



## AWS Machine Learning Blog

# Understanding Amazon SageMaker notebook instance networking configurations and advanced routing options

by Ben Snively | on 24 OCT 2018 | in [Artificial Intelligence](#), [SageMaker](#) | [Permalink](#) | [Comments](#) | [Share](#)

An Amazon SageMaker notebook instance provides a Jupyter notebook app through a fully managed machine learning (ML) Amazon EC2 instance. Amazon SageMaker Jupyter notebooks are used to perform advanced data exploration, create training jobs, deploy models to Amazon SageMaker hosting, and test or validate your models.

The notebook instance has a variety of networking configurations available to it. In this blog post we'll outline the different options and discuss a common scenario for customers.

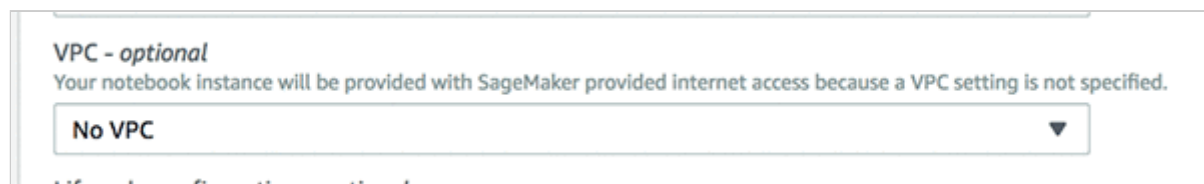
## The basics

Amazon SageMaker notebook instances can be launched with or without your [Virtual Private Cloud](#) (VPC) attached. When launched with your VPC attached, the notebook can either be configured with or without direct internet access.

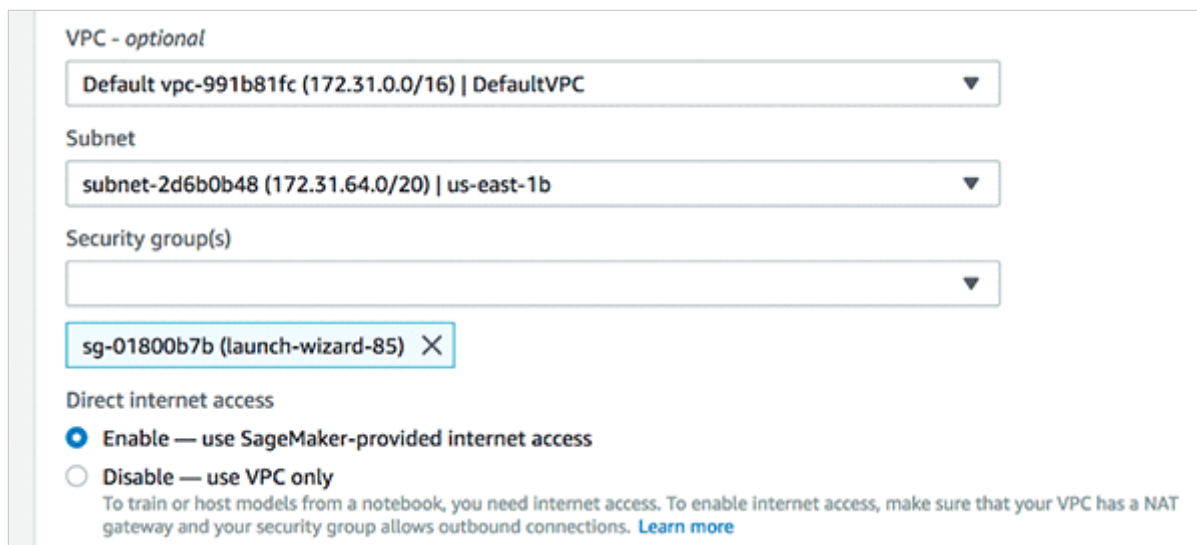
**IMPORTANT NOTE:** Direct internet access means that the Amazon SageMaker service is providing a network interface that allows for the notebook to talk to the internet through a VPC managed by the service.

