 15: Create New Stack



2. Select **Upload a template file.** Click **Choose file** and select "pulsesecure-pps-3-nics-existing-vpc.json" template file for existing VPC. Click **Next**.

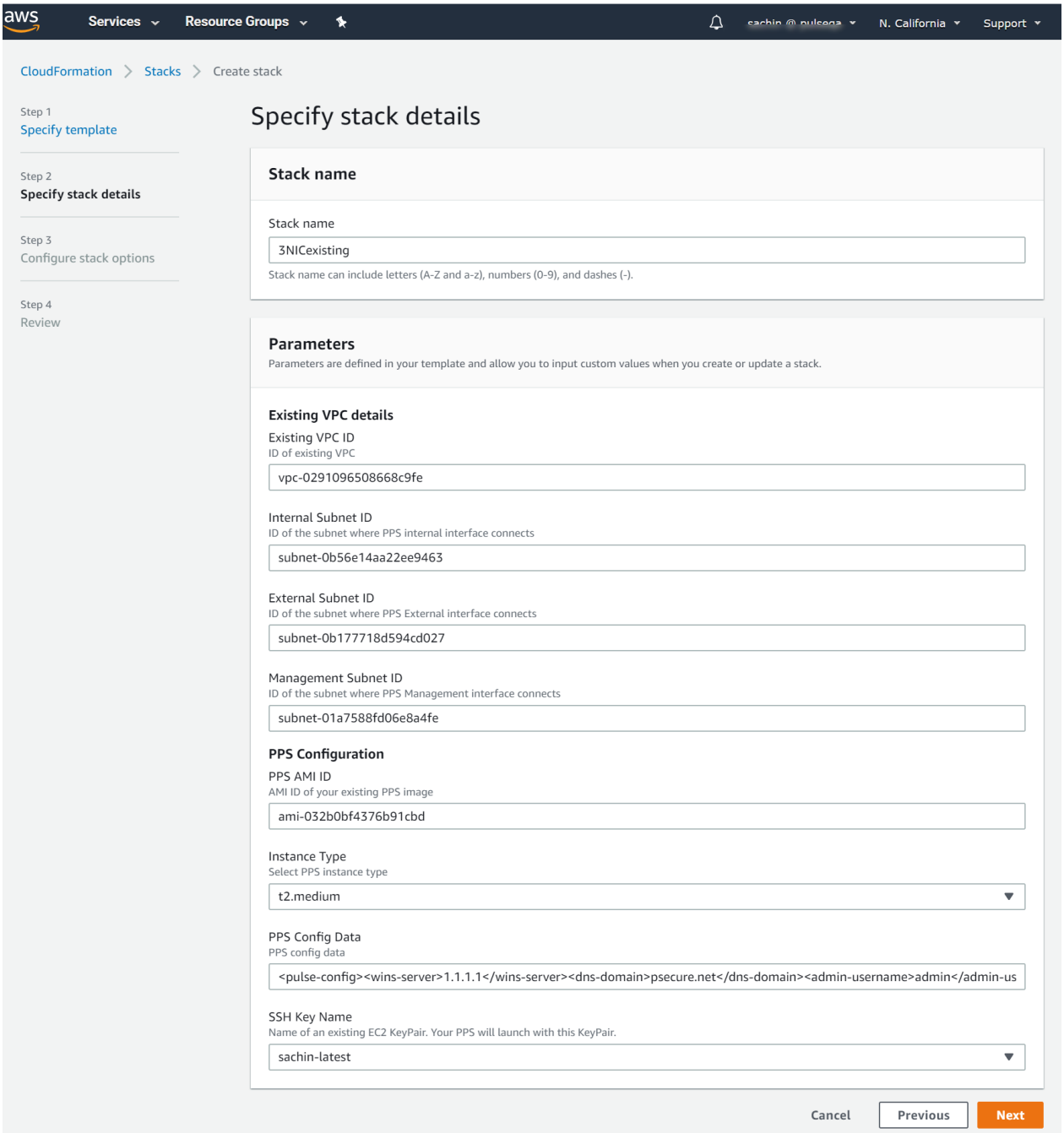
Figure 16: Upload Template



3. In the Specify Stack Details page, fill or modify the following parameters.

Note: Before proceeding with deployment, ensure that the attribute "**accept-license-agreement"** in PPSConfigData is set to "**y**".

Figure 17: Specify Details for Existing Virtual Private Cloud



- **Stack name**: Specify the stack name in which Pulse Policy Secure needs to be deployed
- **Existing VPC ID**: Virtual private cloud ID
- **Internal Subnet ID**: Subnet from which Pulse Policy Secure internal interface needs to lease IP
- **External Subnet ID**: Subnet from which Pulse Policy Secure external interface needs to lease IP
- **Management Subnet ID**: Subnet from which Pulse Policy Secure management interface needs to lease IP
- **PPS AMI ID**: ID of the uploaded AMI file
- **Instance Type**: Size of the instance – t2.medium or t2.large
- **PPS Config Data**: Provisioning parameters in an XML format. For details, see Pulse Policy Secure Provisioning Parameters.
- **SSH Key Name**: This key is used to access PPS 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 http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ec2-key-pairs.html

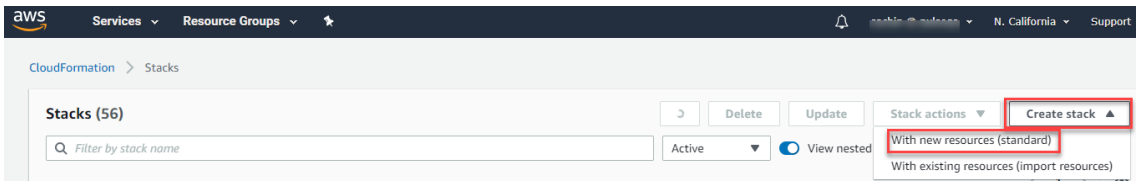
4. Review the specified details and click **Create stack**. Observe the deployed PPS in a few minutes.

Deployment on VM with Two NIC Cards

To deploy Pulse Policy Secure on AWS using the AWS portal, do the following:

1. Select **AWS Services > CloudFormation** and click **Create new stack.**

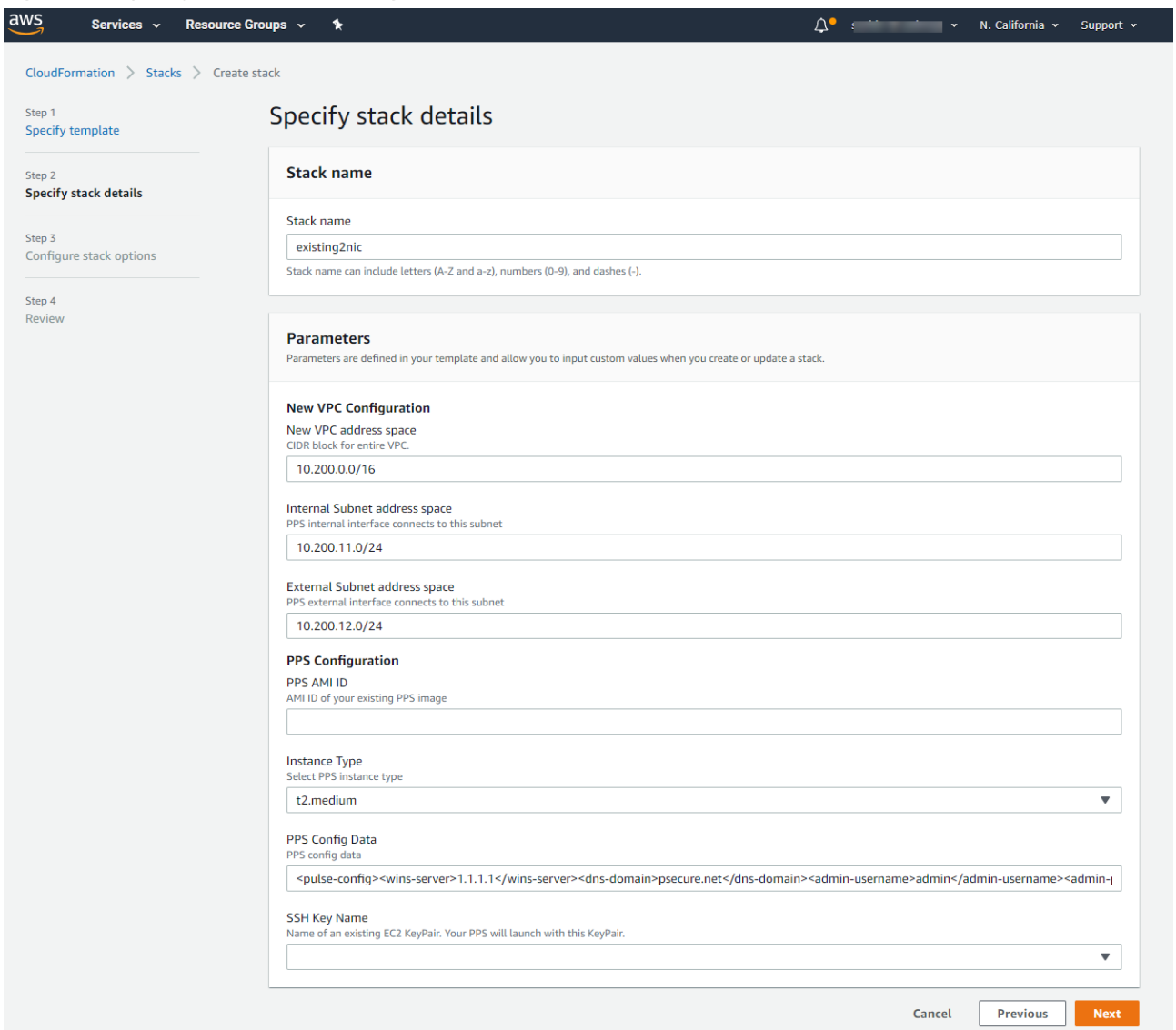
Figure 18: Create New Stack



2. Select **Upload a template to Amazon S3**. Click **Browse** and select "pulsesecure-PPS-2-nics-existing-vpc.json" template file for existing VPC. Click **Next**.
3. In the Specify Stack Details page, fill or modify the following parameters.

