AWS re:Invent

NOV. 29 – DEC. 3, 2021 | LAS VEGAS, NV

NET305

NetDevOps: A modern approach to AWS networking deployments

Sid Chauhan
Principal Solutions Architect
AWS

Ákos Varga
Principal Infrastructure Architect
General Electric

Michael Palmer
Staff Software Engineer
General Electric

We are all in the business of innovation

DevOps?

Increases an organization’s ability to deliver applications and services at high velocity

Faster changes
Automation and rigorous testing
Reduced software delivery lifecycle

Net DevOps?

Increases an organization’s ability to deliver applications and services at high velocity

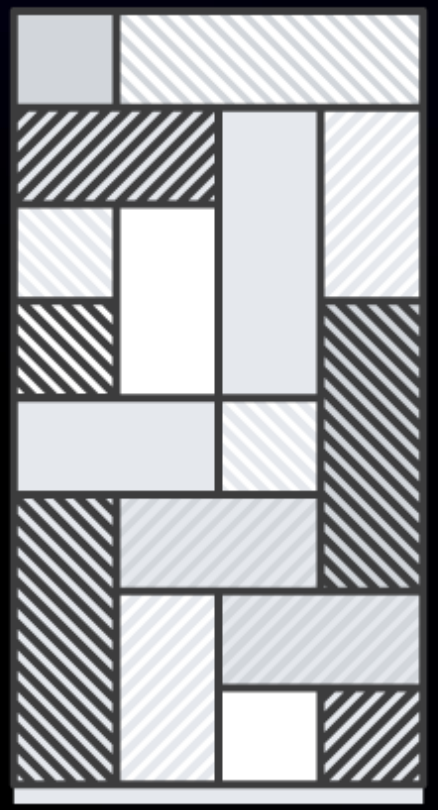
Faster changes
Automation and rigorous testing
Reduced software delivery lifecycle

Today’s session is all about this
+ use cases and GE’s story!

DevOps best practices we must take on

Best practice 1 – Breaking down monoliths

Best practice 1 – Breaking down monoliths



Monolithic architecture

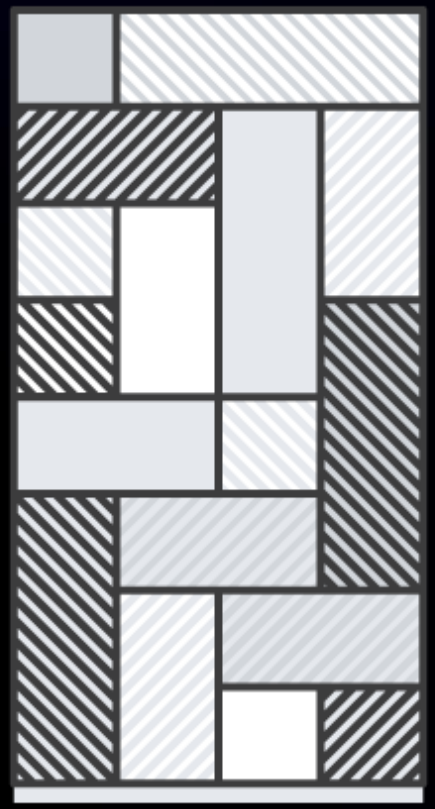
Bigger area of impact – very sensitive to change

More testing and build time

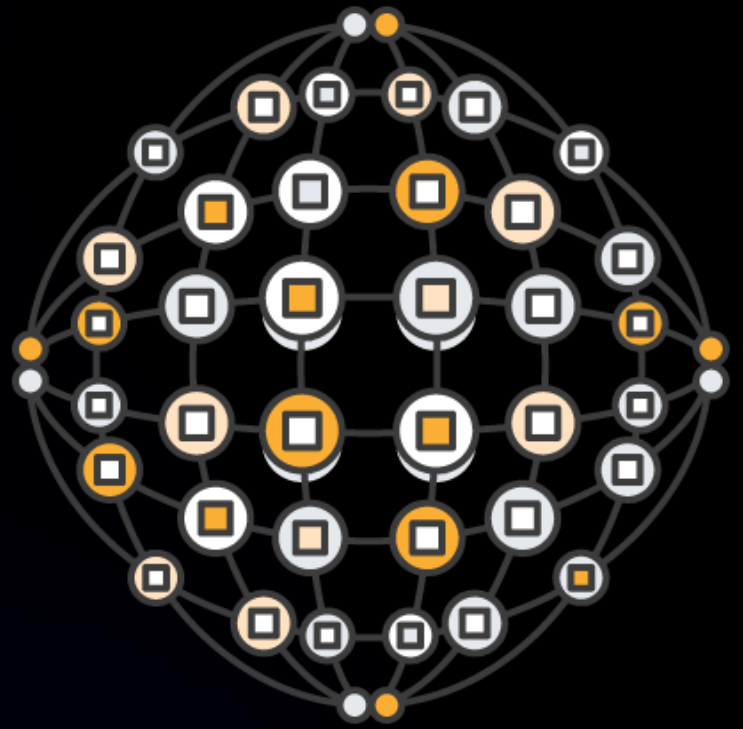
More complexity – difficult to manage and maintain

Slower pace of innovation

Best practice 1 – Breaking down monoliths

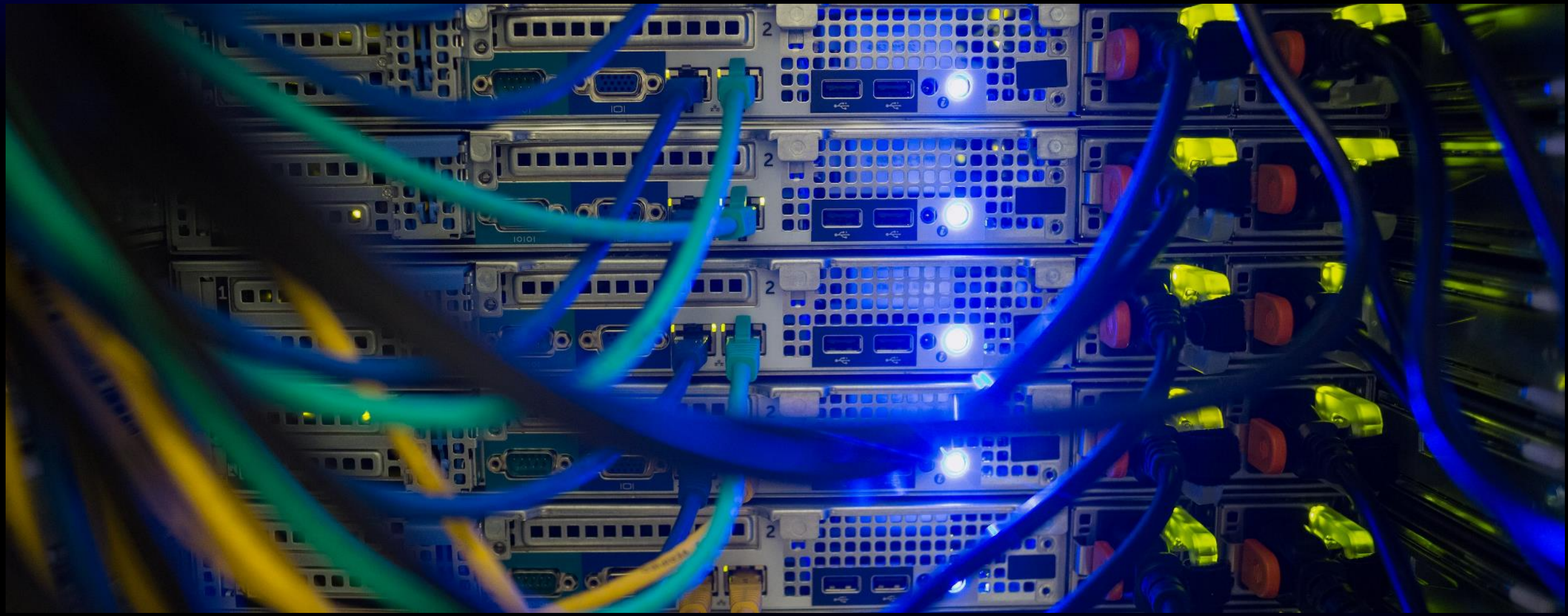


Monolithic architecture



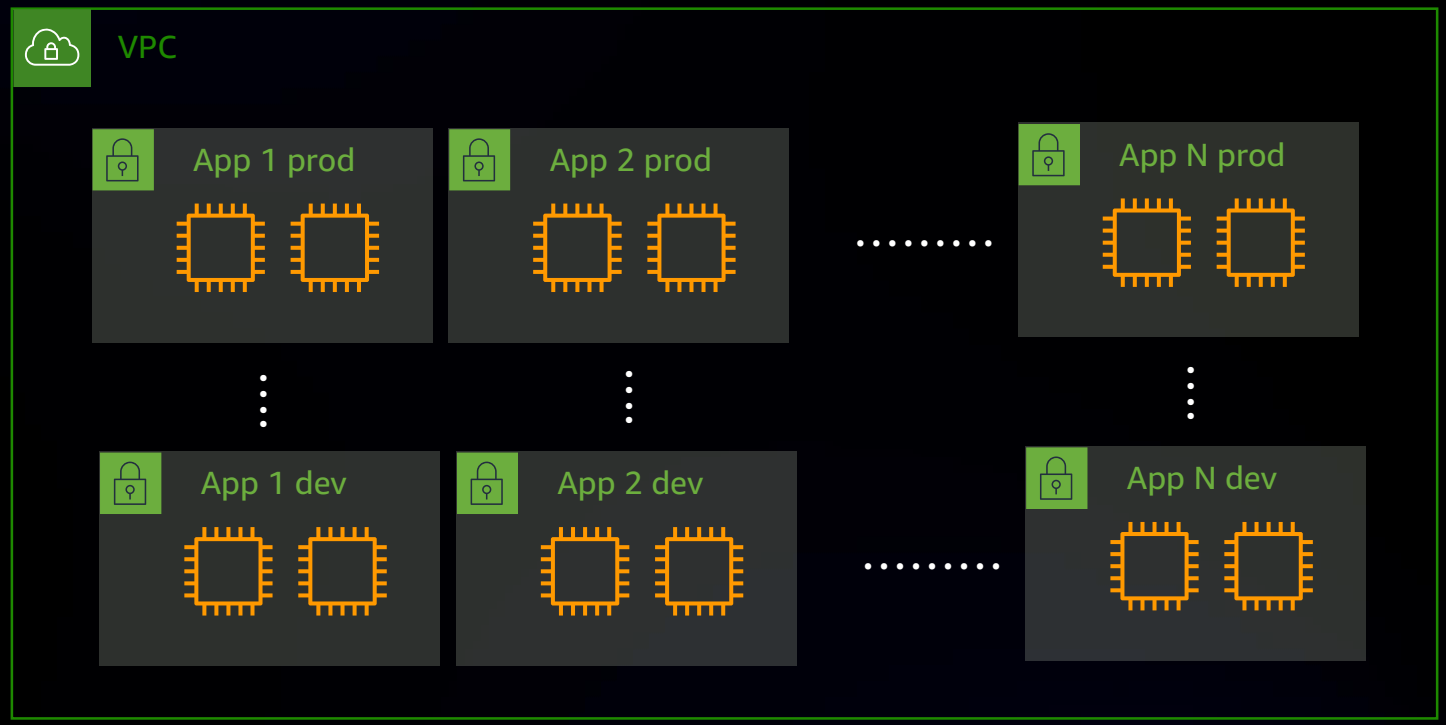
Microservices

Best practice 1 – Breaking down monoliths



Networking monoliths exist!

