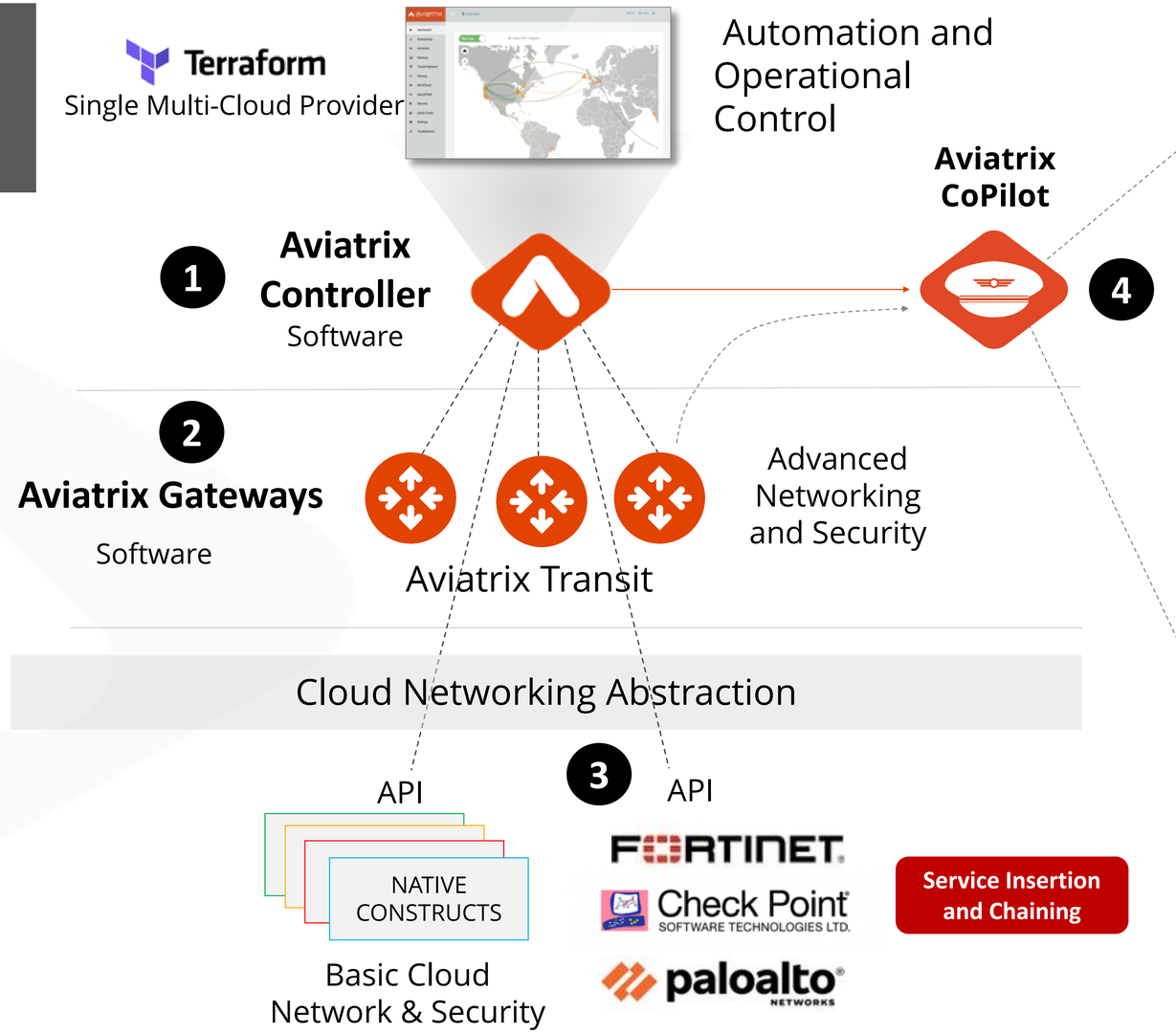
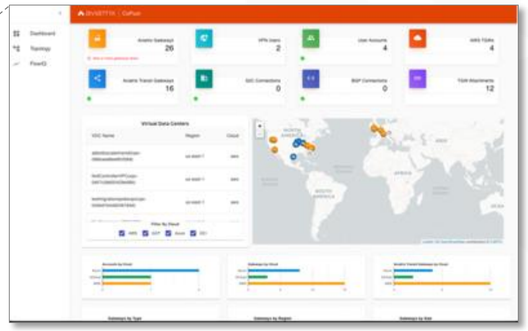
Aviatrix Cloud Network Platform Software

Not a SaaS or Managed Service. It's Yours.





Multi-Cloud Operational Visibility

FlowIQ Multi-Cloud Traffic Flow Analysis

Multi-Cloud Dynamic Topology Mapping

AWS Immersion Day
LAB 2

BUILD AND TROUBLESHOOT YOUR CLOUD NETWORK BACKBONE

Brad Hedlund
Principal Solutions Architect,
Aviatrix Systems

aviatrix

Lab 1 Recap

In Lab 1 you created (3) AWS VPCs in the us-east-1 region, each with an EC2 instance.

There is a 4th VPC called the "Controller VPC" that contains your Aviatrix Controller and Aviatrix CoPilot as EC2 instances

Next, you will use Aviatrix CoPilot to build a multi-region cloud network backbone…

Lab 2 Intro

AWS us-west-2

10.51.0.0/16
Spoke 1
Jump Instance
10.51.0.10
SAP1

10.52.0.0/16
Spoke 2
10.52.0.10
SAP2

10.52.0.0/16
Spoke 3
10.53.0.10
SAP3

Transit
10.50.0.0/16

AWS us-east-1

10.0.0.0/16
VPC A
EC2-A

10.1.0.0/16
VPC B
EC2-B

Controller VPC
Aviatrix CoPilot
Aviatrix Controller

10.2.0.0/16
VPC C (Transit)
EC2-C

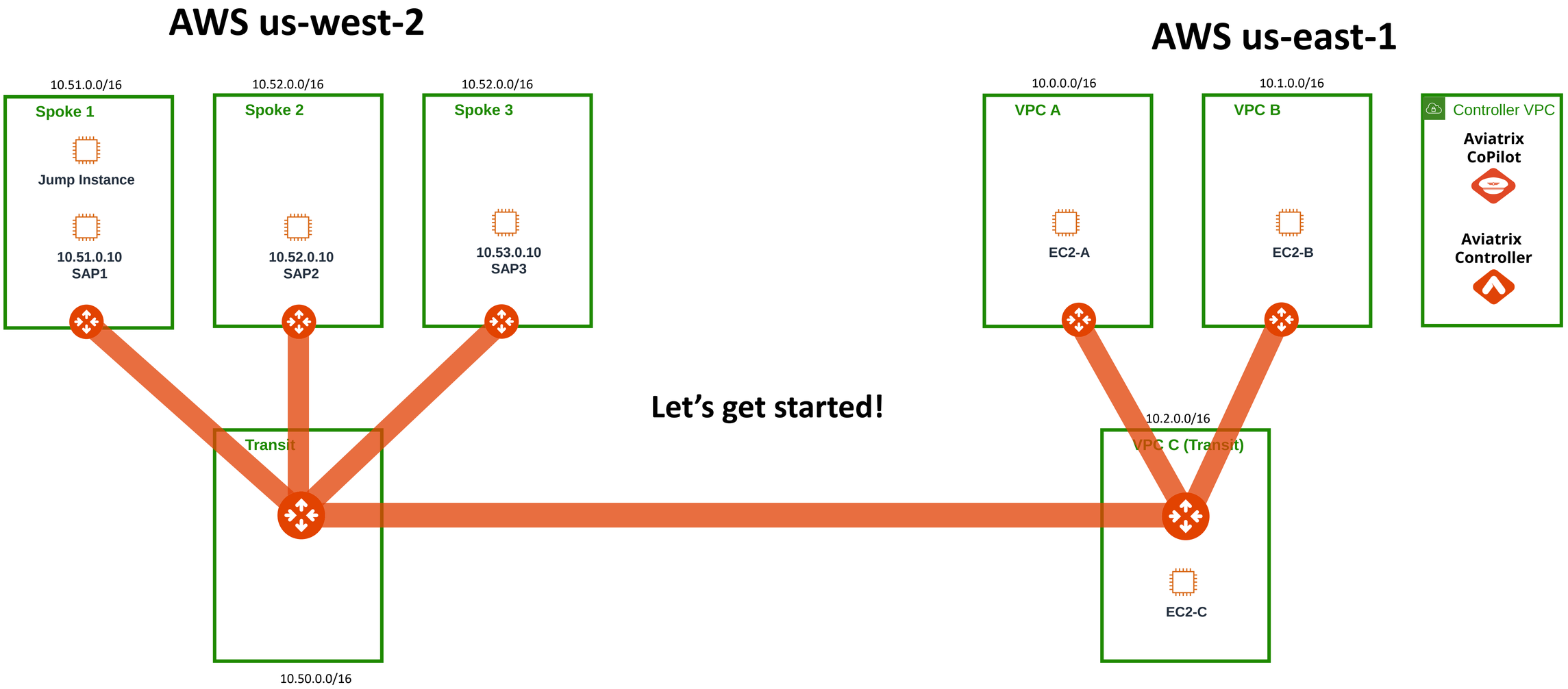
Your AWS lab account also has (4) VPCs in the us-west-2 region, (3) Spoke VPCs and (1) Transit VPC.

The VPCs in us-west-2 already have Aviatrix Gateways deployed (but not connected to anything).

In Lab 2 you will deploy Aviatrix gateways in us-east-1 and connect the two AWS regions together.

Lab 2: Cloud Backbone

When you are done with Lab 2, your cloud network will look like this



Lab 2: Step 2.0

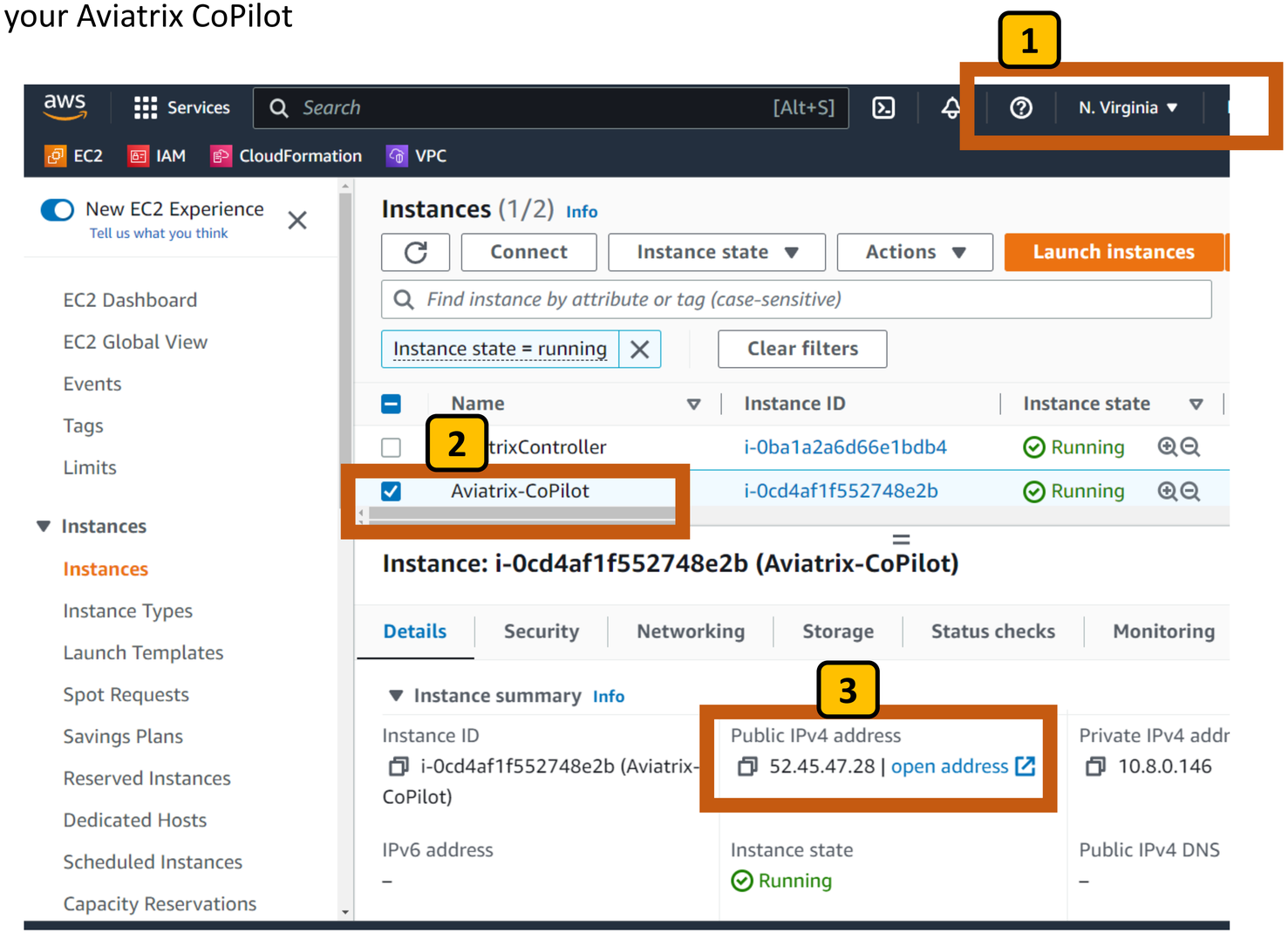
Log in to your Aviatrix CoPilot

Go the us-east-1 **N. Virginia** region in your AWS Console 1

Select the Aviatrix-Copilot EC2 instance 2

On the Details tab, find the Public IP address and click on “open address” 3

This will open a browser tab to log on to the Aviatrix Copilot UI



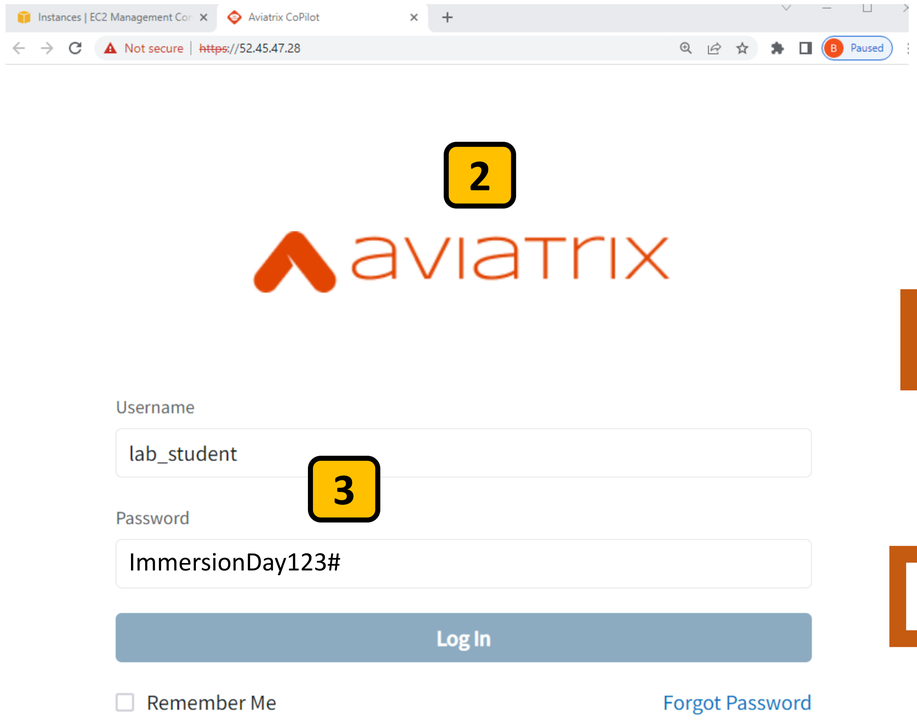
Lab 2: Step 2.1

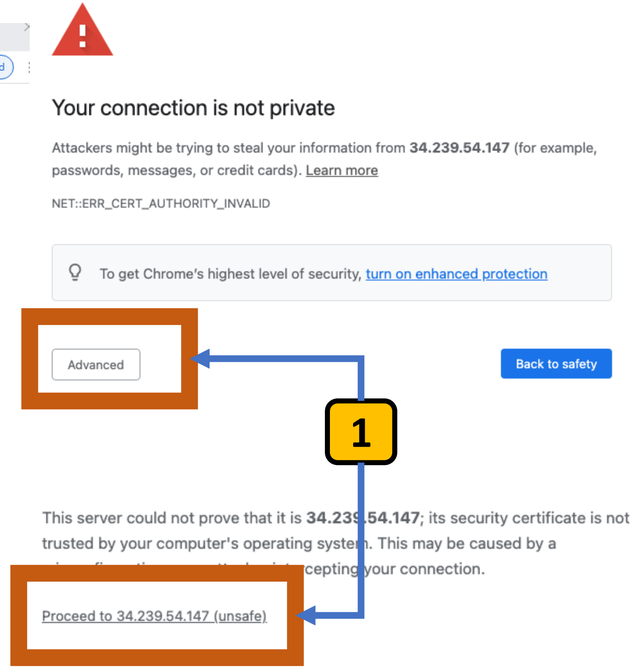
Log in to your Aviatrix CoPilot

Acknowledge the Certificate warning by clicking Advanced, then Proceed (Chrome browser) **1**

You should see the Aviatrix CoPilot logon page **2**

Username = lab_student
Password = ImmersionDay123# **3**

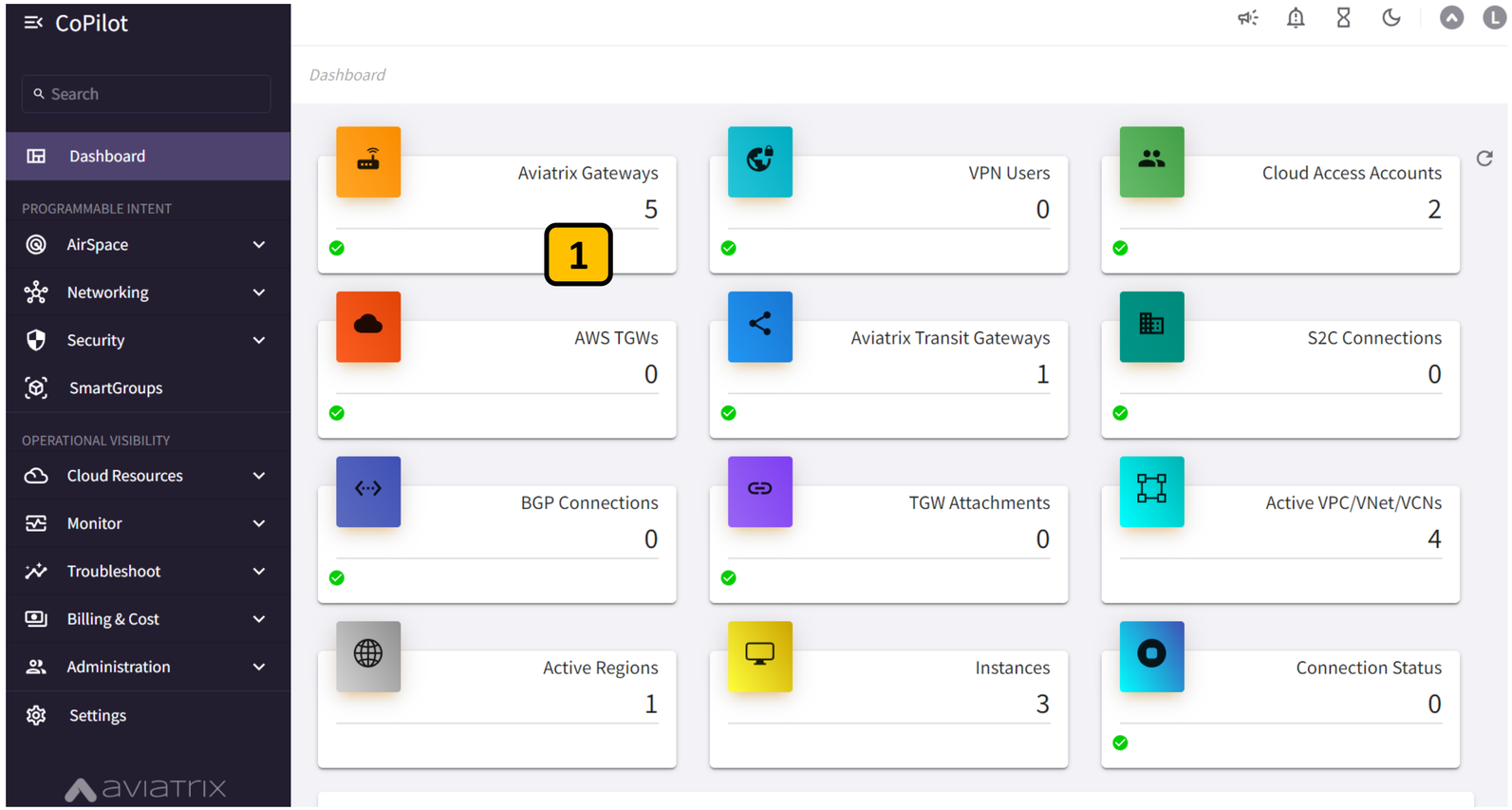




Lab 2: Step 2.2

Log in to your Aviatrix CoPilot

You should now see your Aviatrix CoPilot UI
Take a minute to checkout the main Dashboard page and all the info it provides.



