Operations, Troubleshooting and Visibility

Solutions Engineering

Operational Challenges in Public Cloud

Evidential Data

When working with Cloud Providers, often customer is challenged to prove providers fault/issues

Unfamiliar Toolset

Native cloud lacks familiar tools like ping, packet capture, traceroute

Blackbox – No visibility

Native cloud constructs want you to trust all is well always. No visibility into logs, current state, routing tables, etc.

Infrastructure as Code

Solves agility problem, creates support issues as tier-1 is not able to troubleshoot code problems

A Flat World in Public Cloud

There is a lack of hierarchy in the cloud which means its hard to insert security, control and visibility

Tier-3 becomes Tier-1

Frontline support teams don't have the skill and tools in public cloud requiring senior network engineers to assist with most support issues

Scaling Out

Real problems are experienced when architecture scales out as it very quickly grows to be complex and very hard to troubleshoot

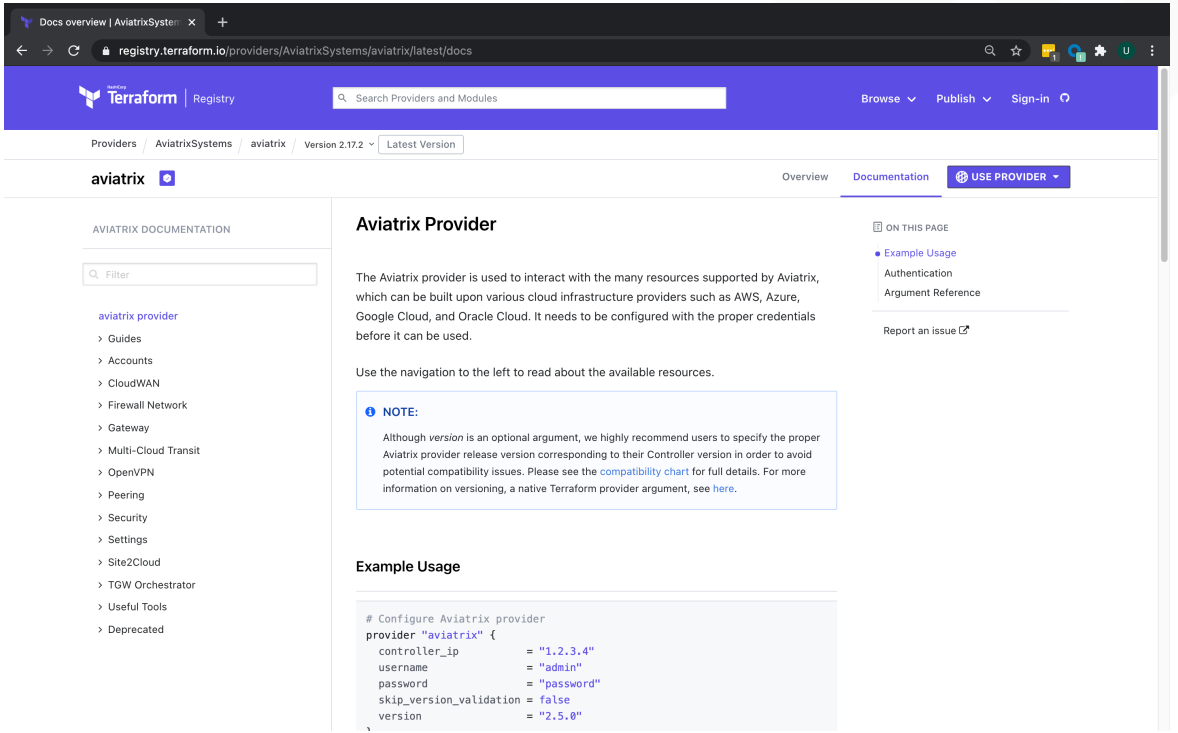
Infrastructure as Code

What it is

- Use Infrastructure as Code to provision and manage any cloud, infrastructure, or service
- Write declarative configuration files – define desired state
- Plan and predict changes
- Create reproducible infrastructure – if resource already exists, it won't recreate it
- Maintains knowledge of resources in a database called **State**
  - State maps config to real world