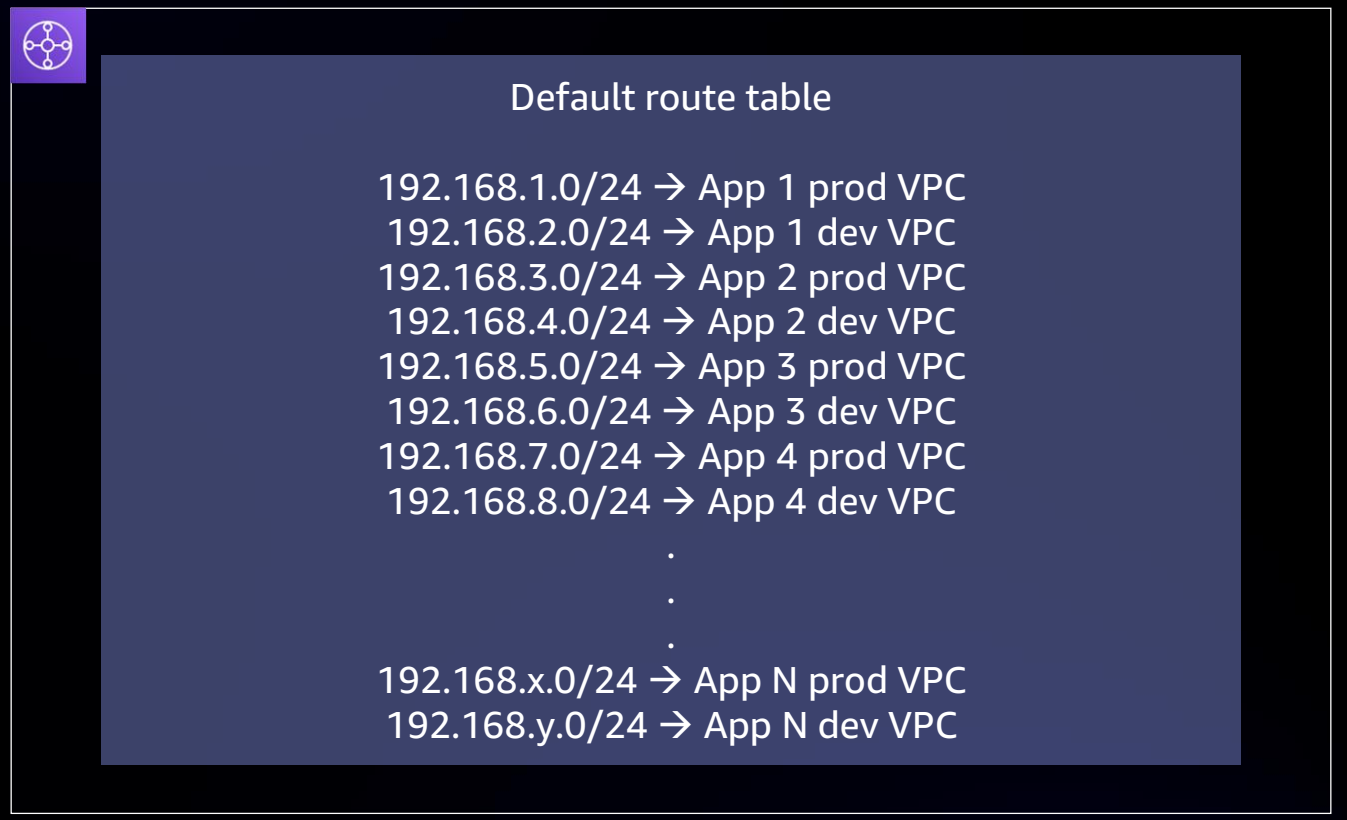
Networking monolith – Example 1



Very large VPCs with all applications and environments in the same VPC

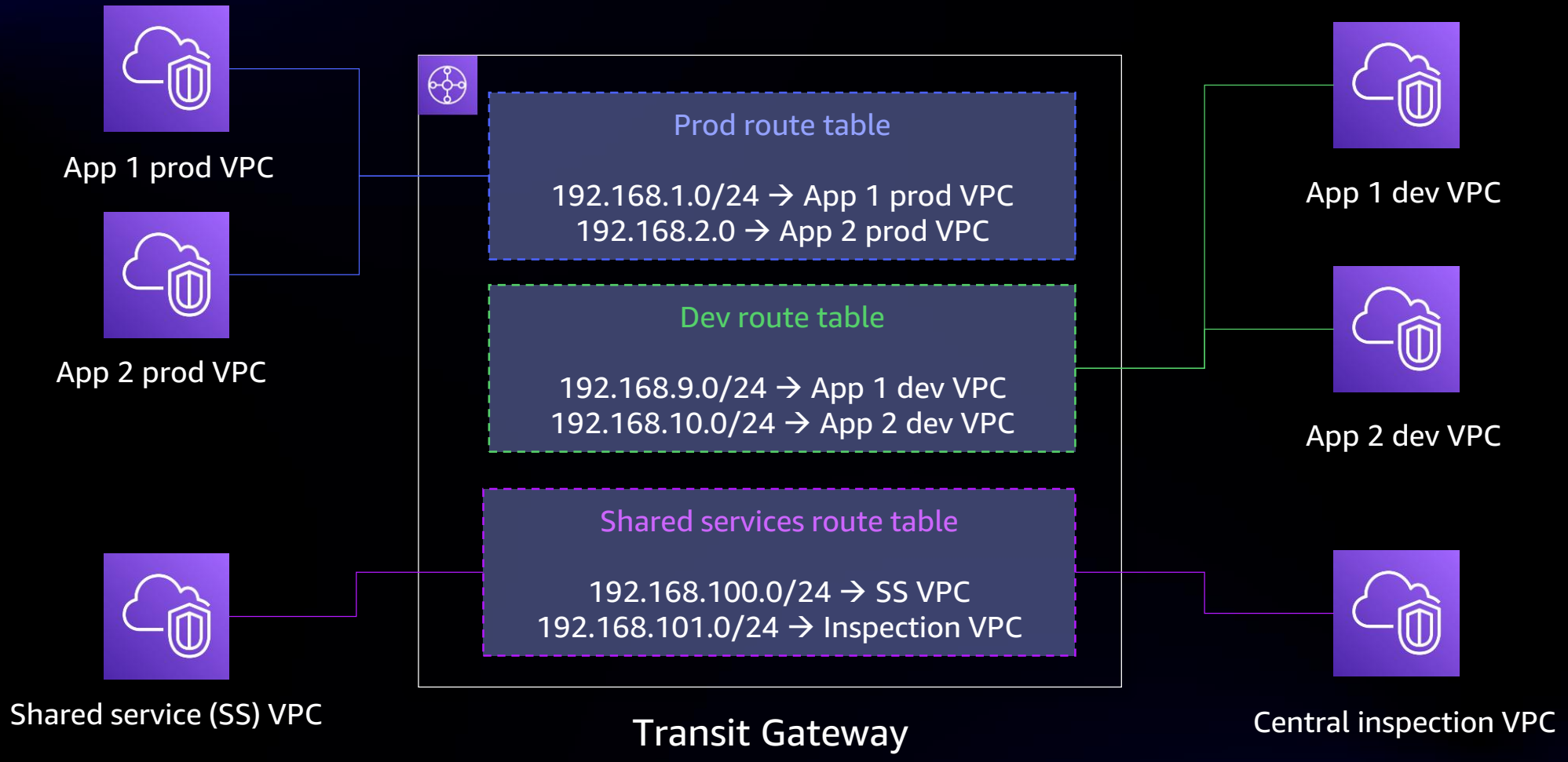
Networking monolith – Example 2

AWS Transit Gateway
route table



Very large routing domains with multiple BU applications and environments in the same route table

Best practice 1 – Breaking down monoliths



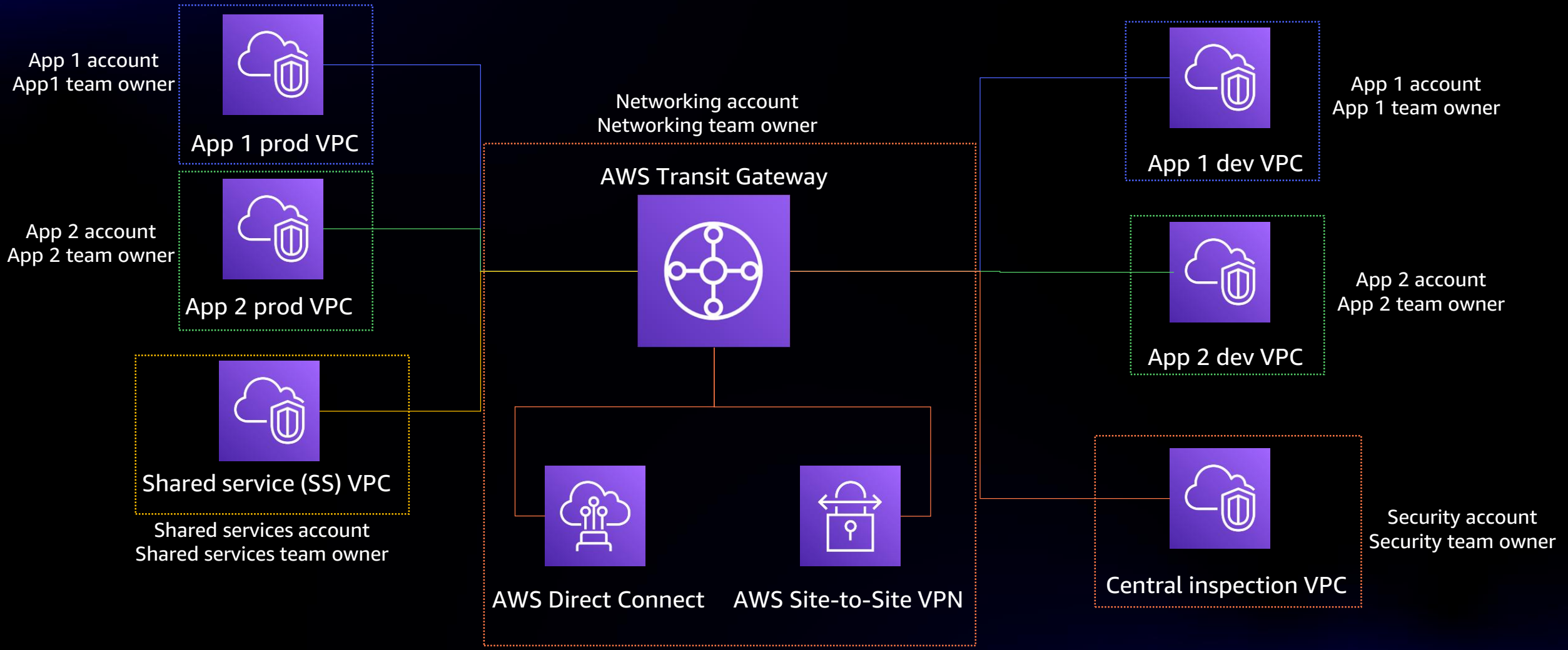
Build modular network architectures that can scale

Best practice 2 – Complete ownership

“Any organization that designs a system will produce a design whose structure is a copy of the organization's communication structure.”

Conway’s law

Best practice 2 – Complete ownership



Best practice 3 – Automation

Best practice 3 – Automation with IaC

Infrastructure as code (IaC)

- Simplified way to create and manage your network resources
- Predictable and repeatable provisioning
- Enables version control

Best practice 3 – Automation with IaC

Infrastructure as code (IaC)

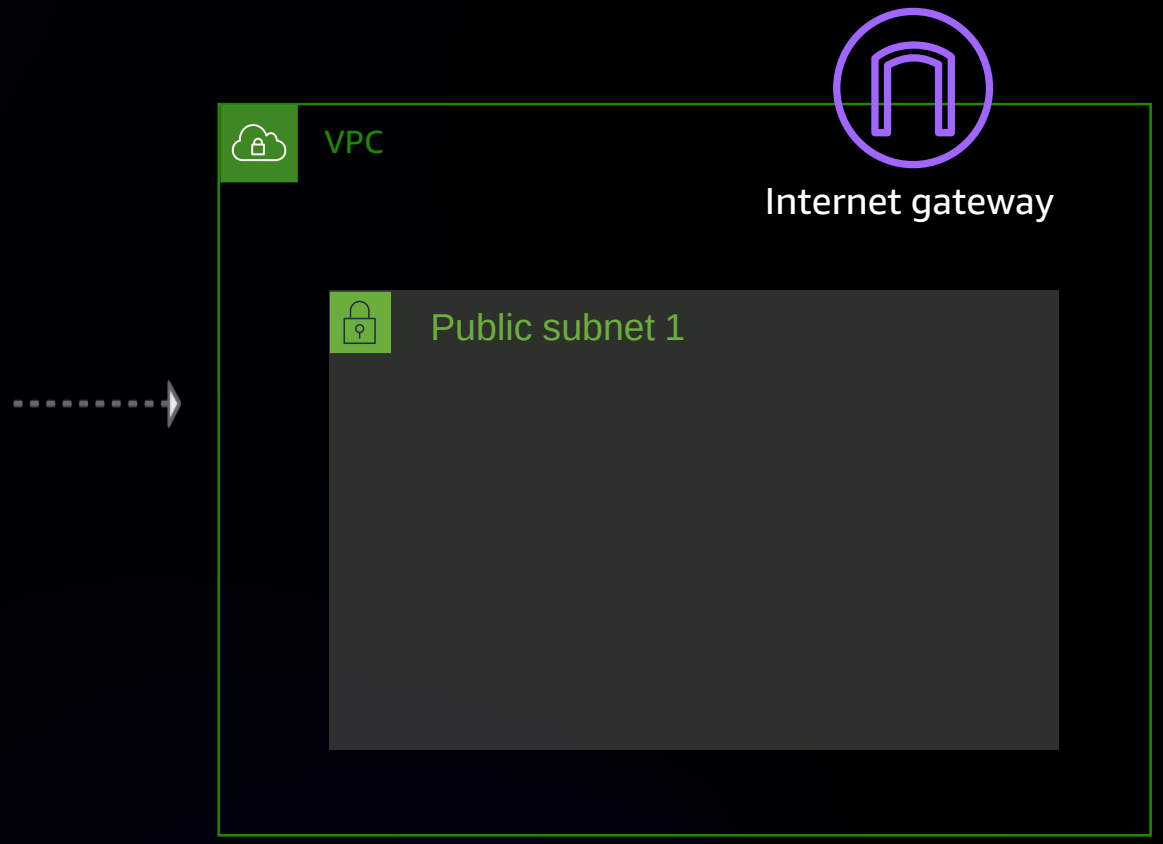
AWS CloudFormation

```yaml
Resources:
  VPC:
    Type: AWS::EC2::VPC
    Properties:
      CidrBlock: !Ref VpcCIDR
      EnableDnsSupport: true
      EnableDnsHostnames: true
      Tags:
        - Key: Name
          Value: !Ref EnvironmentName

  InternetGateway:
    Type: AWS::EC2::InternetGateway
    Properties:
      Tags:
        - Key: Name
          Value: !Ref EnvironmentName

  InternetGatewayAttachment:
    Type: AWS::EC2::VPCGatewayAttachment
    Properties:
      InternetGatewayId: !Ref InternetGateway
      VpcId: !Ref VPC

  PublicSubnet1:
    Type: AWS::EC2::Subnet
    Properties:
      VpcId: !Ref VPC
      AvailabilityZone: !Select [ 0, !GetAZs '' ]
      CidrBlock: !Ref PublicSubnet1CIDR
      MapPublicIpOnLaunch: true
      Tags:
        - Key: Name
          Value: !Sub ${EnvironmentName} Public Subnet (AZ1)
```



Sample CloudFormation template

Best practice 3 – Automation with IaC

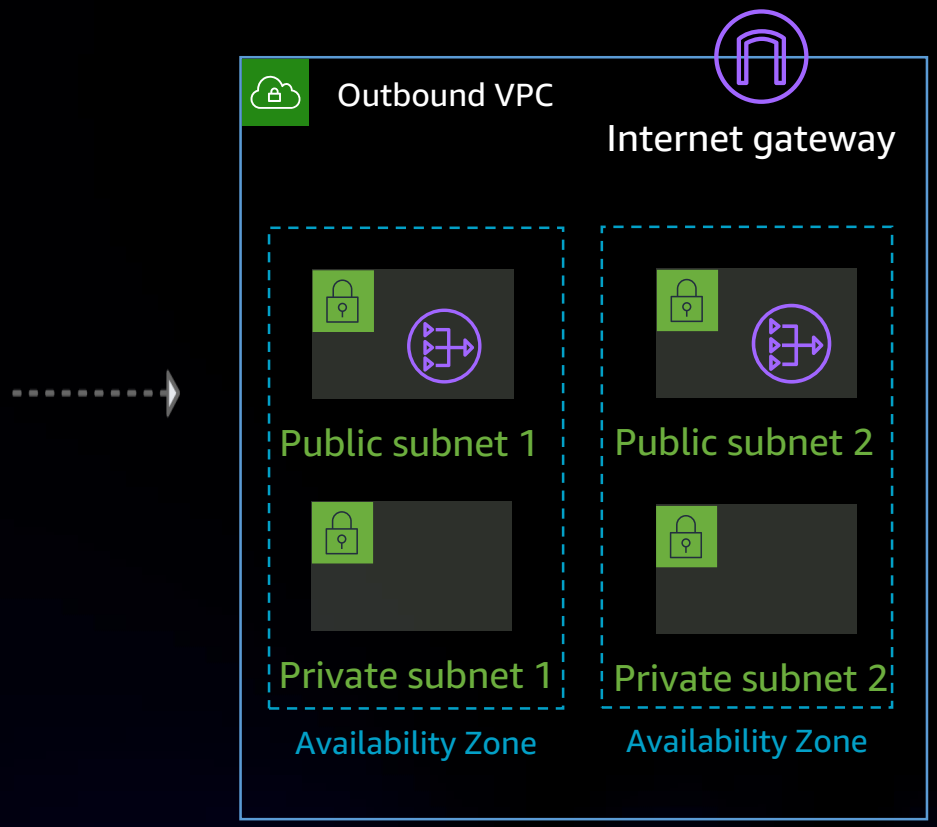
Infrastructure as code!