Note: Before proceeding with deployment, ensure that the attribute "**accept-license-agreement"** in PPSConfigData is set to "**y**".

Figure 19: Specify Details for Existing Virtual Private Cloud



- **Stack name**: Specify the stack name in which Pulse Policy Secure needs to be deployed
- **Existing VPC ID**: Virtual private cloud ID
- **Internal Subnet ID**: Subnet from which Pulse Policy Secure internal interface needs to lease IP
- **External Subnet ID**: Subnet from which Pulse Policy Secure external interface needs to lease IP
- **PPS AMI ID**: ID of the uploaded AMI file
- **Instance Type**: Size of the instance – t2.medium or t2.large
- **PPS Config Data**: Provisioning parameters in an XML format. For details, see Pulse Policy Secure Provisioning Parameters.
- **SSH Key Name**: This key is used to access PPS 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 http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ec2-key-pairs.html

4. Review the specified details and click **Create stack**. Observe the deployed PPS in a few minutes.

Pulse Policy Secure Provisioning Parameters

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

```
<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>
  <enable-license-server><value></enable-license-server>
  <accept-license-agreement><value></accept-license-agreement >
  <enable-rest><value></enable-rest>
</pulse-config>
```

The below table depicts the details of the xml file.

#	Parameter Name	Type	Description
1	primary-dns	IP address	Primary DNS for Pulse Policy Secure
2	secondary-dns	IP address	Secondary DNS for Pulse Policy Secure
3	wins-server	IP address	Wins server for Pulse Policy Secure
4	dns-domain	string	DNS domain of Pulse Policy Secure
5	admin-username	string	admin UI user name
6	admin-password	string	admin UI password
7	cert-common-name	string	Common name for the self-signed certificate generation. This certificate is used as the device certificate of Pulse Policy Secure Random text for the self-certificate generation Organization name for the self-signed certificate generation
8	cert-random-text	string	
9	cert-organization	string	
10	config-download-url	String URL	Http based URL where XML based Pulse Policy

			Secure configuration can be found. During provisioning, Pulse Policy Secure fetches this file and comes up with preloaded configuration. XML based configuration can be present in another VM in AWS cloud or at corporate network which is accessible for Pulse Policy Secure through site to site VPN between AWS and corporate data center
11	config-data	string	base64 encoded XML based Pulse Policy Secure configuration
12	auth-code-license	string	Authentication code that needs to be obtained from Pulse Secure
13	enable-license-server	string	If set to 'y', PPS will be deployed as a License server. If set to 'n', PPS will be deployed as a normal server.
14	accept-license-agreement	string	This value is passed to the instance for configuration at the boot time. By default, this value is set to "n". **This value must be set to "y".**
15	enable-rest	string	If set to 'y', REST API access for the administrator user is enabled.

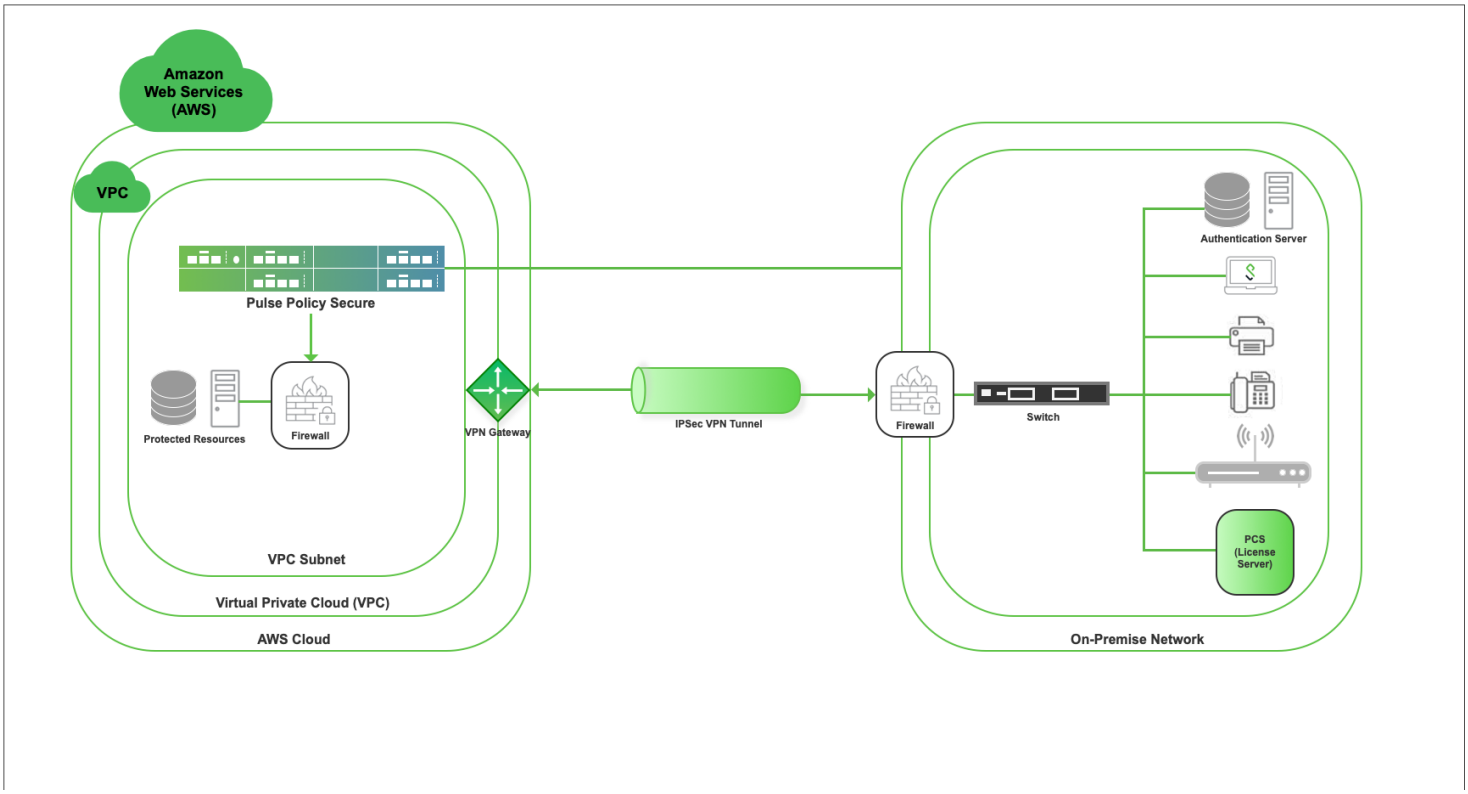
Note: In the above list of parameters, **primary dns, dns domain, admin username, admin password, cert-random name, cert-random text, cert-organization** and **accept-license-agreement** are mandatory parameters. The other parameters are optional parameters.

Configuring Licenses on the Pulse Policy Secure Appliance

In this release, evaluation licenses are provided. To add more licenses, the Pulse Policy Secure administrator needs to leverage the Pulse License server.
The Pulse License server can be made available in the corporate network