Click on any widget to get more info 1

Lab 2: Cloud Backbone: Step 2.3

Network your VPCs in us-east-1 with Aviatrix Spoke and Transit Gateways

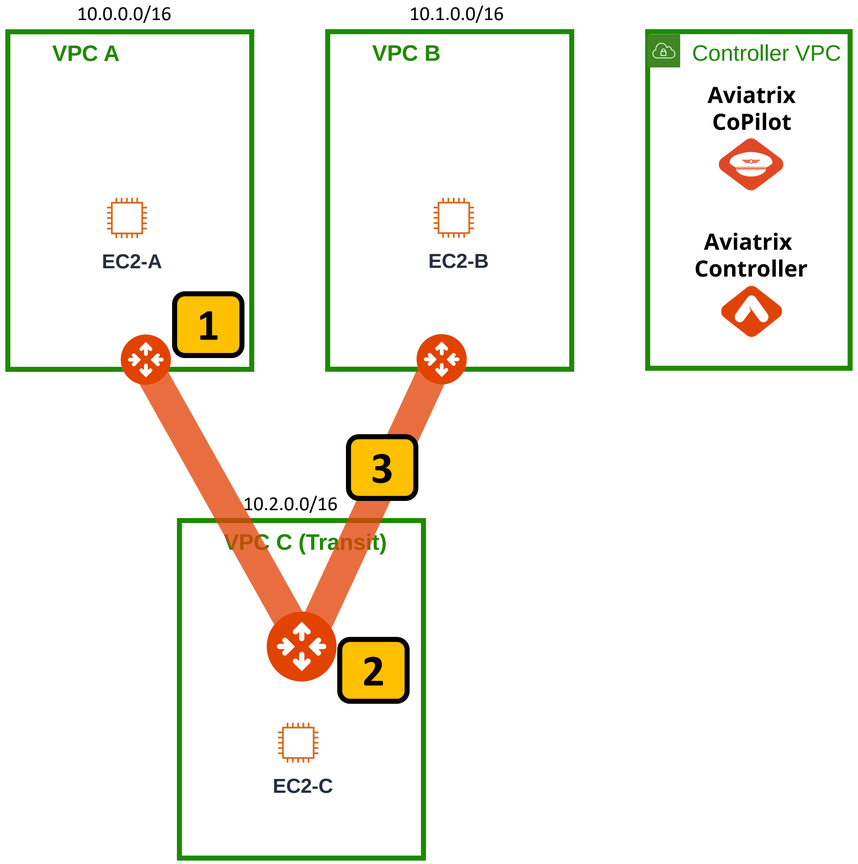
You will start by creating a hub and spoke architecture in your first region, us-east-1.

VPCs A & B will be our Spoke VPCs, where we will deploy Aviatrix Spoke Gateways. 1

VPC C will be the Transit VPC where we will deploy Aviatrix Transit Gateway (the “hub”). 2

We will connect the Spokes to the Transit with secure IPsec connections. 3

AWS us-east-1



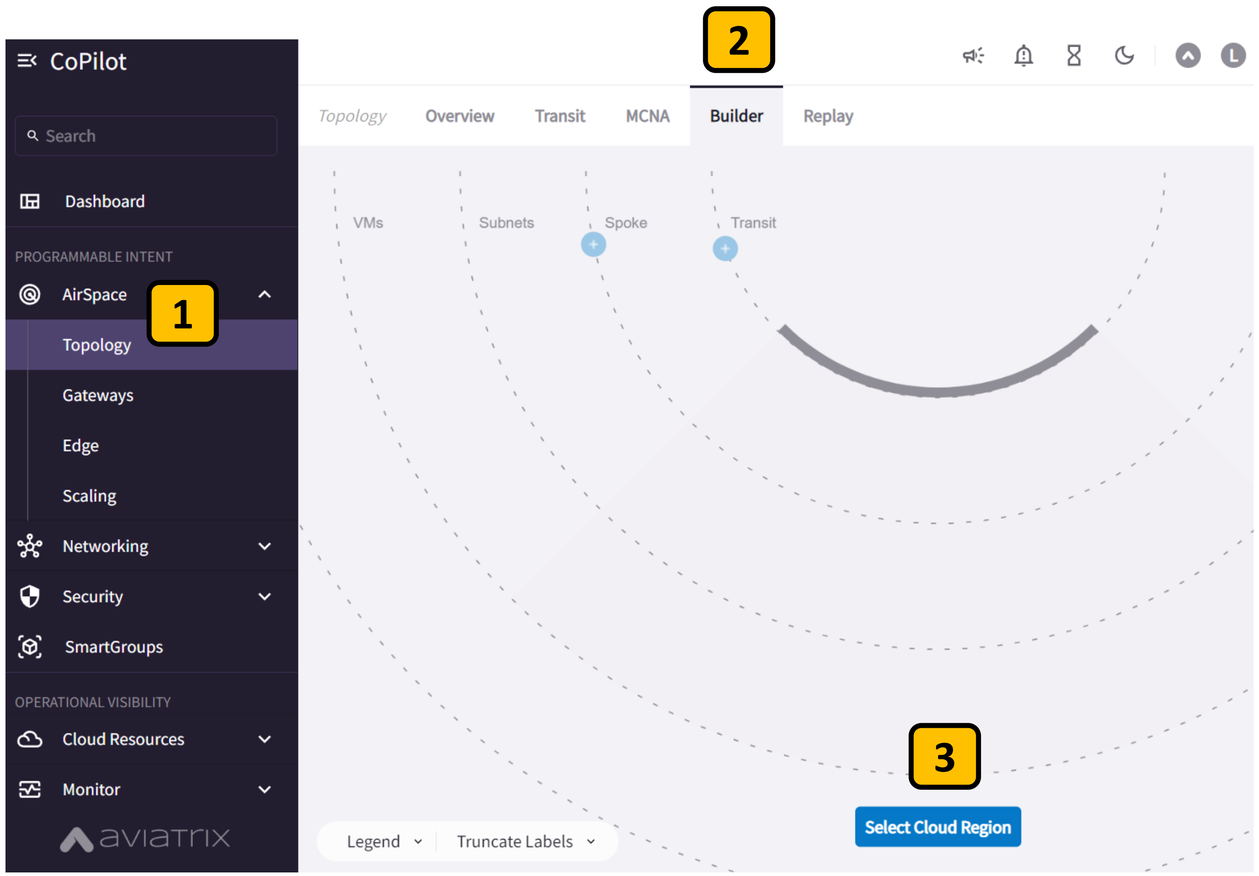
Lab 2: Cloud Backbone: Step 2.4

Use the Topology Builder to launch an Aviatrix Transit Gateway in a Transit VPC

From the CoPilot UI select AirSpace > Topology 1

Select the Builder tab 2

Click on Select Cloud Region 3



Lab 2: Cloud Backbone: Step 2.5

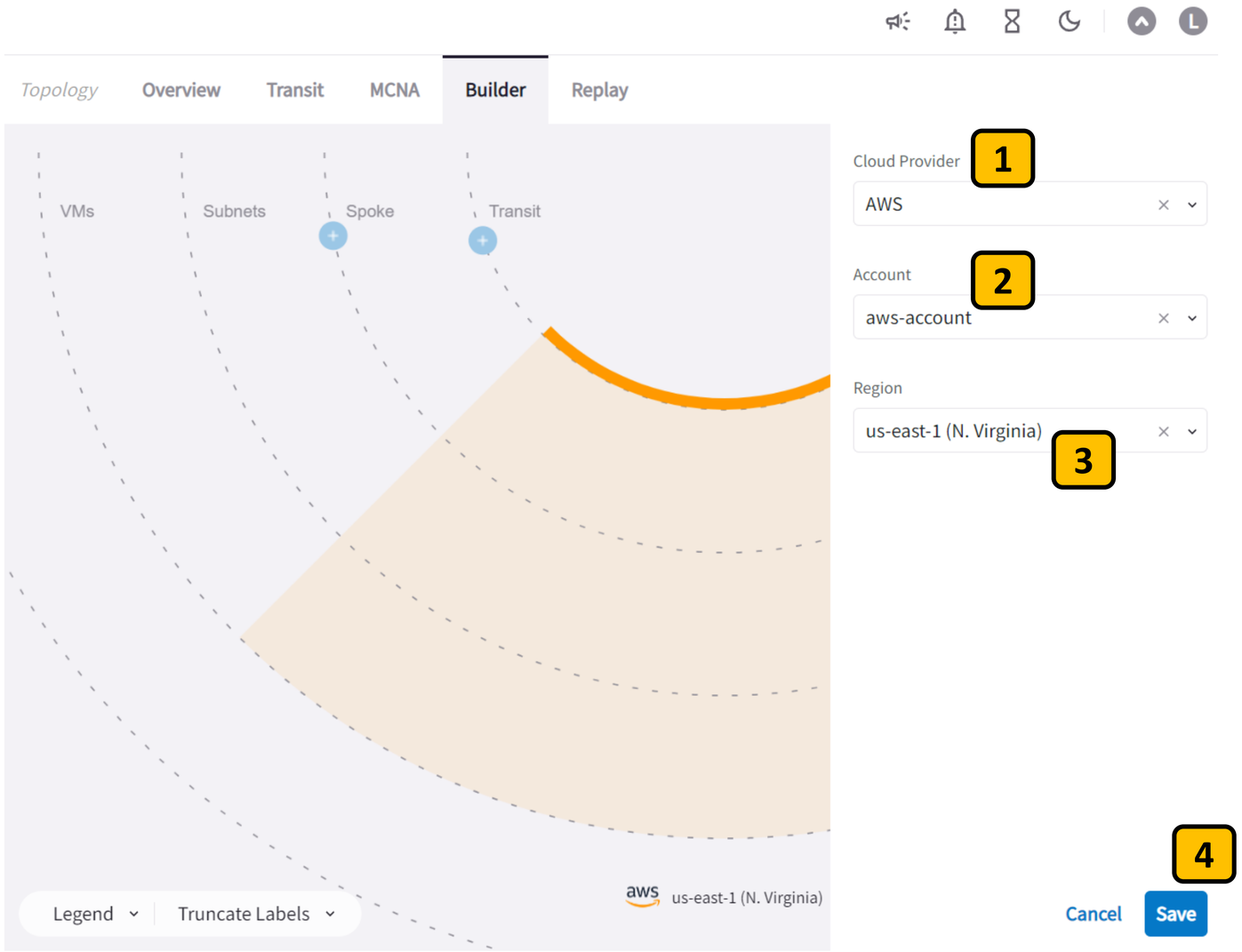
Use the Topology Builder to launch an Aviatrix Transit Gateway in a Transit VPC

Select the Cloud Provider: **AWS** 1

Select the AWS Account: **aws-account** 2

Select the Region: **us-east-1** 3

Click **Save** to start a new Build session in this cloud, account, and region. 4



Lab 2: Cloud Backbone: Step 2.6

Use the Topology Builder to launch an Aviatrix Transit Gateway in a Transit VPC

Click the **blue +** under Transit 1

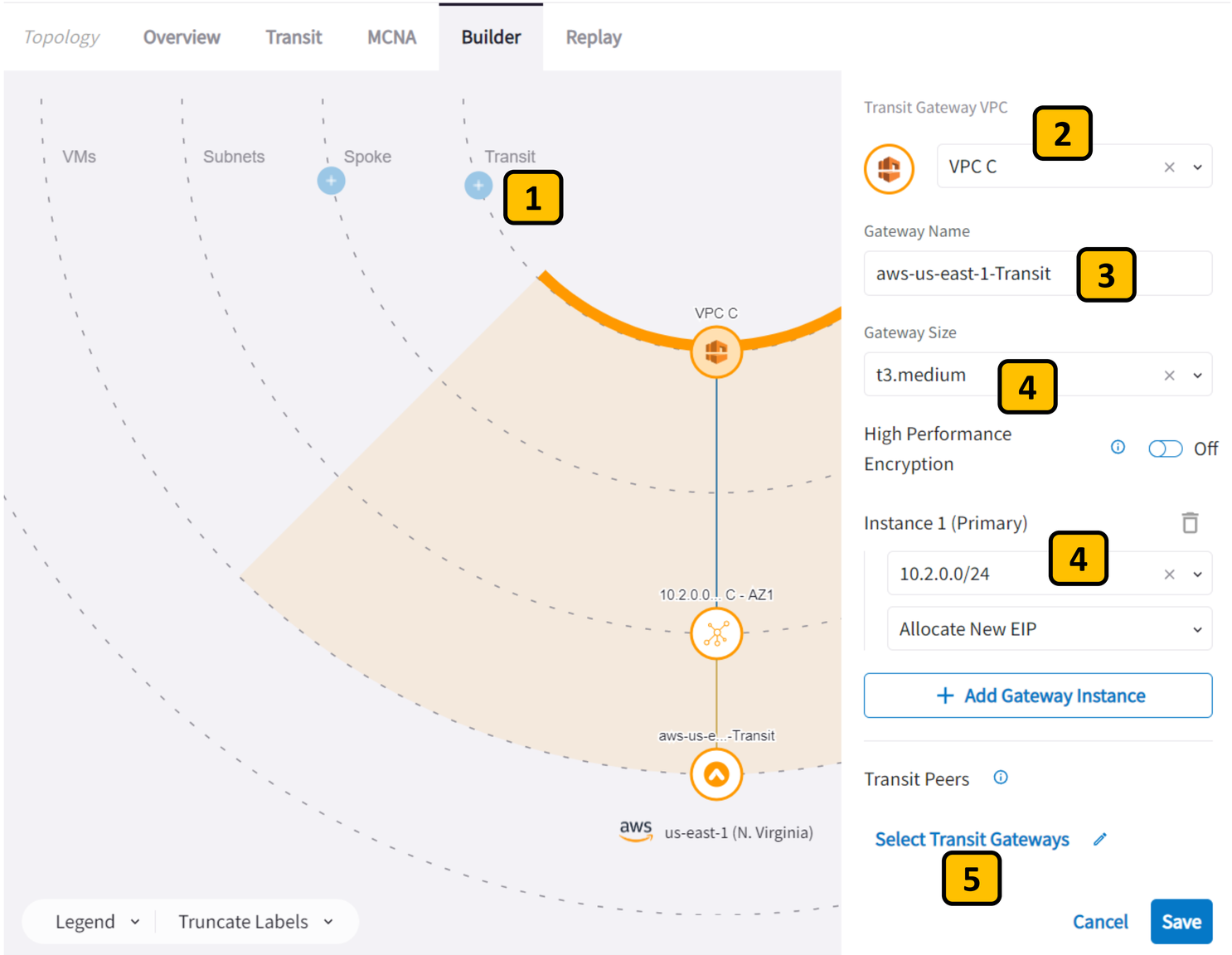
Select **VPC C** to be the Aviatrix Transit Gateway VPC 2

Name the gateway: **aws-us-east-1-Transit** 3

Select **t3.medium** for the gateway instance size 4

Select the 10.2.0.0/24 as the public subnet to deploy the gateway in to 4

Let's peer this Transit to us-west-2 by clicking **Select Transit Gateways** 5



Lab 2: Cloud Backbone: Step 2.7

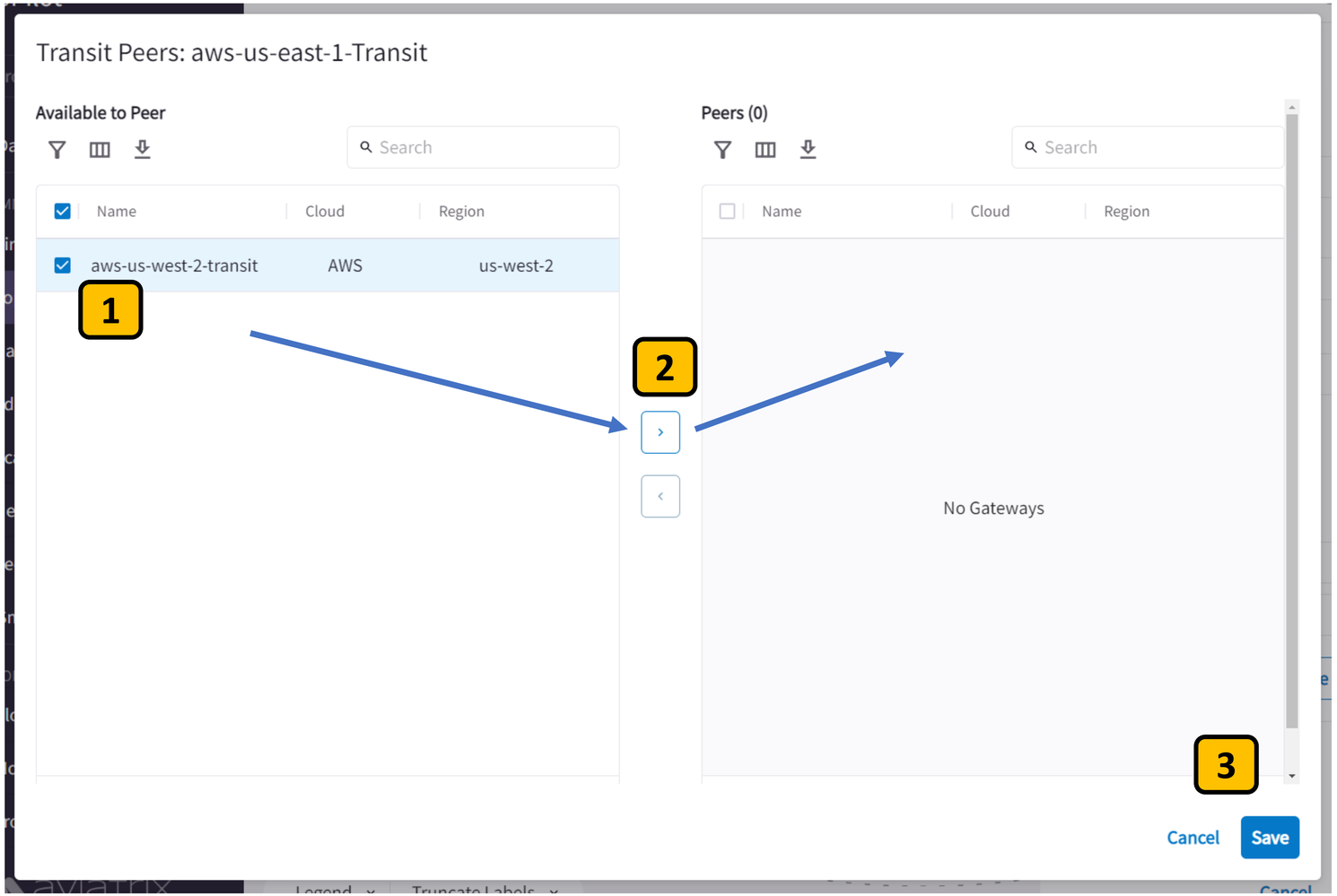
Use the Topology Builder to launch an Aviatrix Transit Gateway in a Transit VPC

You will be asked which Aviatrix Transit you want to peer to. Select the aws-us-west-2-Transit. 1

Click the right arrow to move the aws-us-west-2-Transit to the peering column 2

We have now instructed CoPilot to peer our new Aviatrix Transit in us-east-1 to the Aviatrix Transit in us-west-2.