Aviatrix Terraform Provider

- Multi-lingual entity responsible for API interactions with CSPs
- Exposes resources in those CSPs for any account/subscription that has been onboarded
- Feature parity with Controller code



Aviatrix Terraform Resources – Examples

- # Create an Aviatrix AWS Gateway

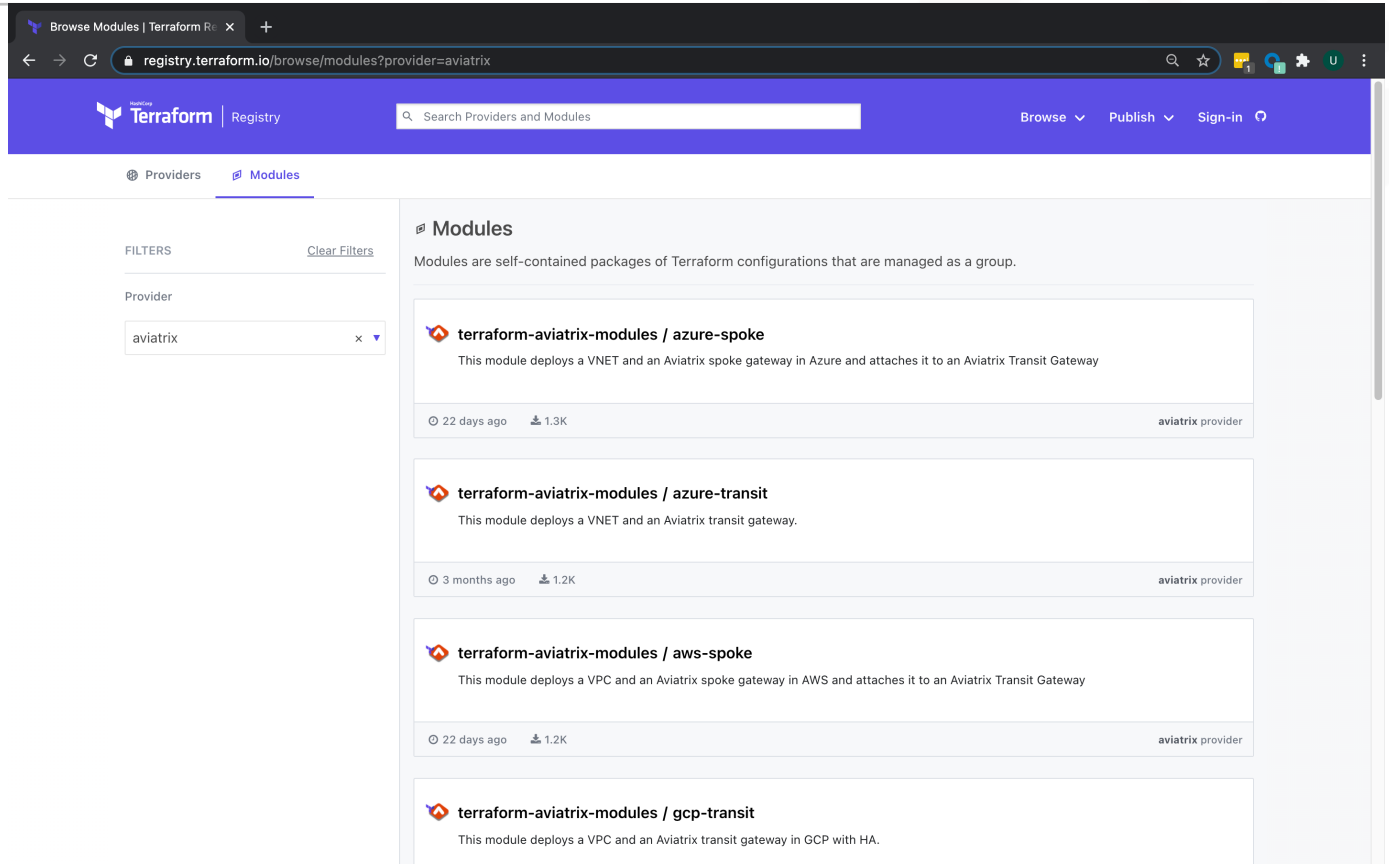
```
resource "aviatrix_gateway"
"test_gateway_aws" {

   cloud_type    = 1

   account_name = "devops-aws"

  gw_name       = "avtx-gw-1"

  vpc_id        = "vpc-abcdef"

  vpc_reg       = "us-west-1"

  gw_size       = "t2.micro"

   subnet        = "10.0.0.0/24"

}
```