Pulse License Server in Corporate Network

Figure 20: Pulse License Server in a Corporate Network



Pulse License Server in Cloud Network

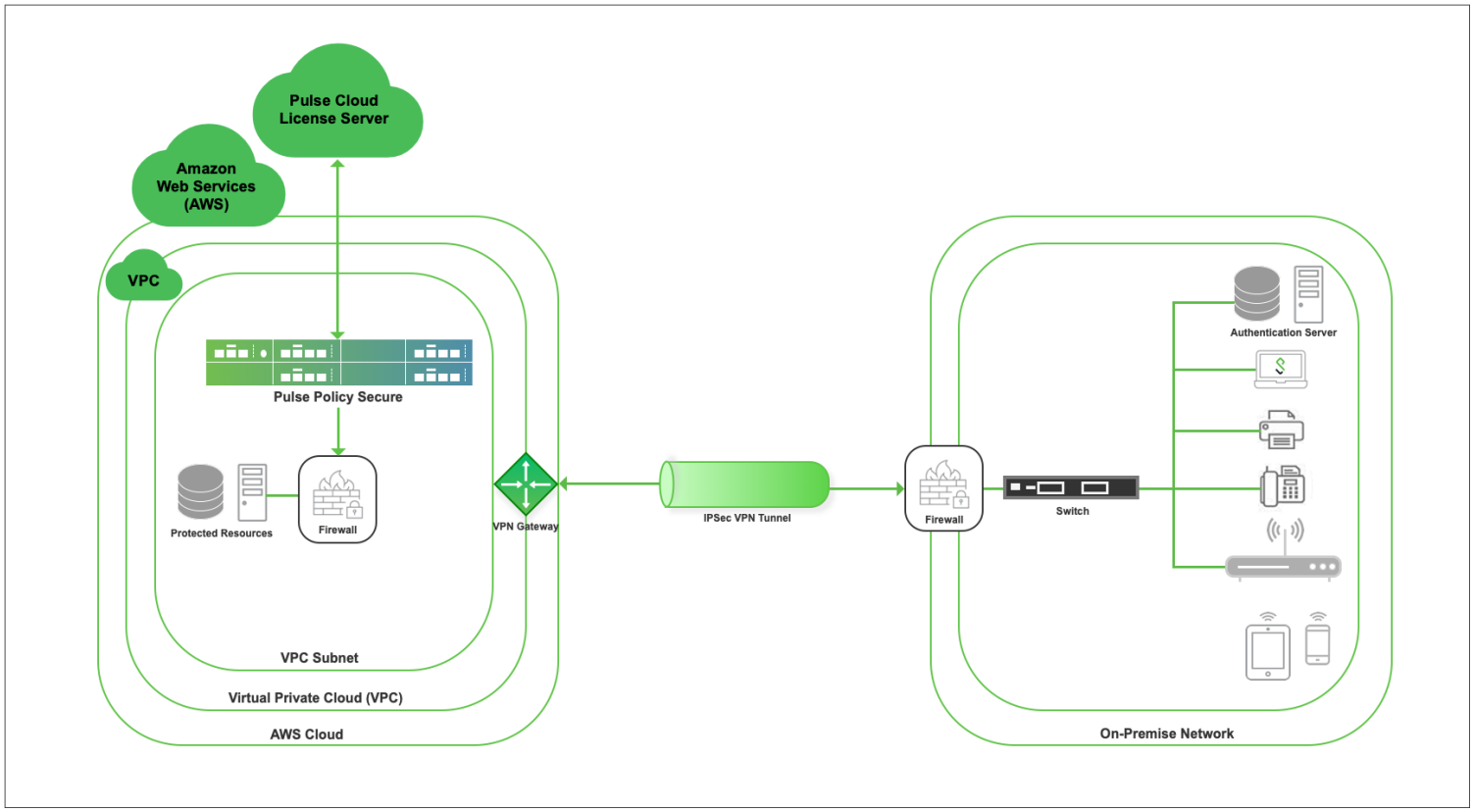
Pulse Policy Secure virtual machines (VM) are enabled to provision licenses through the Pulse Cloud Licensing Service (PCLS). For this, administrator needs to obtain an Authentication code from Pulse Secure Support and apply it in Download Licenses page of PPS admin console. The PPS also periodically sends heartbeat messages to PCLS for auditing purposes.

```
"<pulse-config><primary-dns>8.8.8.8</primary-dns><secondary-dns>8.8.8.9</secondary-dns><wins-server>1.1.1.1</wins-server><dns-domain>psecure.net</dns-d
omain><admin-username>admin</admin-username><admin-password>password</admin-password><cert-common-name>va1.psecure.net</cert-common-name><cert-random-t
ext>fdsfpisonvsfnms</cert-random-text><cert-organisation>Psecure
Org</cert-organisation><config-download-url></config-download-url><config-data></config-data><auth-code-license></auth-code-license><enable-license-ser
ver>n</enable-license-server><accept-license-agreement>n</accept-license-agreement></pulse-config>"
```

The Authentication code can also be specified in the CloudFormation template. When PPS comes up, it automatically fetches the Authentication code.

- Adding Authentication Code in PPS Admin Console
- Including Authentication Code in CloudFormation Template

Figure 21: Pulse License Server in Cloud Network



Adding Authentication Code in PPS 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**.

Including Authentication Code in CloudFormation Template

To include Authentication code in the CloudFormation template:

1. In the CloudFormation template, go to the PPSConfig 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 Policy Secure Virtual Appliance

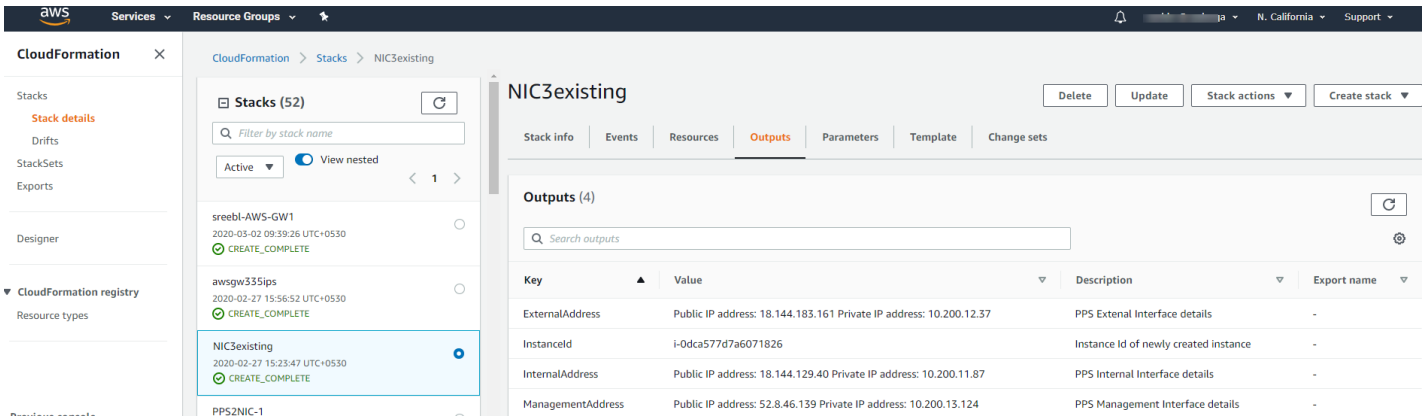
The Pulse Policy Secure virtual appliance can be accessed:

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

Accessing the Pulse Policy Secure Virtual Appliance as an Administrator

In the AWS portal, navigate to CloudFormation section. Select the stack where PPS is deployed and then click on the 'Outputs' tab. Note down the PPS management, internal and external address from the table as shown in Figure 17.

Figure 22: Accessing PPS Virtual Appliance



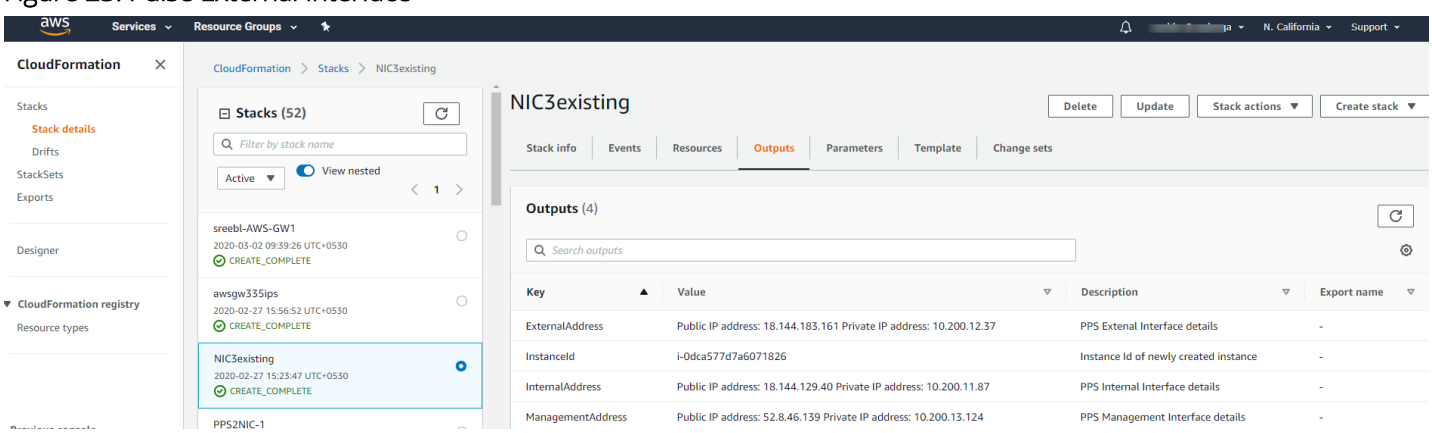
Use the credentials provided in the provisioning parameters to log in as the administrator https://<PPS-IP>/admin. The default PPS admin UI user configured in the CloudFormation config file is: user 'admin' and password 'password1234'.

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

Accessing the Pulse Policy Secure Virtual Appliance as an End User

After successfully deploying PPS on AWS, go to the Outputs section and copy the Pulse External Interface details.