Click Save 3



Lab 2: Cloud Backbone: Step 2.8

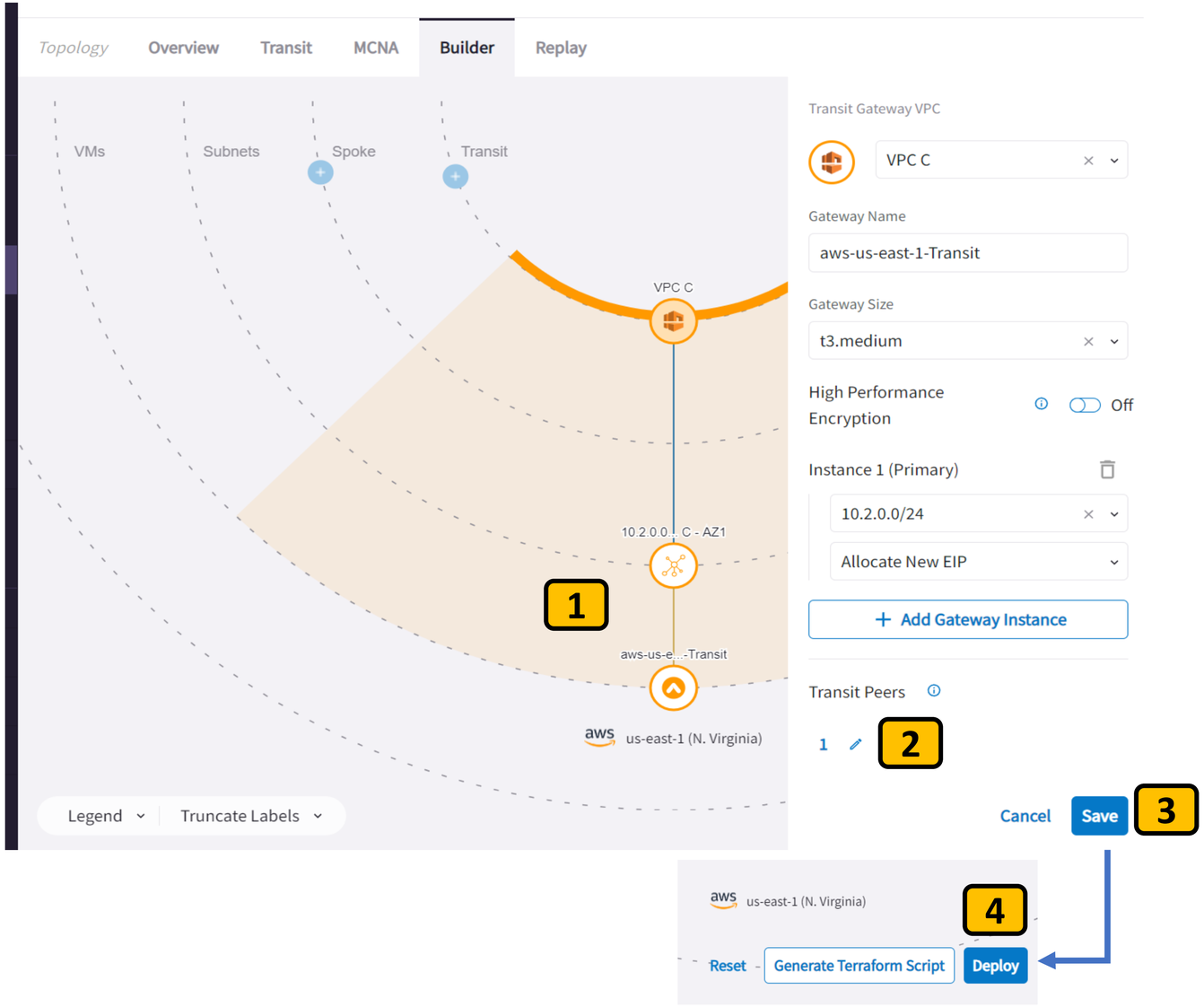
Use the Topology Builder to launch an Aviatrix Transit Gateway in a Transit VPC

You build session should now look like this. 1

After deployment, CoPilot will peer this Aviatrix Transit Gateway to (1) other transit that you specified, aws-us-west-2-Transit 2

We are ready to deploy, so..
Click Save. 3

Then click Deploy 4



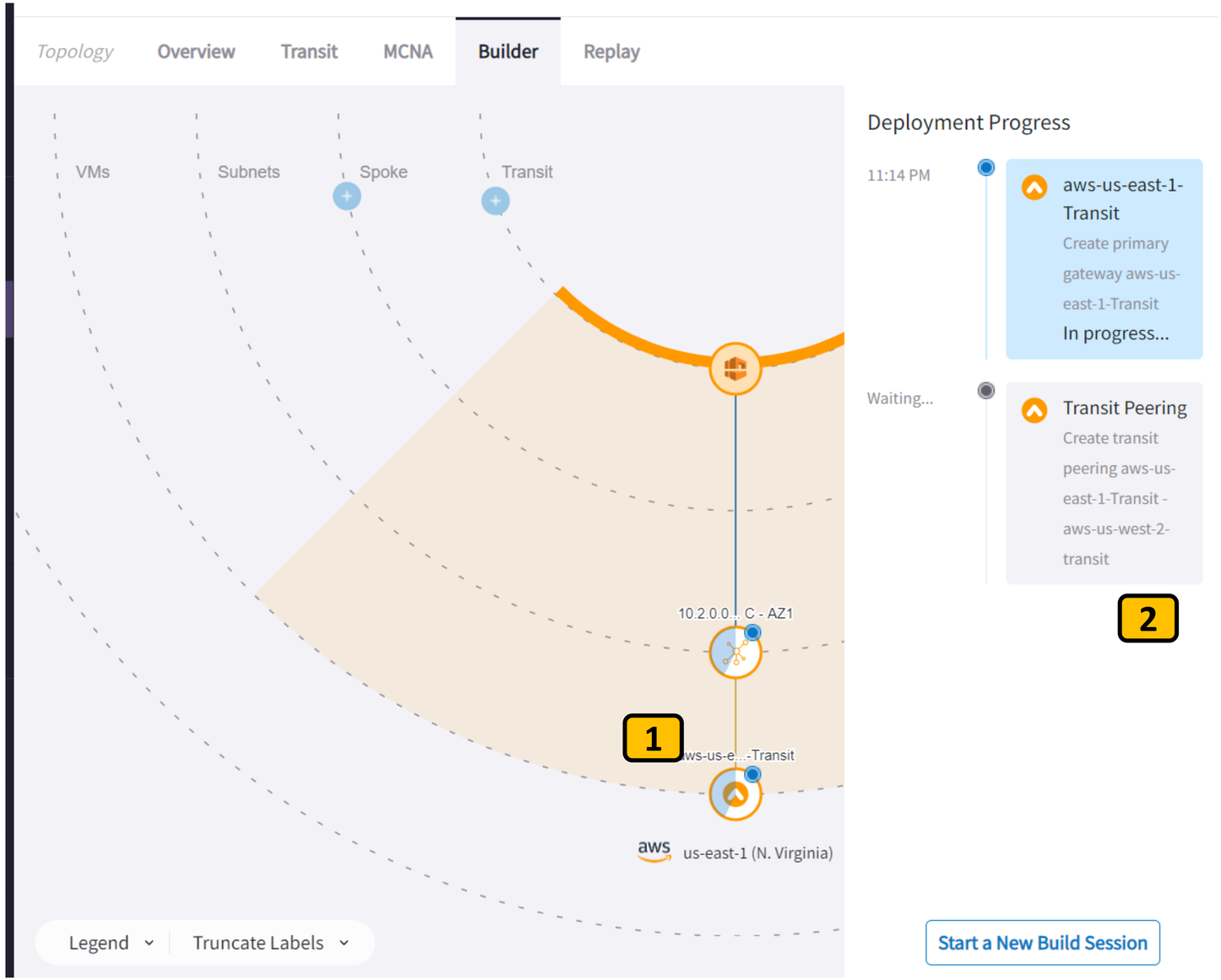
Lab 2: Cloud Backbone: Step 2.9

Use the Topology Builder to launch an Aviatrix Transit Gateway in a Transit VPC

Observe the Deployment Progress of your Aviatrix Transit Gateway in us-east-1. **1**

After the gateway is deployed, CoPilot will peer this transit to us-west-2, just like you asked it to. **2**

The total deployment time should take about 5 minutes.

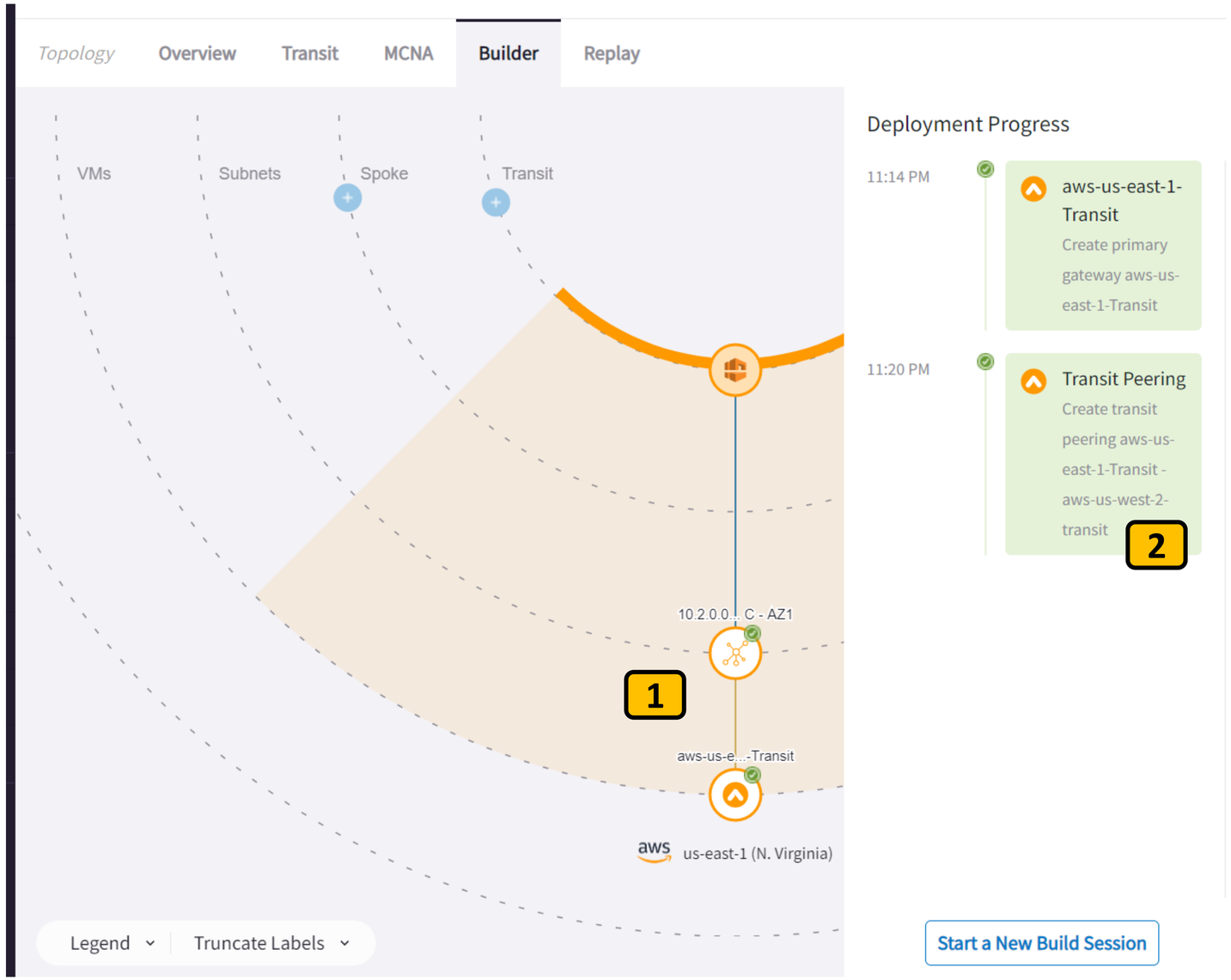


Lab 2: Cloud Backbone: Step 2.10

Use the Topology Builder to launch an Aviatrix Transit Gateway in a Transit VPC

Deploy Success will look like this. 1

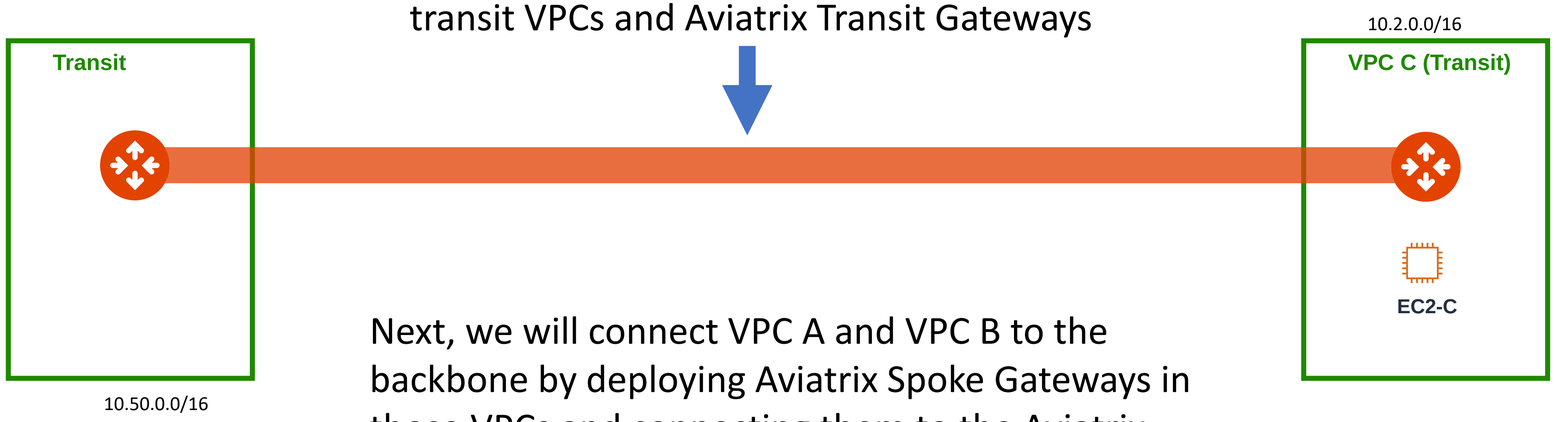
You have now successfully created a Transit VPC and Gateway in us-east-1 and peered it to the Transit VPC and Gateway in us-west-2 2



Lab 2: Cloud Backbone: Progress Check

Your backbone architecture now looks like *this*

You have peered the two AWS regions using transit VPCs and Aviatrix Transit Gateways



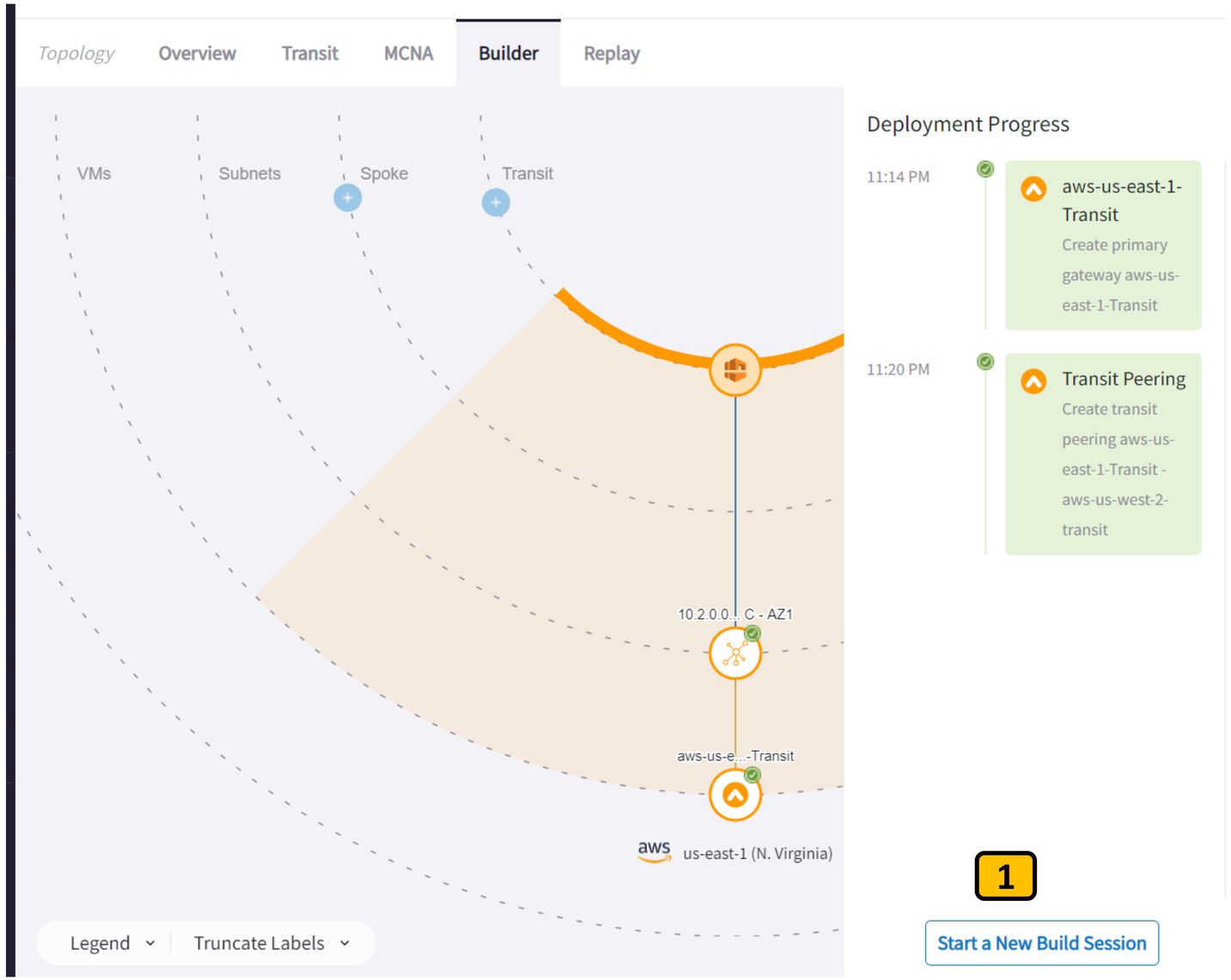
Next, we will connect VPC A and VPC B to the backbone by deploying Aviatrix Spoke Gateways in those VPCs and connecting them to the Aviatrix Transit Gateway in VPC C.

Lab 2: Cloud Backbone: Step 2.11

Use the Topology Builder to launch Aviatrix Spoke Gateways and connect them to Transit

Now let's deploy Aviatrix Spoke Gateways in VPC A and VPC B using the same Topology Builder tool.

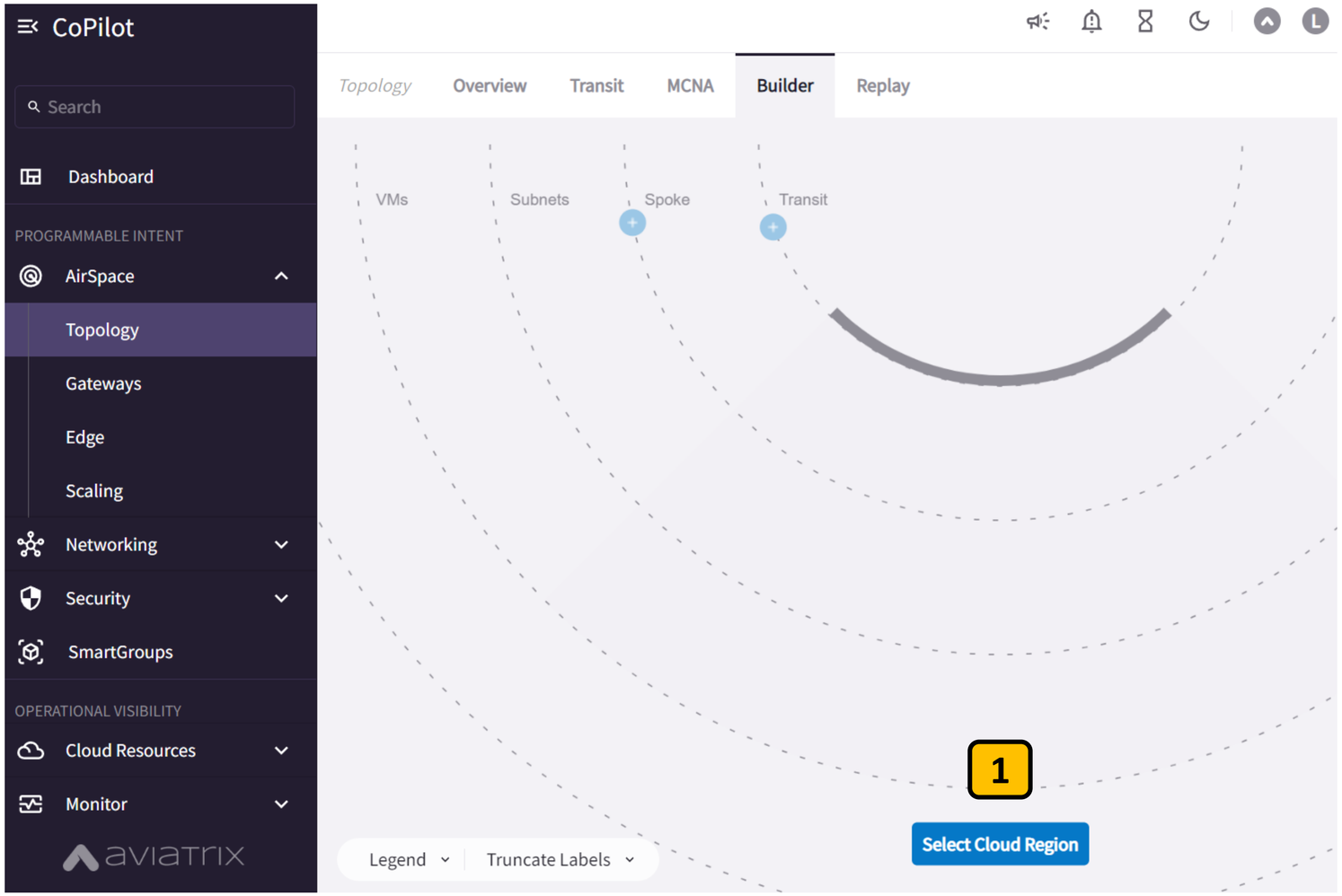
Click Start a New Build Session 1



Lab 2: Cloud Backbone: Step 2.12

Use the Topology Builder to launch Aviatrix Spoke Gateways and connect them to Transit