AWS CloudFormation

AWS Cloud Development Kit
(AWS CDK)

```
const egressVPC = new ec2.Vpc(this, 'Egress VPC', {
  cidr: "10.0.1.0/26",
  //natGateways: 1, add this to limit numer of deployed NAT gateways
  subnetConfiguration: [{
      cidrMask: 28,
      name: 'Public - EgressVPC SubNet',
      subnetType: SubnetType.PUBLIC,
    },
    {
      cidrMask: 28,
      name: 'Private - EgressVPC SubNet',
      subnetType: SubnetType.PRIVATE,
    },
  ]
});
```

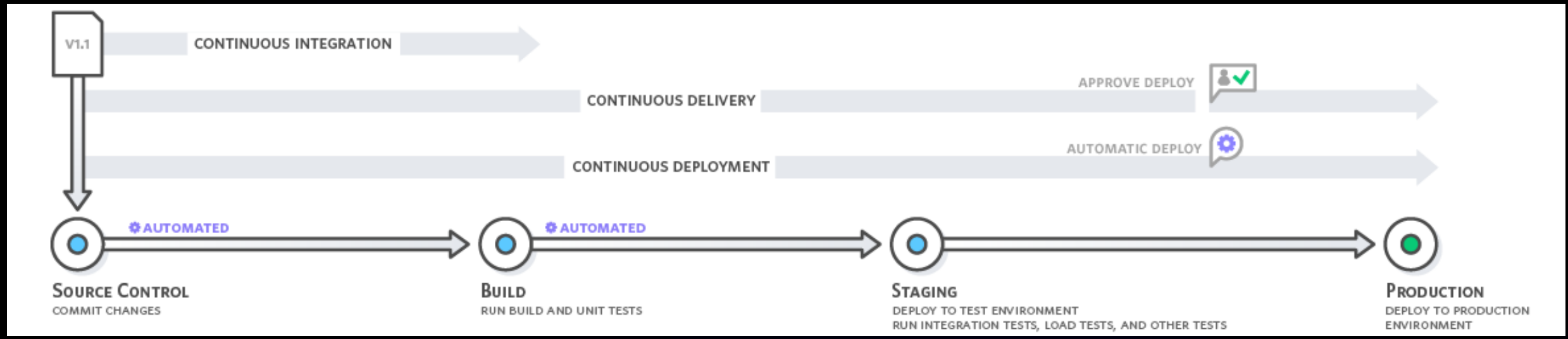
Sample AWS CDK Python code



Where to deploy – Prod?

Best practice 4 – CI/CD

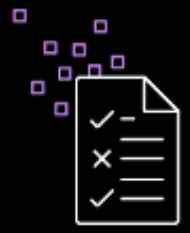
- Continuous integration
- Continuous delivery and deployment



Unit tests

Code inspection for security, config, enterprise guardrails verification

https://github.com/aws-cloudformation/cfn-lint



AWS CloudFormation Guard – Validate cloud environments with policy as code

https://github.com/aws-cloudformation/cloudformation-guard

https://github.com/stelligent/cfn_nag

aws

What can you check for?

- Firewall rules (security group, network firewall, web application firewall, etc.) validation
- Flow log settings
- Public subnets and public access

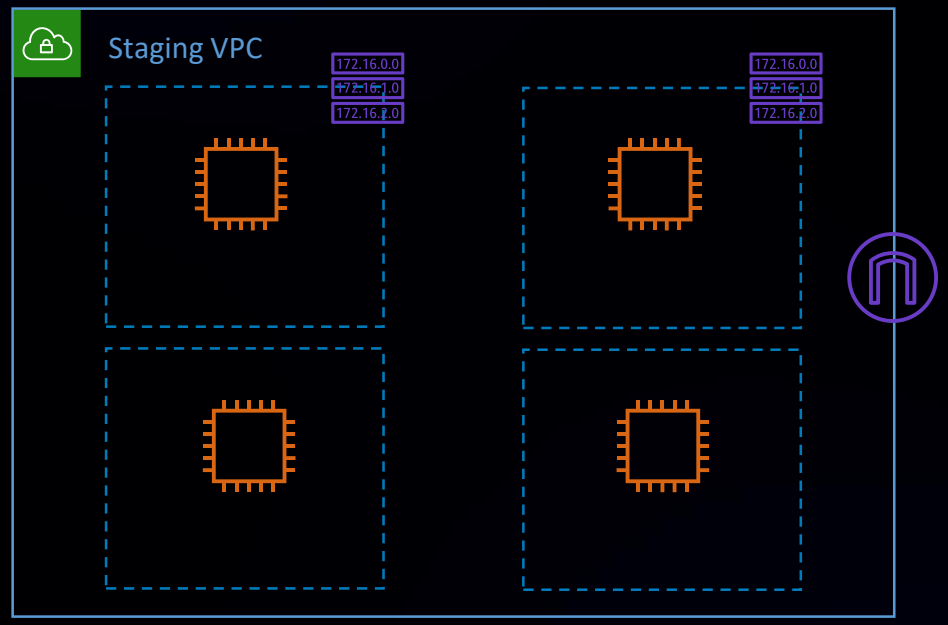
What can you check for?

- Firewall rules (security group, network firewall, web application firewall, etc.) validation
- Flow log settings
- Public subnets and public access
- CIDR allocations
- Peering establishments