- # Create an Aviatrix Azure Gateway

```
resource "aviatrix_gateway"
"test_gateway_azure" {

  cloud_type    = 8

  account_name = "devops-azure"

  gw_name       = "avtx-gw-azure"

  vpc_id        = "gateway:test-gw-123"

  vpc_reg       = "West US"

  gw_size       = "Standard_D2"

  subnet        = "10.13.0.0/24"

}
```

Aviatrix Terraform Modules

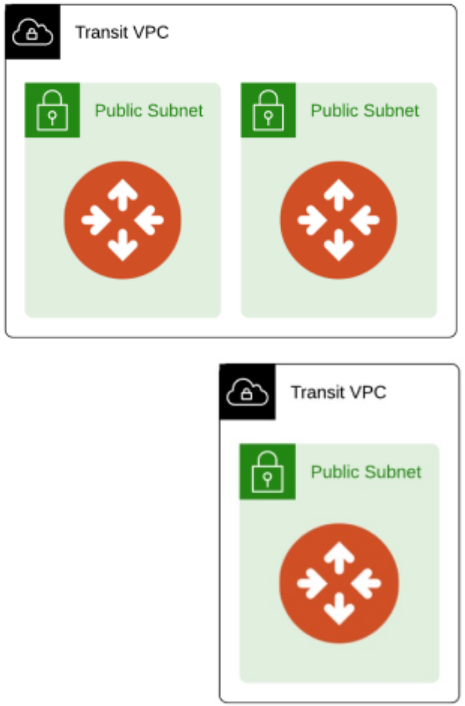
- ***"Repeatable++"***
- Similar to the concepts of libraries, packages, or modules found in most programming languages
- Provide many of the same benefits
- ~10X reduction in lines of code
- Can be found on Terraform Registry



Aviatrix Terraform Module – Example

- # Create a VPC and a set of Aviatrix transit gateways.

```
module "transit_aws_1" {

  source  = "terraform-aviatrix-modules/mc-transit/aviatrix"

  version = ”1.1.2”

  cloud   = “aws”

  cidr    = "10.1.0.0/20"

  region  = "eu-west-1"

  account = "AWS-account"

}

ha_gw set to true by default
```



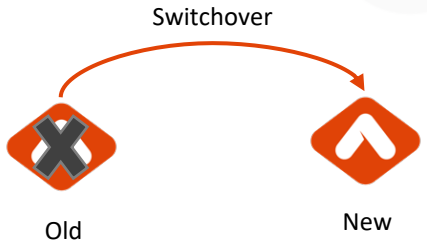
Aviatrix Controller High Availability (HA)

Aviatrix Controller High Availability (HA)

- Very important: Controller is not in the data path
- If Controller is down → Data Plane still functions
- Your cloud network is still up and running
- Do not compare on-prem to cloud
  - Hardware devices cannot be replaced / software is more flexible
  - Cloud operating models are different
  - Cloud processes are different
  - We need a fresh and different look to solve

Aviatrix Controller HA Process

- Takes minutes to switch over to new controller
  - Depends on factors such as AWS latency, instance type, size of the DB, etc.
- Previous controller is terminated
- All existing configuration is restored
- New Private IP is assigned (new AZ)
- New controller stays at the same version as previous