In the Topology Builder, click **Select Cloud Region** 1



Lab 2: Cloud Backbone: Step 2.13

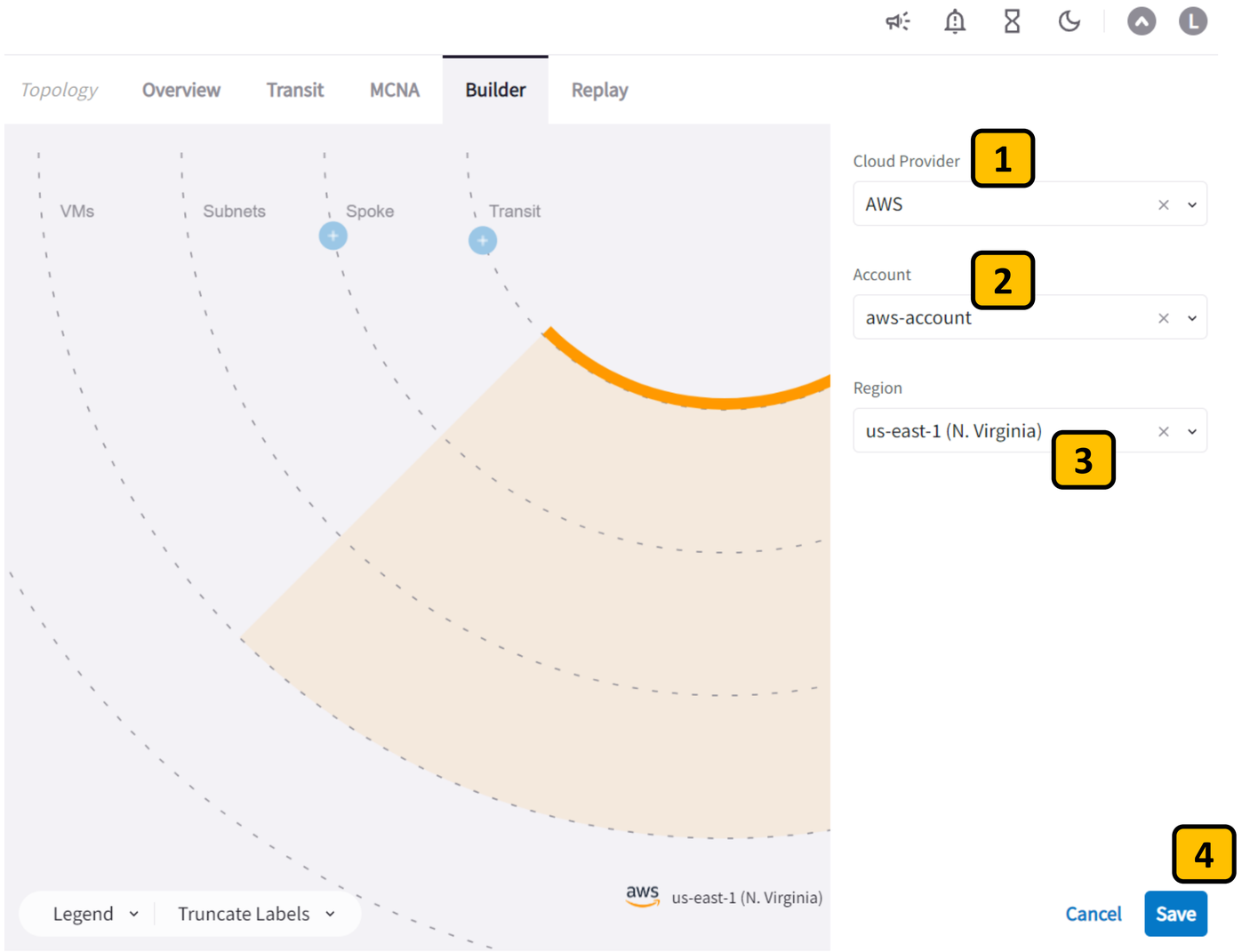
Use the Topology Builder to launch Aviatrix Spoke Gateways and connect them to Transit

Select the Cloud Provider: **AWS** 1

Select the AWS Account: **aws-account** 2

Select the Region: **us-east-1** 3

Click **Save** to start a new Build session in this cloud, account, and region. 4



Lab 2: Cloud Backbone: Step 2.14

Use the Topology Builder to launch Aviatrix Spoke Gateways and connect them to Transit

Click the **blue +** under Spoke 1

Select **VPC A** to be the Spoke VPC 2

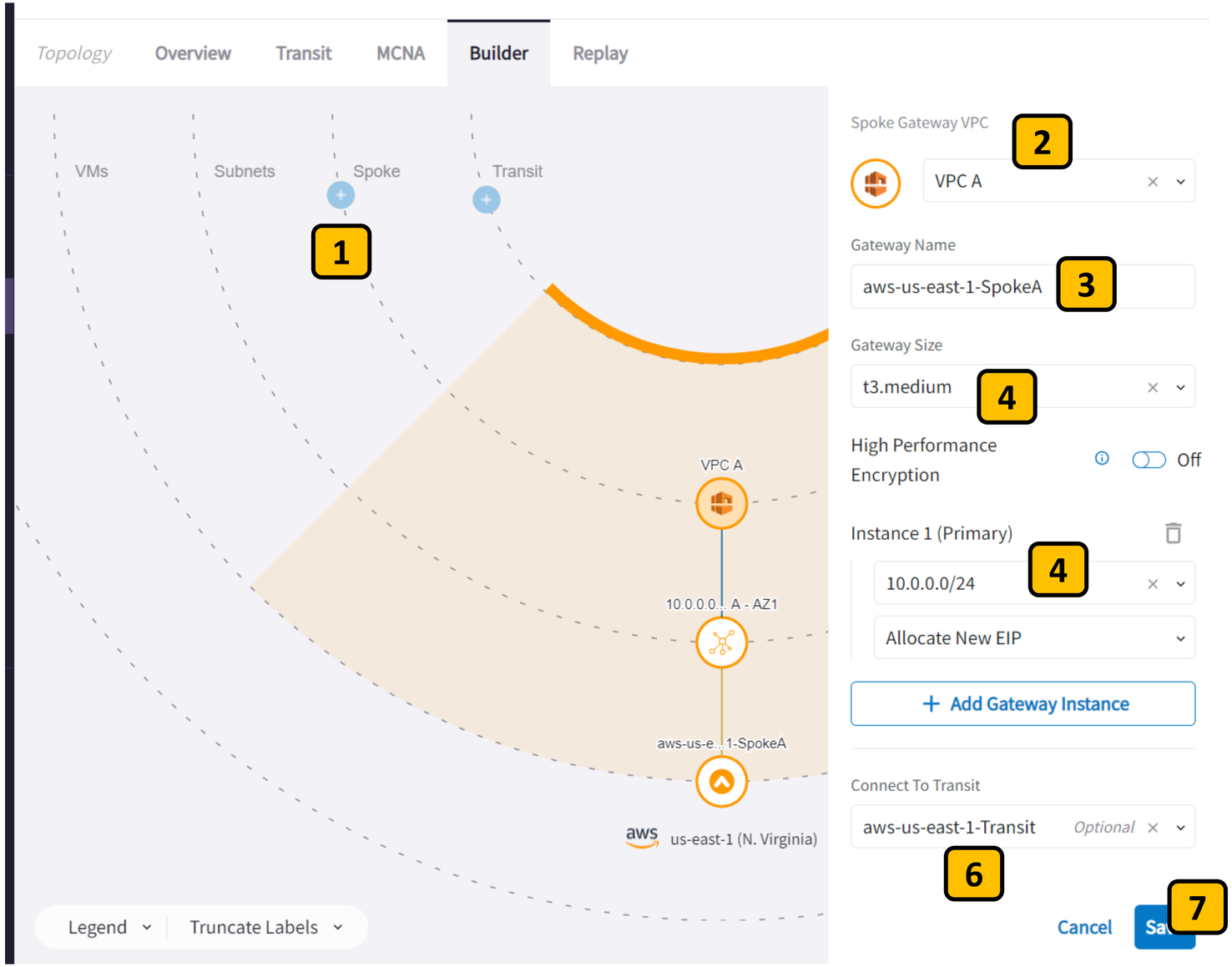
Name the Spoke gateway: **aws-us-east-1-SpokeA** 3

Select **t3.medium** for the gateway instance size 4

Select the 10.0.0.0/24 as the public subnet to deploy the gateway in to 5

Click Connect to Transit and select the aws-us-east-1-Transit. 6

Click Save 7



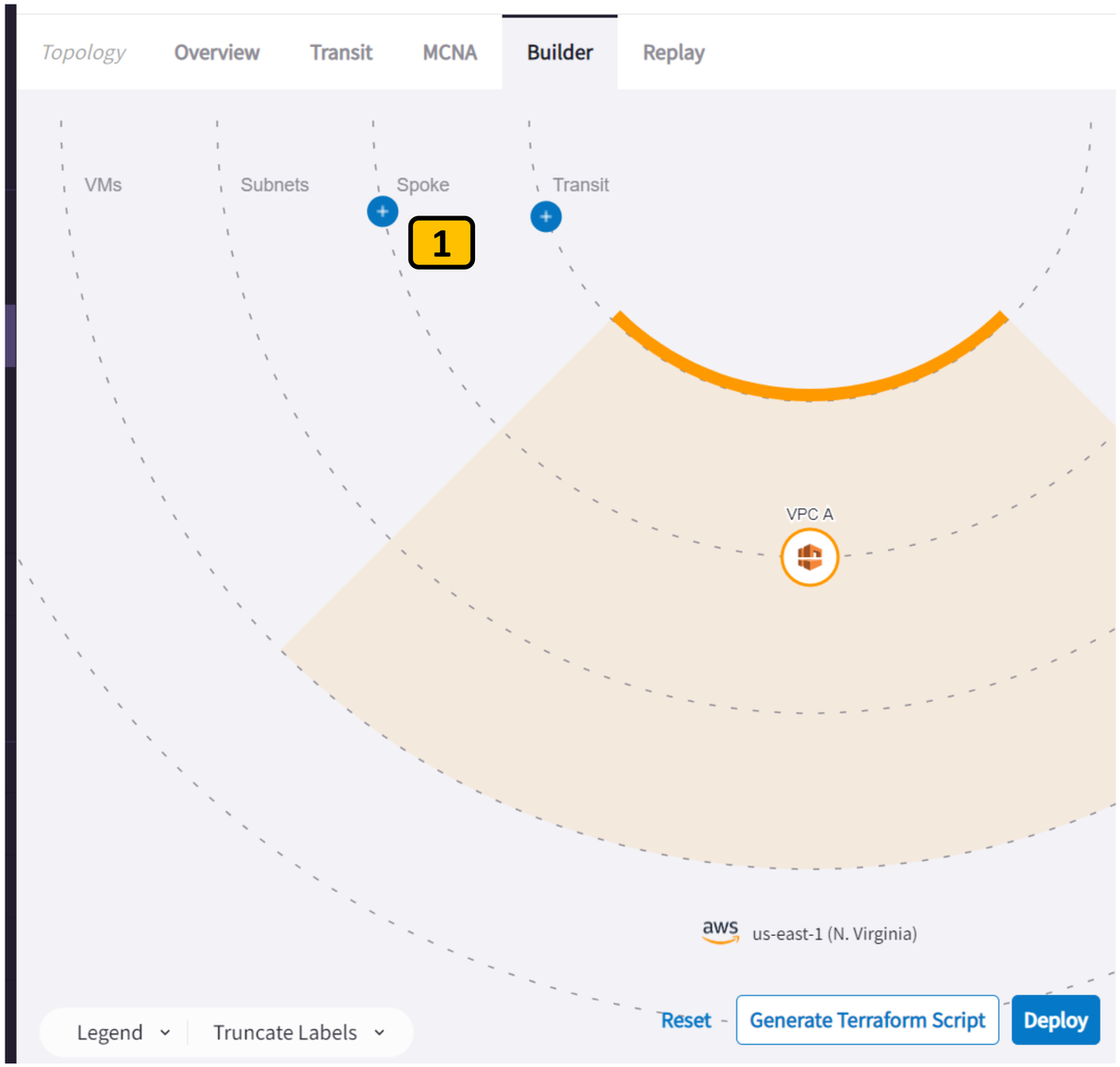
Lab 2: Cloud Backbone: Step 2.15

Use the Topology Builder to launch Aviatrix Spoke Gateways and connect them to Transit

Don’t click Deploy yet!

Let’s add VPC B to this build session to get another Spoke Gateway deployed all in the same build session.

Click the blue+ under Spoke



Lab 2: Cloud Backbone: Step 2.16

Use the Topology Builder to launch Aviatrix Spoke Gateways and connect them to Transit

Select **VPC B** to be a Spoke VPC 1

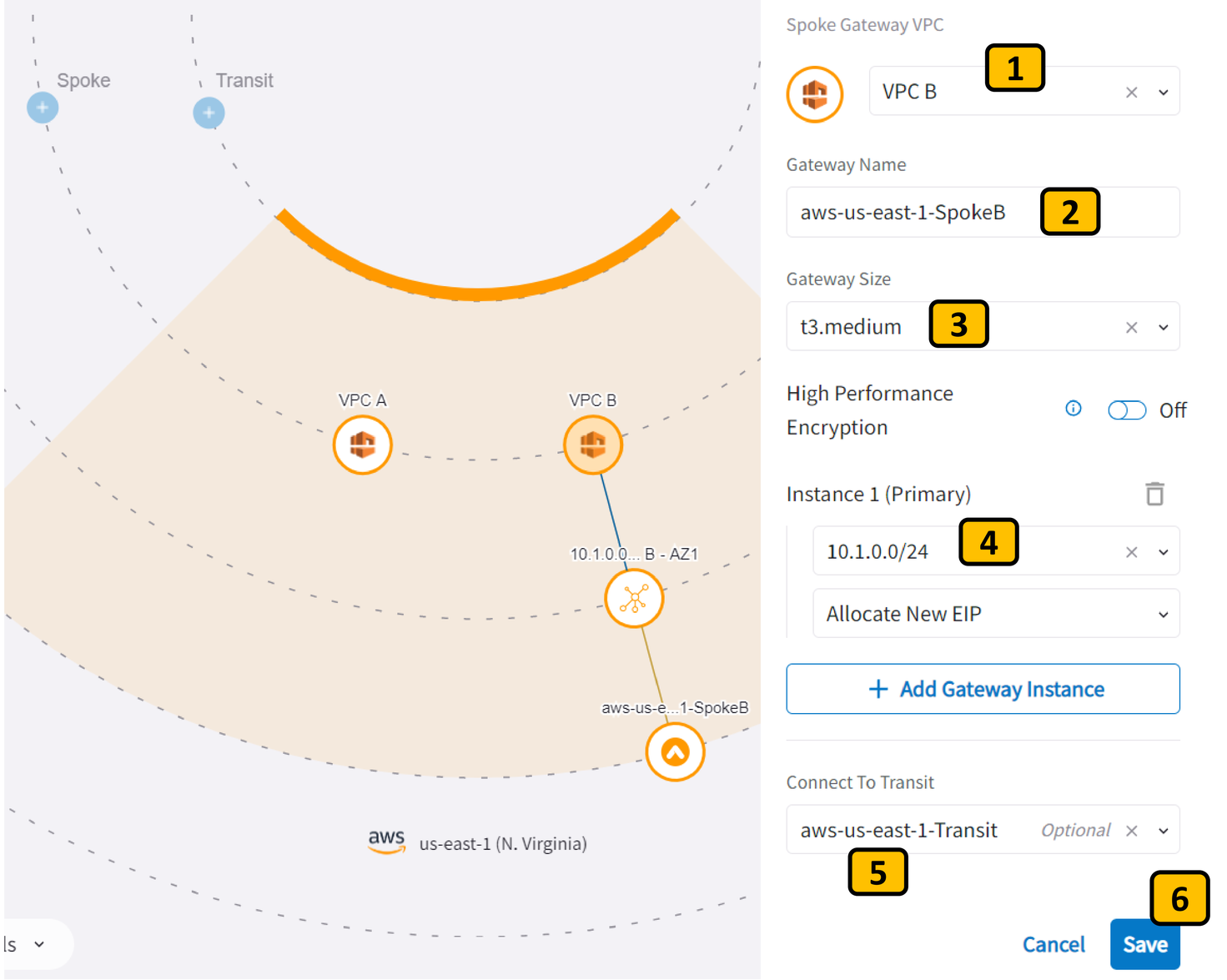
Name the Spoke gateway:
aws-us-east-1-SpokeB 2

Select **t3.medium** for the gateway instance size 3

Select the 10.1.0.0/24 as the public subnet to deploy the gateway in to 4

Click Connect to Transit and select the aws-us-east-1-Transit. 5

Click Save 6

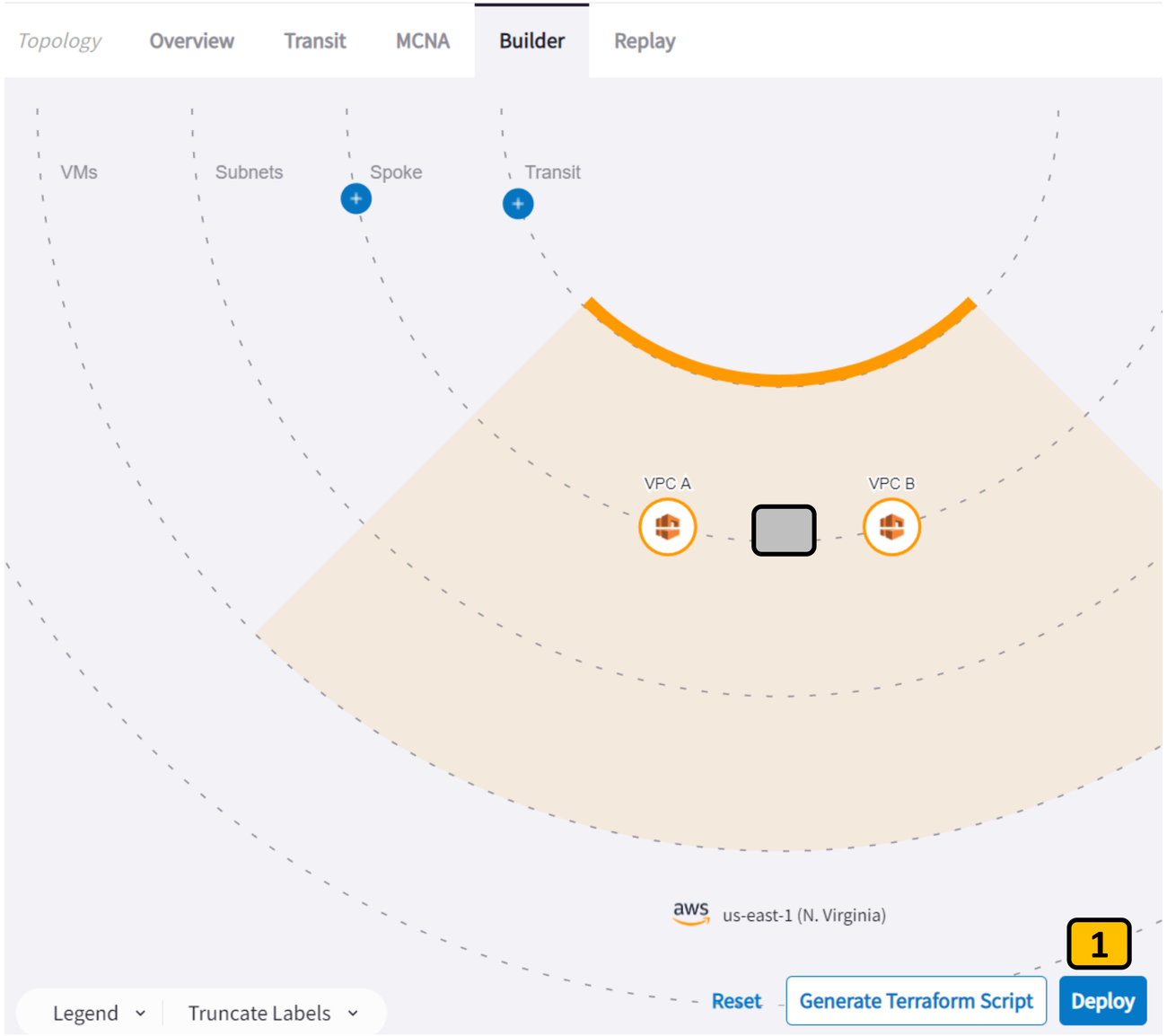


Lab 2: Cloud Backbone: Step 2.17

Use the Topology Builder to launch Aviatrix Spoke Gateways and connect them to Transit

Optional: If you want to double-check or change details of your Spoke gateway deployment settings, click the VPC A or VPC B icons