Integration test

CHANGE YOUR NETWORK INFRASTRUCTURE INTO A TEST/STAGING ENVIRONMENT

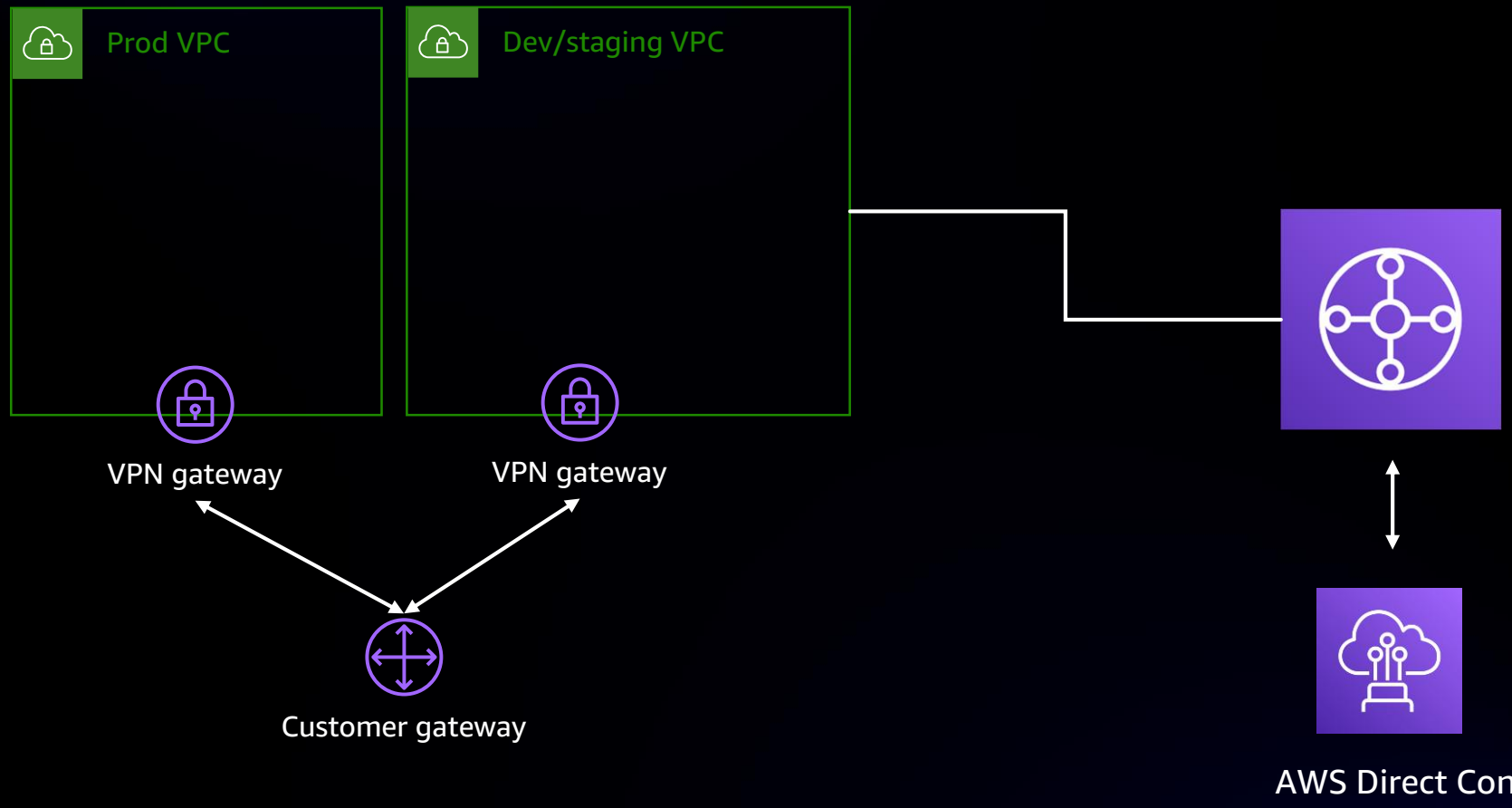
Deploy



Test + validate

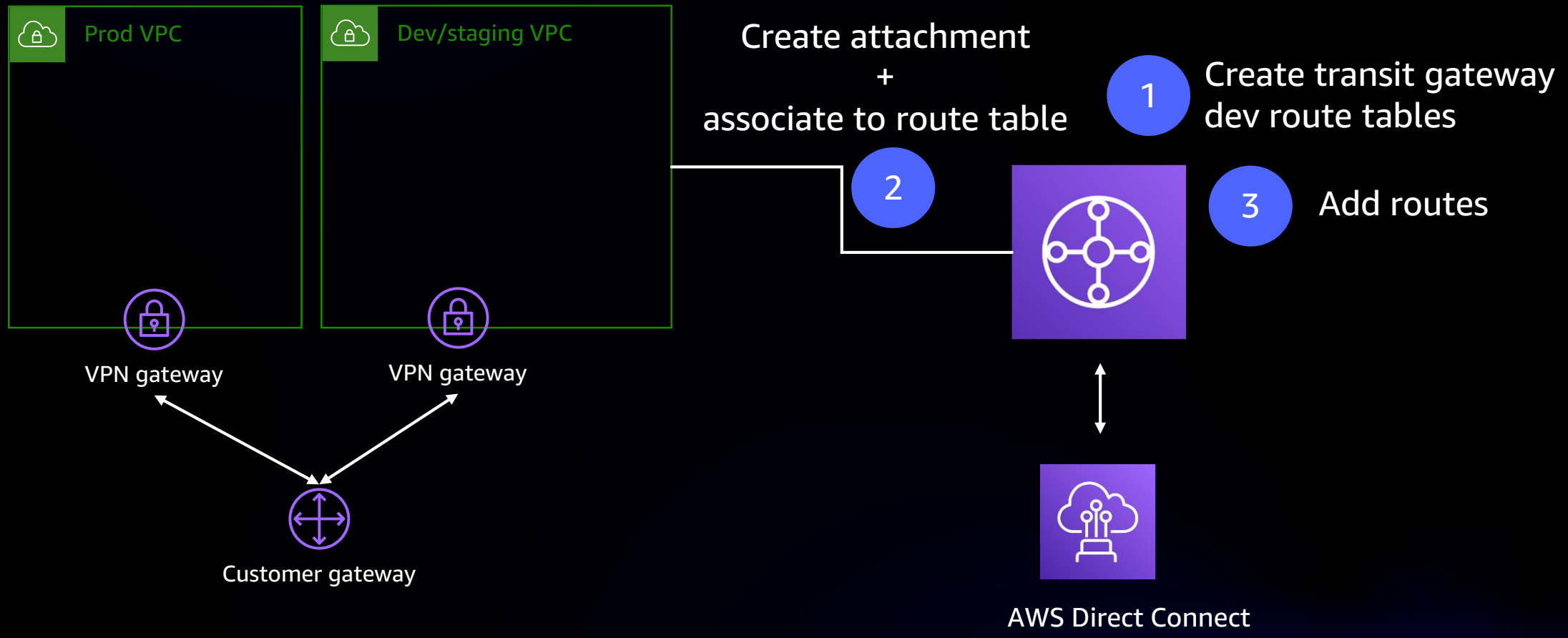
CI/CD network deployment – Example 1

MIGRATING SITE-TO-SITE VPN TO TRANSIT GATEWAY AND DIRECT CONNECT



CI/CD network deployment – Example 1

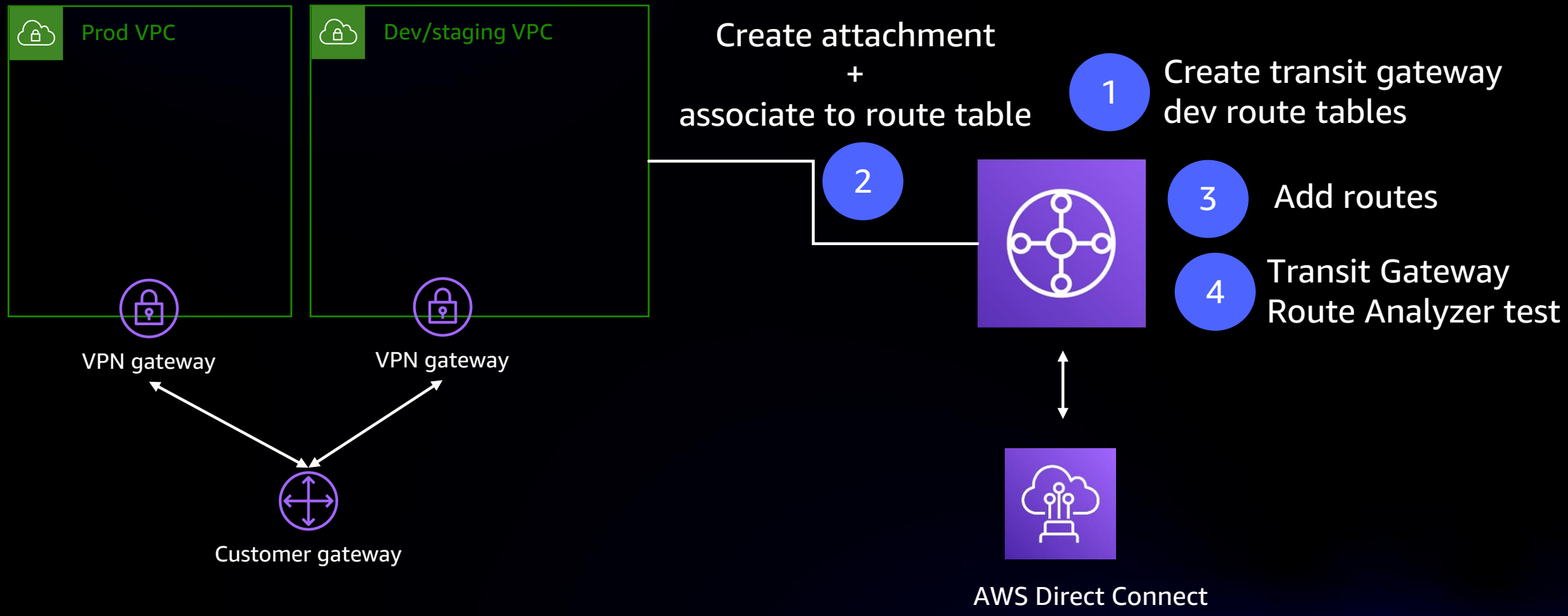
MIGRATING SITE-TO-SITE VPN TO TRANSIT GATEWAY AND DIRECT CONNECT



Remember - Code validation!
Multi-AZ VPC attachment

CI/CD network deployment – Example 1

MIGRATING SITE-TO-SITE VPN TO TRANSIT GATEWAY AND DIRECT CONNECT



Remember - Code validation!

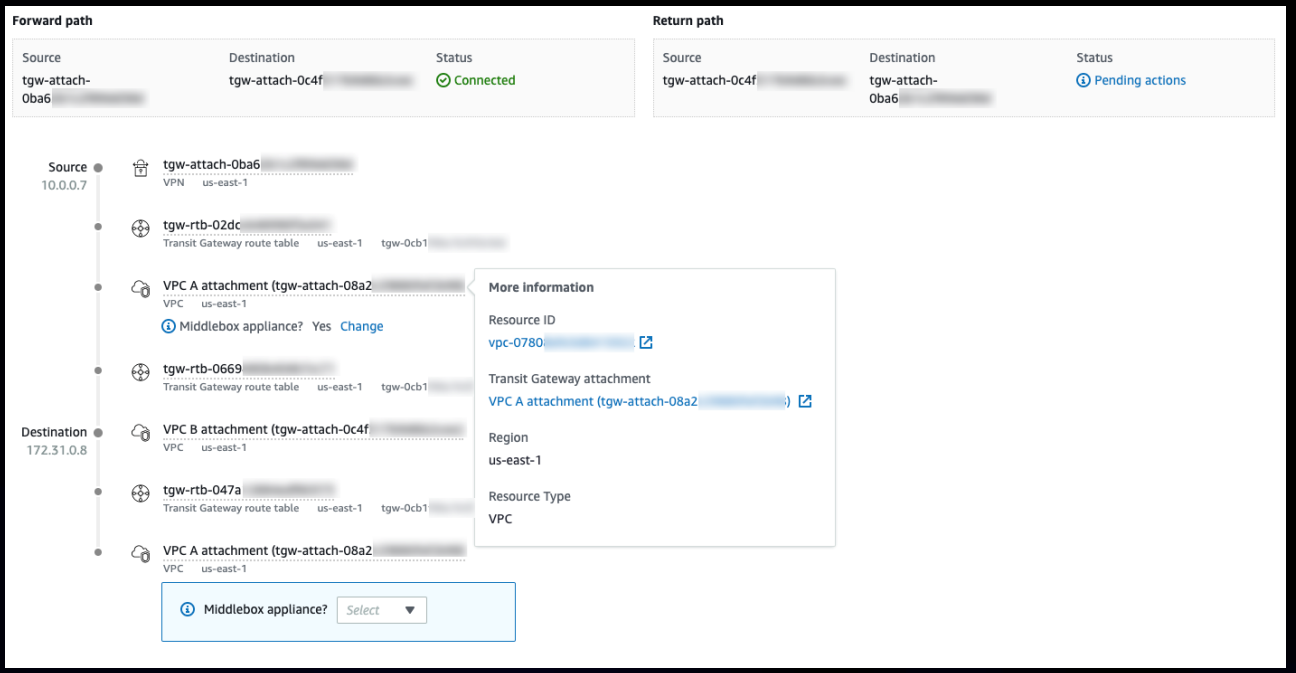
Multi-AZ VPC attachment

Environment isolation

Transit Gateway – Route Analyzer

PART OF TRANSIT GATEWAY NETWORK MANAGER

- Verify the **Transit Gateway route table** configuration
- Diagnose route-related issues that are causing traffic disruption in your global network



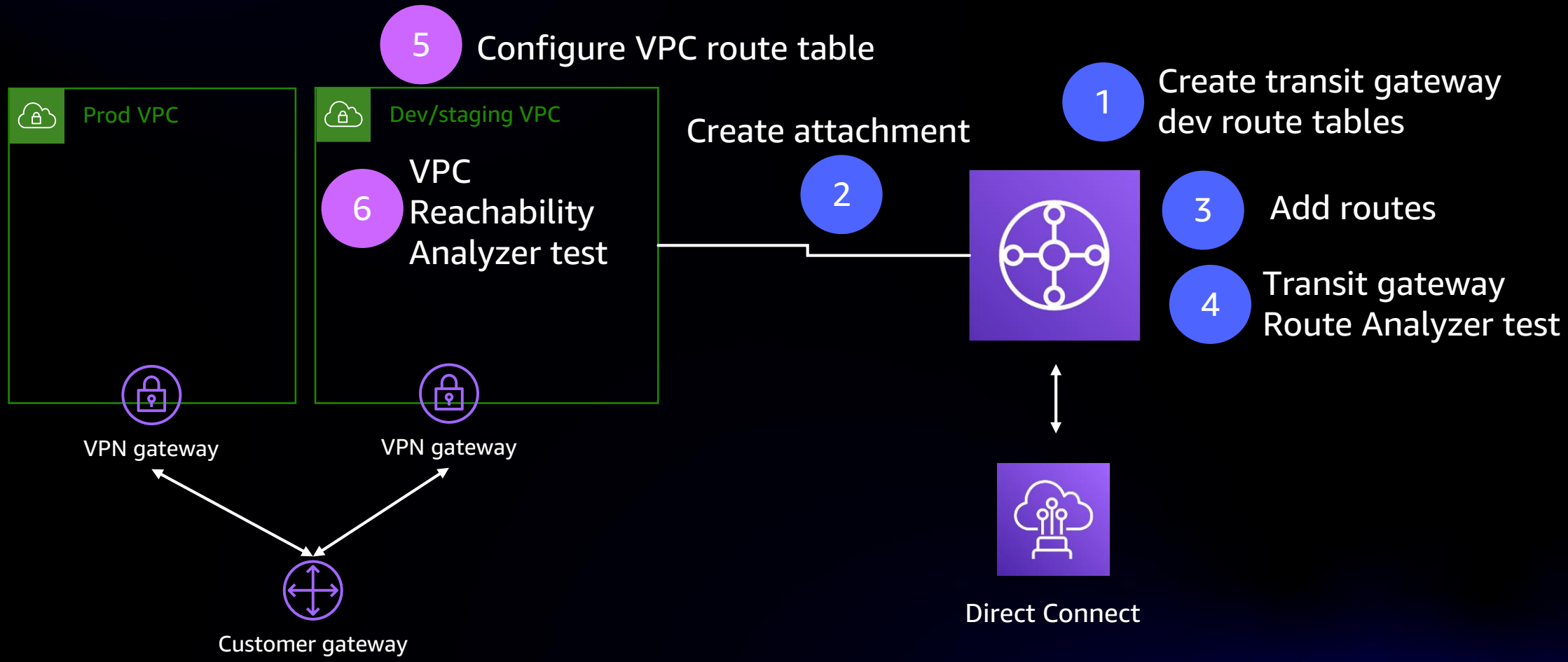
SCAN ME

https://amzn.to/2J8vDbi

Introducing Route Analyzer!

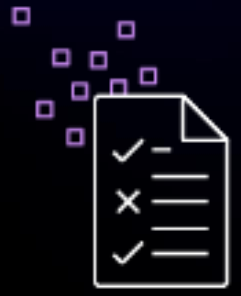
CI/CD network deployment – Example 1

MIGRATING AWS SITE-TO-SITE VPN TO TRANSIT GATEWAY AND DIRECT CONNECT

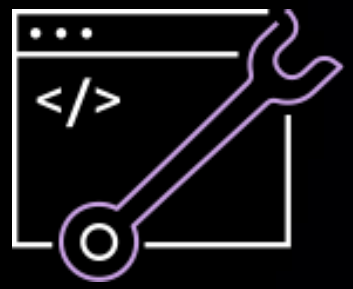


VPC Reachability Analyzer

CONFIGURATION AND NETWORK REACHABILITY ANALYSIS



Automated validation



Ensure configuration matches intent

Analyzes subnet and gateway route tables (but not transit gateway route tables)

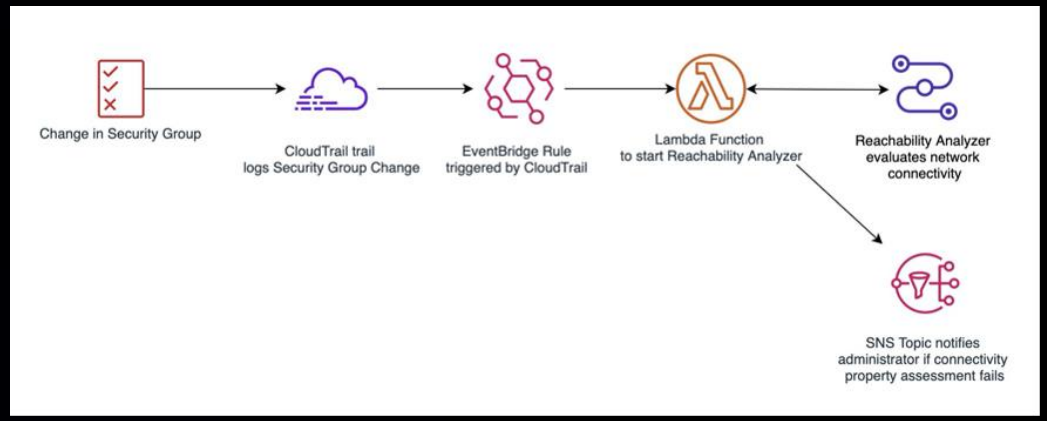
- Define source and destination components
- Optionally define intermediate components

Reference Blogs

VPC REACHABILITY ANALYZER

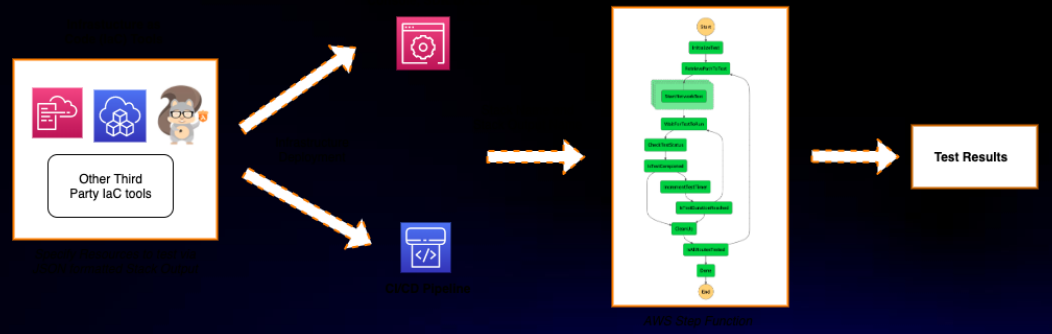
SCAN ME

Automating connectivity assessments with VPC Reachability Analyzer
https://go.aws/3nTEMER