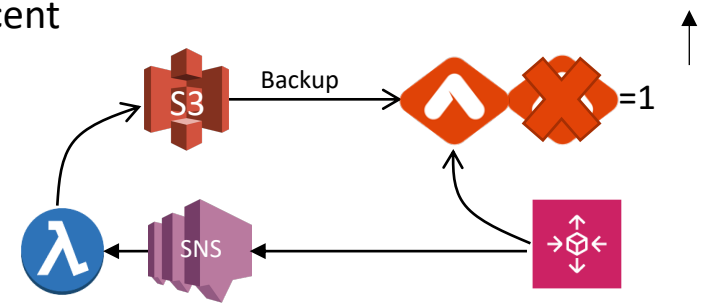


https://docs.aviatrix.com/HowTos/controller_ha.html

https://github.com/AviatrixSystems/Controller-HA-for-AWS/

Aviatrix Controller HA Process

- Aviatrix Controller HA operates by relying on an AWS Auto Scaling Group
- The Auto Scaling Group has a desired capacity of 1
- If the Controller EC2 instance is stopped or terminated, it will be automatically re-deployed by the Auto Scaling Group
- An AWS Lambda script is notified via SNS when new instances are launched by the Auto Scaling Group
- This script handles configuration restore using the most recent Controller backup file, stored in S3



Upgrade Process

Important Points to Remember for Upgrade

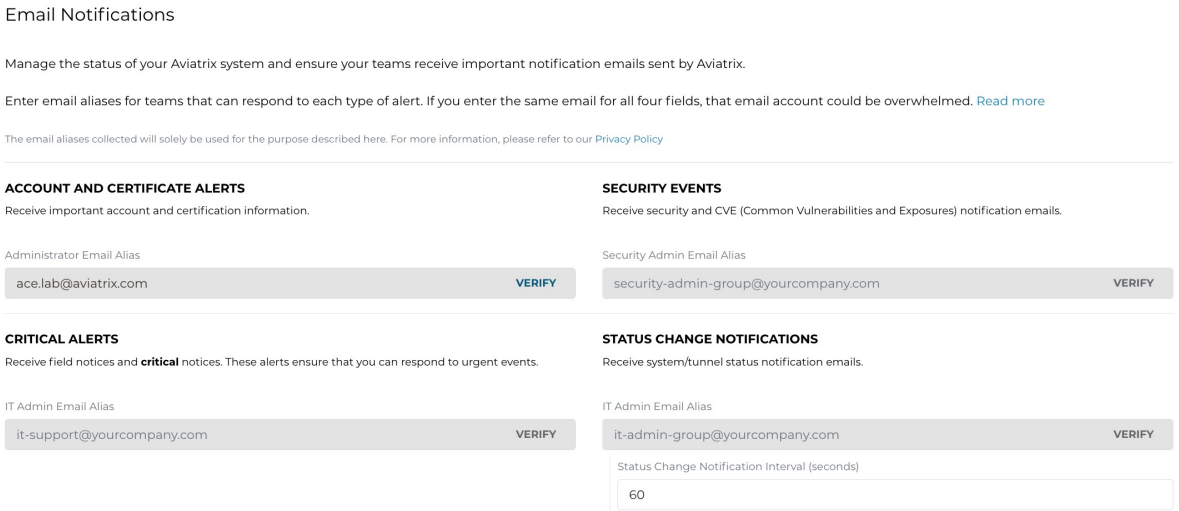
- Controller is in management/control plane
- Upgrade process is designed in a modern / cloud native / born in the cloud way that is hitless and much faster than on-prem upgrades
- Upgrade time depends on many factors
  - Customer Data Point: 500 GWs environment → ~22 min to upgrade
- You can perform the following operations on selected gateways:
  - Perform a Platform Software Upgrade Dry Run
  - Perform a Gateway Software Upgrade Dry Run
  - Upgrade the Platform Software
  - Upgrade the Gateway Software
  - Roll Back the Gateway Software
  - Upgrade the Gateway Image
- All details at https://docs.aviatrix.com/HowTos/selective_upgrade.html

Best Practices for Upgrade

- Use Staging environment to test new features out
- Perform upgrades in a maintenance window
- Read the documentation before planning your upgrades
  - https://docs.aviatrix.com/HowTos/UCC_Release_Notes.html
  - https://docs.aviatrix.com/Support/support_center_operations.html#pre-op-procedures
  - https://docs.aviatrix.com/HowTos/inline_upgrade.html
  - https://docs.aviatrix.com/HowTos/field_notices.html
- Stage the upgrade with a Dry Run to assess the following:
  - Reachability between the controller and the release server
  - Reachability between the controller and gateways
  - Free memory on controller and gateways
  - Disk space on controller and gateways
  - CPU low enough on controller and gateways
- Only if all green, should you proceed with upgrade