Using the Amazon SageMaker console, these are the three options:

1. No customer VPC is attached.

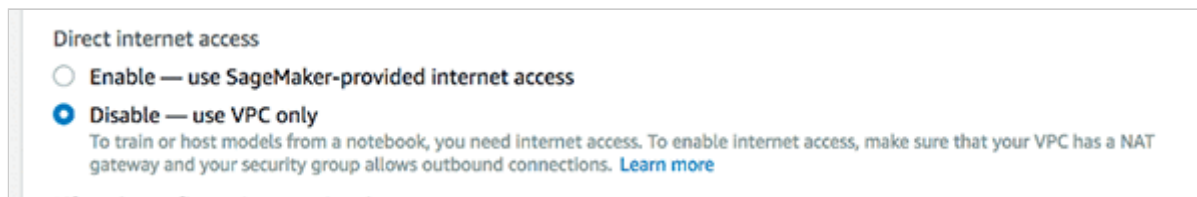


## 2. Customer VPC is attached with direct internet access.



The screenshot shows the 'VPC - optional' section of the SageMaker console. It includes three dropdown menus: 'VPC' set to 'Default vpc-991b81fc (172.31.0.0/16) | DefaultVPC', 'Subnet' set to 'subnet-2d6b0b48 (172.31.64.0/20) | us-east-1b', and 'Security group(s)' set to 'sg-01800b7b (launch-wizard-85)'. Below these, the 'Direct internet access' section has two radio buttons: 'Enable — use SageMaker-provided internet access' (selected) and 'Disable — use VPC only'. A note below the radio buttons states: 'To train or host models from a notebook, you need internet access. To enable internet access, make sure that your VPC has a NAT gateway and your security group allows outbound connections. [Learn more](#)'.