SCAN ME

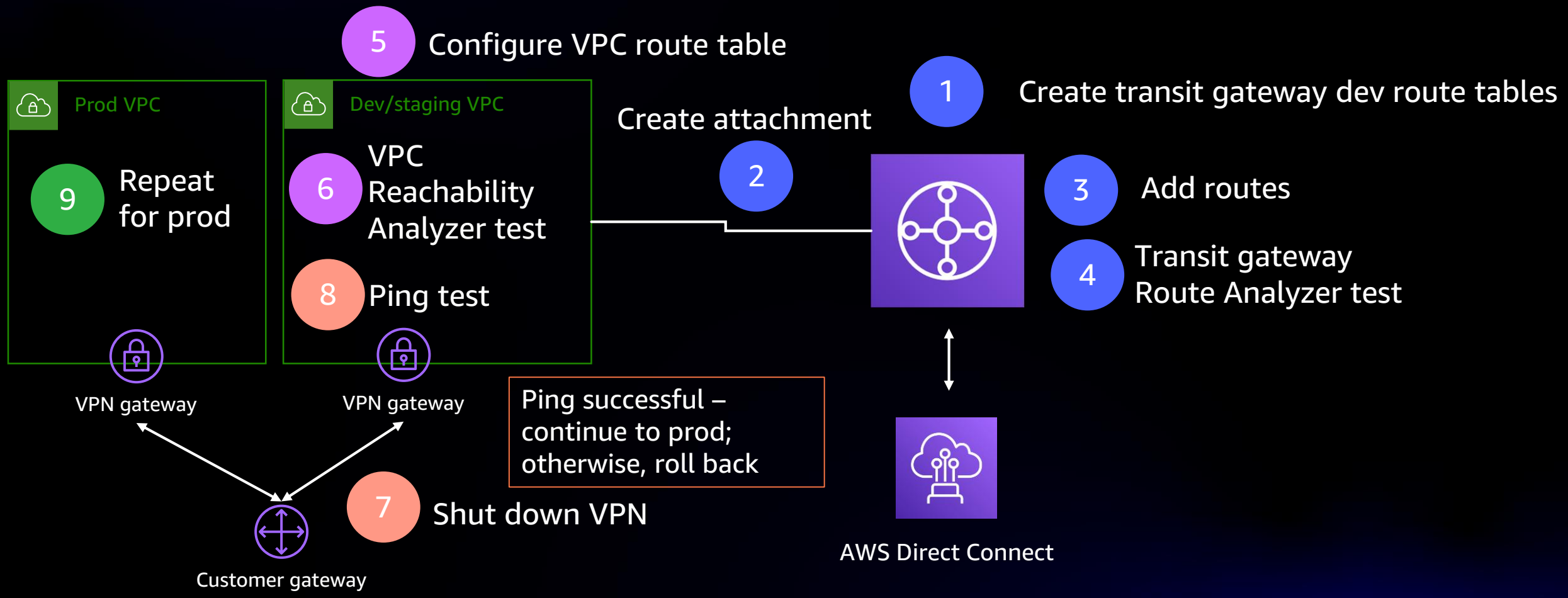
Integrating network connectivity testing with infrastructure deployment
https://go.aws/3k0tQEk



aws

CI/CD network deployment – Example 1

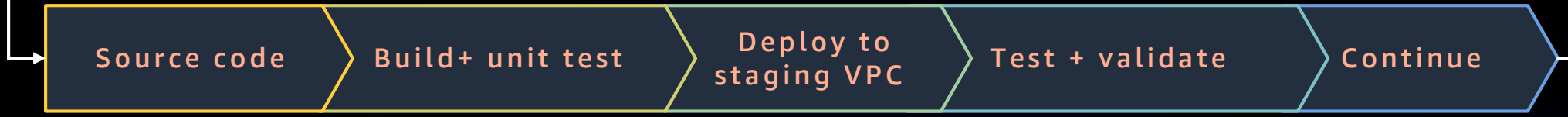
MIGRATING SITE-TO-SITE VPN TO TRANSIT GATEWAY AND DIRECT CONNECT



Phase 1 – Transit gateway (TGW) configuration change pipeline

Phase 2 – VPC configuration change pipeline

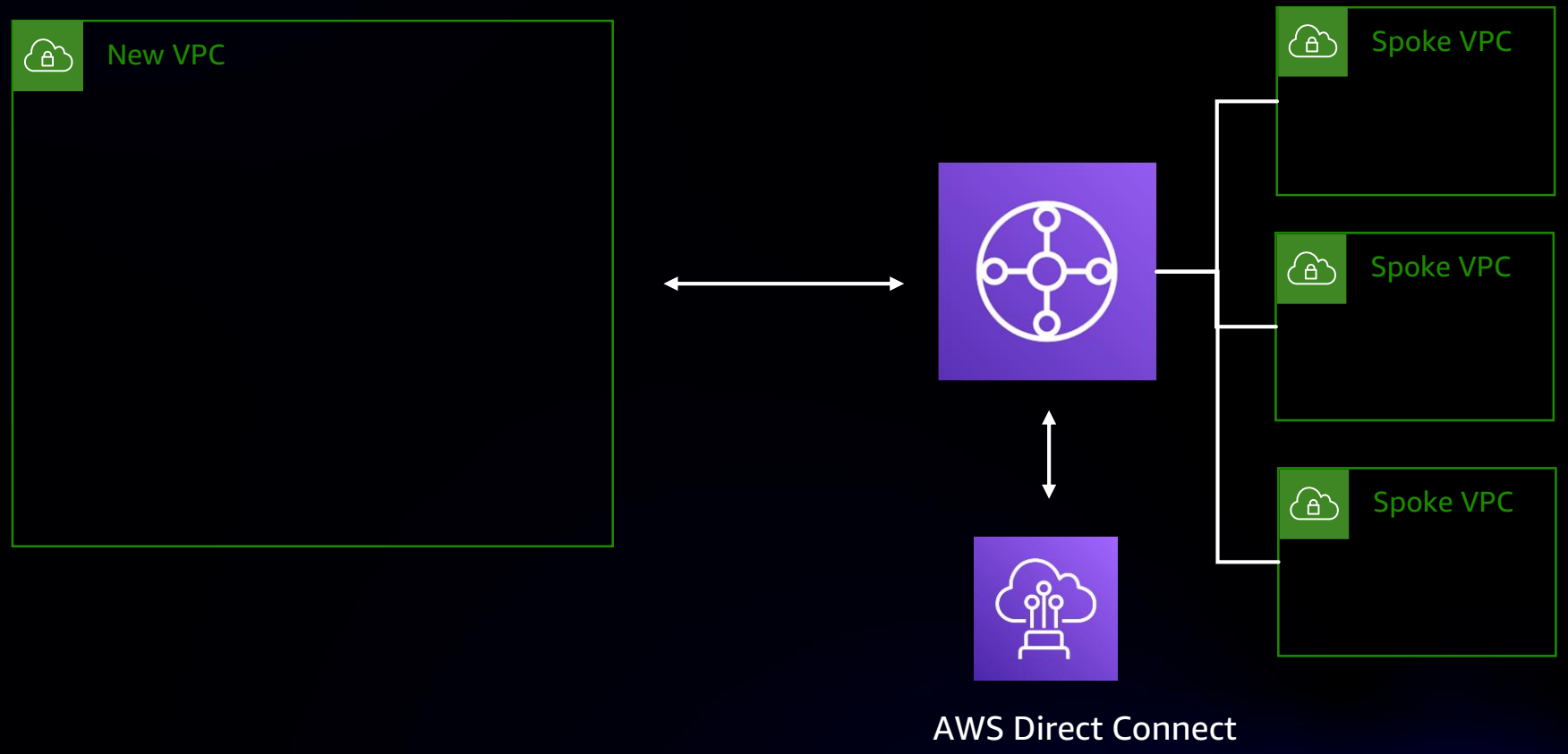
Phase 3 – Cutover change pipeline



Move to production VPC cutover

CI/CD network deployment – Example 2

CREATING A NEW VPC AND ADDING IT TO EXISTING NETWORK SETUP



CI/CD network deployment – Example 2

CREATING A NEW VPC AND ADDING IT TO EXISTING NETWORK SETUP

1

App team requests a VPC via service catalog or custom UI

Enter custom details like
- BU
- App
- Environment
- Network mask
- Subnets
- SG rules
- Etc.

Custom CloudFormation template or AWS CDK script generated

2

Transit Gateway configuration pipeline

Source code → Build+ unit test → Deploy → Test + validate → Continue

New VPC

Private subnet 1

Private subnet 2

Private subnet 3

Private subnet 4

Availability Zone

Availability Zone

3 Create and configure the VPC

AWS Transit Gateway

Networking Account

4 Initiate an attachment request

Note: Transit Gateway has to be shared with spoke account during account creation pipeline

CI/CD network deployment – Example 2

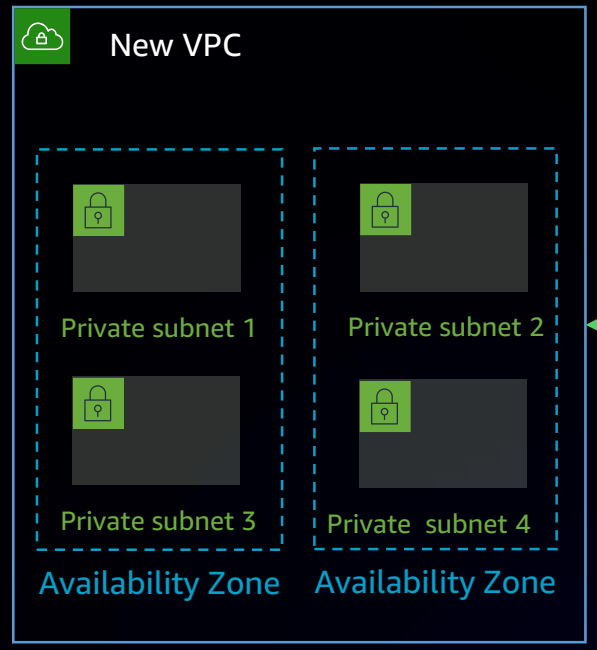
CREATING A NEW VPC AND ADDING IT TO EXISTING NETWORK SETUP

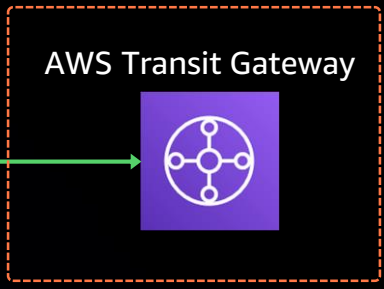
Trigger attachment pipeline in central networking account (approval optional)

SCAN ME

Automating AWS Transit Gateway attachments to a transit gateway in a central account

https://go.aws/3mDtHZ8





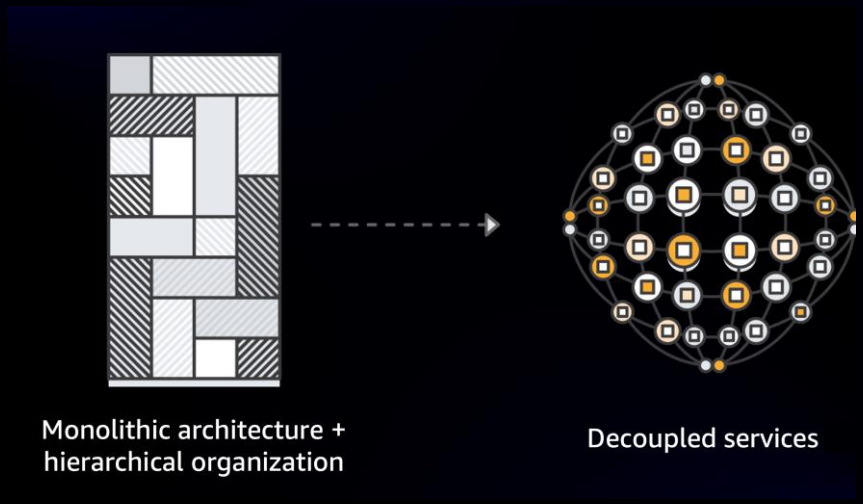
6 Associate attachment to a route table

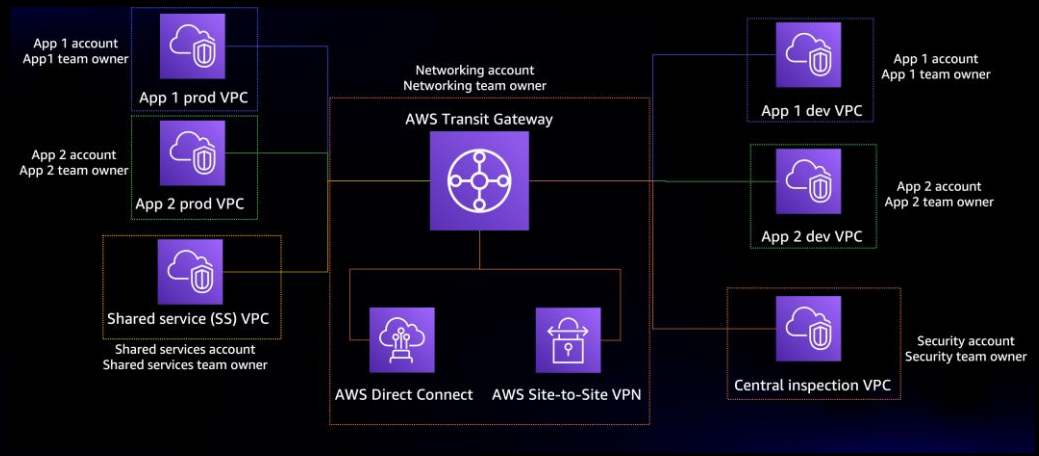
7 Configure routes/prefix-lists or enable propagation