Figure 23: Pulse External Interface



Accessing the Pulse Policy Secure Virtual Appliance using SSH Console

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

On Linux and Mac OSX

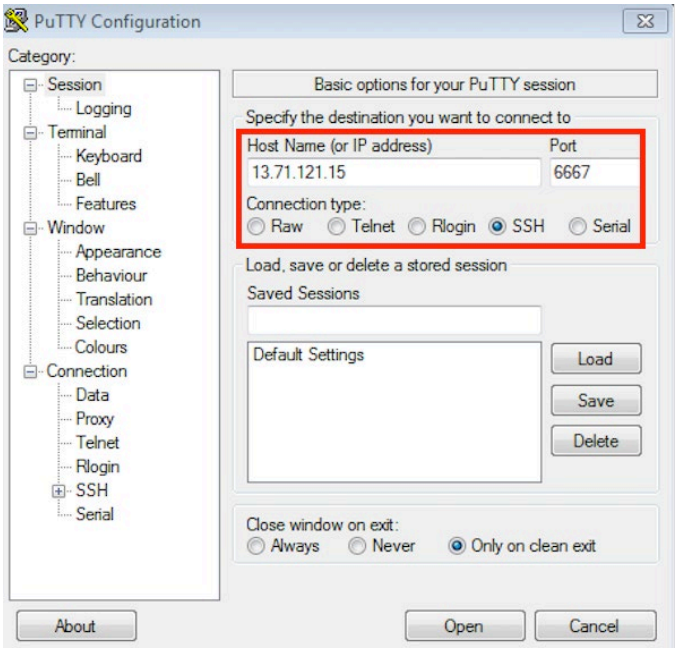
Execute the following command:

```
ssh -i <rsa-public-key-file> <PPS-Management-Interface-PublicIP> -p 6667
```

On Windows

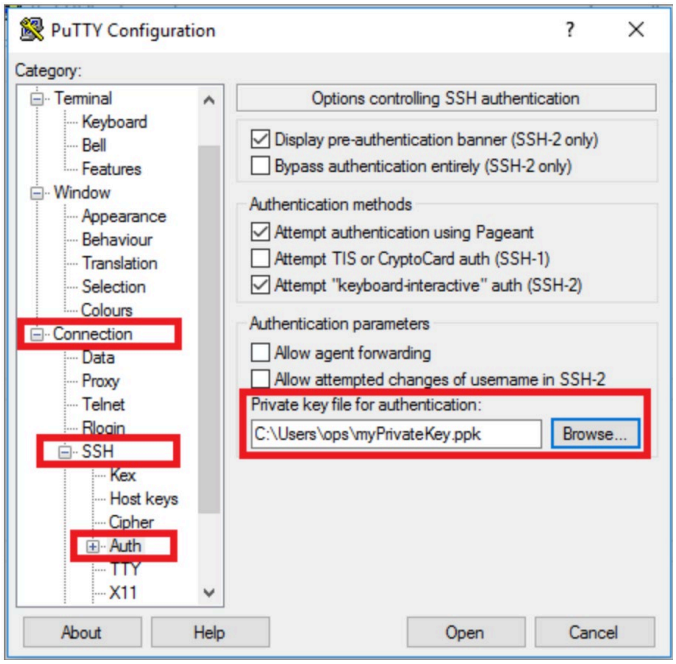
1. Launch the Putty terminal emulator.
2. In the Session category:
   - Enter the host name or IP address.
   - Enter the port number.
   - Select the connection type as SSH.

Figure 24: Putty Configuration – Basic Options



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

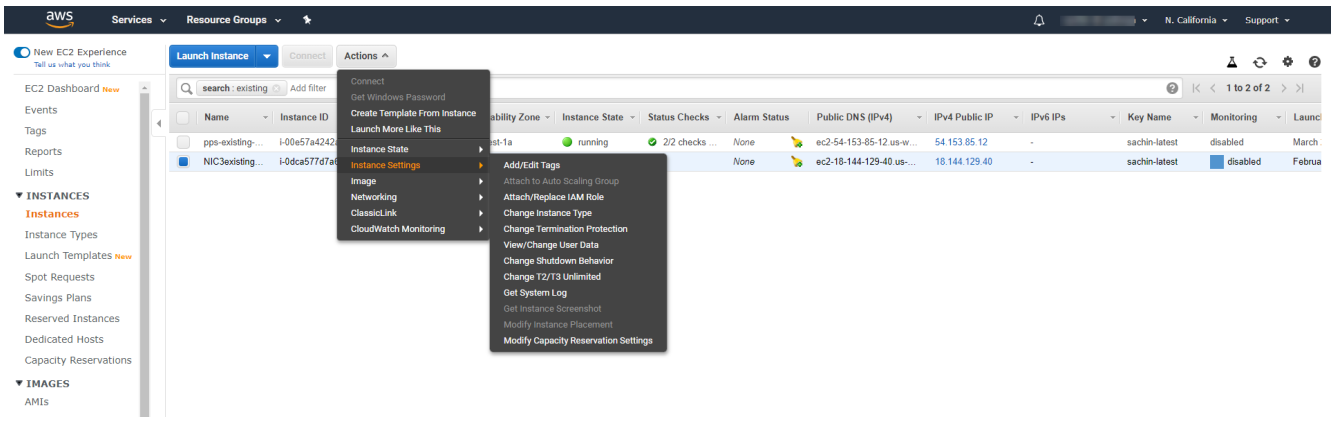
Figure 25: Putty Configuration – SSH Authentication



System Operations

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

Figure 26: System Operations



On the AWS portal, select **AWS Services > Launch Instance**. From the **Actions** menu, select **Instance State**.

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

Network Configuration

IP Address Assignment for Internal, External and Management Interfaces

Each interface in AWS can have private and public IP addresses. Sample CloudFormation Templates provided by Pulse Policy Secure creates the Pulse Policy 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 AWS can be seen at: https://docs.aws.amazon.com/AmazonVPC/latest/UserGuide/vpc-ip-addressing.html

IP Addressing Modes

When Pulse Policy Secure gets deployed by using the sample templates provided by Pulse Secure, Pulse Policy Secure comes up with multiple interfaces. If you take an example of a template “pulsesecure-PPS-3-nics.zip” provided by Pulse Secure, you notice the following things.

PPS external interface and PPS management interface have both Elastic and Private IP addresses.

Modifying Network Parameters After Deployment

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

Controlling the Selection of Internal, External and Management Interfaces

Sample CloudFormation template, provided by Pulse Secure, requests AWS fabric to create three Network Interfaces. While running this template, AWS fabric creates interfaces named eth0, eth1 and eth2 and attaches them to PPS 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.

Interface Name	PPS Interface
eth0	internal interface
eth1	external interface
eth2	management interface

Then, question is how you can control the order of network interfaces named eth0, eth1 and eth2 created through CloudFormation template?
The Pulse Policy 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.

```
"EC2Instance": {
 "Type": "AWS::EC2::Instance",
 "Properties": {
  "ImageId": {"Ref": "PPSImageAMIId"},
  "KeyName": {"Ref": "KeyName"},
  "InstanceType": {"Ref": "InstanceType"},
  "NetworkInterfaces": [
   {"NetworkInterfaceId": {"Ref": "Eth0"}, "DeviceIndex": "0"},
   {"NetworkInterfaceId": {"Ref": "Eth1"}, "DeviceIndex": "1"},
   {"NetworkInterfaceId": {"Ref": "Eth2"}, "DeviceIndex": "2"}
  ],
  "Tags"     : [
   {"Key": "Name",
    "Value": {"Fn::Join": [ "-", [ { "Ref": "AWS::StackName" }, "PPSvAWS" ] ] }
   }
   ],
   "UserData": {"Fn::Base64": {"Fn::Join": ["", [{"Ref": "PPSConfigData"}]]}}
 }
},
```

PPS 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:

Interface Name	PPS Interface	Network ID
eth0	internal interface	nic1
eth1	external interface	nic2
eth2	management interface	nic3

Backing up Configs and Archived Logs on S3 Bucket

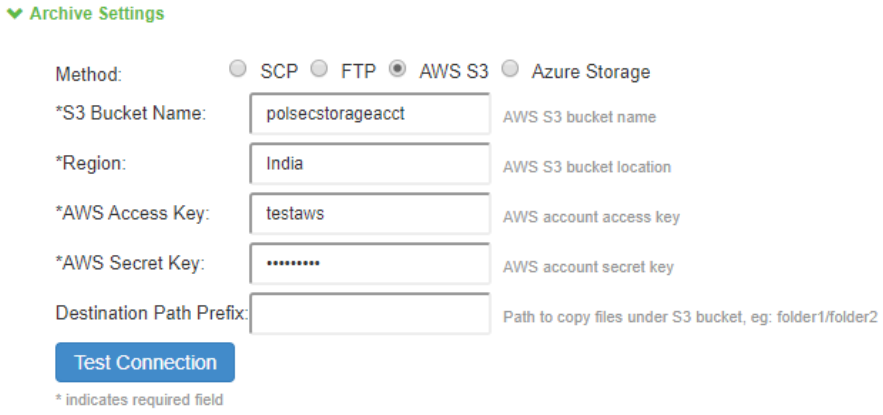
Pulse Policy Secure supports pushing configs and archived logs to the servers that support SCP and FTP protocols. In the AWS deployment, Pulse Policy Secure now supports pushing configs and archived logs to the S3 bucket.