## 3. Customer VPC is attached without direct internet access.

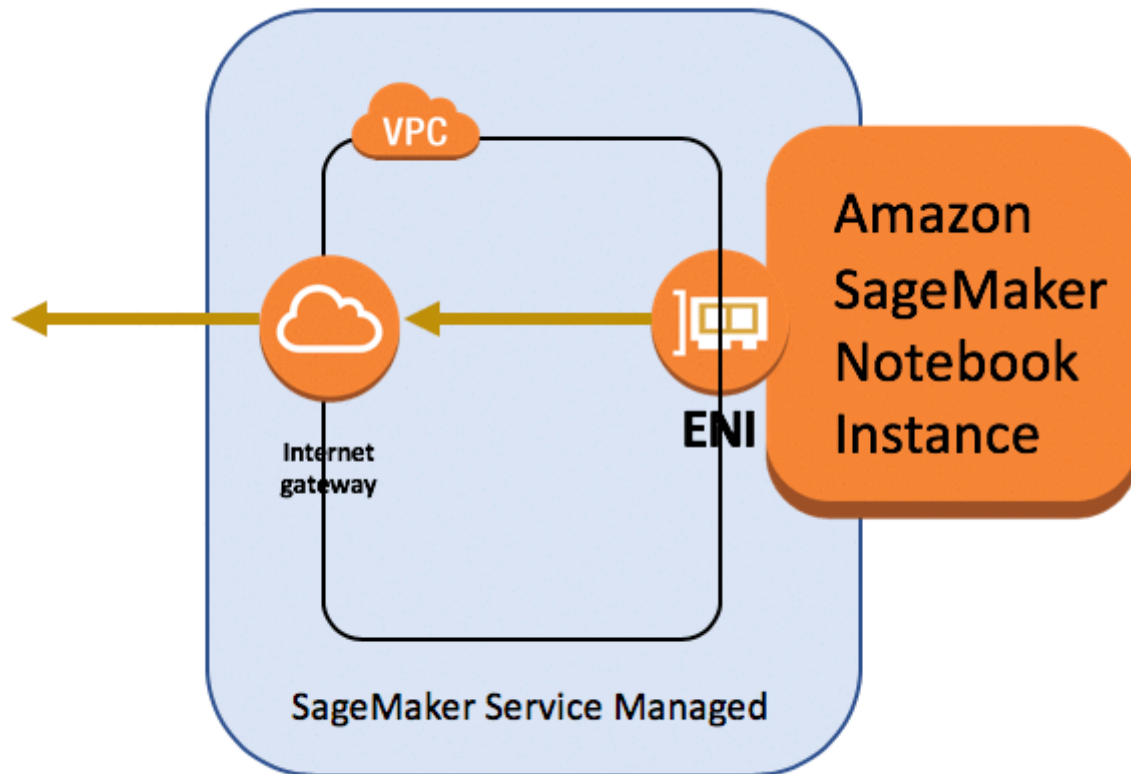


The screenshot shows the 'Direct internet access' section of the SageMaker console. It has two radio buttons: 'Enable — use SageMaker-provided internet access' (unselected) and 'Disable — use VPC only' (selected). A note below the radio buttons states: 'To train or host models from a notebook, you need internet access. To enable internet access, make sure that your VPC has a NAT gateway and your security group allows outbound connections. [Learn more](#)'.

## What does it really mean?

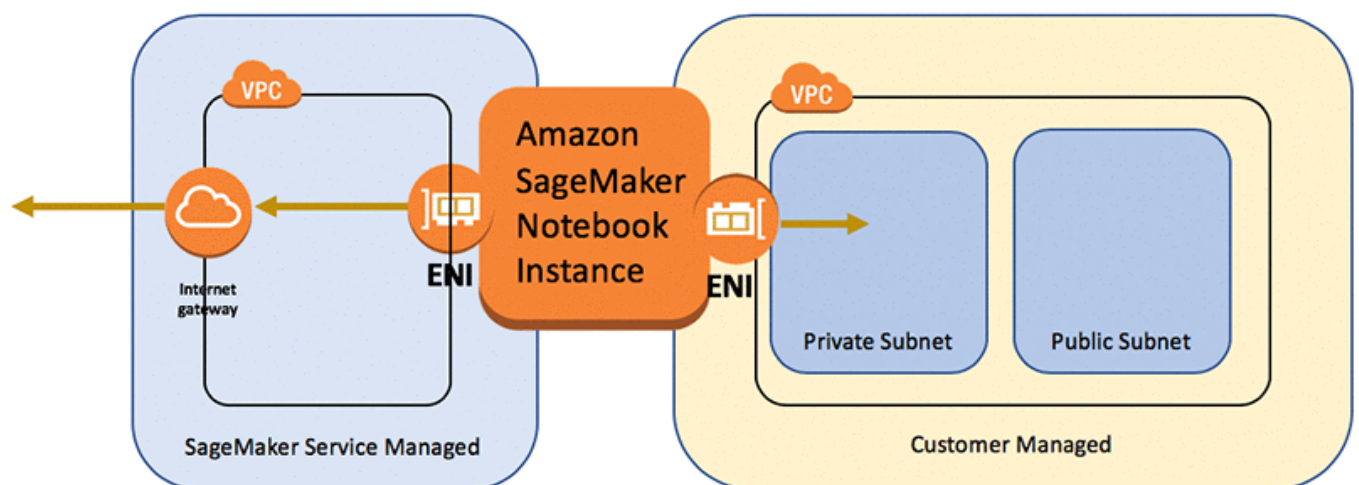
Each of the three options automatically configures the network interfaces on the managed EC2 instance with a set of routing configurations. In certain situations, you might want to modify these settings to route specific IP address ranges to a different network interface. Next, we'll step through each of these default configurations:

## 1. No attached customer VPC (1 network interface)



In this configuration, all the traffic goes through the single network interface. The notebook instance is running in an Amazon SageMaker managed VPC as shown in the above diagram.

## 2. Customer attached VPC with direct internet access (2 network interfaces)



In this configuration, the notebook instance needs to decide which network traffic should go down either of the two network interfaces.