Click Deploy to begin your Spoke VPC deployment in us-east-1 1



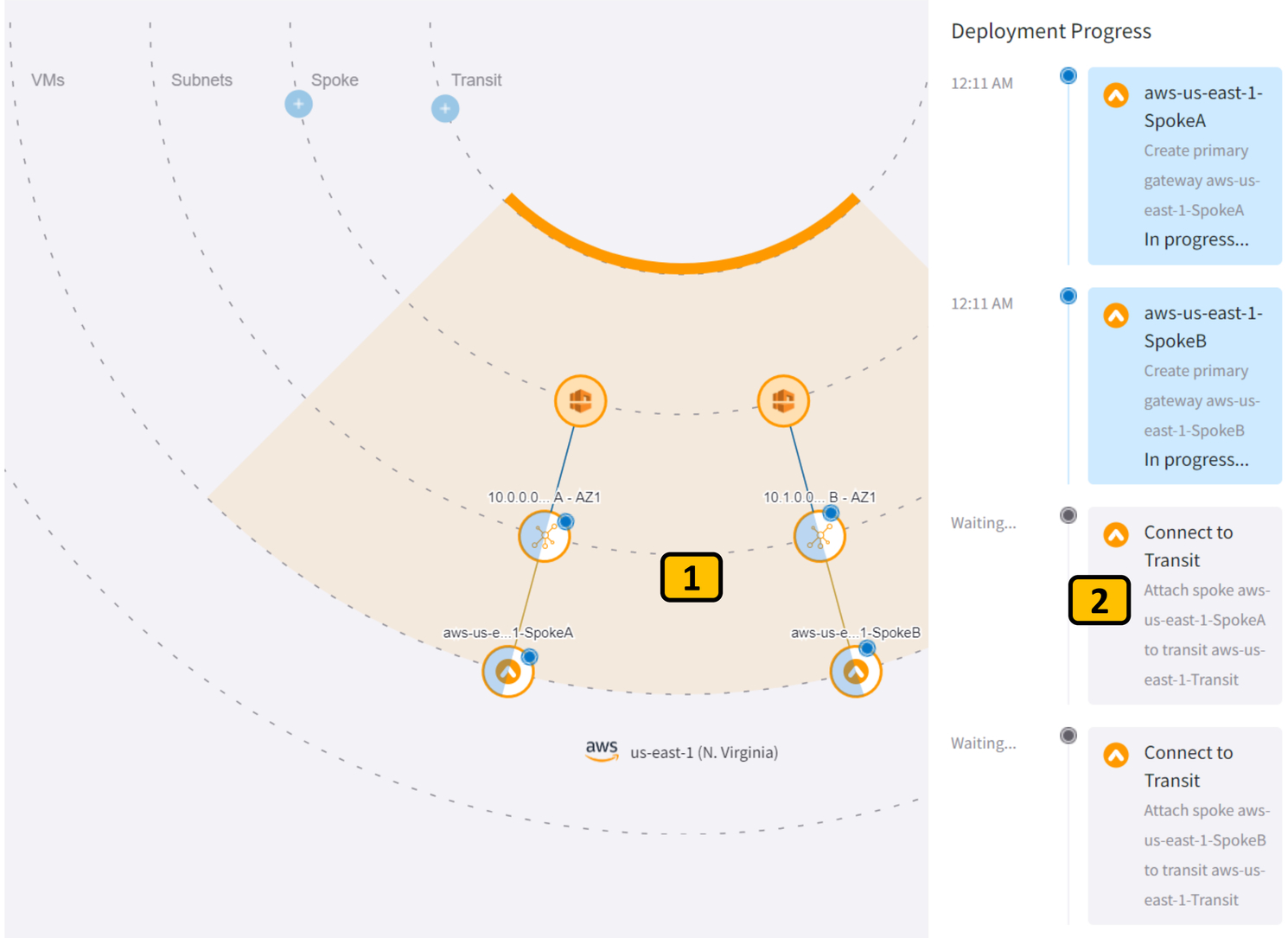
Lab 2: Cloud Backbone: Step 2.18

Use the Topology Builder to launch Aviatrix Spoke Gateways and connect them to Transit

Observe the Deployment Progress of your Aviatrix Spoke Gateways in us-east-1. **1**

After the Spoke gateways are deployed, CoPilot will connect them to the Aviatrix Transit gateway you created earlier. **2**

The total deployment time should take about 5 minutes.



Lab 2: Cloud Backbone: Step 2.19

Enable Connected Transit

Go to AirSpace > Gateways > Transit Gateways

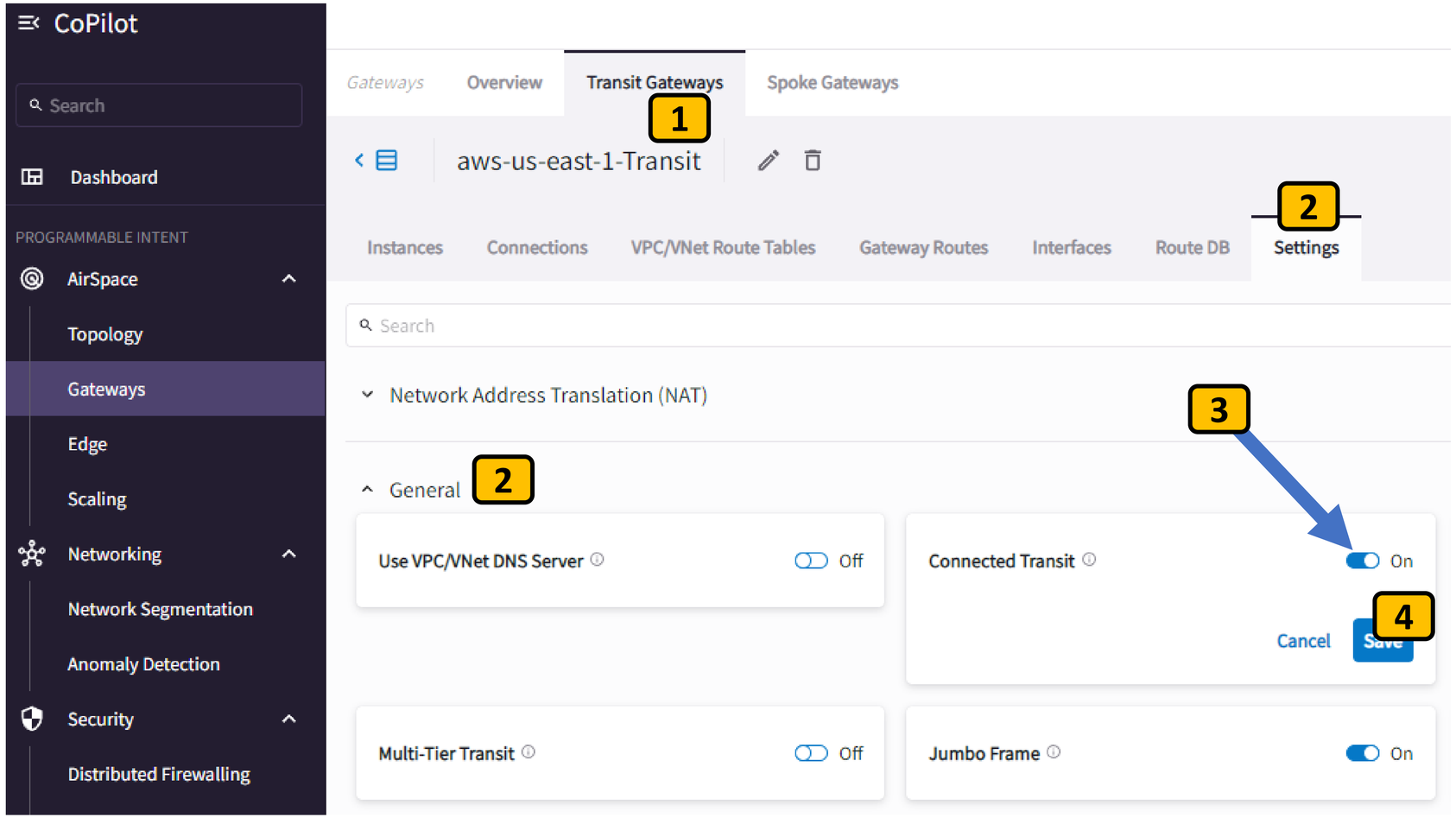
Click on the Transit Gateway **aws-us-east-1-Transit** 1

Select **Settings** 2

Toggle **Connected Transit** to On 3

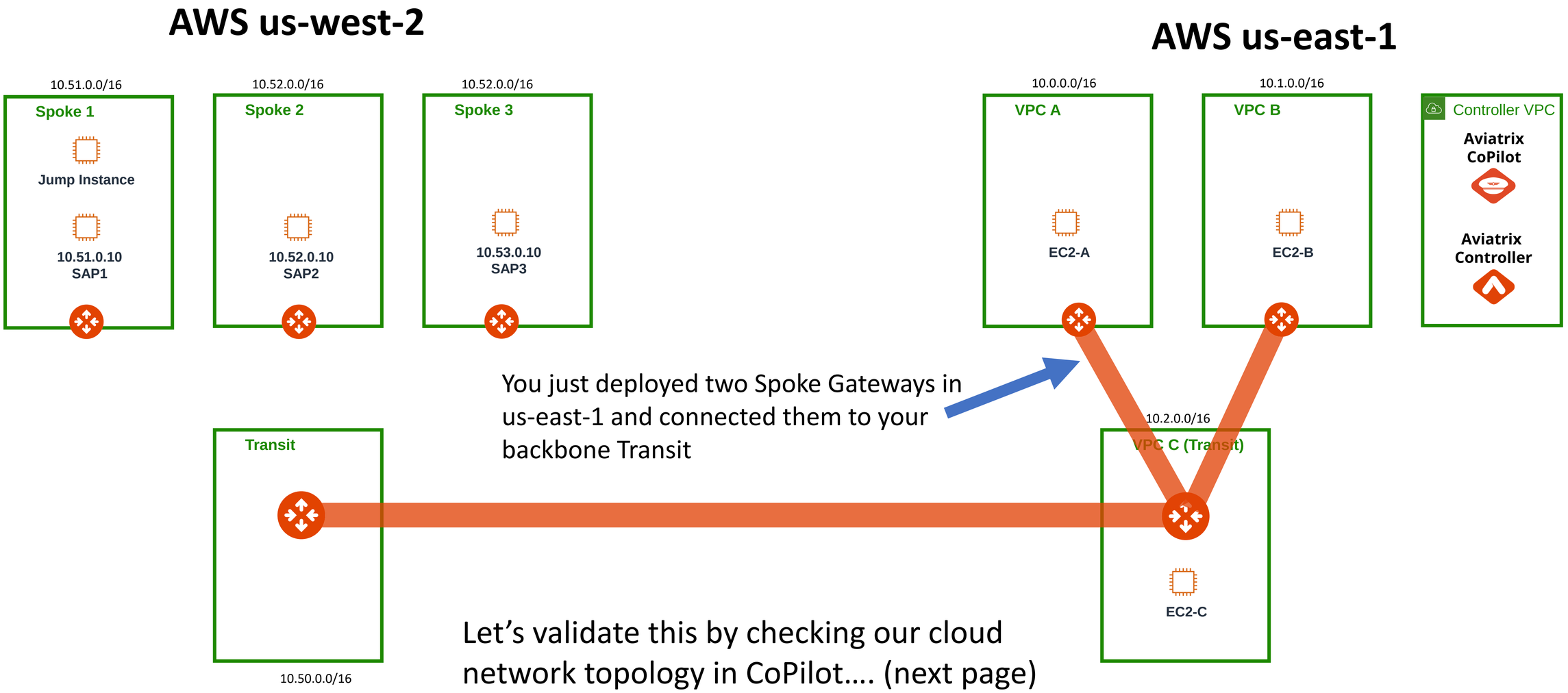
Click Save 4

This tells the Transit Gateway that it can forward traffic between Spoke Gateways



Lab 2: Cloud Backbone: Progress Check

Your backbone architecture now looks like *this*



You just deployed two Spoke Gateways in us-east-1 and connected them to your backbone Transit

Let's validate this by checking our cloud network topology in CoPilot…. (next page)

Lab 2: Cloud Backbone: Step 2.20

Check your cloud network topology in Aviatrix CoPilot

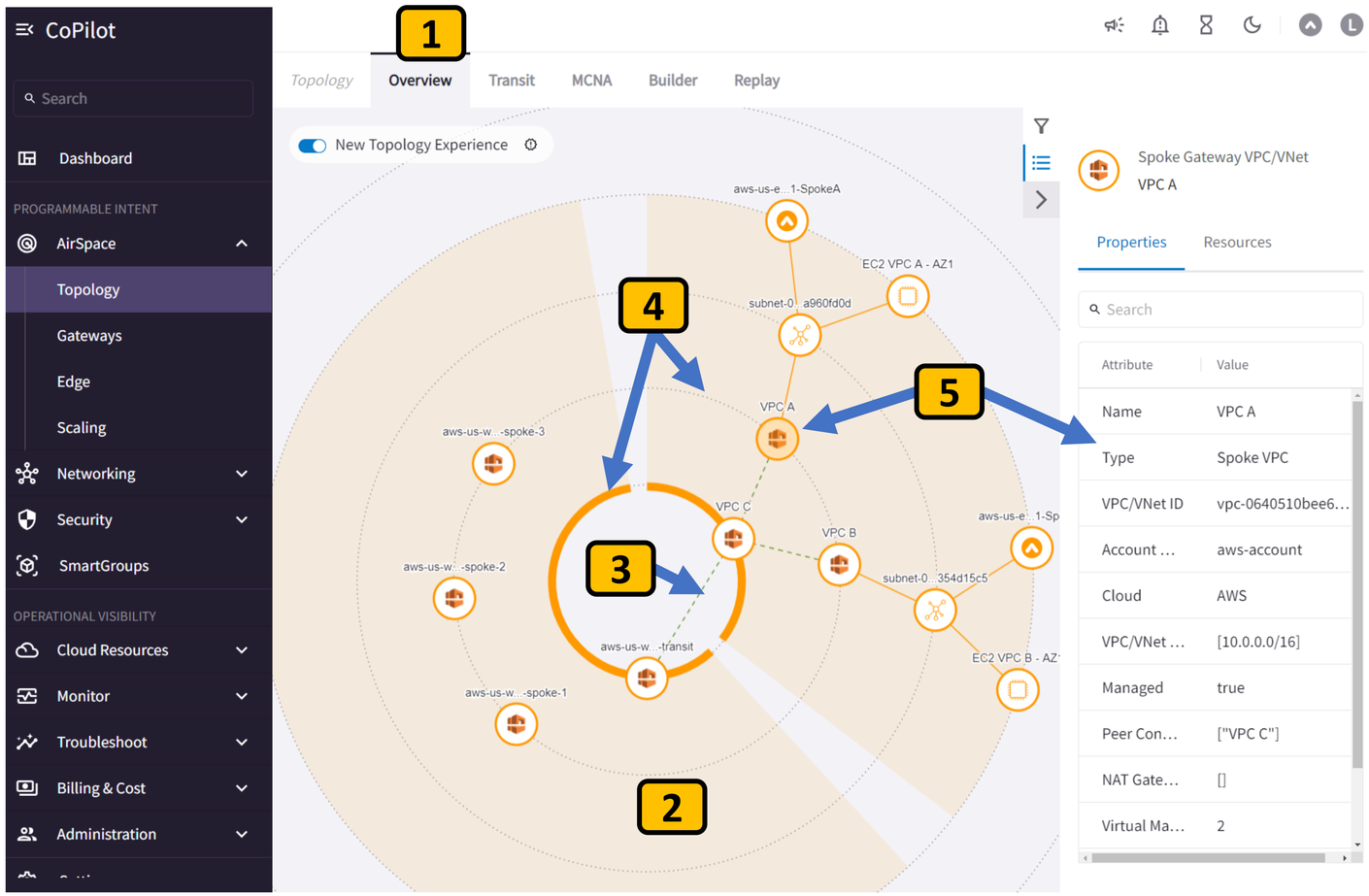
Go to AirSpace > Topology > Overview 1

Each piece of orange pie represents an AWS region and the resources inside it 2

Your backbone is represented by the dashed line 3

The inner most circle of your topology represents Transit VPCs. The second circle represents Spoke VPCs. 4

Click on any VPC to expand (or hide) its contents and see details in the right-side panel. 5



NOTE: If you don't see your EC2 instances from Lab 1 in your topology, see the next slide for steps to resolve that.

Lab 2: Cloud Backbone: Step 2.20b

Set your Fetch Instance task frequency in CoPilot

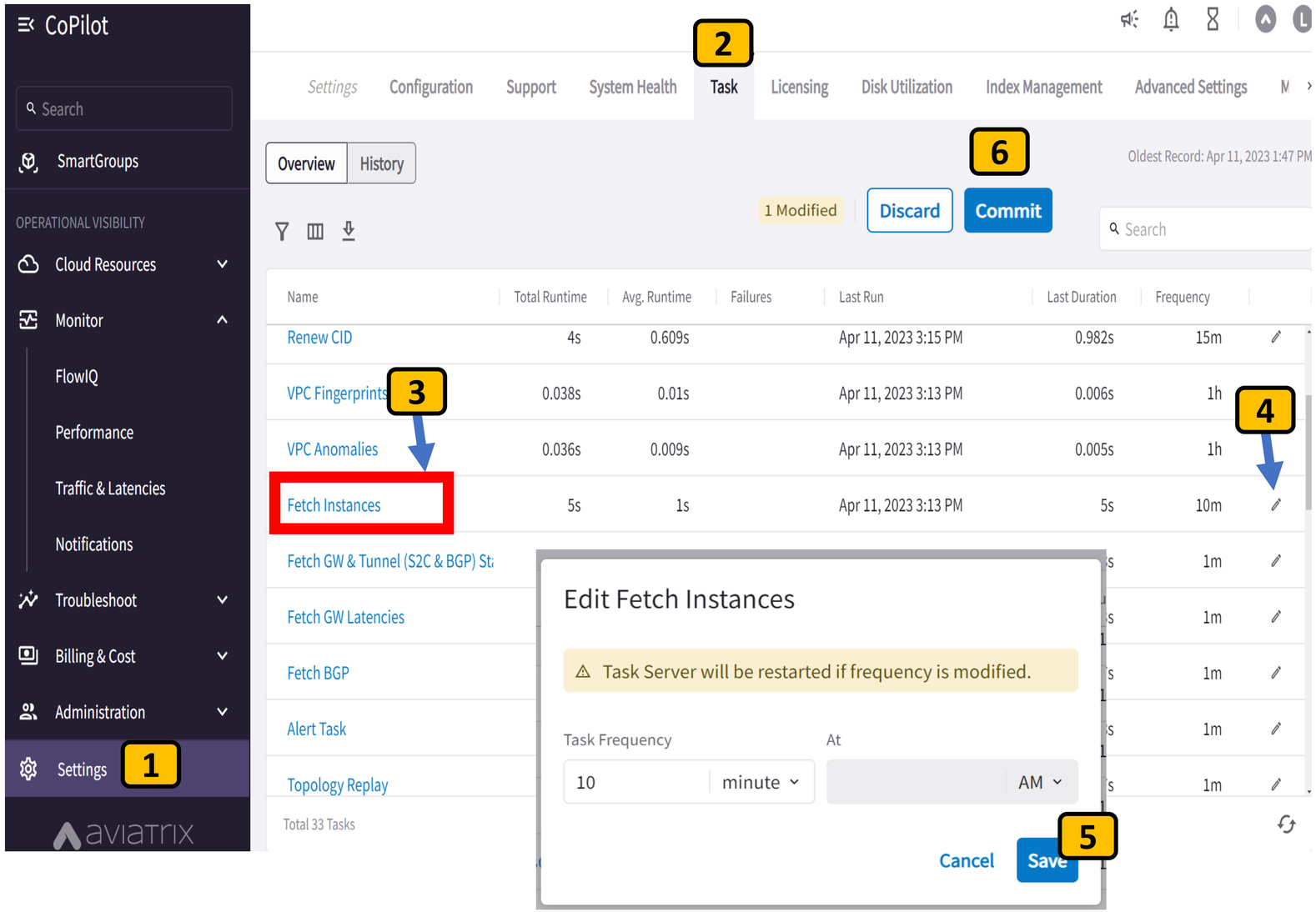
If you are not able to see your EC2 instances from Lab 1 in the CoPilot topology above…

Navigate to Settings 1
Select Task 2

Find the Fetch Instances task 3
Click the Pencil to edit this task 4

Set the task to run every 10 minutes and click Save 5

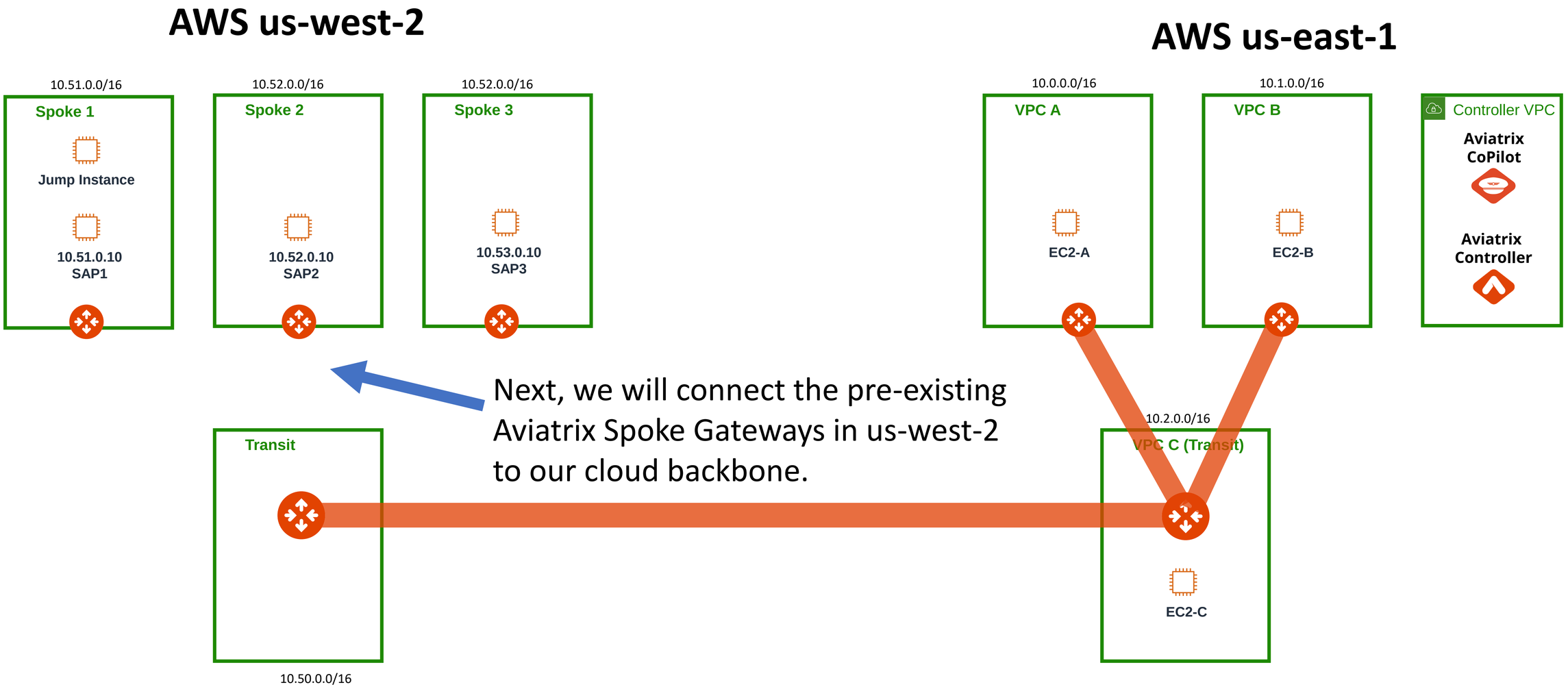
Be sure to Commit your change 6



This will cause CoPilot to fetch your instances every 10 minutes, starting Now.