Configuring Backup Configs and Archived Logs via PPS Admin Console

To configure backing up configs and archived logs:

1. Log into the Pulse Policy Secure admin console.
2. Navigate to **Maintenance > Archiving > Archiving Servers**.
3. In the Archive Settings section, select the **AWS** option and configure S3 Bucket Name, AWS Access Key, AWS Secret Key, S3 Bucket Location and Destination Path Prefix.

Figure 27: AWS Archive Settings



Parameter	Description
S3 Bucket Name	To create an S3 bucket: 1. Sign in to the AWS Management Console and open the Amazon S3 console at https://console.aws.amazon.com/s3/. 2. Select **Create bucket**. 3. In the **Bucket name** field, type a unique DNS-compliant name for your new bucket. For more details about S3 bucket name, refer https://docs.aws.amazon.com/AmazonS3/latest/gsg/CreatingABucket.html
Region	S3 bucket location.
AWS Access Key	To create AWS Access Key and AWS Secret Key: 1. Sign in to the AWS Management Console and open the IAM console at https://console.aws.amazon.com/iam/. 2. In the navigation bar on the upper right, select your user name, and then select **My Security Credentials**. 3. On the **AWS IAM Credentials** tab, in the **Access keys for CLI, SDK, and API access** section, select **Create access key**. 4. Then select **Download .csv file** to save the access key ID and secret access key to a .csv file on your computer. When you create an access key, the key pair (access key ID and secret access key) is active by default, and you can use the pair right away. For more details, refer https://aws.amazon.com/premiumsupport/knowledge-center/create-

	access-key/
AWS Secret Key	See the procedure described for AWS Access Key. For more details, refer https://help.bittitan.com/hc/en-us/articles/115008255268-How-do-I-find-my-AWS-Access-Key-and-Secret-Access-Key-
Dest Path Prefix (Optional)	Path to copy files under S3 bucket.

Configuring Backup Configs and Archived Logs via REST

Setting AWS as Archive Logs Backup

REQUEST

```
PUT /api/v1/configuration/system/maintenance/archiving/settings HTTP/1.1
Content-Type: application/json
(
    "archive-path": "folder1/folder2",
    "directory": "ap-south-1",
    "method": "AWS",
    "Password-cleartext": "xkjdsklukjkwej",
    "server": "S3-server-storage-bucket",
    "user-name": "ADDDDDFVFFFQXXXXXA"
)
```

Mapping of keys in POST body:

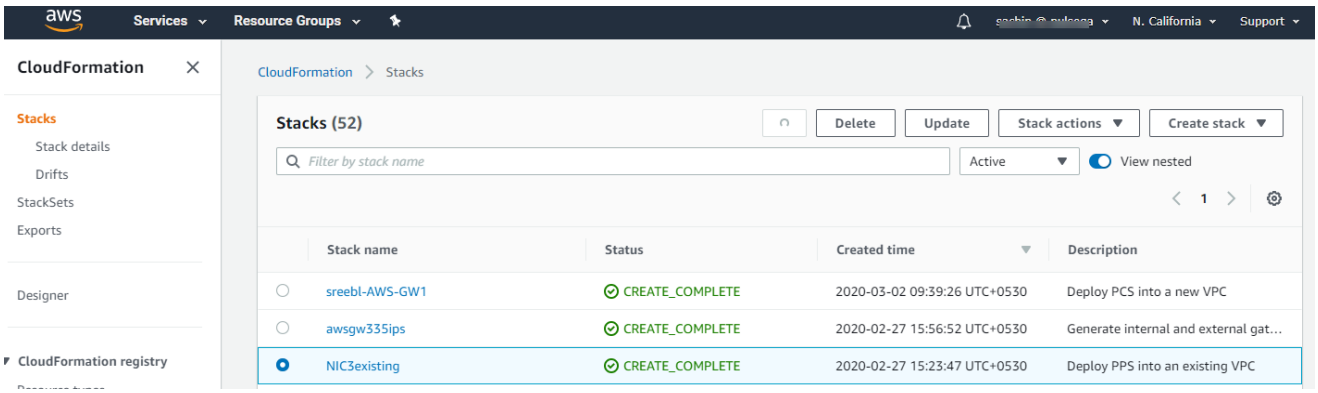
archive-path	Destination path prefix
directory	Region
method	method (AWS)
Password-cleartext	AWS Secret key
server	S3 Bucket Name
user-name	AWS Access key

Decommissioning Pulse Policy Secure

To decommission Pulse Policy Secure, perform the following steps:

1. Select **AWS Services > CloudFormation.**
2. Click **Actions**. From the drop-down list displayed, select **Delete Stack**.

Figure 28: Delete Stack



Pricing

The cost of running this product is combination of License cost and AWS infrastructure cost. It will be very difficult to find out AWS infrastructure cost for this product as it may vary with Regions/Country/Time. Hence, we recommend using "AWS Calculator" which is available online to calculate the cost of running this product. https://calculator.s3.amazonaws.com/index.html

Here are resources that are created during deployment. Highlighted ones are chargeable in AWS.

Resources	Category	Chargeable
PPS VM (t2.medium / t2.xlarge / t2.2xlarge)	Compute	Yes
Virtual Private Cloud with four subnets	Networking	No
Three NICs named PPSInternalNIC, PPSExternalNIC and PPSManagementNIC	Networking	No
Three Elasti Public IPs for internal, external and management interfaces	Networking	Yes
Three Security Groups named SGInternal, SGExternal and SGManagement	Networking	No
Route table	Networking	No
PPS IMG file of size 40GB in S3 bucket	Storage	Yes
PPS Snapshot file of size 40GB in Elastic block store	Storage	Yes

Limitations

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

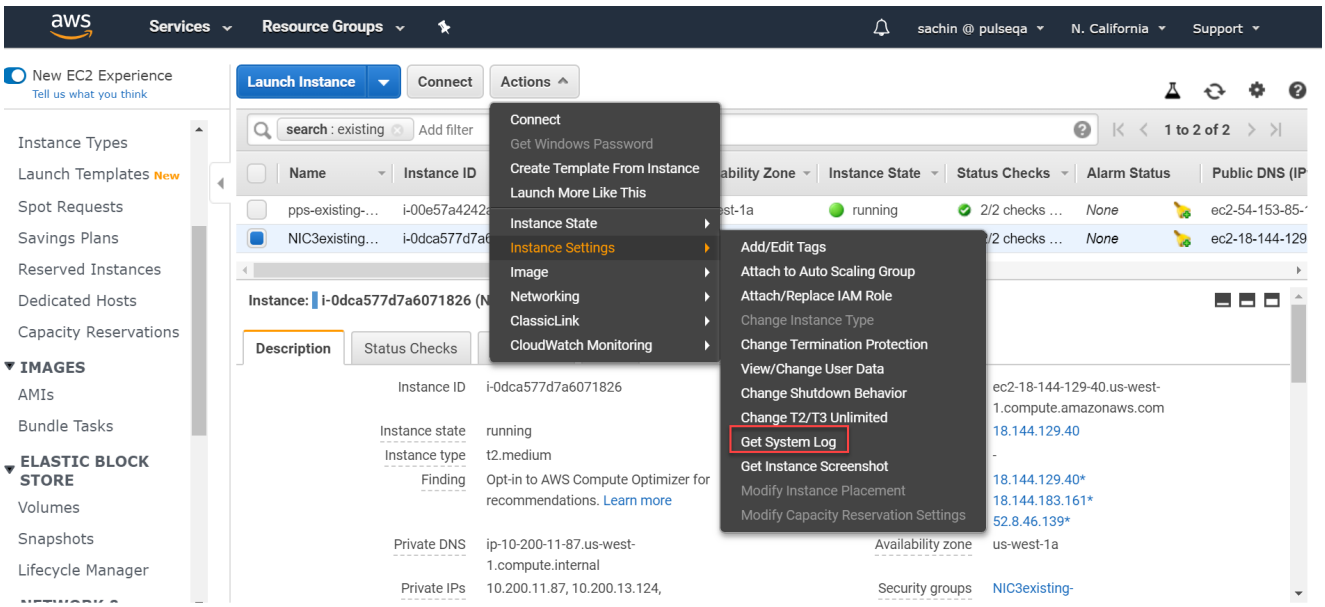
- IP address (private) of the interfaces should not be changed
- IPV6 is not supported