If we look at an example where we launched into a VPC with a 172.31.0.0/16 CIDR range and look at the OS-level route information, we see this.

```

jupyter

sh-4.2$ route -n
Kernel IP routing table
Destination      Gateway         Genmask         Flags Metric Ref    Use Iface
0.0.0.0          172.16.32.1    0.0.0.0         UG    0      0      0 eth0
169.254.0.0      0.0.0.0        255.255.255.0   U      0      0      0 veth_def_agent
169.254.169.254  0.0.0.0        255.255.255.255 UH    0      0      0 eth0
172.17.0.0       0.0.0.0        255.255.0.0     U      0      0      0 docker0
172.31.0.0       172.31.64.1    255.255.0.0     UG    0      0      0 eth2
172.31.64.0      0.0.0.0        255.255.240.0   U      0      0      0 eth2
sh-4.2$

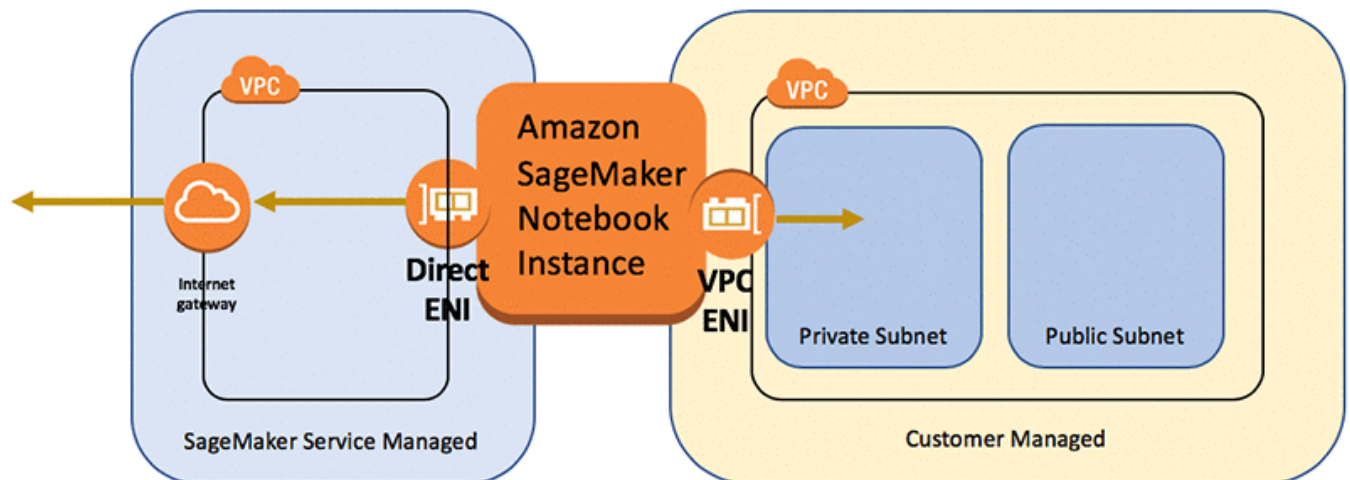
```

Looking at this route table, we can see:

- 172.31.0.0/16 traffic will use eth2 interface.
- Some Docker and metadata routes.
- All other traffic will use the eth0 interface.

For simplicity, we'll focus on the eth0 and eth2 configurations and not the other Docker/ec2 metadata entries. This shows us the following configuration:

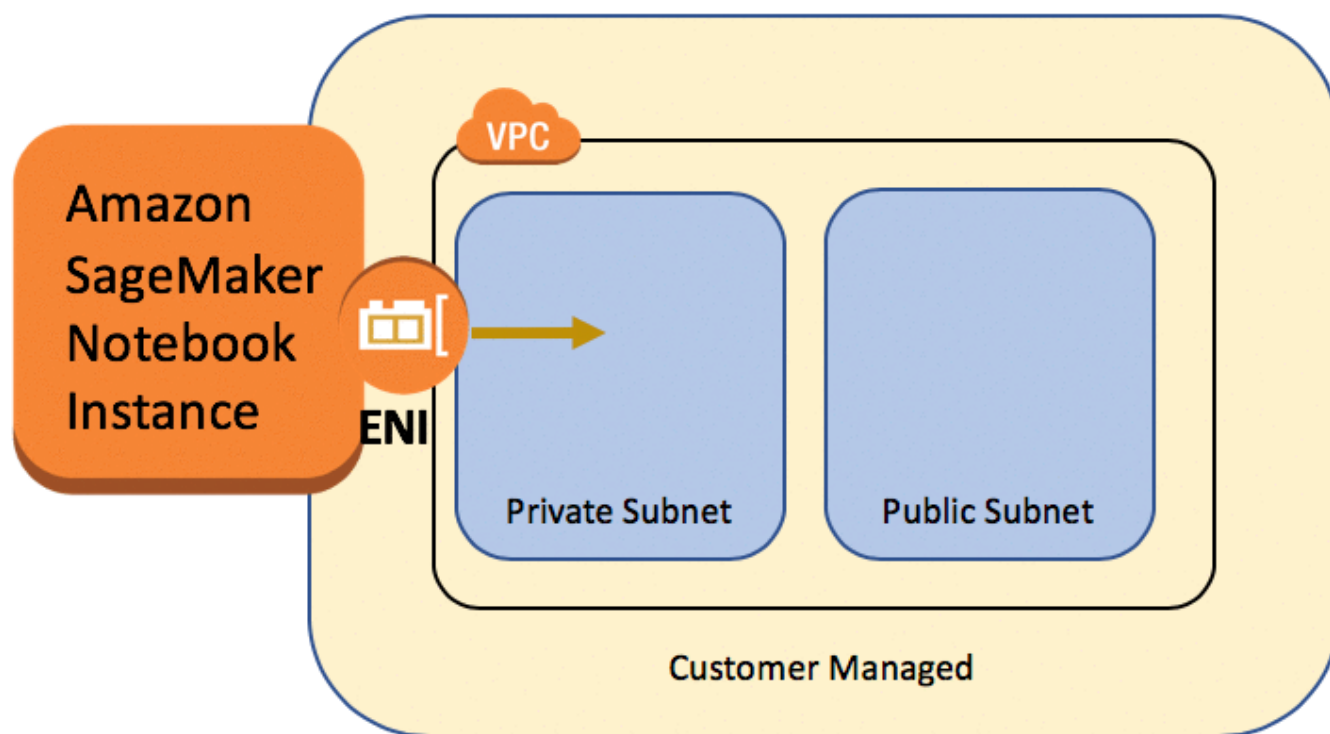
Endpoint	Interface Name	Route:
172.31.0.0/16	eth2	VPC ENI
Everything Else*	eth0	Direct ENI



This default setting uses the internet network interface (eth0) for all traffic except for the CIDR range for the customer attached VPC (eth2). This setting sometimes needs to be overwritten when interacting with either on-premises or peered-VPC resources.

### 3. Customer attached VPC without direct internet access.

**IMPORTANT NOTE:** In this configuration, the notebook instance can still be configured to access the internet. The network interface that gets launched only has a private IP address. What that means is that it needs to either be in a *private subnet* with a NAT or to access the internet back through a virtual private gateway. If launched into a public subnet, it won't be able to speak to the internet through an internet