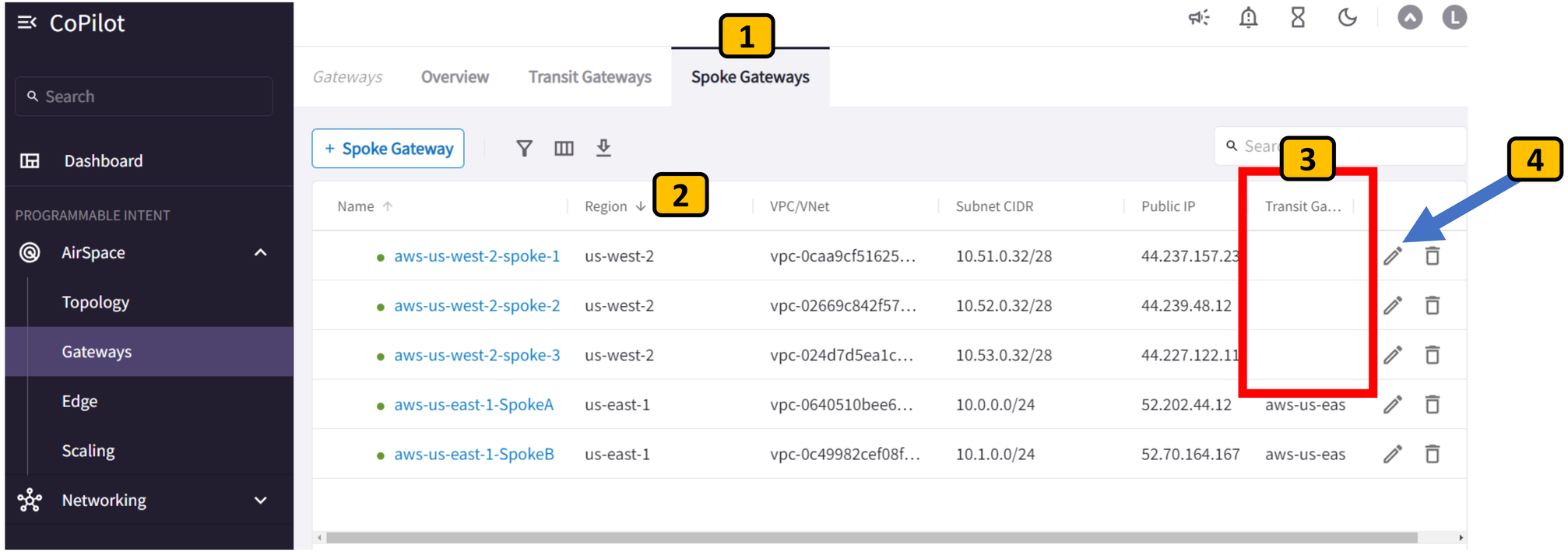
Lab 2: Cloud Backbone: Progress Check

Your backbone architecture now looks like *this*



Lab 2: Cloud Backbone: Step 2.21

Connect Spoke Gateways to Transit using CoPilot



Go to AirSpace > Gateways > Spoke Gateways 1

Sort by Region 2

Notice your us-west-2 Spoke gateways are not connected to a Transit 3

Select the edit Pencil for the Spoke Gateway aws-us-west-2-spoke-1 4

Lab 2: Cloud Backbone: Step 2.22

Connect Spoke Gateways to Transit using CoPilot

On the Edit pop-up window select: Attach to Transit Gateway 1

Select the **aws-us-west-2-transit** 2

FYI: If you wanted to add an active-active HA Spoke Gateway, you can do that here. 3
(don't do that now as our lab EIP limits are already used up)

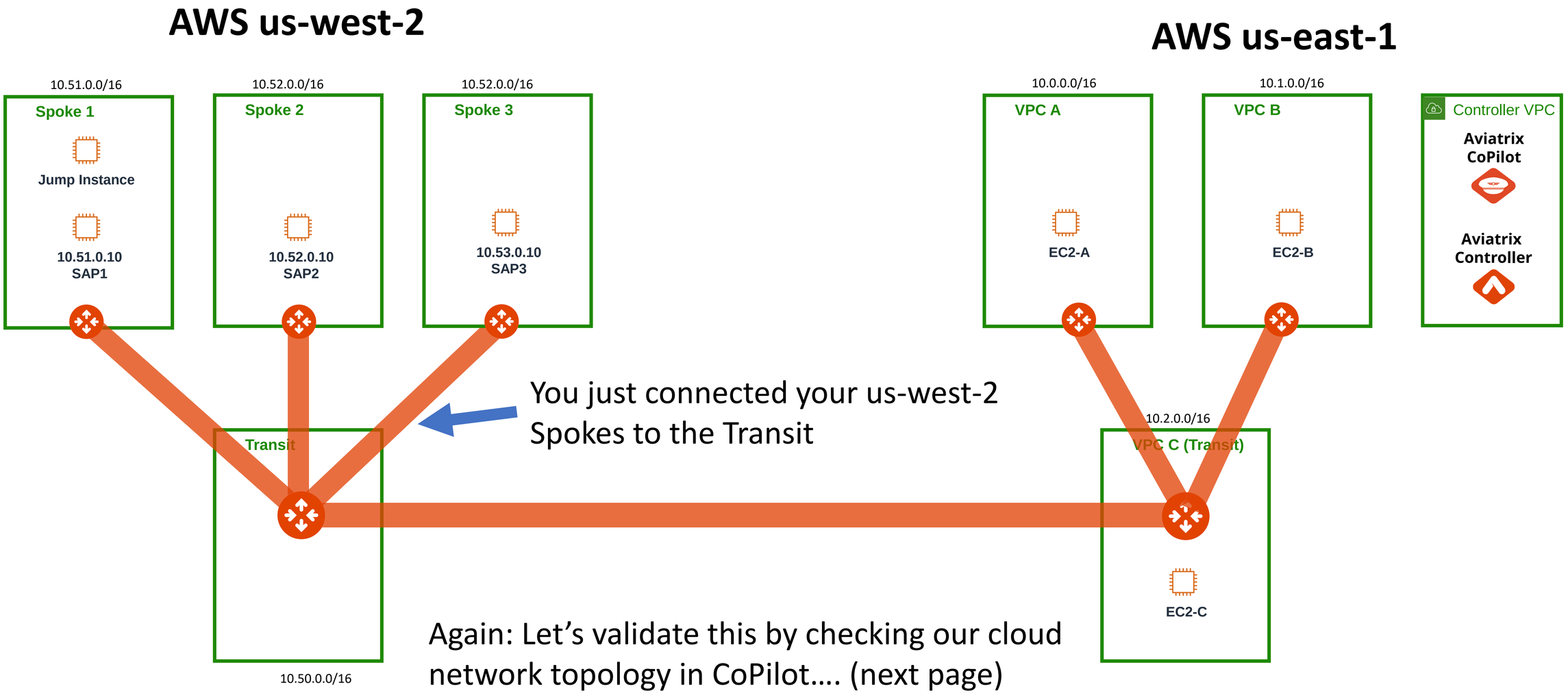
Click Save 4

Repeat this process for the remaining Spoke Gateways in us-west-2

Easy, right?

Lab 2: Cloud Backbone: Progress Check

Your backbone architecture now looks like *this*



You just connected your us-west-2 Spokes to the Transit

Again: Let's validate this by checking our cloud network topology in CoPilot…. (next page)

Lab 2: Cloud Backbone: Step 2.23

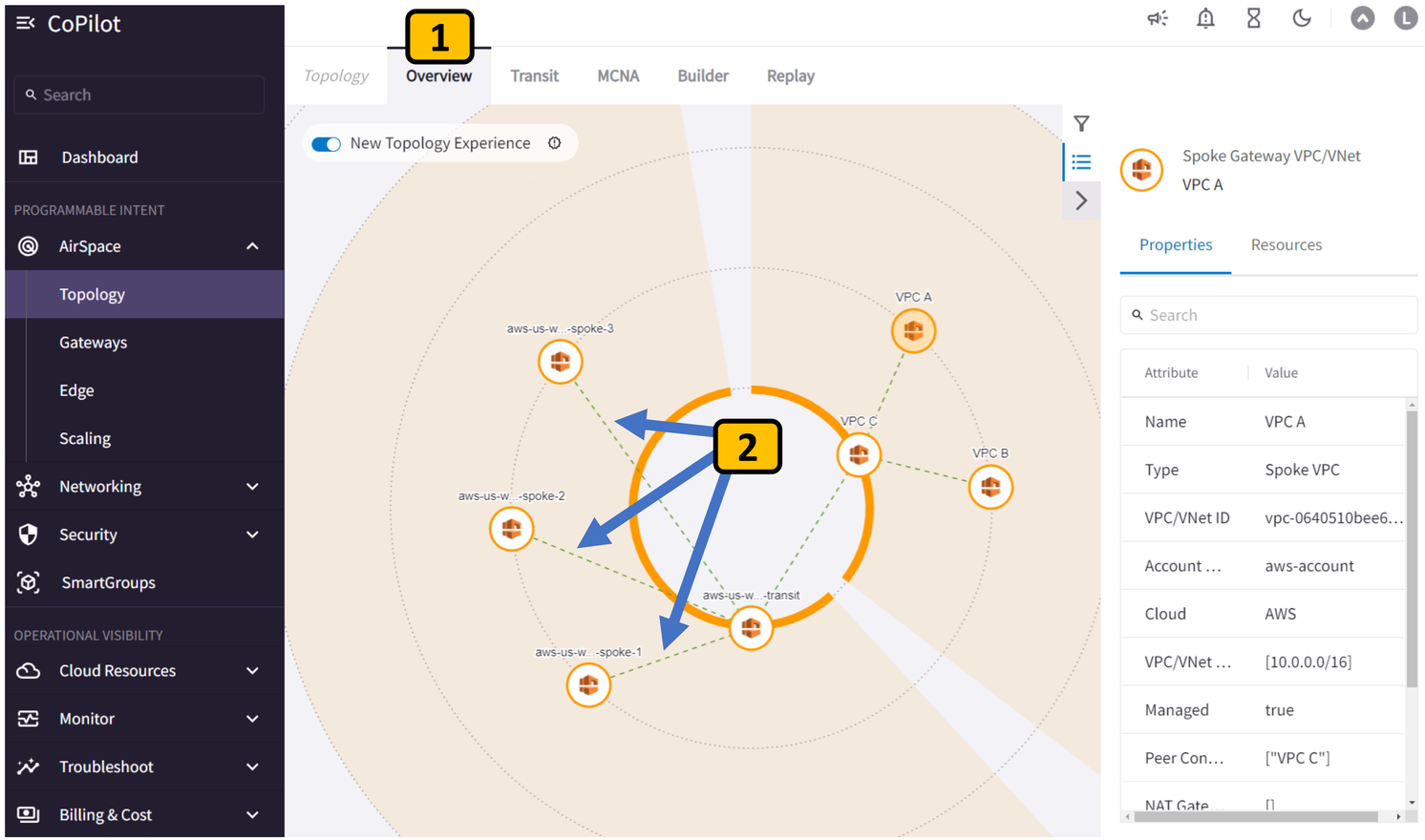
Check your cloud network topology in CoPilot

Go to AirSpace > Topology > Overview 1

Notice your us-west-2 Spokes are now connect to your Transit in us-west-2 2

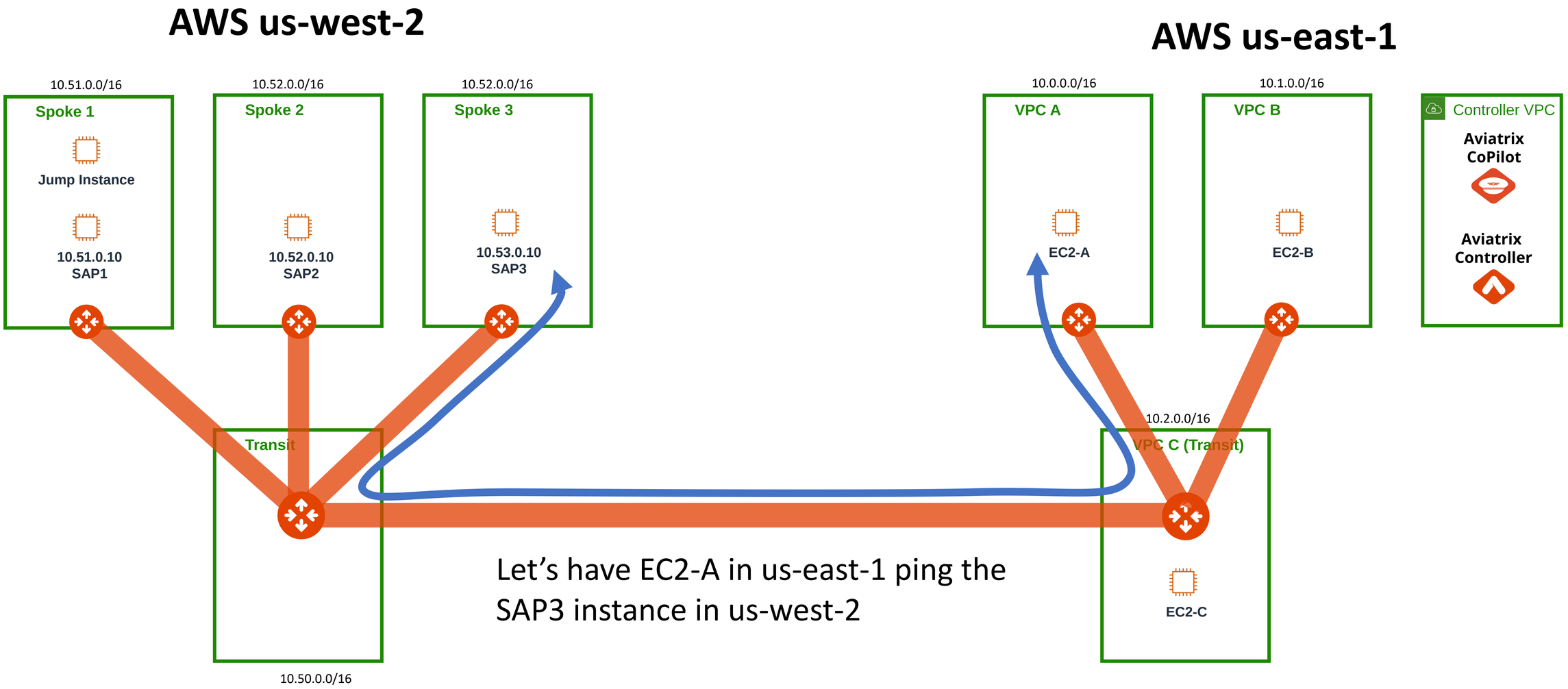
Success! We have our multi-region cloud network backbone!

Now let's test the connectivity and observe network traffic …(next)



Lab 2: Cloud Backbone: Test

Test Connectivity



Let's have EC2-A in us-east-1 ping the SAP3 instance in us-west-2

Lab 2: Cloud Backbone: Step 2.24

Access the console of instance EC2-A in us-east-1

From the AWS Console make sure you're in the correct region 1

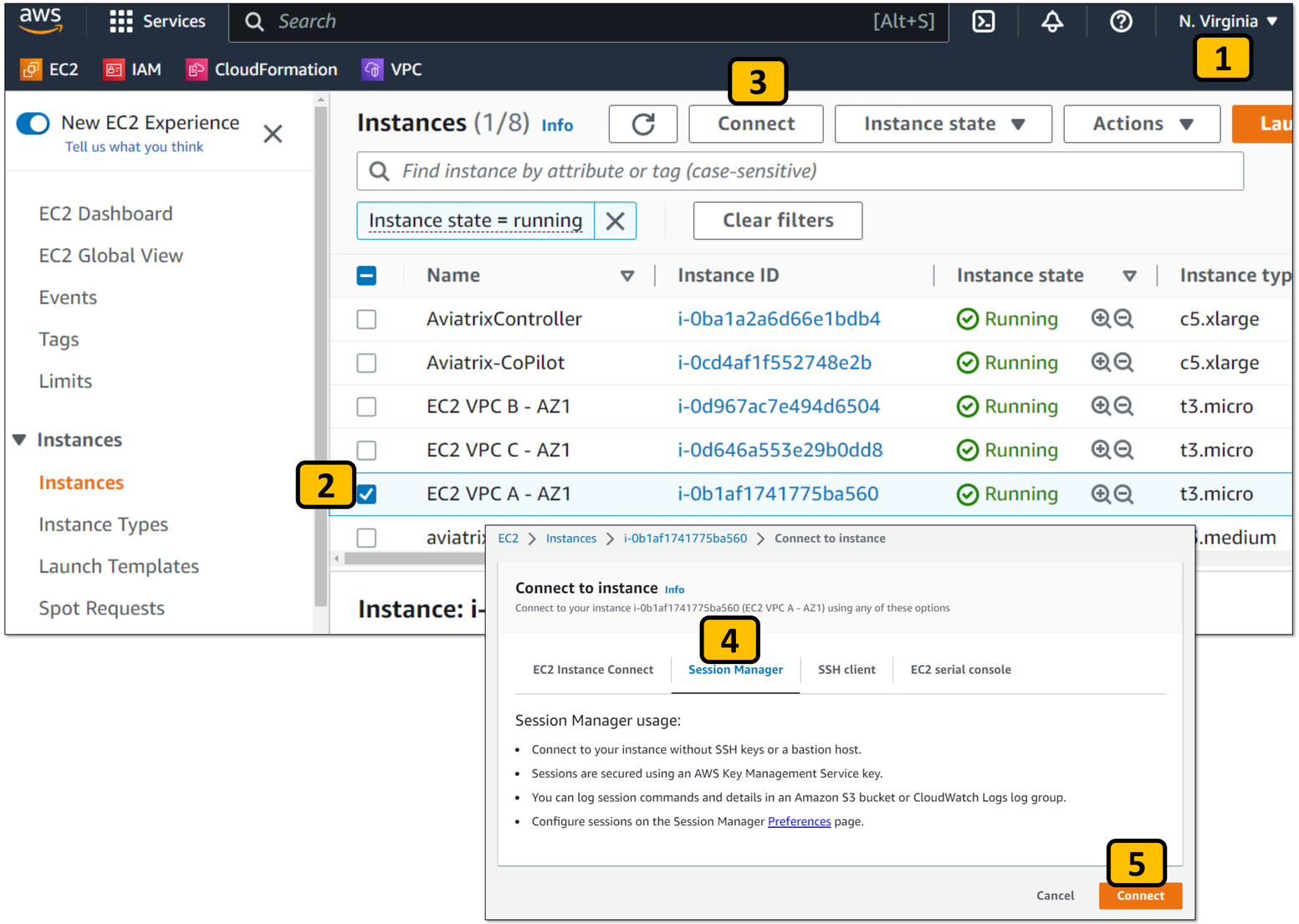
Select the **EC2 VPC A – AZ1** instance 2