Troubleshooting

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

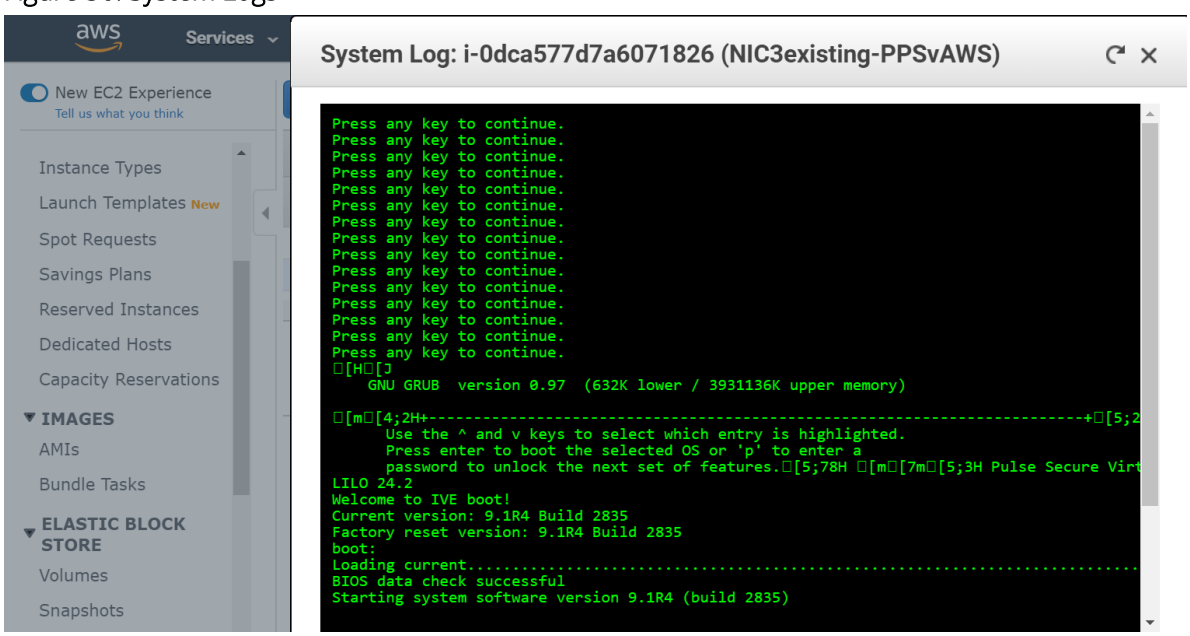
1. Select AWS Services > Instances > Launch Instance.
2. From the list displayed, select Instance Settings > Get System Log.

Figure 29: Boot Diagnostics



The system logs window is displayed.

Figure 30: System Logs

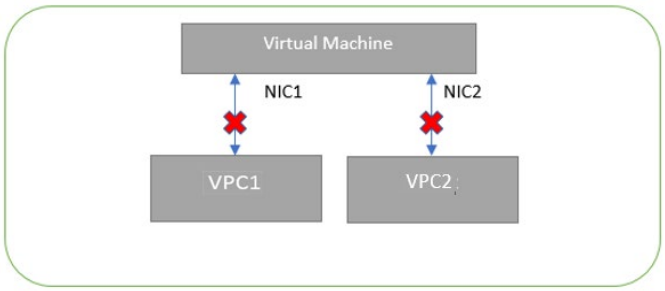


Frequently Asked Questions

Appendix A: Security Group (SG)

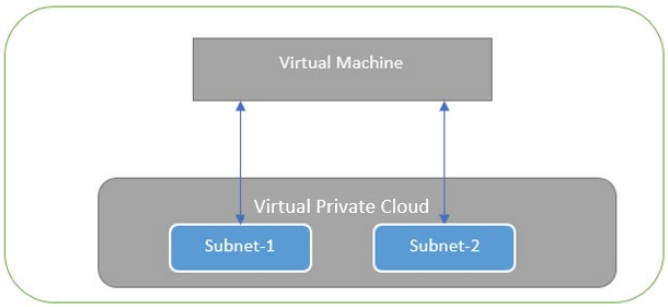
AWS has a limitation where virtual machine with multiple network interfaces cannot connect to different Virtual Private Cloud (VPCs). For example, a VM with two NICs, NIC1 and NIC2, will not be able to connect to VPC1 and VPC2 respectively.

Figure 31: Virtual Machine with two NICs Connecting to VPC1 and VPC2



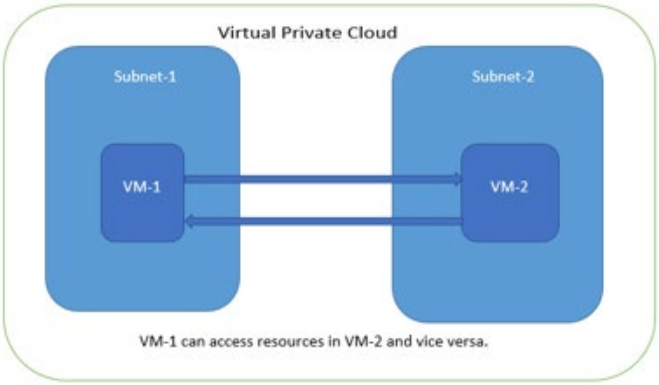
AWS supports a virtual machine with multiple NICs to connect to different Subnets under a same Virtual Private Cloud. For example, a VM with two NICs, NIC1 and NIC2, can connect to 'Subnet1' and 'Subnet2' where these subnets exist under a same Virtual Private Cloud respectively.

Figure 32: Virtual Machine with two NICs Connecting to Subnet1 and Subnet2



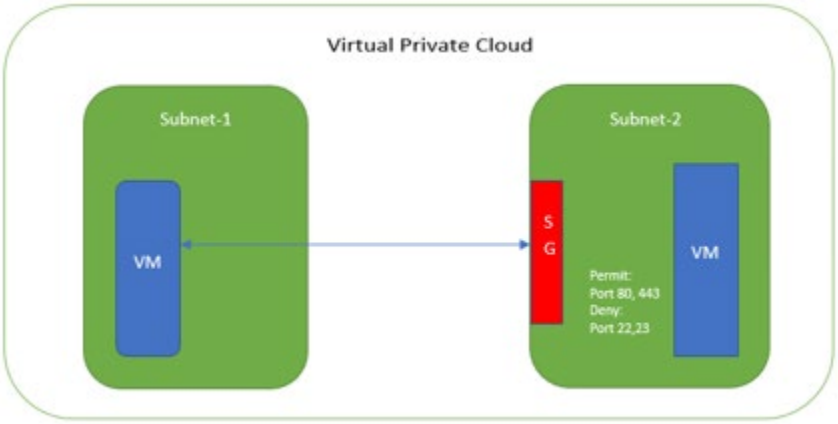
AWS provides isolation between different VPCs. But it does not provide the same kind of isolation when it comes to subnets in the same VPC. For example, consider a VPC has two subnets, Subnet1 and Subnet2. And consider two VMs, VM-1 and VM-2, which are connected to Subnet1 and Subnet2 respectively. In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

Figure 33: 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 "Security Group" provided by AWS.

Figure 34: Traffic Filtering by AWS Support Group



Pulse Policy Secure, when provisioned through the CloudFormation template provided by Pulse Secure, creates four subnets under a virtual private cloud named "PPSVirtualNetwork". The four Subnets are:

1. PPSInternalSubnet
2. PPSExternalSubnet
3. PPSManagementSubnet

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

1. SGExternalSubnet
2. SGInternalSubnet
3. SGManagementSubnet

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

1. The list of SG Inbound/Outbound rules created "Stack-PPSvExtSG" are:

Figure 35: Stack-PPSvExtSG - Inbound Rules

Inbound rules — Edit inbound rules

Type	Protocol	Port range	Source	Description - optional
HTTP	TCP	80	0.0.0.0/0	-
PostgreSQL	TCP	5432	0.0.0.0/0	-
Custom TCP	TCP	11122 - 11123	0.0.0.0/0	-
Custom TCP	TCP	49	0.0.0.0/0	-
Custom UDP	UDP	1812 - 1813	0.0.0.0/0	-
Custom TCP	TCP	601	0.0.0.0/0	-
Custom UDP	UDP	67	0.0.0.0/0	-
Custom UDP	UDP	162	0.0.0.0/0	-
Custom UDP	UDP	3799	0.0.0.0/0	-
HTTPS	TCP	443	0.0.0.0/0	-
All ICMP - IPv4	ICMP	All	0.0.0.0/0	-
Custom UDP	UDP	514	0.0.0.0/0	-

Figure 36: Stack-PPSvExtSG - Outbound Rules

Inbound rules | Outbound rules | Tags

Outbound rules — Edit outbound rules

Type	Protocol	Port range	Destination	Description - optional
All traffic	All	All	127.0.0.1/32	-

2. The list of SG Inbound/Outbound rules created "Stack-PPSvIntSG" are:

Figure 37: Stack-PPSvIntSG - Inbound Rules

Inbound rules Edit inbound rules

Type	Protocol	Port range	Source	Description - optional
HTTP	TCP	80	0.0.0.0/0	-
Custom TCP	TCP	6667	0.0.0.0/0	-
PostgreSQL	TCP	5432	0.0.0.0/0	-
Custom TCP	TCP	11122 - 11123	0.0.0.0/0	-
Custom TCP	TCP	49	0.0.0.0/0	-
Custom UDP	UDP	1812 - 1813	0.0.0.0/0	-
Custom TCP	TCP	601	0.0.0.0/0	-
Custom UDP	UDP	67	0.0.0.0/0	-
Custom UDP	UDP	162	0.0.0.0/0	-
Custom UDP	UDP	3799	0.0.0.0/0	-
HTTPS	TCP	443	0.0.0.0/0	-
All ICMP - IPv4	ICMP	All	0.0.0.0/0	-
Custom UDP	UDP	514	0.0.0.0/0	-

Figure 38: Stack-PPSvIntSG - Outbound Rules

Inbound rules | Outbound rules | Tags

Outbound rules Edit outbound rules

Type	Protocol	Port range	Destination	Description - optional
All traffic	All	All	0.0.0.0/0	-