Support Portal

- Aviatrix customers may visit Support portal – https://support.aviatrix.com to access:
  - Knowledge Base with videos
  - Documentation
  - Community
  - History of tickets
  - CSP outage tracker
- Sign up for Email Notifications



Aviatrix Sandbox Starter Tool

Build your own MCNA Transit at ~$1/hr

- Goal: Fast path for Customers and Partners to deploy Aviatrix multi-cloud transit foundation with minimal cost
- Turn-key solution to deploy Aviatrix Controller + MCNA in AWS and Azure + test instances with extreme simplicity and flexibility
- Can be deployed in 3 different ways:
  - Local (BYO Docker)
  - AMI
  - AMI with Terraform module

Description	Unit Cost	Quantity	Hourly Cost	Cost for 8 hours	Cost for 24 hours
Aviatrix Controller in AWS (t3.large)	$0.09	1	$0.09		
Aviatrix Gateway in AWS (t2.micro)	$0.01	3	$0.03		
Test instances in AWS (t2.micro)	$0.01	2	$0.02		
Aviatrix Encrypted Peering (AWS)	$0.23	2	$0.46		
Total Cost for AWS-only Transit + 2 Spokes			$0.60	$4.80	$14.40

Extending into Azure

Description	Unit Cost	Quantity	Hourly Cost	Cost for 8 hours	Cost for 24 hours
Aviatrix Gateway in Azure (B1s)	$0.01	3	$0.03		
Aviatrix Encrypted Peering (Azure)	$0.23	2	$0.46		
Aviatrix Transit Peering (between AWS and Azure)	$0.70	1	$0.70		
Total Cost for MCNA (including minimal network egress charges)			$1.19	$9.52	$28.56