Click on **Connect** 3

Select the **Session Manager** tab 4

Click on **Connect** 5

This will open a console on that instance where we can ping… (next)



Lab 2: Cloud Backbone: Step 2.25

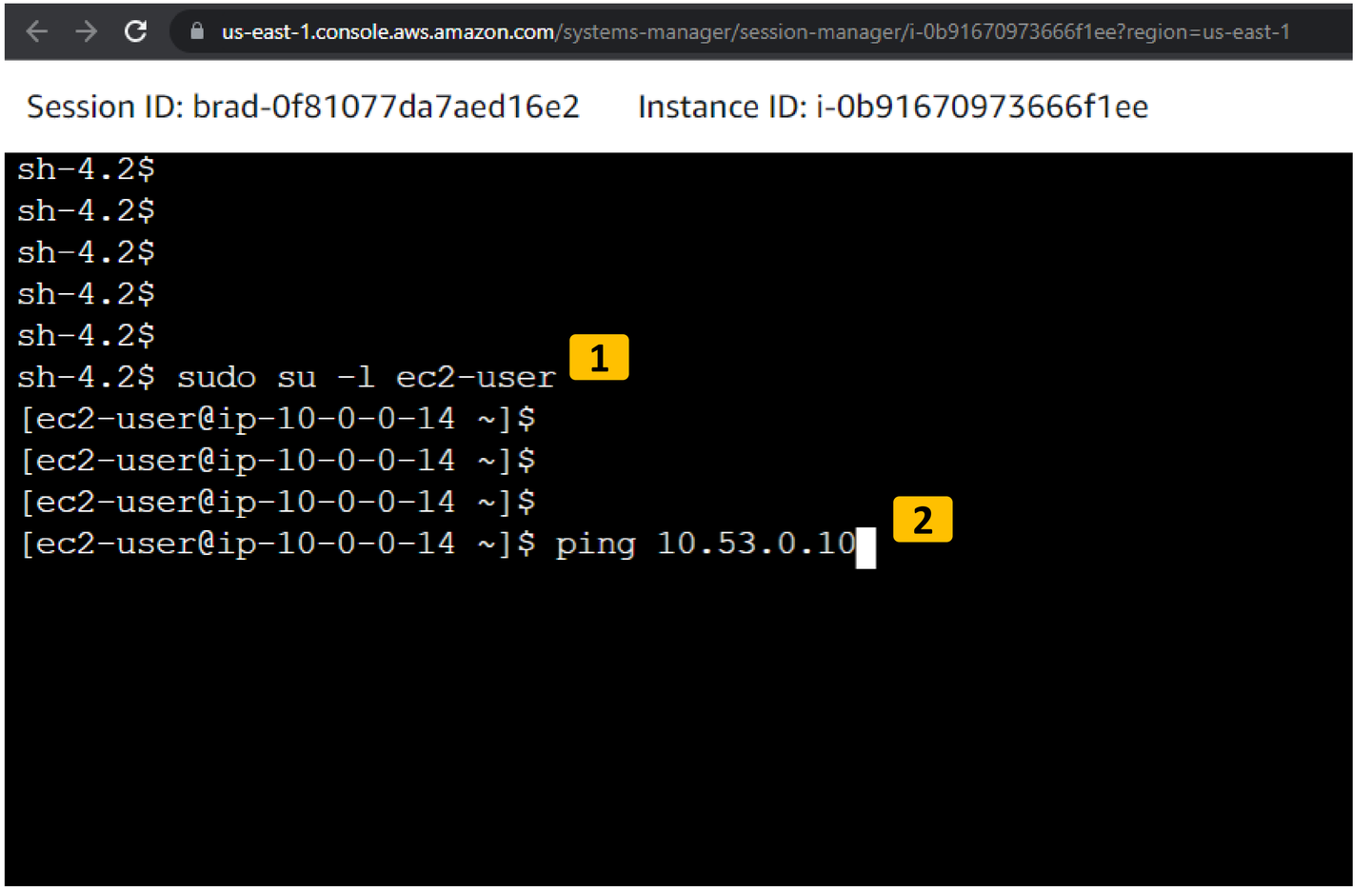
Ping test from EC2-A to SAP3

From the instance console type:
sudo su –l ec2-user 1

(that's a dash lowercase L)

Now let's try to ping SAP3 instance in the other region…

Enter the command:
ping 10.53.0.10 2



Lab 2: Cloud Backbone: Step 2.26

Did the ping work?

OH SNAP! It's not working!? 1

This SHOULD be working

Major bummer. Now we must manually analyze every route table and security group configuration, right?

No, we don't! **Let's use CoPilot to do the troubleshooting for us!** (next)

us-east-1.console.aws.amazon.com/systems-manager/session-manager/i-0b91670973666f1ee?region=us-east-1

Session ID: brad-0f81077da7aed16e2 Instance ID: i-0b91670973666f1ee

```
sh-4.2$
sh-4.2$
sh-4.2$
sh-4.2$
sh-4.2$
sh-4.2$
sh-4.2$ sudo su -l ec2-user
Last login: Mon Feb 27 05:54:11 UTC 2023 on pts/0
[ec2-user@ip-10-0-0-14 ~]$
[ec2-user@ip-10-0-0-14 ~]$
[ec2-user@ip-10-0-0-14 ~]$
[ec2-user@ip-10-0-0-14 ~]$
[ec2-user@ip-10-0-0-14 ~]$
[ec2-user@ip-10-0-0-14 ~]$ ping 10.53.0.10
PING 10.53.0.10 (10.53.0.10) 56(84) bytes of data.
```

1

Lab 2: Cloud Backbone: Step 2.27

Troubleshoot connectivity problems with AppIQ in CoPilot

Let's use the AppIQ troubleshooting tool in Copilot to find the problems.

Go to Troubleshoot > AppIQ > **FlightPath** 1

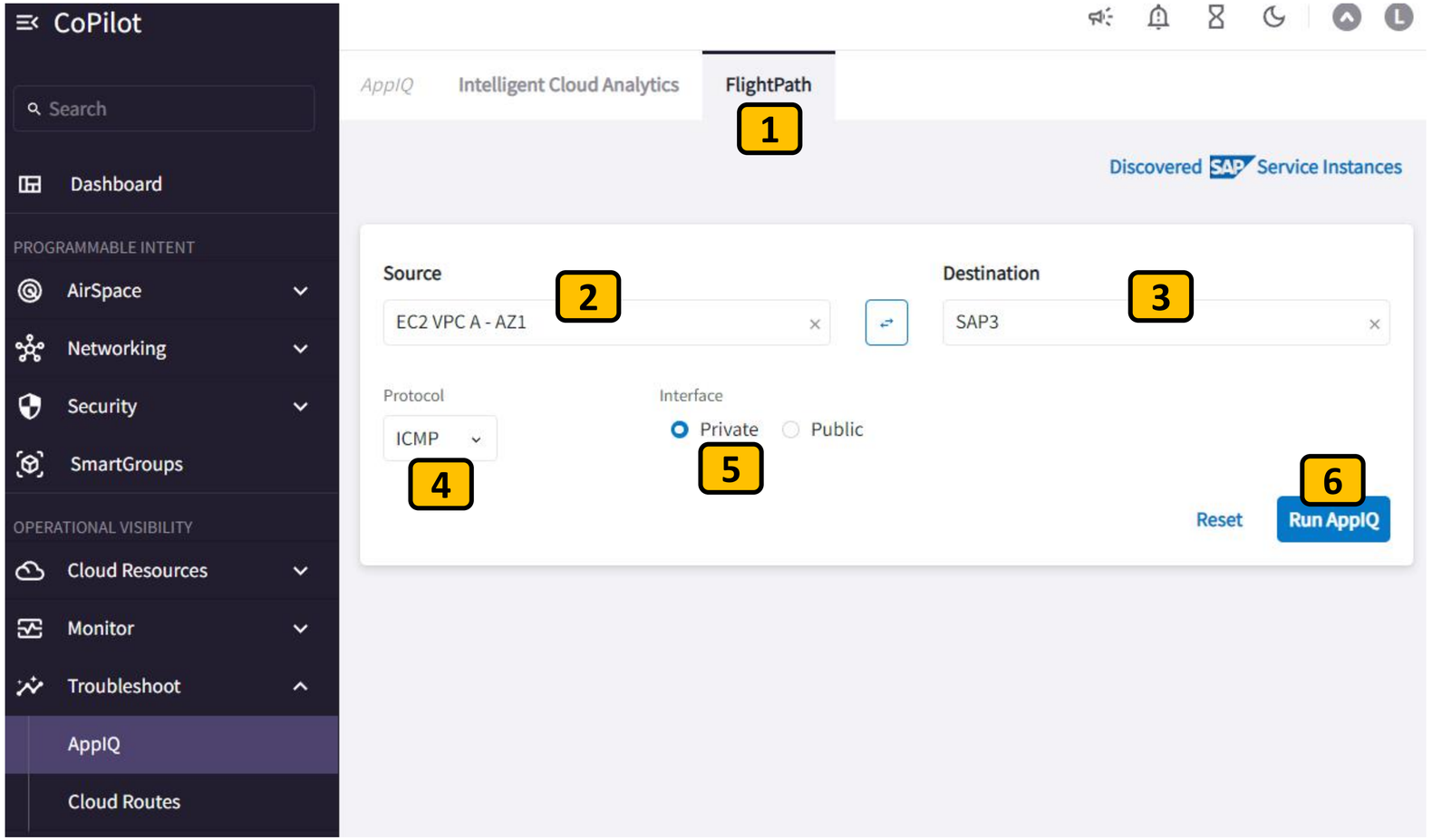
Select **EC2 VPC A – AZ1** as the *Source* instance 2

Select **SAP3** as the *Destination* instance 3

For Protocol select **ICMP** 4

Select **Private** 5

Click **Run AppIQ** 6



CoPilot will now analyze all gateways, route tables, security groups, and NACLs for ICMP to succeed between these two instances

Lab 2: Cloud Backbone: Step 2.28a

View the AppIQ report to find the problem

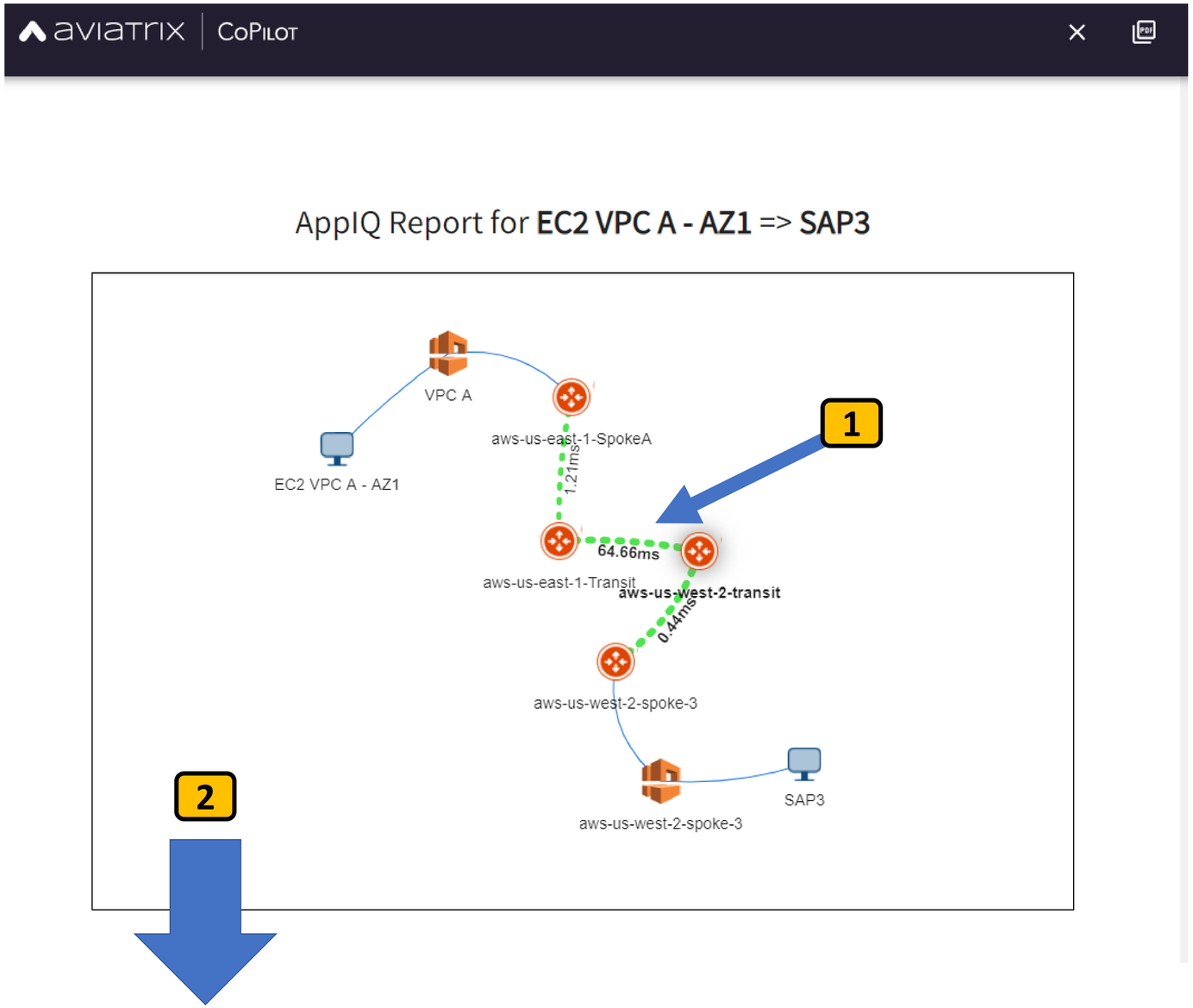
The AppIQ report gives a bunch of useful information.

The AppIQ report begins by showing us the network topology between these two instances, including link health and latency. 1

Green lines between the Aviatrix Gateways means their connection to each other is good.

So far so good.

Scroll down to see the rest of the report 2



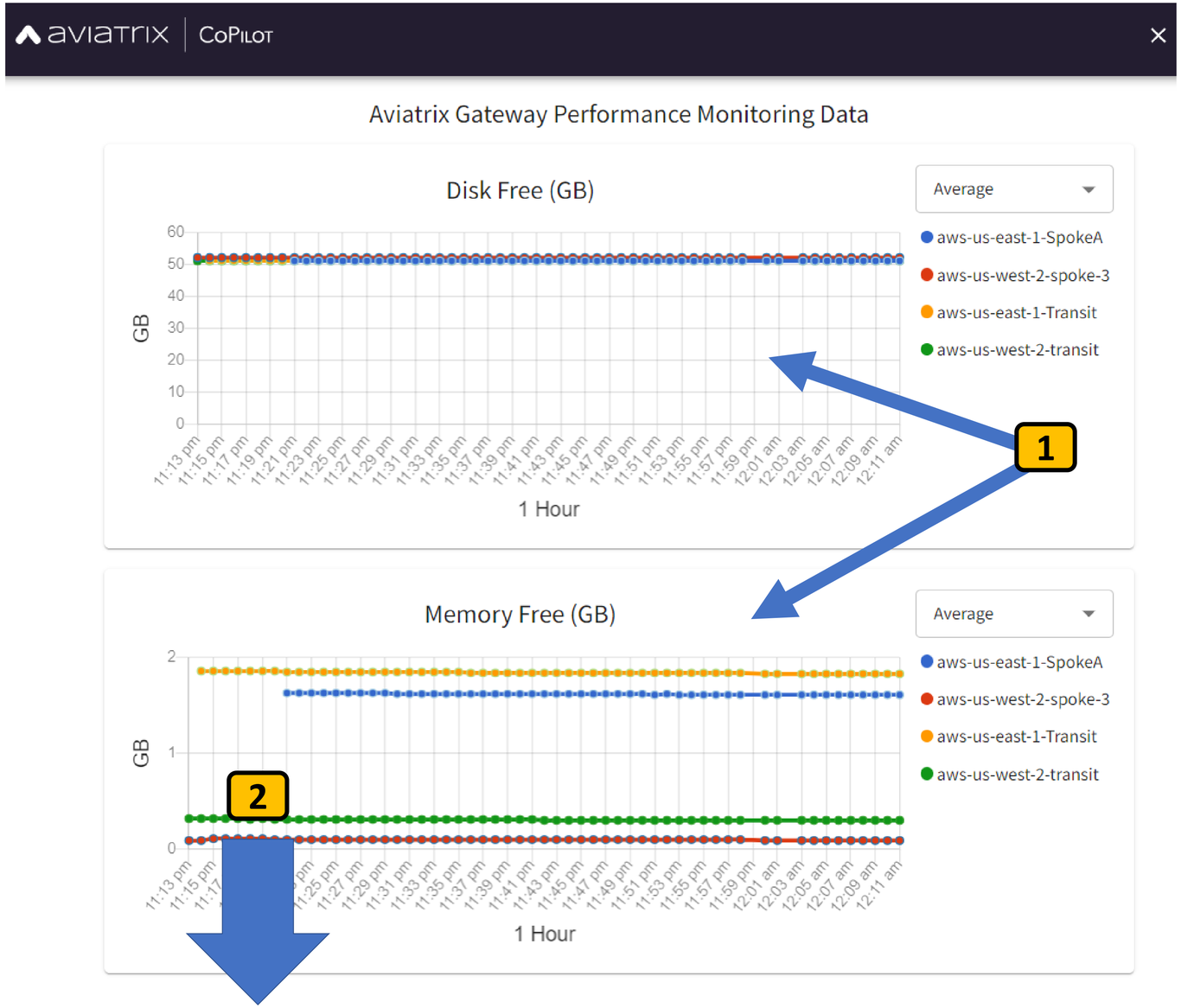
Lab 2: Cloud Backbone: Step 2.28b

View the AppIQ report to find the problem

Next, AppIQ will give us a **ton of metrics concerning the health and utilization** of the Aviatrix Gateways carrying the traffic between these two instances.

So far so good..

Scroll down to see the rest of the report 2



Lab 2: Cloud Backbone: Step 2.28c

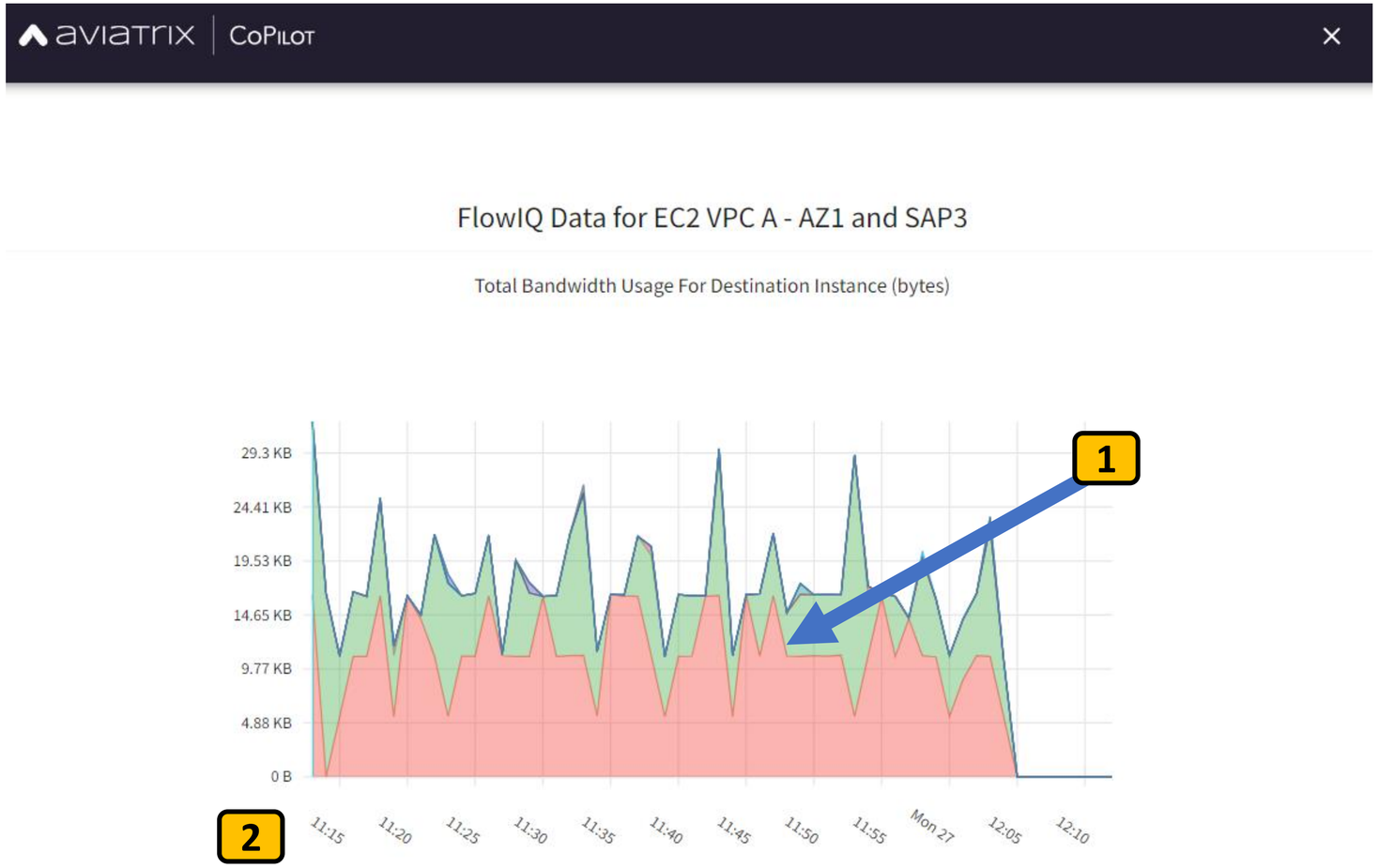
View the AppIQ report to find the problem

Next, AppIQ will **search the network traffic history** sent from the Source and Destination instances.