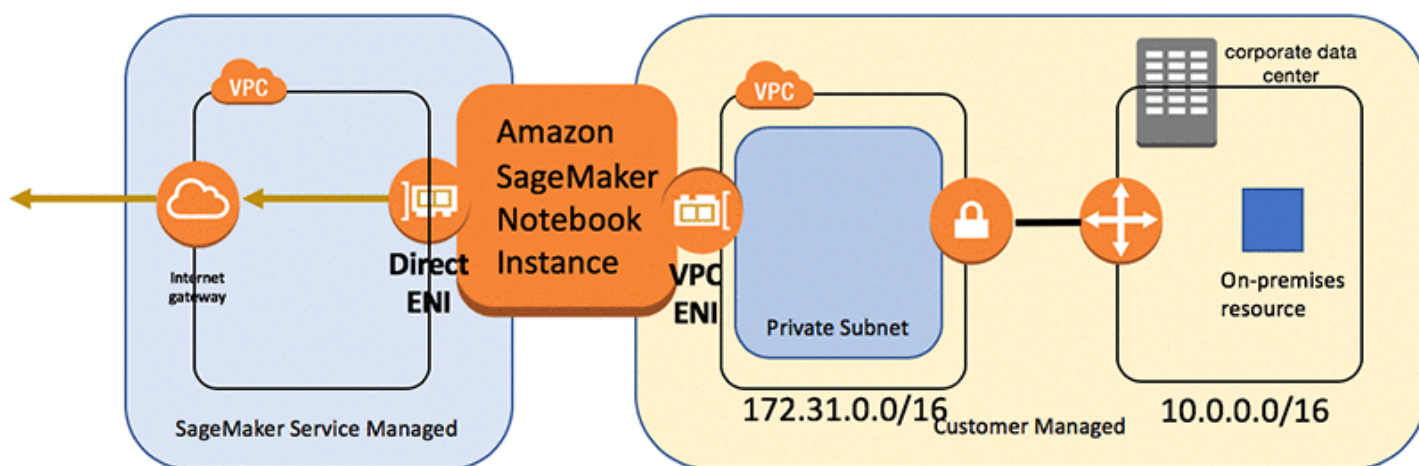


## Common customer patterns

Accessing on-premises resources from an Amazon SageMaker instance with direct internet access:

Suppose we have the following configuration:

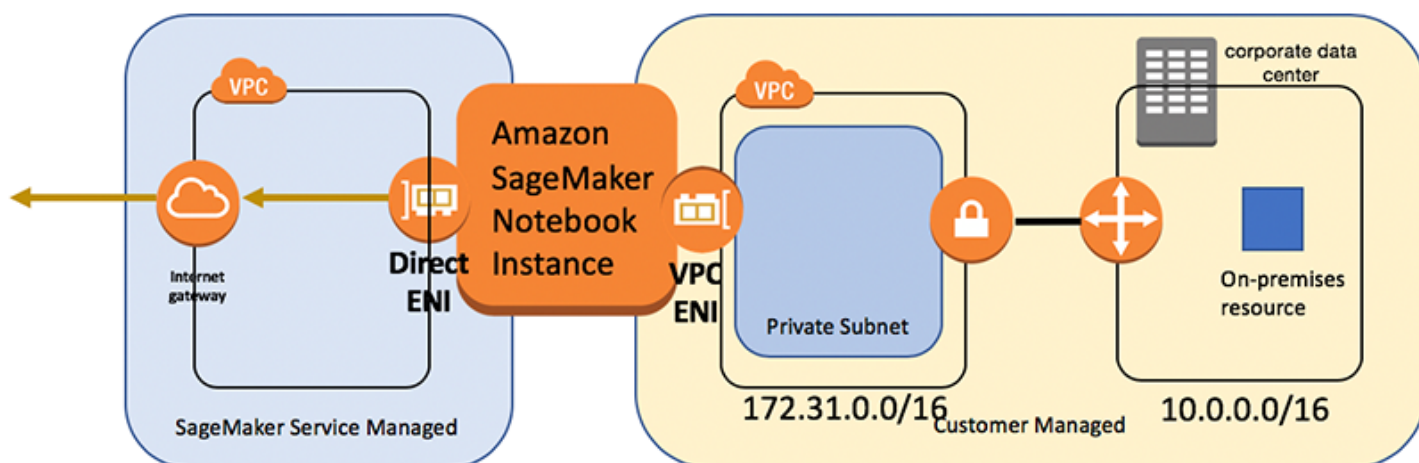
Endpoint	Interface Name	Route:
172.31.0.0/16	eth2	VPC ENI
Everything Else*	eth0	Direct ENI



If we try to access the on-premises resource in the 10.0.0.0/16 CIDR range, it will get routed by the OS through the eth0 internet interface. This interface doesn't have the connection back on-premises and won't allow us to communicate with on-premises resources.

To route back on premises, we'll want to update the route table to have the following:

Endpoint	Interface Name	Route:
172.31.0.0/16	eth2	VPC ENI
10.0.0.0/16	eth2	VPC ENI
Everything Else*	eth0	Direct ENI





To do this, we can perform the following commands from a terminal on the Amazon SageMaker notebook instance:

```
sh-4.2$ sudo ip route add 10.0.0.0/16 via 172.31.64.1 dev eth2
```