5 Accept attachment request

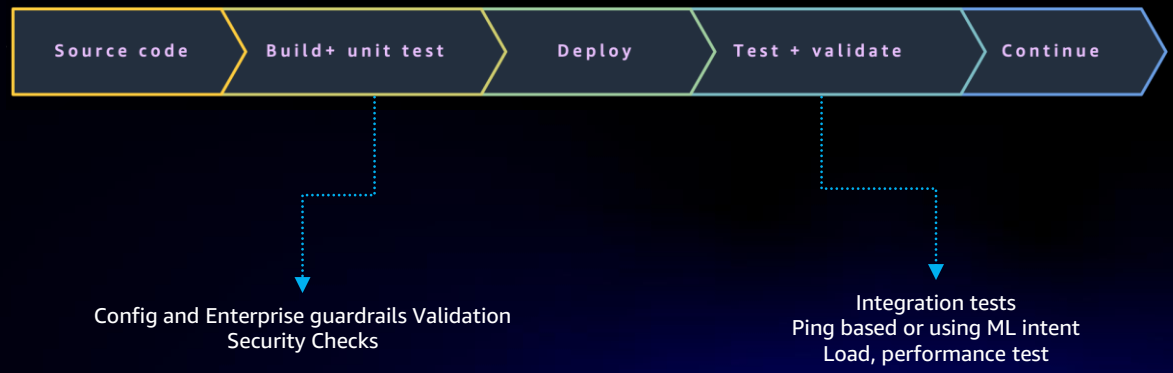
NetDevOps

LET'S RECAP






```
const egressVPC = new ec2.Vpc(this, 'Egress VPC', {
  cidr: "10.0.1.0/26",
  //natGateways: 1, add this to limit numer of deployed NAT gateways
  subnetConfiguration: [{
      cidrMask: 28,
      name: 'Public - EgressVPC SubNet',
      subnetType: SubnetType.PUBLIC,
    },
    {
      cidrMask: 28,
      name: 'Private - EgressVPC SubNet',
      subnetType: SubnetType.PRIVATE,
    },
  ]
});
```



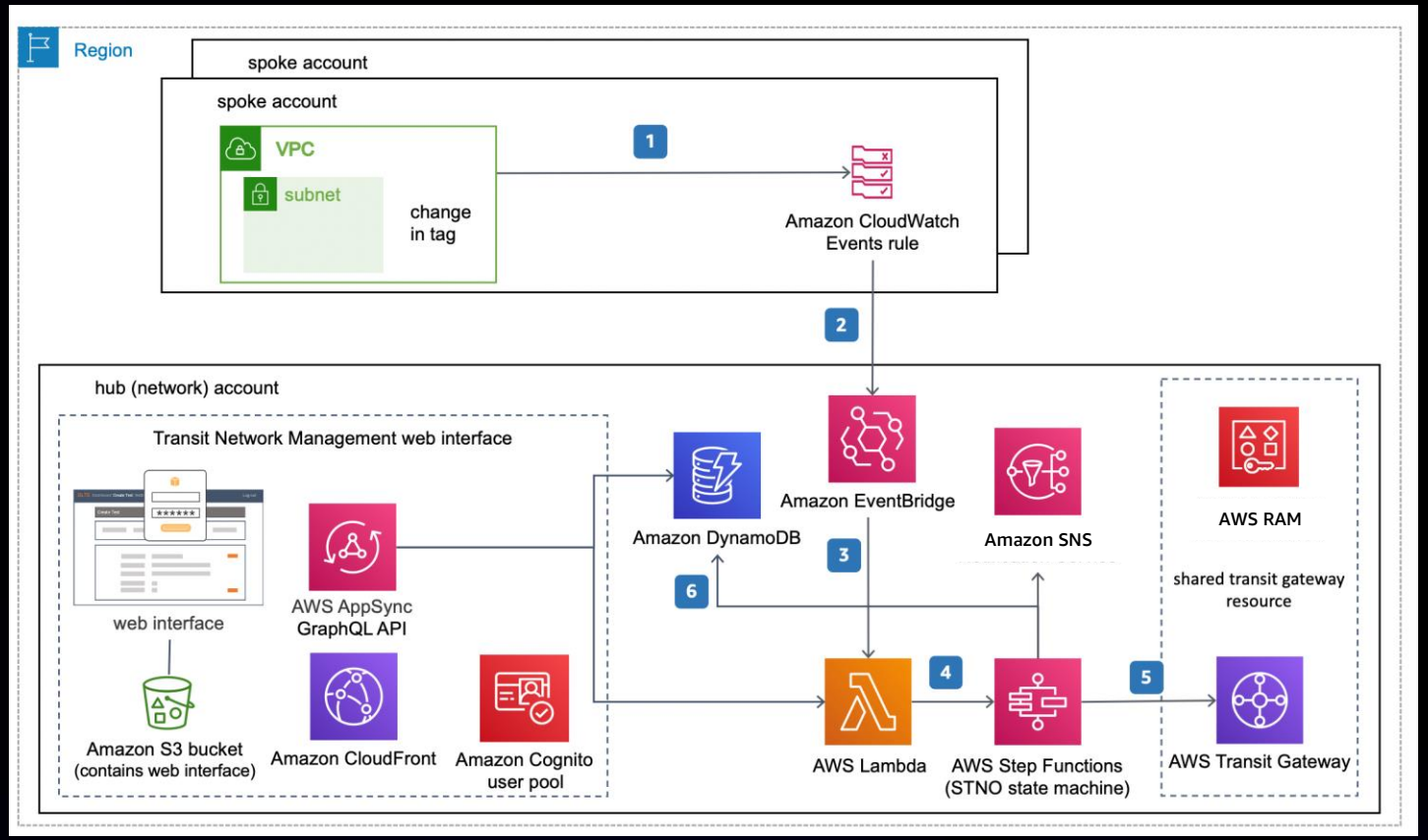
AWS solutions to get you started

STNO (SERVERLESS TRANSIT NETWORK ORCHESTRATOR)

- **Automates** the process of setting up and **managing transit networks** in distributed AWS environments
- Provides a **web interface** to help **control, audit, and approve** (transit) network changes

SCAN ME

https://go.aws/3k04I0K



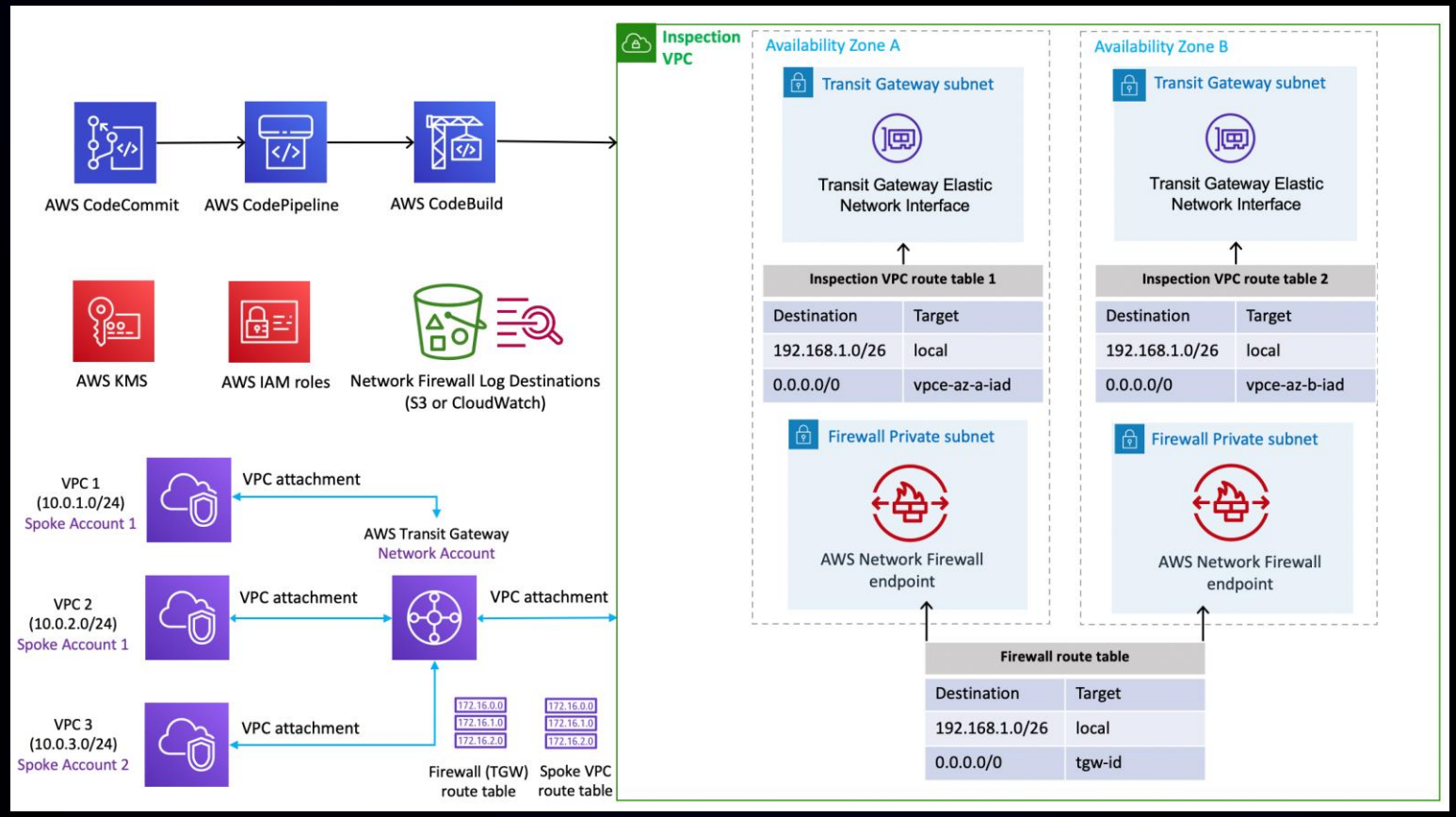
AWS solutions to get you started

AWS NETWORK FIREWALL DEPLOYMENT AUTOMATIONS FOR AWS TRANSIT GATEWAY

Automatically deploy changes to AWS Network Firewall

Centrally manage your Network Firewall

Audit and track changes to Network Firewall



SCAN ME

https://go.aws/3EIe3Sq

aws

Network automation at GE

Agenda

Background

Benefits of automation

Automation today

Migration to Transit Gateway

Automation tomorrow

General Electric: Who we are

- Global
- Company of companies
- 174,000 employees
- Partnered early on with AWS
  - Company challenge – move 50 apps in 60 days (Nov 2013)
- Today – utilize almost every AWS service

Cloud Hosting: Who we are

Cloud Hosting is a horizontal

- 50 DevOps engineers with multi-platform pro-level cloud certifications
- 1 UX brand manager

Work closely with several teams

- Network engineering and operations
- DNS engineering and operations
- Cyber, compliance, and threat intelligence
- Identity
- GE business units (customers)

Cloud Hosting: What we do

- VPC builds/decommissions (products)
  - Connected and disconnected to/from our network
  - Bundled security controls
- Cloud billing
- Cloud education
- Cloud networking
- Cloud identity
- Build tools/microservices to protect/govern GE (and make all our lives easier)
- Automate, automate, automate

Cloud Hosting: Philosophy

- Customer independence but with guardrails
- Accelerate innovation
- Stay on the bleeding edge
- Continue to push boundaries
- Cloud native first
- Automation is a fabric of our services

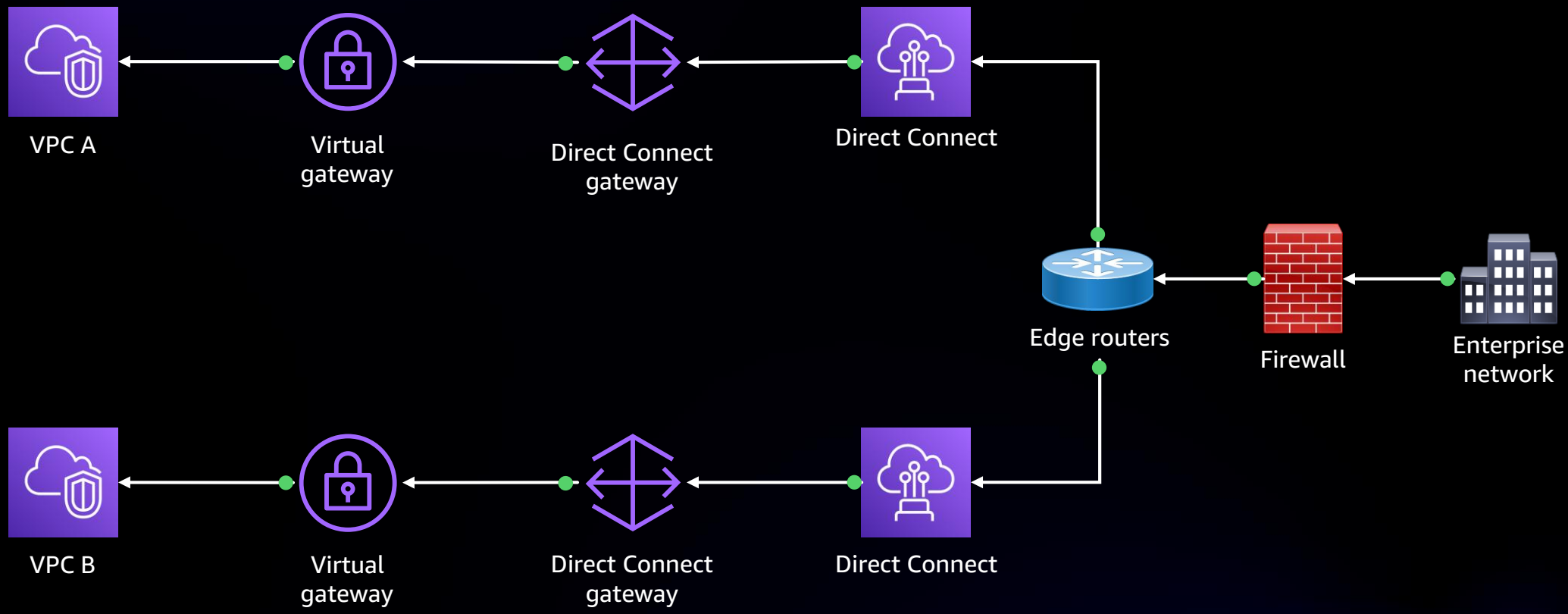
Ordering platform

- Internal team orders connected/disconnected VPC
- Requires approval
- Cloud Hosting team receives order
- Build AWS resources
- Build network resources
- Quality assurance checks
- Account released
- Later – account decommissioned