3. The list of SG Inbound/Outbound rules created "Stack-PPSvMgmtSG" are:

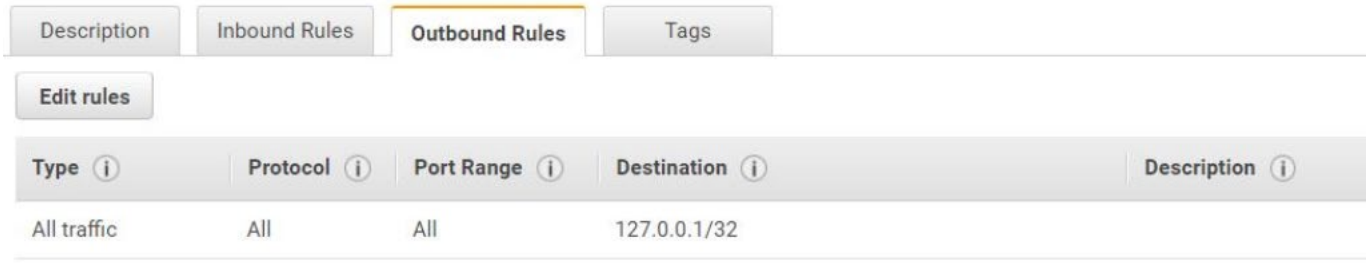
Figure 39: Stack-PPSvMgmtSG - Inbound Rules

Description | **Inbound Rules** | Outbound Rules | Tags

Edit rules

Type	Protocol	Port Range	Source	Description
HTTP	TCP	80	0.0.0.0/0	
Custom TCP Rule	TCP	830	0.0.0.0/0	
HTTPS	TCP	443	0.0.0.0/0	
All ICMP - IPv4	All	N/A	0.0.0.0/0	

Figure 40: Stack-PPSvMgmtSG - Outbound Rules



Appendix B: Pulse Policy Secure CloudFormation Template

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

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

The template has following sections:

Parameters	This section defines the parameters used for deploying PPS on AWS. It contains parameter name, its default value and the mouse-over help text that is displayed when mouse is placed over the parameter in AWS Web portal. The parameters defined here are displayed in the Custom Deployment page of AWS portal.
Resources	This section defines resource types that are deployed or updated in a resource group.
Outputs	This section defines the public IP address, private IP address and primary private IP address returned after successful deployment of PPS on AWS.

Parameters

Key Name: This is the name of the PPS Storage Account where the PPS VA image (.ami file) is stored.

```
"Parameters" : {

  "KeyName": {
    "Type": "AWS::EC2::KeyPair::KeyName",
    "Default": "",
    "AllowedPattern" : "[-_ a-zA-Z0-9]*",
    "Description": "Name of an existing EC2 KeyPair. Your PPS will launch with this KeyPair.",
    "ConstraintDescription": "Must be the name of an existing EC2 KeyPair."
  },
```

PPS Image AMI ID: This is the ID of the uploaded AMI file.

```
"PPSImageAMIId" : {
      "Type" : "String",
      "Description" : "AMI ID of your existing PPS image"
    },
```

Instance Type: This specifies the size of the instance – t2.medium or t2.large

```
"InstanceType": {
      "Description": "Select PPS instance type",
      "Type": "String",
      "Default": "t2.medium",
      "AllowedValues": [
        "t2.medium",
        "t2.xlarge",
        "t2.2xlarge"
```

```
    ],
    "ConstraintDescription": "Must be an allowed EC2 instance type."
  },
```

PPS Config Data: 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 AWS cloud or in the corporate network which is accessible for Pulse Policy Secure through site-to-site VPN between AWS and the corporate data center.
Pulse Policy Secure accepts the following parameters as provisioning parameters:

- primary-dns
- secondary-dns
- wins-server
- dns-domain
- admin-username
- admin-password
- cert-common-name
- cert-random-text
- cert-organization
- config-download-url
- config-data
- auth-code-license
- enable-license-server
- accept-license-agreement
- enable-rest

For details about these parameters, see Pulse Policy Secure Provisioning Parameters.

```
"PPSConfigData" : {
      "Type" : "String",
      "Description" : "PPS config data",
      "Default" : "<pulse-config><wins-server>1.1.1.1</wins-server><dns-domain>psecure.net</dns-domain><admin-
username>admin</admin-username><admin-password>password1234</admin-password><cert-common-
name>va1.psecure.net</cert-common-name><cert-random-text>fdsfpisonvsfnms</cert-random-text><cert-
organisation>Psecure Org</cert-organisation><config-download-url></config-download-url><config-data></config-
data><auth-code-license></auth-code-license><enable-license-server>n</enable-license-server><accept-license-
agreement>y</accept-license-agreement></pulse-config>"
    },
```

VPC CIDR: It is a valid CIDR range of the form x.x.x.x/x for entire VPC.

```
"VPCCIDR": {
      "Description": "CIDR block for entire VPC.",
      "Type": "String",
      "Default": "10.200.0.0/16",
      "AllowedPattern": "^(([0-9]|[1-9][0-9]|1[0-9]{2}|2[0-4][0-9]|25[0-5])\\.){3}([0-9]|[1-9][0-9]|1[0-9]{2}|2[0-4][0-9]|25[0-
5])(\\V([0-9]|[1-2][0-9]|3[0-2]))$",
      "ConstraintDescription": "Must be a valid CIDR range of the form x.x.x.x/x."
    },
```

Internal Subnet CIDR: Subnet from which Pulse Policy Secure Internal Interface needs to lease IP.

```
"InternalSubnetCIDR": {
```

```
        "Description": "PPS internal interface connects to this subnet",
        "Type": "String",
        "Default": "10.200.11.0/24",
        "AllowedPattern": "^(([0-9]|[1-9][0-9]|1[0-9]{2}|2[0-4][0-9]|25[0-5])\\.){3}([0-9]|[1-9][0-9]|1[0-9]{2}|2[0-4][0-9]|25[0-
5])(\\V([0-9]|[1-2][0-9]|3[0-2]))$",
        "ConstraintDescription": "CIDR block parameter must be in the form x.x.x.x/x"
      },
```

External Subnet CIDR: Subnet from which Pulse Policy Secure External Interface needs to lease IP.

```
"ExternalSubnetCIDR": {
        "Description": "PPS external interface connects to this subnet",
        "Type": "String",
        "Default": "10.200.12.0/24",
        "AllowedPattern": "^(([0-9]|[1-9][0-9]|1[0-9]{2}|2[0-4][0-9]|25[0-5])\\.){3}([0-9]|[1-9][0-9]|1[0-9]{2}|2[0-4][0-9]|25[0-
5])(\\V([0-9]|[1-2][0-9]|3[0-2]))$",
        "ConstraintDescription": "CIDR block parameter must be in the form x.x.x.x/x"
      },
```

Management Subnet CIDR: Subnet from which Pulse Policy Secure Management Interface needs to lease IP.

```
"ManagementSubnetCIDR": {
        "Description": "PPS management interface connects to this subnet",
        "Type": "String",
        "Default": "10.200.13.0/24",
        "AllowedPattern": "^(([0-9]|[1-9][0-9]|1[0-9]{2}|2[0-4][0-9]|25[0-5])\\.){3}([0-9]|[1-9][0-9]|1[0-9]{2}|2[0-4][0-9]|25[0-
5])(\\V([0-9]|[1-2][0-9]|3[0-2]))$",
        "ConstraintDescription": "CIDR block parameter must be in the form x.x.x.x/x"
      }

  },
```

Resources

VPC:

```
"VPC" : {
        "Type" : "AWS::EC2::VPC",
```

IntSubnet: This block is responsible for creating subnet. The created subnet is applied to PPS Internal interface.

```
"IntSubnet" : {
        "Type" : "AWS::EC2::Subnet",
```

ExtSubnet: This block is responsible for creating subnet. The created subnet is applied to PPS External interface.

```
"ExtSubnet" : {
        "Type" : "AWS::EC2::Subnet",
```

MgmtSubnet: This block is responsible for creating subnet. The created subnet is applied to PPS Management interface.

```
"MgmtSubnet" : {
        "Type" : "AWS::EC2::Subnet",
```

InternetGateway:

```
"InternetGateway" : {
        "Type" : "AWS::EC2::InternetGateway",
```

AttachGateway:

```
"AttachGateway" : {
      "Type" : "AWS::EC2::VPCGatewayAttachment",
```

PublicSubnetRouteTable:

```
"PublicSubnetRouteTable" : {
      "Type" : "AWS::EC2::RouteTable",
```

PublicSubnetRoute:

```
"PublicSubnetRoute" : {
      "Type" : "AWS::EC2::Route",
```

ExtSubnetRouteTableAssociation:

```
"ExtSubnetRouteTableAssociation" : {
      "Type" : "AWS::EC2::SubnetRouteTableAssociation",
```

MgmtSubnetRouteTableAssociation:

```
"MgmtSubnetRouteTableAssociation" : {
      "Type" : "AWS::EC2::SubnetRouteTableAssociation",
```