User guide: https://community.aviatrix.com/t/g9hx9jh

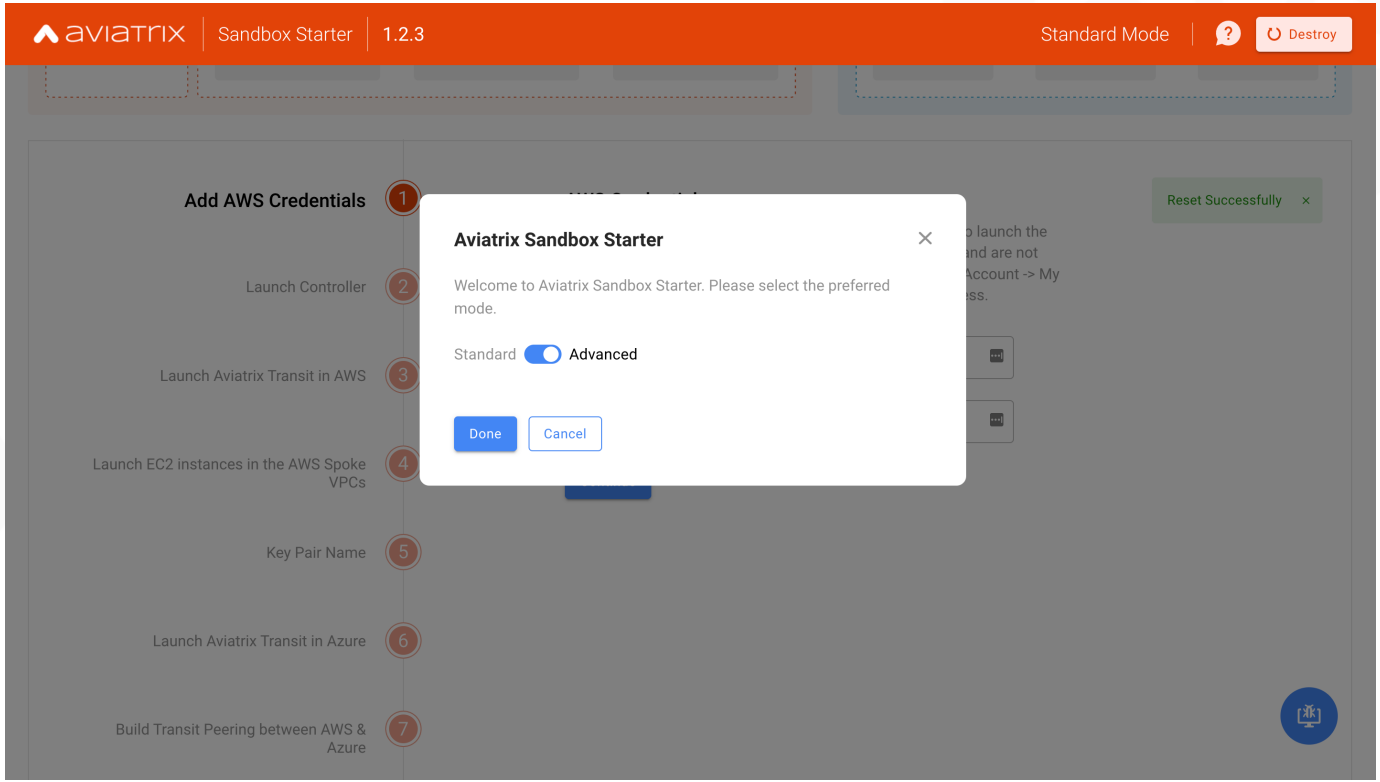
What Sandbox Starter Tool Builds

- Controller launch (Metered or BYOL)
  - VPC and all networking
  - Security Groups
  - Key pairs
  - IAM roles and policies (only if they don't already exist)
  - EC2 instance
  - Username and password
  - Software upgrade
  - AWS account onboarding
  - Configuring License (BYOL)

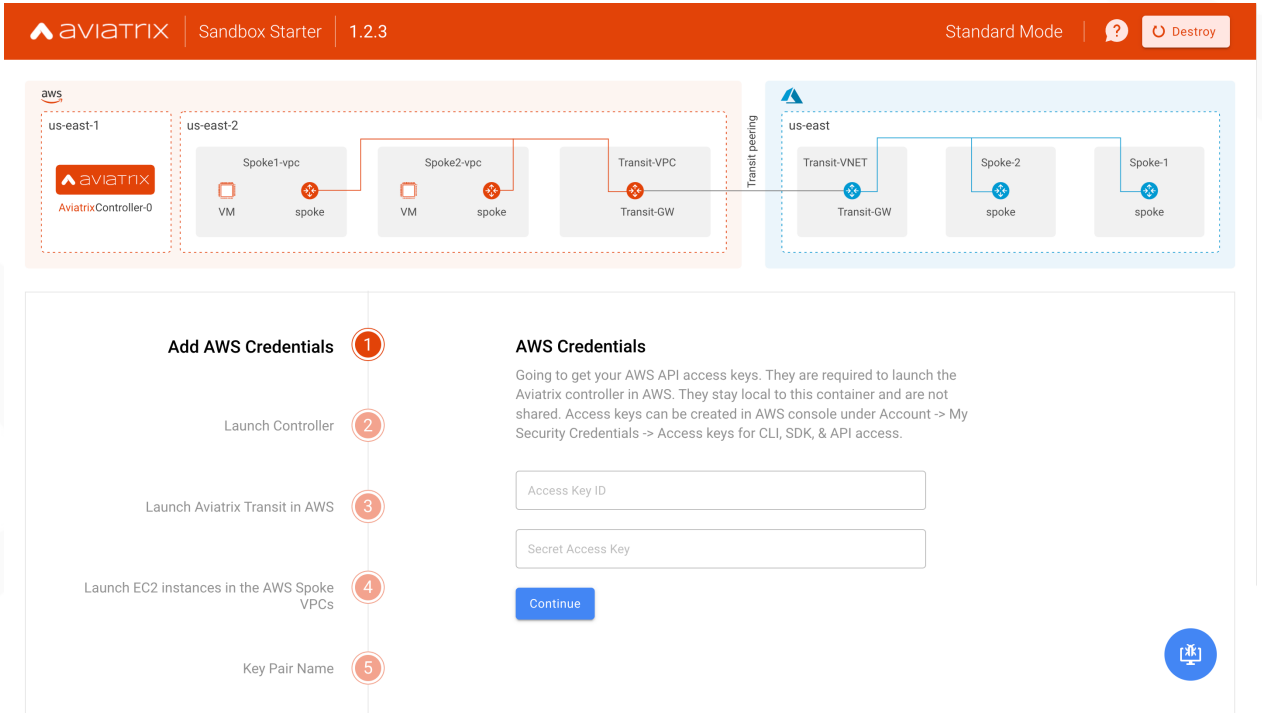
- MCNA launch
  - Azure account onboarding
  - AWS VPCs and Azure VNets
    - Spoke and Transit
  - ActiveMesh Transit in AWS
    - Spoke gateways
    - Transit gateways
    - Spoke attachment to Transit
  - Same ActiveMesh Transit in Azure
  - Transit peering between AWS and Azure