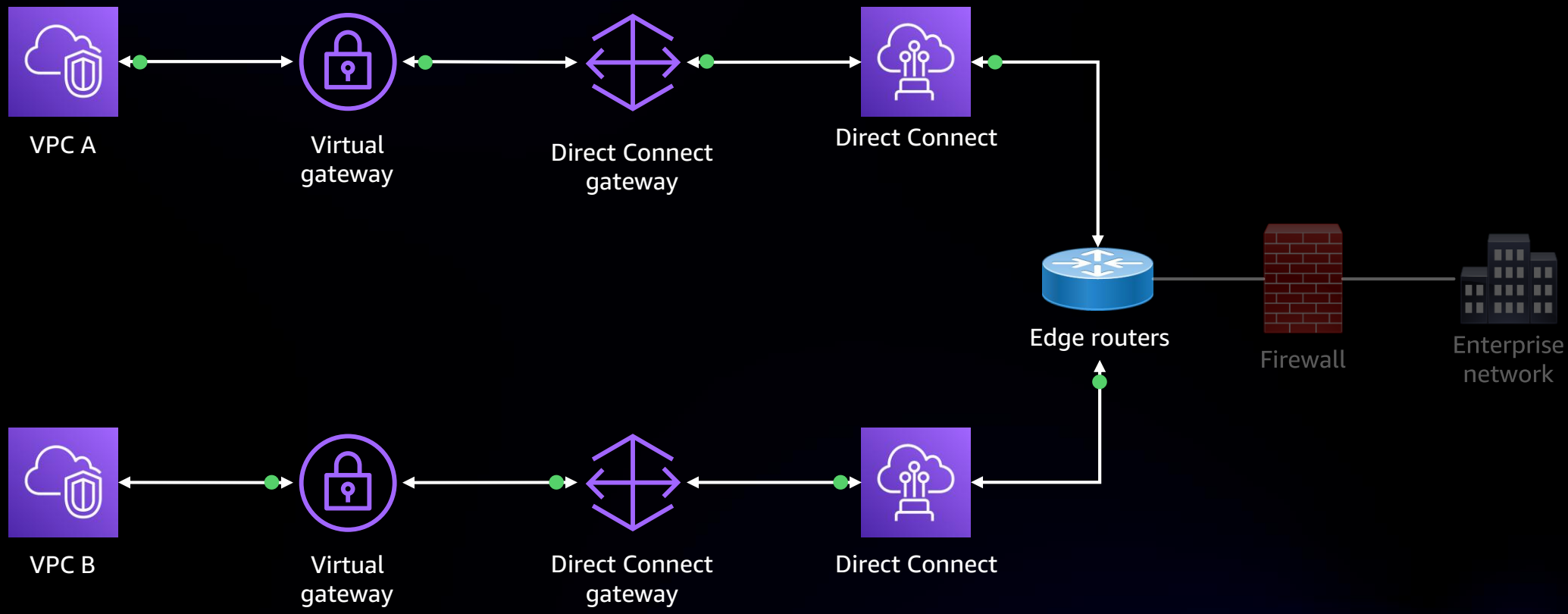
Today’s network connectivity

Global presence

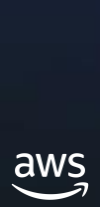
Today’s architecture



VPC-to-VPC traffic



Automation



Automation

- Less overhead
- Less room for human error
- Updating company’s primary cloud edge routers
- Network builds sped up from 6 weeks to 1 week – no dependencies on network or firewall teams
- Some initial pushback at the start
- Lots of testing required

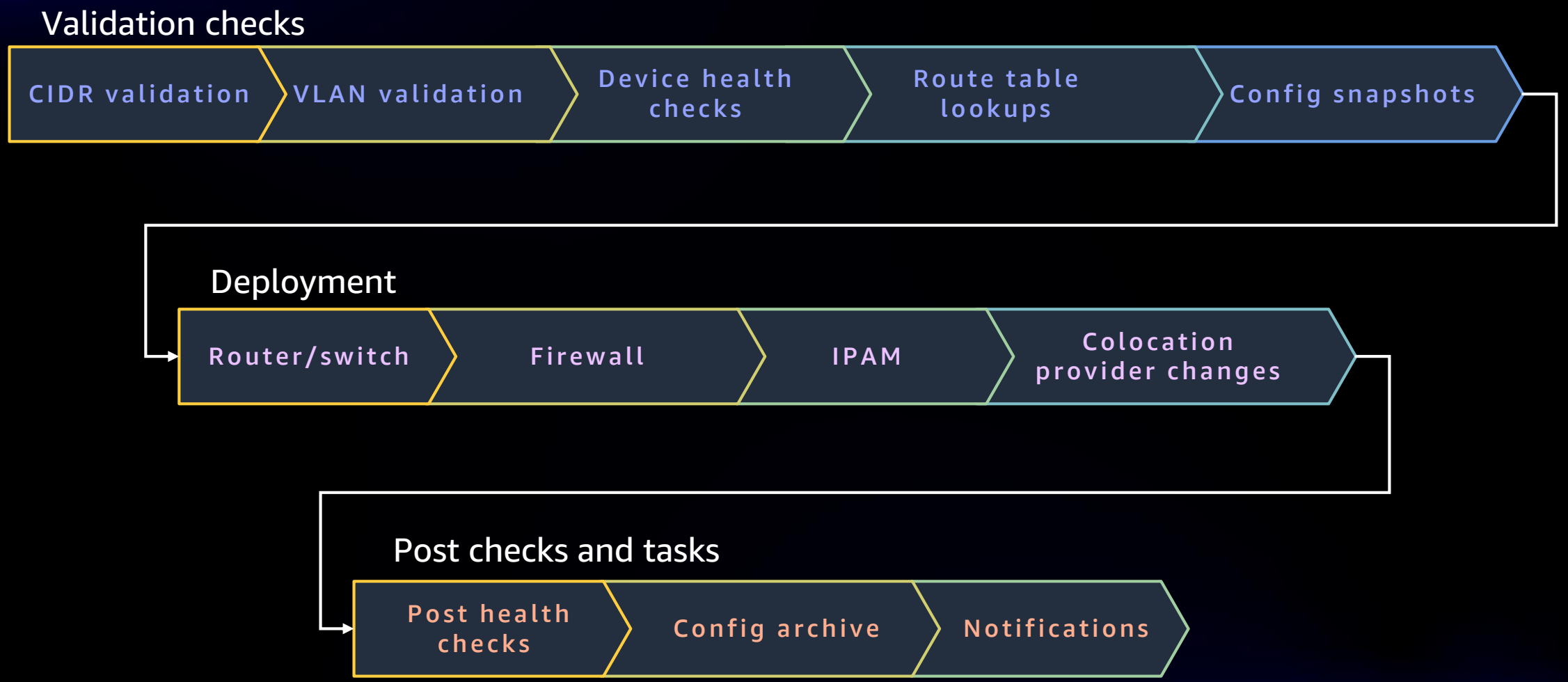
Tools we use

aws

Use case #1: CloudNet playbook

- Automation playbook that connects accounts to the enterprise network
- Network device configurations
- Firewall
- IPAM
- Colocation provider

Use case #1: CloudNet playbook

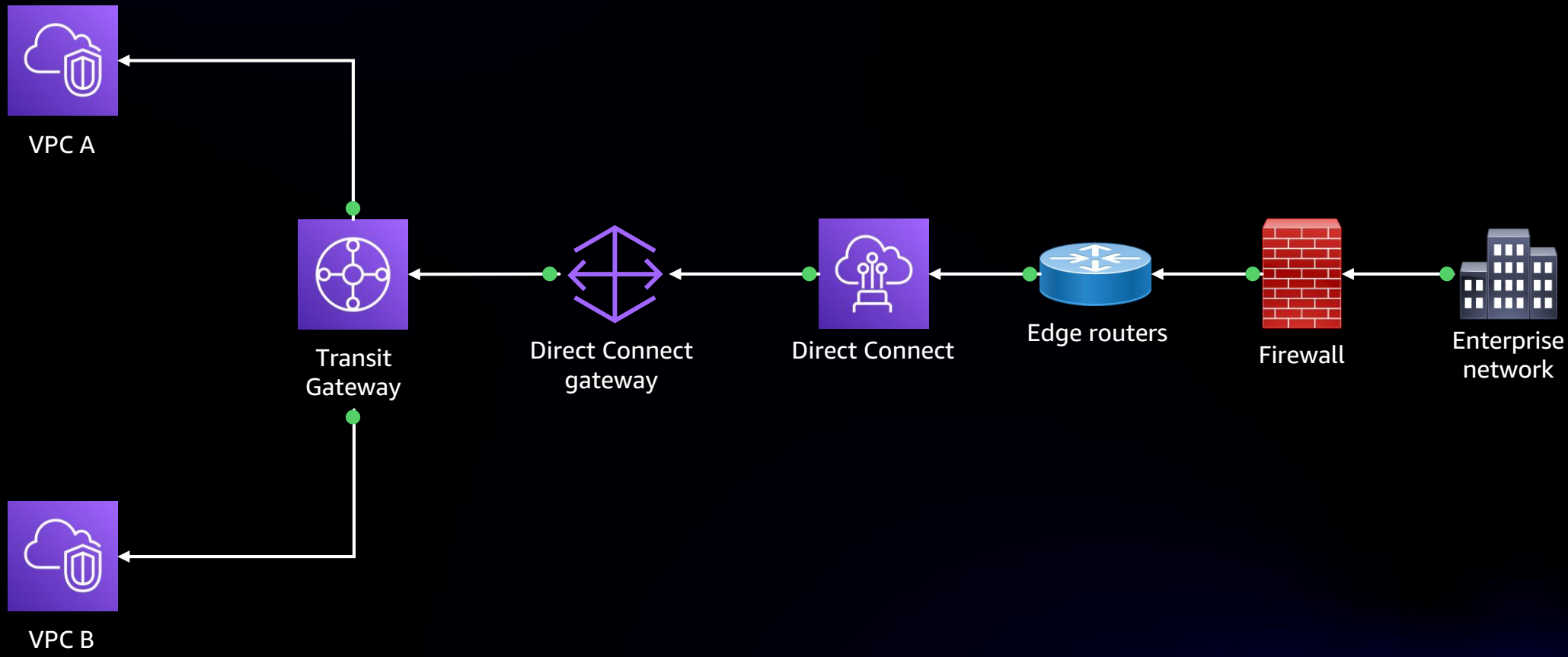


Tomorrow's network connectivity

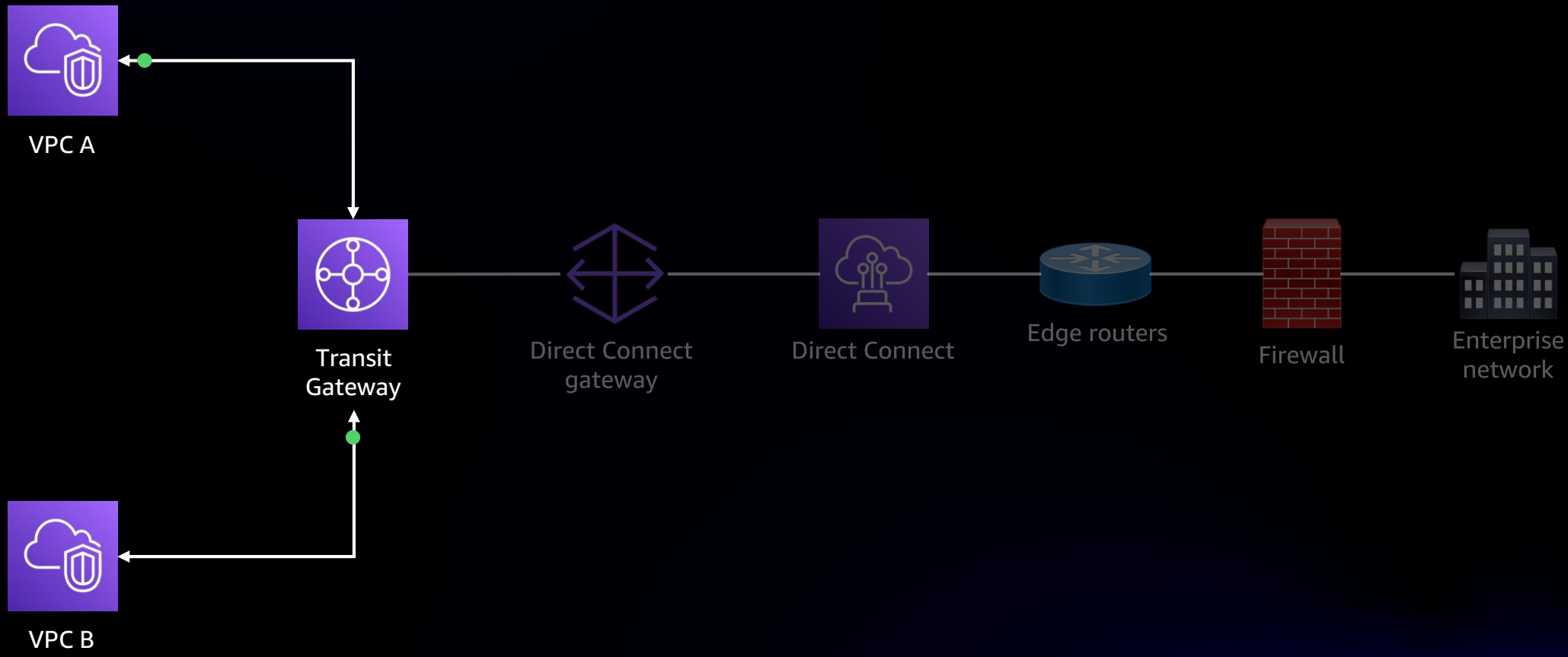
Migration to Transit Gateway

- Opportunity to centralize connectivity model
- Simplifies peering configurations
- Makes life easier for network engineering/operations
- Keeps cloud traffic within the cloud
- Reduces Direct Connect utilization
- Improves latency
- Enables fine-grained routing control within AWS
- Enables cloud-native firewalls