EIP1:

```
"EIP1" :  {
  "Type" : "AWS::EC2::EIP",
```

EIPAssoc1:

```
"EIPAssoc1" : {
  "Type" : "AWS::EC2::EIPAssociation",
```

EIP2:

```
"EIP2" :  {
  "Type" : "AWS::EC2::EIP",
```

EIPAssoc2:

```
"EIPAssoc2" : {
  "Type" : "AWS::EC2::EIPAssociation",
```

PPSvExternalSecurityGroup:

```
"PPSvExternalSecurityGroup": {
  "Type": "AWS::EC2::SecurityGroup",
```

PPSvInternalSecurityGroup:

```
"PPSvInternalSecurityGroup": {
  "Type": "AWS::EC2::SecurityGroup",
```

PPSvManagementSecurityGroup:

```
"PPSvManagementSecurityGroup": {
  "Type": "AWS::EC2::SecurityGroup",
```

EC2Instance:

```
"EC2Instance" : {
  "Type" : "AWS::EC2::Instance",
```

Eth0:

```
"Eth0" : {
  "Type" : "AWS::EC2::NetworkInterface",
```

Eth1:

```
"Eth1" : {
  "Type" : "AWS::EC2::NetworkInterface",
```

Eth2:

```
"Eth2" : {
  "Type" : "AWS::EC2::NetworkInterface",
```

Outputs

The Outputs section defines the public IP address, private IP address and primary private IP address that is displayed on successful deployment of PPS on AWS.

```
"Outputs" : {
    "InstanceId" : {
      "Value" : { "Ref" : "EC2Instance" },
      "Description" : "Instance Id of newly created instance"
    },
    "ManagementAddress" : {
      "Value" : { "Fn::Join" : [" ", [ "Public IP address:", { "Ref" : "EIP2" }, "Private IP address:", { "Fn::GetAtt" : ["Eth2",
"PrimaryPrivateIpAddress"] }]]},
      "Description" : "PPS Management Interface details"
    },
    "ExternalAddress" : {
      "Value" : { "Fn::Join" : [" ", [ "Public IP address:", { "Ref" : "EIP1" }, "Private IP address:", { "Fn::GetAtt" : ["Eth1",
"PrimaryPrivateIpAddress"] }]]},
      "Description" : "PPS Extenal Interface details"
    },
    "InternalAddress" : {
       "Value" : { "Fn::Join" : [" ", [ "Public IP address:", { "Ref" : "EIP0" }, "Private IP address:", { "Fn::GetAtt" : ["Eth0",
"PrimaryPrivateIpAddress"] }]]},
       "Description" : "PPS Internal Interface details"
      }
  }
}
```

Appendix C: Pulse Policy Secure CloudFormation Template for an Existing Virtual Private Cloud

Pulse Secure provides sample CloudFormation template files to deploy Pulse Policy Secure Virtual Appliance on AWS. Users can modify this to make it suitable for their need. Visit https://www.pulsesecure.net and download the pulsesecure-pps-3-nics.zip file, and unzip it to get **pulsesecure-pps-3-nics-existing-vpc.json**.

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

The template has following sections:

Parameters	This section defines the parameters used for deploying PPS on AWS. It contains parameter name, its default value and the mouse-over help text that is displayed when mouse is placed over the parameter in AWS Web portal. The parameters defined here are displayed in the Custom Deployment page of AWS portal.
Resources	This section defines resource types that are deployed or updated in a resource group.
Outputs	This section defines the public IP address and FQDN returned after successful deployment of PPS on AWS.

Parameters

Key Name: This is the name of the PPS Storage Account where the PPS VA image (.ami file) is stored.

```
 "Parameters" : {

  "KeyName": {
     "Type": "AWS::EC2::KeyPair::KeyName",
     "Default": "",
     "AllowedPattern" : "[-_ a-zA-Z0-9]*",
     "Description": "Name of an existing EC2 KeyPair. Your PPS will launch with this KeyPair.",
     "ConstraintDescription": "Must be the name of an existing EC2 KeyPair."
   },
```

PPS Image AMI ID: This is the ID of the uploaded AMI file.

```
"PPSImageAMIId" : {
      "Type" : "String",
      "Description" : "AMI ID of your existing PPS image"
    },
```

Instance Type: This specifies the size of the instance – t2.medium or t2.large

```
"InstanceType": {
      "Description": "Select PPS instance type",
      "Type": "String",
      "Default": "t2.medium",
      "AllowedValues": [
         "t2.medium",
         "t2.xlarge",
         "t2.2xlarge"
```

```
    ],
    "ConstraintDescription": "Must be an allowed EC2 instance type."
  },
```

PPS Config Data: 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 AWS cloud or in the corporate network which is accessible for Pulse Policy Secure through site-to-site VPN between AWS and the corporate data center.
Pulse Policy Secure accepts the following parameters as provisioning parameters:

- primary-dns
- secondary-dns
- wins-server
- dns-domain
- admin-username
- admin-password
- cert-common-name
- cert-random-text
- cert-organization
- config-download-url
- config-data
- auth-code-license
- enable-license-server
- accept-license-agreement
- enable-rest

For details about these parameters, see Pulse Policy Secure Provisioning Parameters.

```
"PPSConfigData" : {
      "Type" : "String",
      "Description" : "PPS config data",
      "Default" : "<pulse-config><wins-server>1.1.1.1</wins-server><dns-domain>psecure.net</dns-domain><admin-
username>admin</admin-username><admin-password>password1234</admin-password><cert-common-
name>va1.psecure.net</cert-common-name><cert-random-text>fdsfpisonvsfnms</cert-random-text><cert-
organisation>Psecure Org</cert-organisation><config-download-url></config-download-url><config-data></config-
data><auth-code-license></auth-code-license><enable-license-server>n</enable-license-server><accept-license-
agreement>y</accept-license-agreement></pulse-config>"
    },
```

VPCID: This is the ID of the existing VPC.

```
"VpcId" : {
      "Type" : "String",
      "Description" : "ID of existing VPC"
    },
```

SubnetIntID: This is the ID of the subnet to which PPS Internal interface connects.

```
"SubnetIntId" : {
      "Type" : "String",
      "Description" : "ID of the subnet where PPS internal interface connects"
    },
```

SubnetExtId: This is the ID of the subnet to which PPS External interface connects.

```
"SubnetExtId" : {
      "Type" : "String",
      "Description" : "ID of the subnet where PPS External interface connects"
    },
```

SubnetMgmtId: This is the ID of the subnet to which PPS Management interface connects.

```
"SubnetMgmtId" : {
      "Type" : "String",
      "Description" : "ID of the subnet where PPS Management interface connects"
    }
```

Resources

EIP1:

```
"EIP1" :  {
  "Type" : "AWS::EC2::EIP",
```

EIPAssoc1:

```
"EIPAssoc1" : {
  "Type" : "AWS::EC2::EIPAssociation",
```

EIP2:

```
"EIP2" :  {
  "Type" : "AWS::EC2::EIP",
```

EIPAssoc2:

```
"EIPAssoc2" : {
  "Type" : "AWS::EC2::EIPAssociation",
```

PPSvExternalSecurityGroup:

```
"PPSvExternalSecurityGroup": {
  "Type": "AWS::EC2::SecurityGroup",
```

PPSvInternalSecurityGroup:

```
"PPSvInternalSecurityGroup": {
  "Type": "AWS::EC2::SecurityGroup",
```

PPSvManagementSecurityGroup:

```
"PPSvManagementSecurityGroup": {
  "Type": "AWS::EC2::SecurityGroup",
```

EC2Instance:

```
"EC2Instance" : {
```

```
 "Type" : "AWS::EC2::Instance",
 "DependsOn" : ["EIPAssoc0", "EIPAssoc1", "EIPAssoc2"],
```

Eth0:

```
"Eth0" : {
 "Type" : "AWS::EC2::NetworkInterface",
```

Eth1:

```
"Eth1" : {
 "Type" : "AWS::EC2::NetworkInterface",
```

Eth2:

```
"Eth2" : {
 "Type" : "AWS::EC2::NetworkInterface",
```

Outputs

The Outputs section defines the public IP address, private IP address and primary private IP address that is displayed on successful deployment of PPS on AWS.

```
"Outputs" : {
   "InstanceId" : {
     "Value" : { "Ref" : "EC2Instance" },
     "Description" : "Instance Id of newly created instance"
   },
   "ManagementAddress" : {
     "Value" : { "Fn::Join" : [" ", [ "Public IP address:", { "Ref" : "EIP2" }, "Private IP address:", { "Fn::GetAtt" : ["Eth2",
"PrimaryPrivateIpAddress"] }]]},
     "Description" : "PPS Management Interface details"
   },
   "ExternalAddress" : {
     "Value" : { "Fn::Join" : [" ", [ "Public IP address:", { "Ref" : "EIP1" }, "Private IP address:", { "Fn::GetAtt" : ["Eth1",
"PrimaryPrivateIpAddress"] }]]},
     "Description" : "PPS Extenal Interface details"
   },
   "InternalAddress" : {
      "Value" : { "Fn::Join" : [" ", [ "Public IP address:", { "Ref" : "EIP0" }, "Private IP address:", { "Fn::GetAtt" : ["Eth0",
"PrimaryPrivateIpAddress"] }]]},
      "Description" : "PPS Internal Interface details"
     }
  }
}
```

References

AWS documentation: https://aws.amazon.com/documentation/

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.