Sandbox Starter Tool Modes

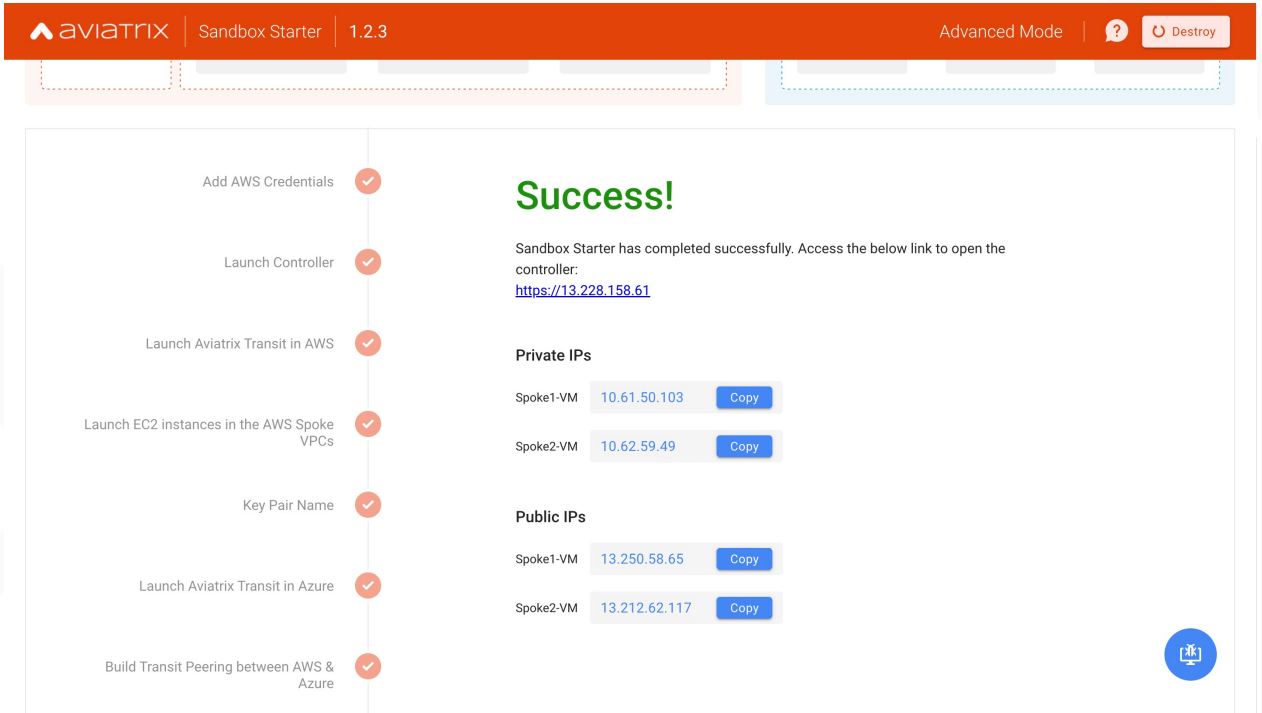
- Standard Mode
  - Fixed regions, resource names, and CIDR blocks
- Advanced Mode
  - Customizable regions, resource names, and CIDR blocks



Sandbox Starter Tool Workflow Start



Sandbox Starter Tool Workflow Completion



Next: Let’s Design Together