Tomorrow’s architecture



VPC-to-VPC traffic



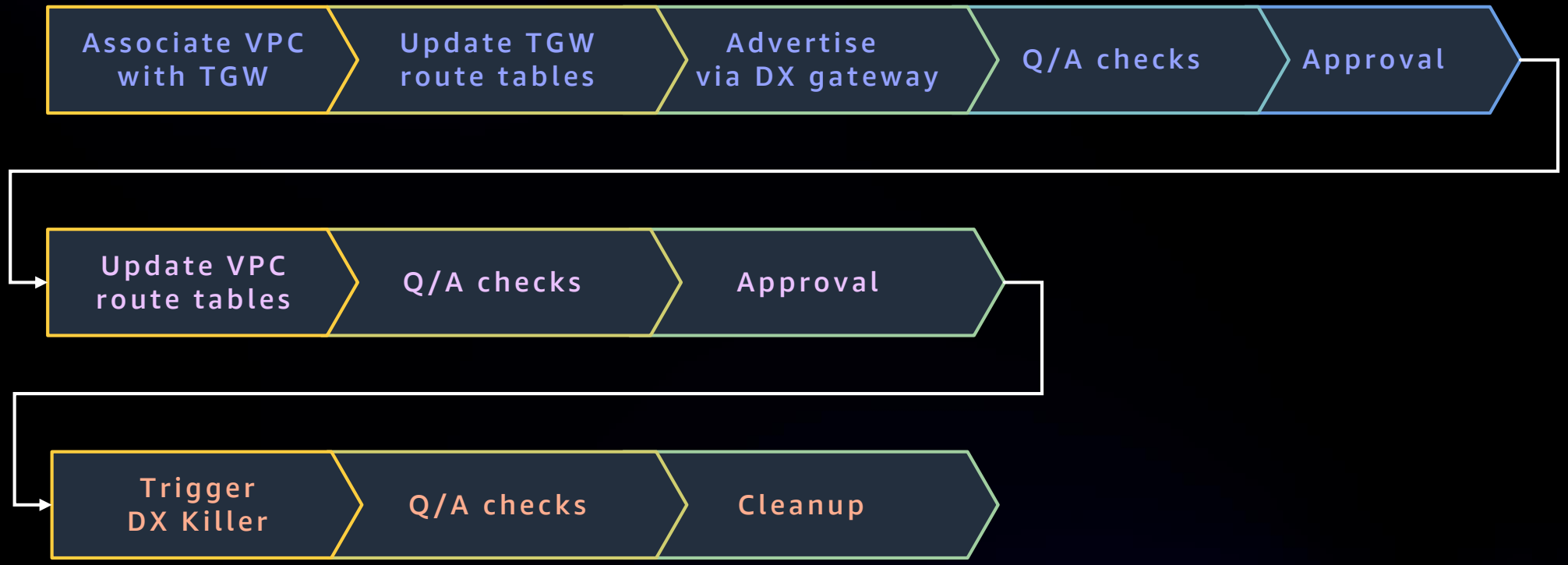
Use case #2: Transit Gateway playbook

- Works with new or existing VPCs
- Minimizes risk of network outage on migrations
- Supports rollback in the event of a connectivity issue
- Propagates BGP route updates to child accounts
- Asymmetric routing

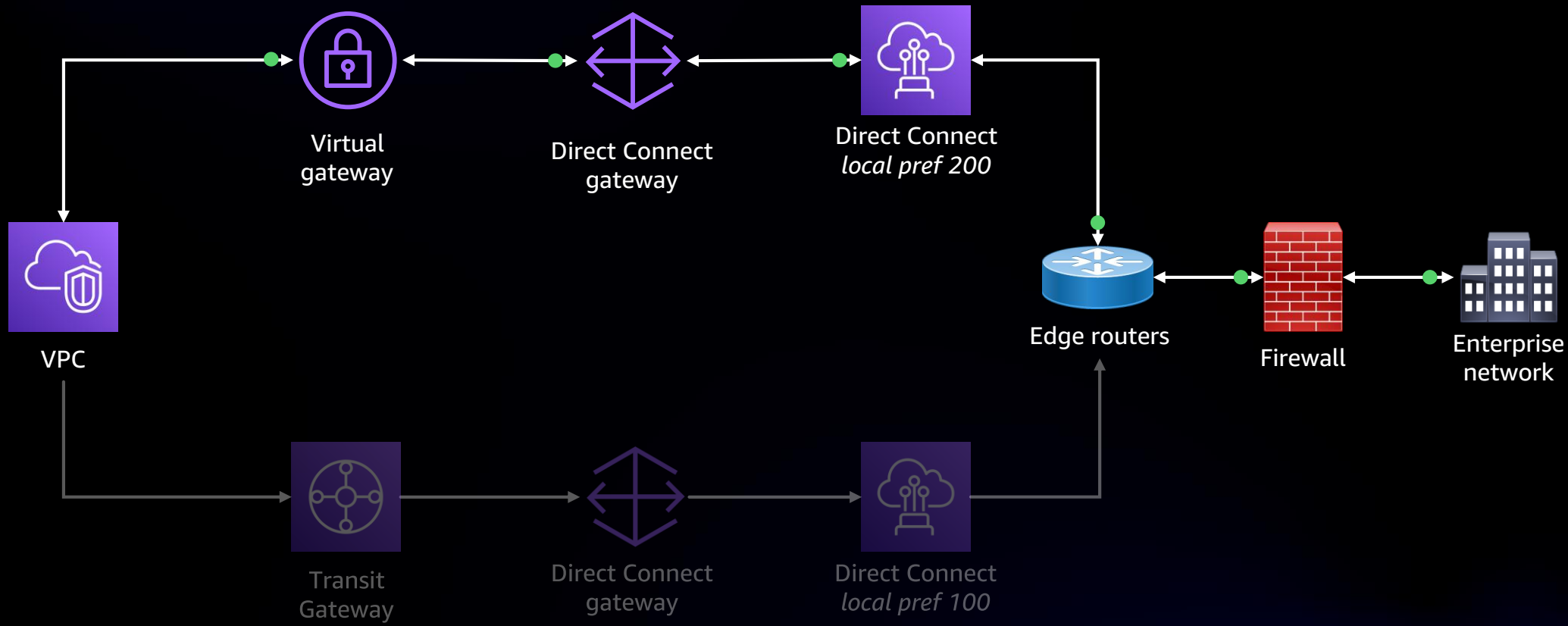
Use case #2: AWS resources automated

- Transit Gateway
- Transit Gateway route table
- Transit Gateway Network Manager
- AWS RAM
- VPC-managed prefix list
- VPC route table
- Direct Connect gateway
- Direct Connect virtual interface

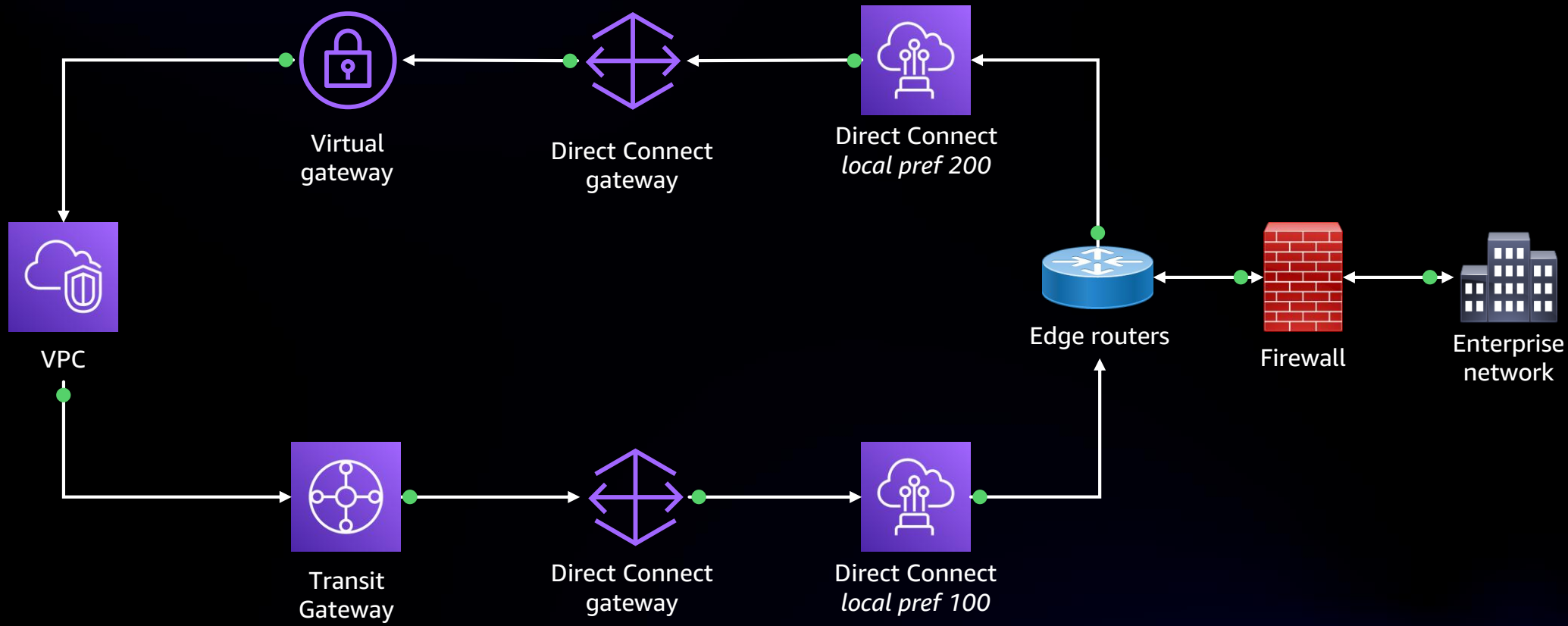
Use case #2: Playbook steps



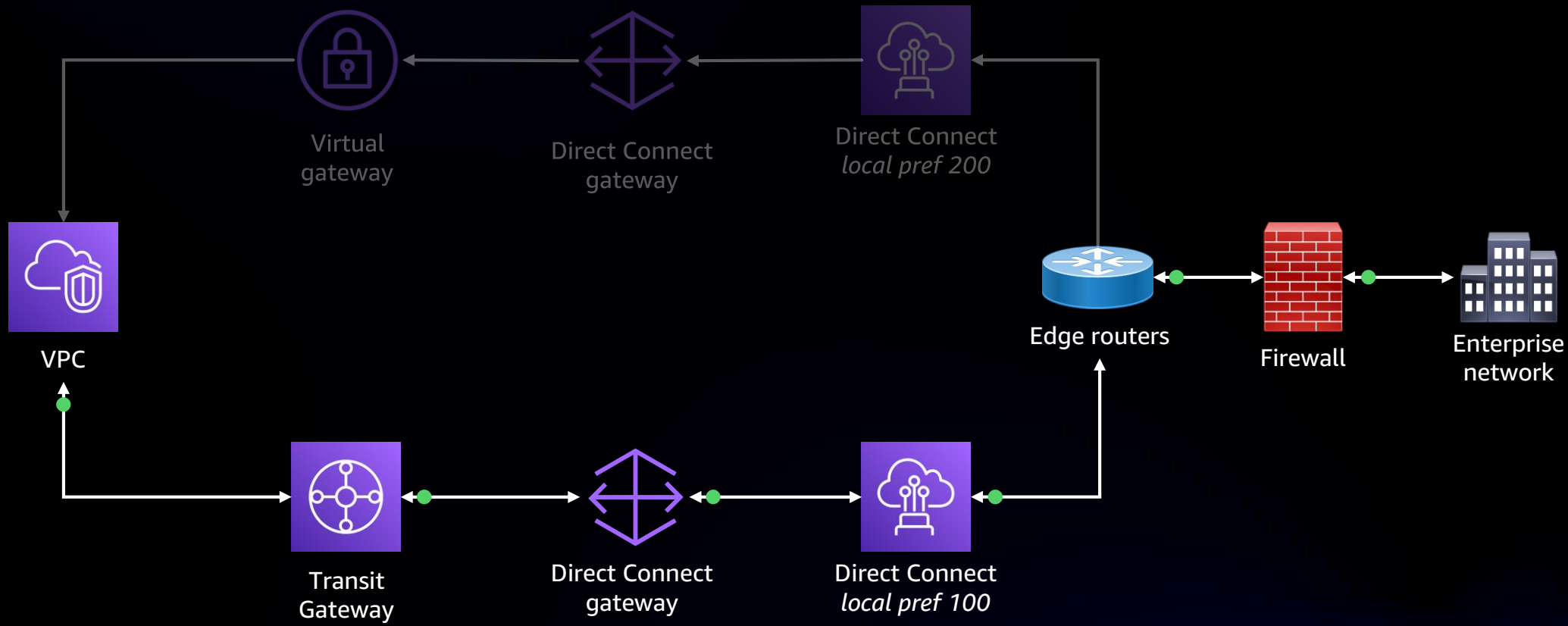
Pre-migration routing



Routing during migration



Post-migration routing



Use case #3: DX Killer playbook

- Avoids outage when disassociating VGW from VPC/DXGW
- Network engineers do not need to be present for migrations
- Supports rollback in case of connectivity issues
- Puts full control of routing in our hands
  - Ensures TGW BGP peer is ready to accept the traffic
  - Shuts down BGP peers of existing interfaces
- Achieve zero-downtime migrations – requires traffic to traverse same firewall

Use case #3: DX Killer playbook

Takeaways

- Automation is great
- But there is risk – test, test, test!
- And then test some more!
- Log everything
- Have a manual intervention plan
- Write documentation

Thank you!

Sid Chauhan
linkedin.com/in/sidaws/

Ákos Varga
linkedin.com/in/akos-varga-/

Michael Palmer
linkedin.com/in/michaeldpalmer/