Have they been talking to ANYBODY? 1

Yes they have …
So far so good..

Scroll down to see the rest of the report



Lab 2: Cloud Backbone: Step 2.28d

View the AppIQ report to find the problem

Next, AppIQ will **search the network traffic history** to see if the two instances have ever talked to each other in the past.

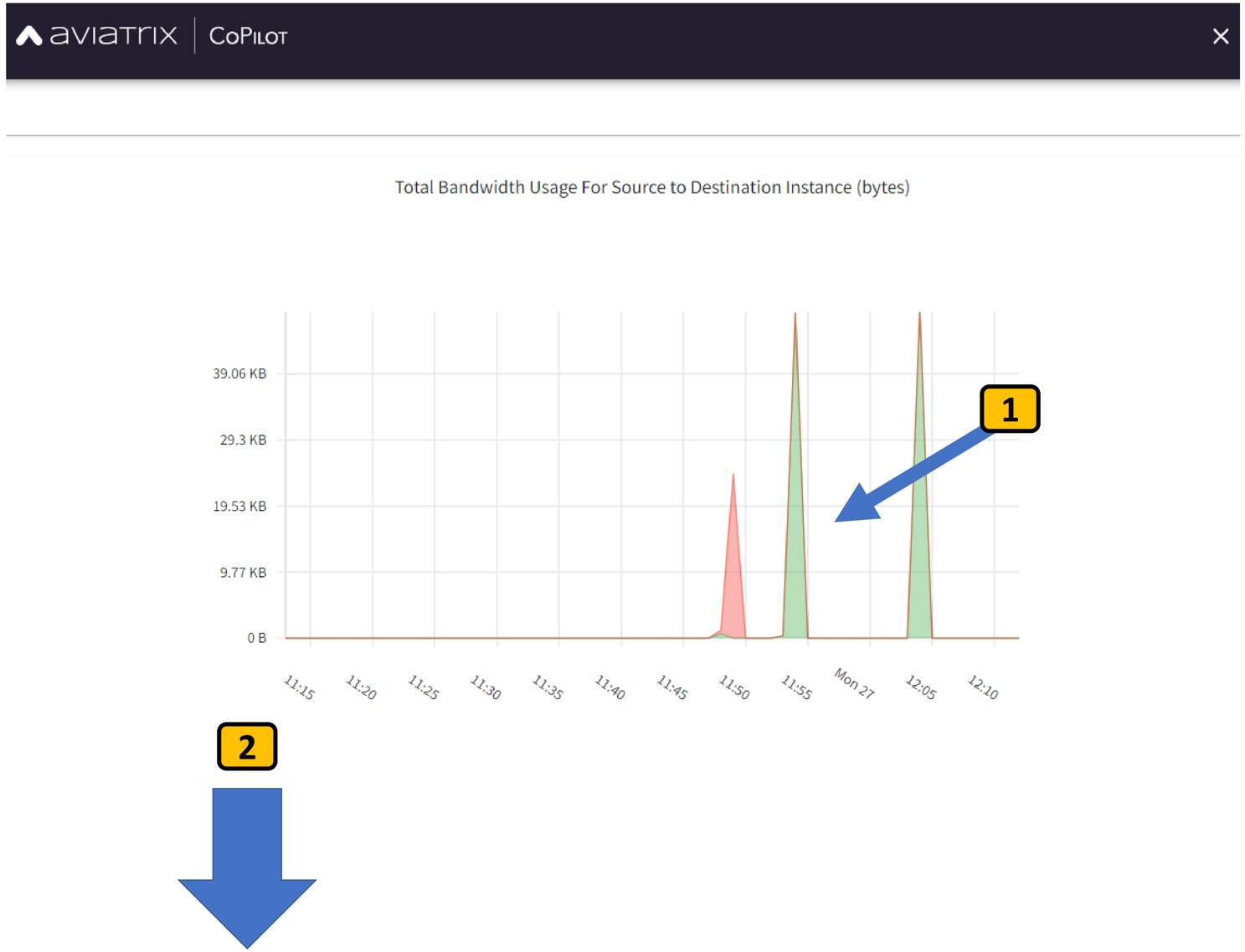
Have they ever talked to EACH OTHER? 1

Yes they have …
So far so good..

This information really helps us to know that our cloud network and these instances are working properly.

So what's the problem?

Scroll down to see the rest of the report 2



Lab 2: Cloud Backbone: Step 2.28e

View the AppIQ report to find the problem

Next, AppIQ will **check for proper configuration** of all routing table and NACL and security groups in this path.

AHA! **CoPilot found the problem**! 1

CoPilot found that there is no entry in the **inbound security group rules** for the destination instance SAP3 that would allow ICMP to work! 2

CoPilot even gives us a direct link to the security group with the problem.

Click on the security group link.
This will bring us to that security group in the AWS console to fix it. 3

Flightpath Data for Destination Instance: **SAP3**

Network ACL: ✓ Pass

ACL ID: acl-09bdab531c452d44a

Direction	Protocol	Port	CIDR	Allow/Deny
Outbound	ALL	ALL	0.0.0.0/0	allow
Outbound	ALL	ALL	0.0.0.0/0	deny
Inbound	ALL	ALL	0.0.0.0/0	allow
Inbound	ALL	ALL	0.0.0.0/0	deny

Security Groups: ✗ Fail 1

Group Name: immersion-sap-sg-3 (sg-02989109d681b5007) 3

2

Inbound Rules				
Type	Protocol	Port Range	Source	Description
http	tcp	80	10.0.0.0/8	Allow local HTTP inbound
http	tcp	80	172.16.0.0/16	Allow local HTTP inbound
ssh	tcp	22	10.0.0.0/8	Allow local ssh inbound
ssh	tcp	22	172.16.0.0/16	Allow local ssh inbound
Outbound Rules				
Type	Protocol	Port Range	Destination	Description
All traffic	ALL	ALL	0.0.0.0/0	Allow all outbound

Lab 2: Cloud Backbone: Step 2.29

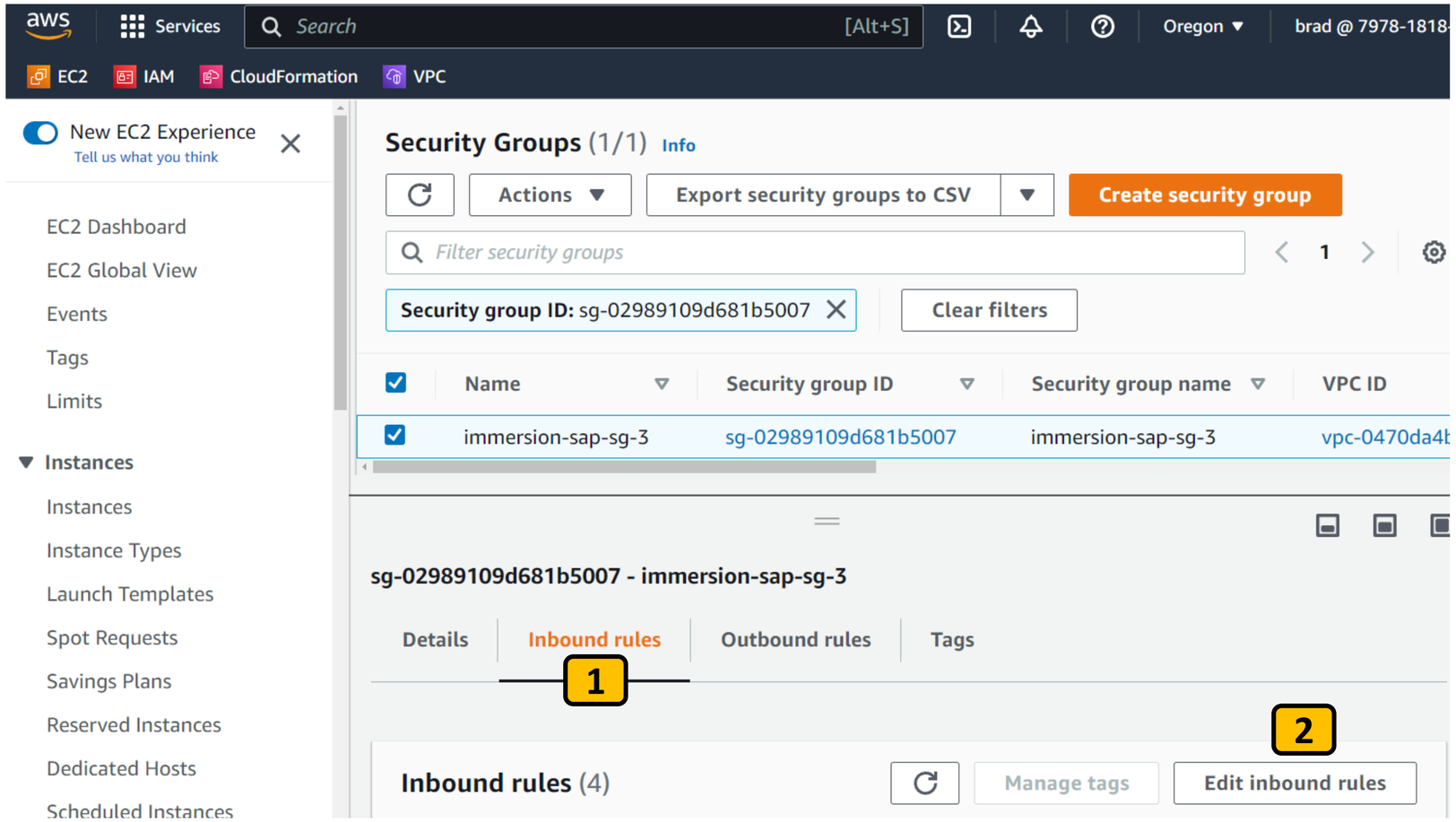
Resolve the connectivity problem

Clicking the link in AppIQ brought us right to the security group to fix it.

Let's fix it.

Click on **Inbound Rules** 1

Click on **Edit inbound rules** 2

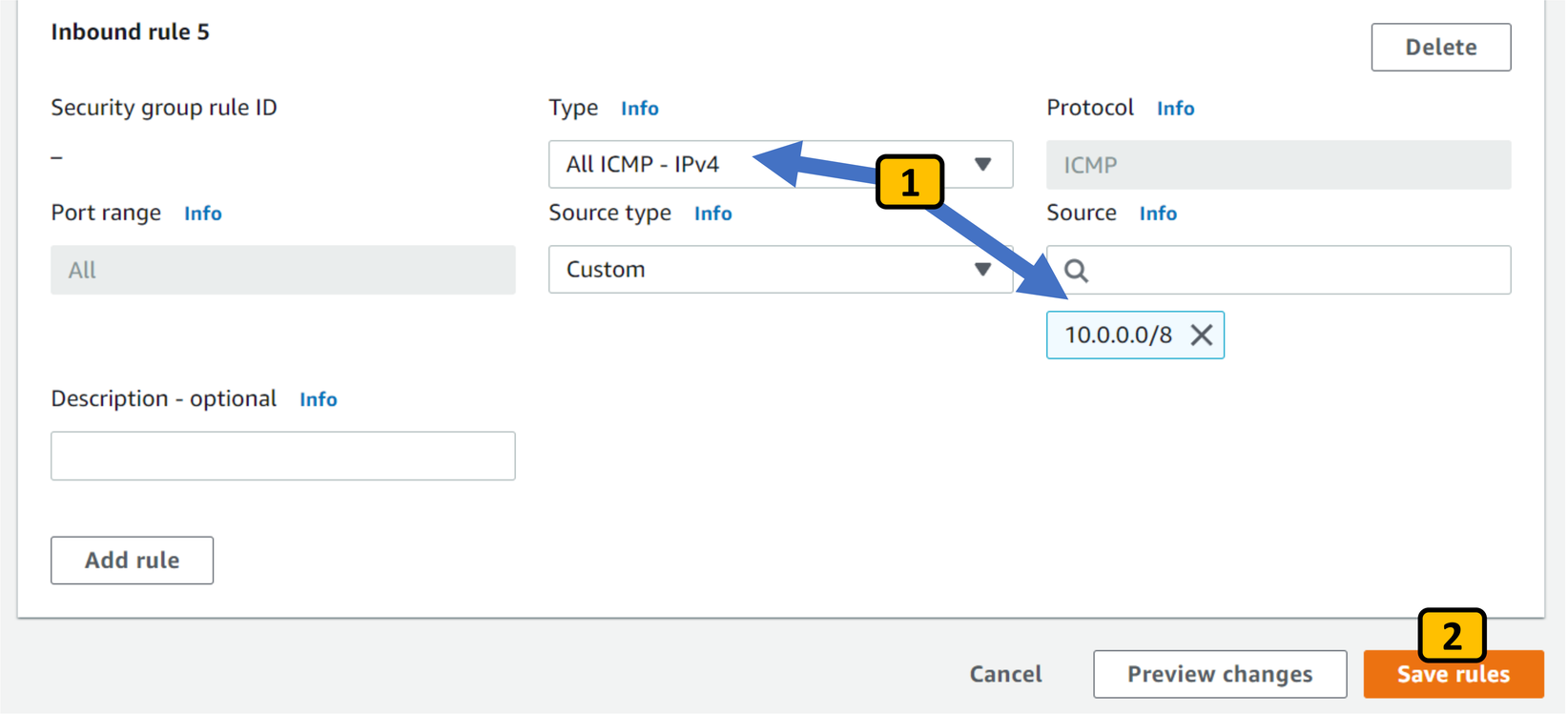


Lab 2: Cloud Backbone: Step 2.30

Resolve the connectivity problem

Add a rule that allows All ICMP v4 from the 10.0.0.0/8 IP range. 1

Click on **Save rules** 2



Lab 2: Cloud Backbone: Step 2.31

Retest connectivity

Now let's try that ping again to see if we fixed the problem.

Go back to the console of the EC2-A instance and ping the SAP3 instance again at 10.53.10 **1**

SUCCESS!! **2**

Look how quickly we were able to solve that problem. It wasn't the network's fault. It was a misconfigured security group.

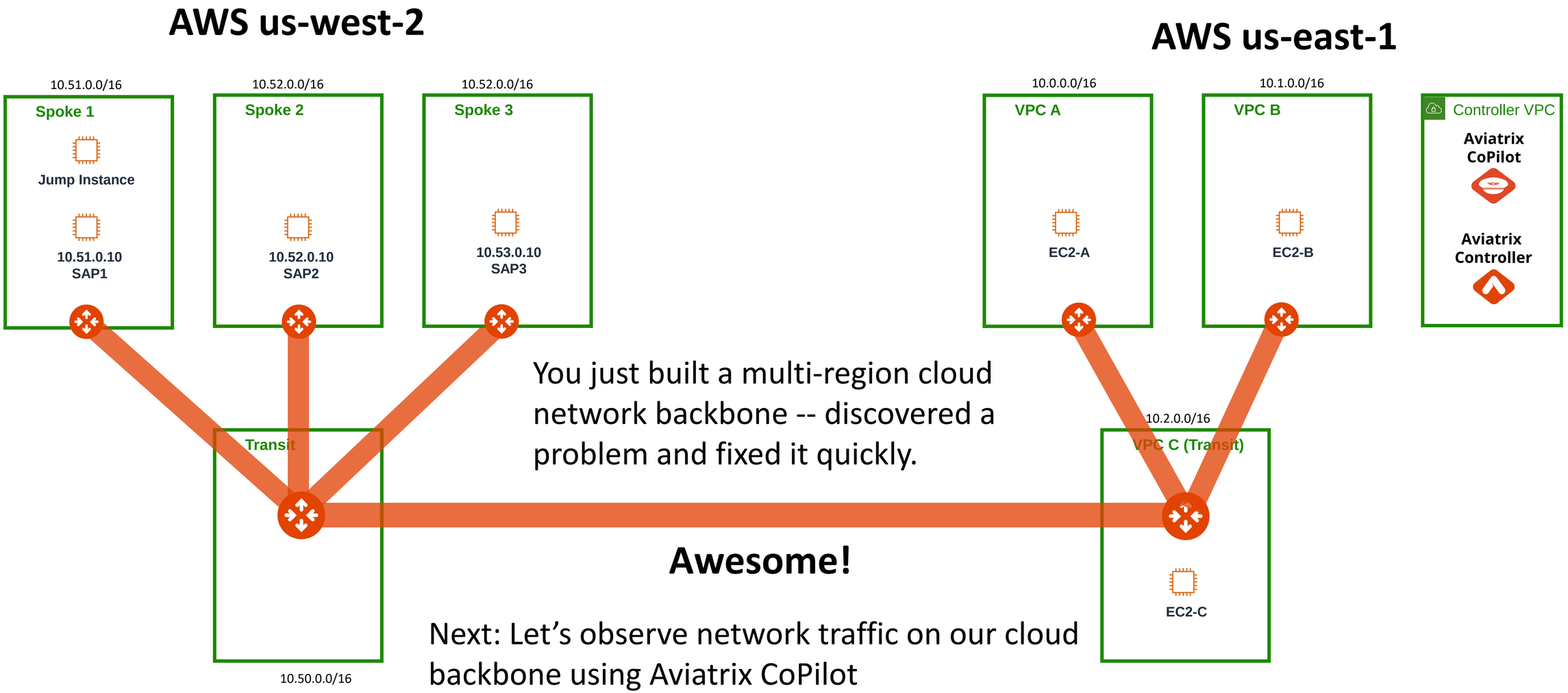
Feel free to ping around and continue testing your network.

Session ID: brad-059347c8e9174a00e Instance ID: i-0b91670973666f1ee

```
sh-4.2$
sh-4.2$
sh-4.2$
sh-4.2$ sudo su -l ec2-user
Last login: Mon Feb 27 05:54:31 UTC 2023 on pts/0
[ec2-user@ip-10-0-0-14 ~]$
[ec2-user@ip-10-0-0-14 ~]$
[ec2-user@ip-10-0-0-14 ~]$
[ec2-user@ip-10-0-0-14 ~]$
[ec2-user@ip-10-0-0-14 ~]$
[ec2-user@ip-10-0-0-14 ~]$                                  1
[ec2-user@ip-10-0-0-14 ~]$ ping 10.53.0.10
PING 10.53.0.10 (10.53.0.10) 56(84) bytes of data.
64 bytes from 10.53.0.10: icmp_seq=1 ttl=60 time=63.1 ms
64 bytes from 10.53.0.10: icmp_seq=2 ttl=60 time=63.0 ms
64 bytes from 10.53.0.10: icmp_seq=3 ttl=60 time=62.9 ms
64 bytes from 10.53.0.10: icmp_seq=4 ttl=60 time=63.3 ms
64 bytes from 10.53.0.10: icmp_seq=5 ttl=60 time=62.9 ms    2
64 bytes from 10.53.0.10: icmp_seq=6 ttl=60 time=62.9 ms
64 bytes from 10.53.0.10: icmp_seq=7 ttl=60 time=63.0 ms
64 bytes from 10.53.0.10: icmp_seq=8 ttl=60 time=63.0 ms
64 bytes from 10.53.0.10: icmp_seq=9 ttl=60 time=62.9 ms
64 bytes from 10.53.0.10: icmp_seq=10 ttl=60 time=63.1 ms
^C
--- 10.53.0.10 ping statistics ---
10 packets transmitted, 10 received, 0% packet loss, time 9011ms
rtt min/avg/max/mdev = 62.911/63.054/63.365/0.258 ms
[ec2-user@ip-10-0-0-14 ~]$
```

Lab 2: Cloud Backbone: Progress Check

Congratulations – You've completed Lab 2



You just built a multi-region cloud network backbone -- discovered a problem and fixed it quickly.

Awesome!

Next: Let's observe network traffic on our cloud backbone using Aviatrix CoPilot