Now if we look at the route table by entering "route -n" we see the route:

```
sh-4.2$ route -n
Kernel IP routing table
Destination      Gateway          Genmask          Flags Metric Ref    Use    Iface
0.0.0.0          172.16.32.1     0.0.0.0          UG        0      0      0     eth0
10.0.0.0          172.31.64.1     255.255.0.0      UG        0      0      0     eth2
169.254.0.0       0.0.0.0         255.255.255.0    U         0      0      0    veth_def_age
169.254.169.254   0.0.0.0         255.255.255.255  UH        0      0      0     eth0
172.17.0.0        0.0.0.0         255.255.0.0      U         0      0      0    docker0
172.31.0.0        172.31.64.1     255.255.0.0      UG        0      0      0     eth2
172.31.64.0       0.0.0.0         255.255.240.0    U         0      0      0     eth2
```

We see a route for 10.0.0.0 with mask 255.255.0.0 (which is the same as 10.0.0.0/16) going through the VPC routing IP address (172.31.64.1).

There is still one issue.

If we restart the notebook the changes won't persist. Only the changes made to the ML storage volume are persisted with a stop/start. Generally, for the package and files to be persisted, they need to be under "/home/ec2-user/SageMaker". In this case, we'll use a different feature: [lifecycle configuration](#) to add the route any time the notebook starts.

We can create a lifecycle policy as shown in the following diagram:

## Create lifecycle configuration

### Configuration setting

Name

Alphanumeric characters and "-", no spaces. Maximum 63 characters.

### Scripts

**Start notebook** | Create notebook

This script will be run each time an associated notebook instance is started, including during initial creation. If the associated instance is already started, it will be run the next time it is stopped and started.

```
1 #!/bin/bash
2
3 set -e
4
5 sudo ip route add 10.0.0.0/16 via 172.31.64.1 dev eth2
```

Now we can create our notebooks with this lifecycle configuration:

gateway and your security group allows outbound connections. [Learn more](#)

**Lifecycle configuration - optional**  
Customize your notebook environment with default scripts and plugins.

**Route10CidrThroughVPC** ▼

**Encryption key - optional**

With this setup when the notebook gets created or when it's stopped and restarted, we will have the networking configuration we expect:



```
sh-4.2$ route -n
Kernel IP routing table
Destination      Gateway         Genmask         Flags Metric Ref    Use Iface
0.0.0.0          172.16.32.1    0.0.0.0         UG    0      0      0 eth0
10.0.0.0         172.31.64.1    255.255.0.0     UG    0      0      0 eth2
169.254.0.0      0.0.0.0        255.255.255.0   U     0      0      0 veth_def_agent
169.254.169.254  0.0.0.0        255.255.255.255 UH    0      0      0 eth0
172.17.0.0       0.0.0.0        255.255.0.0     U     0      0      0 docker0
172.31.0.0       172.31.64.1    255.255.0.0     UG    0      0      0 eth2
172.31.64.0      0.0.0.0        255.255.240.0   U     0      0      0 eth2
sh-4.2$
```

## Conclusion

Amazon SageMaker Jupyter notebooks are used to perform advanced data exploration, create model training jobs, deploy models to Amazon SageMaker hosting, and test or validate your models. This drives the need for the notebook instances to have various networking configurations available to it. Knowing how these configurations can be adapted allows you to integrate with existing resources in your organization and enterprise.

---

## About the Author



**Ben Snively is an AWS Public Sector Specialist Solutions Architect.** He works with government, non-profit, and education customers on big data/analytical and AI/ML projects, helping them build solutions using AWS.

## Resources

[Getting Started](#)







[What's New](#)

[Top Posts](#)

[Official AWS Podcast](#)

[AWS Case Studies](#)

## Follow

-  [Twitter](#)
-  [Facebook](#)
-  [LinkedIn](#)
-  [Twitch](#)
-  [RSS Feed](#)
-  [Email Updates](#)

## Related Posts

---

[Identifying worker labeling efficiency using Amazon SageMaker Ground Truth](#)

[Millennium Management: Secure machine learning using Amazon SageMaker](#)

[Build machine learning-powered business intelligence analyses using Amazon QuickSight](#)

[In the News: AWS Powers Real-time Statistics for Guinness Six Nations](#)

[7 defunct restaurant brands we can learn from](#)

[Amazon Comprehend now supports multi-label custom classification](#)

[Unlocking the Value of Your Contact Center Data with TrueVoice Speech Analytics from Deloitte](#)

[Building a business intelligence dashboard for your Amazon Lex bots](#)