
cisco Live!
Let's go

The bridge to possible

# Zero Touch Provisioning & Configuration Management of Cisco FTD in Azure

## Using Terraform and Ansible

Ed McNicholas, Sr. Cybersecurity Architect
@don't have one

Cisco Live!

DEVNET-2150

# Cisco Webex App

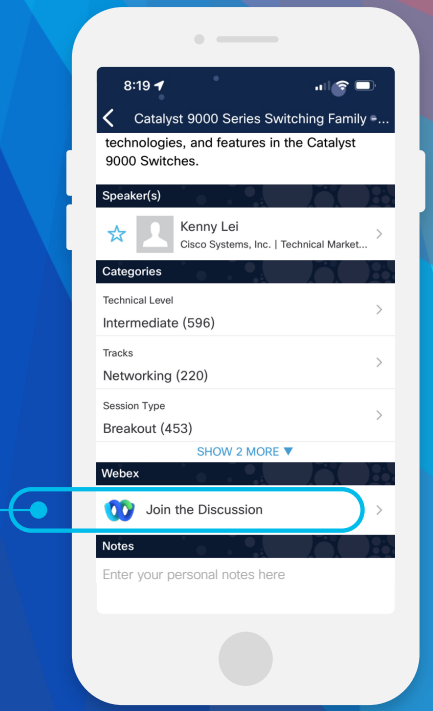
## Questions?

Use Cisco Webex App to chat with the speaker after the session

## How

1. Find this session in the Cisco Live Mobile App
2. Click “Join the Discussion”
3. Install the Webex App or go directly to the Webex space
4. Enter messages/questions in the Webex space

## Webex spaces will be moderated by the speaker until February 23, 2024.

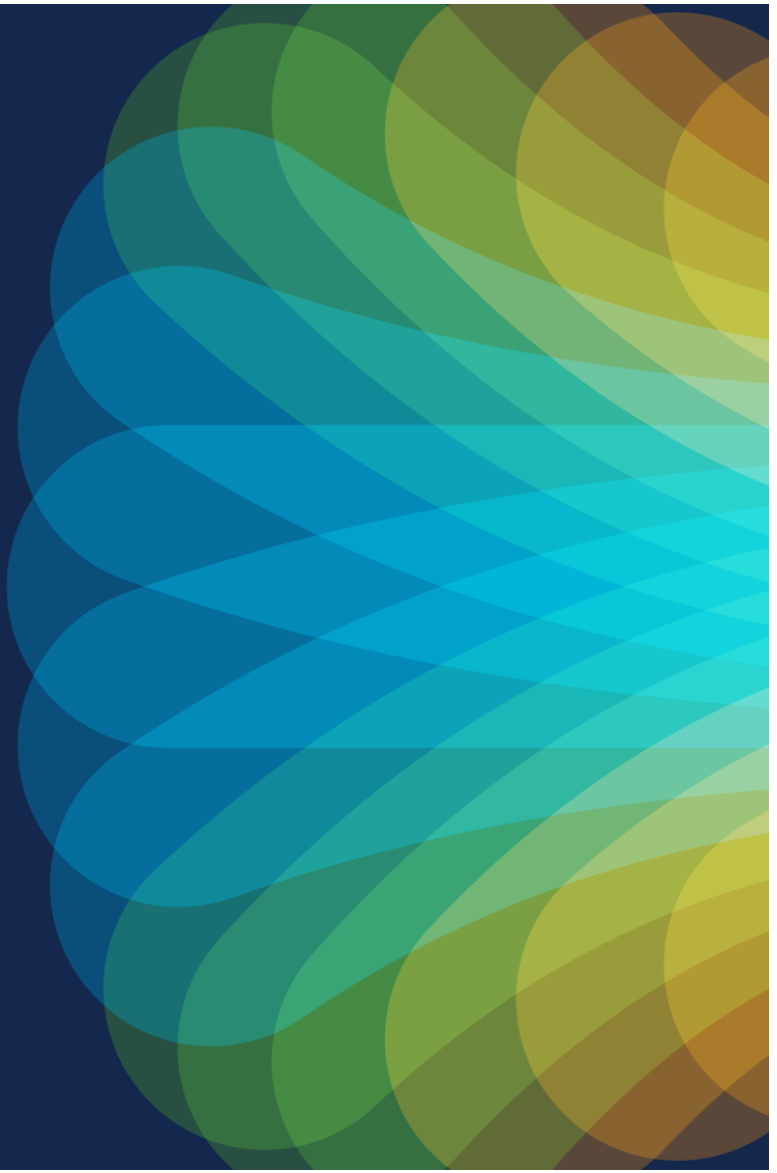



https://ciscolive.ciscoevents.com/ciscolivebot/#BRKXXX-xxxx

# Agenda

- Introduction
- Azure
- Terraform
- Ansible
- Conclusion

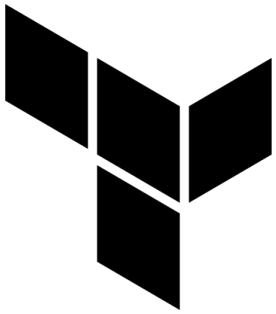
# Introduction



cisco Live!

# Automating FTD in Azure

Azure



Terraform

Ansible

Secure Firewall

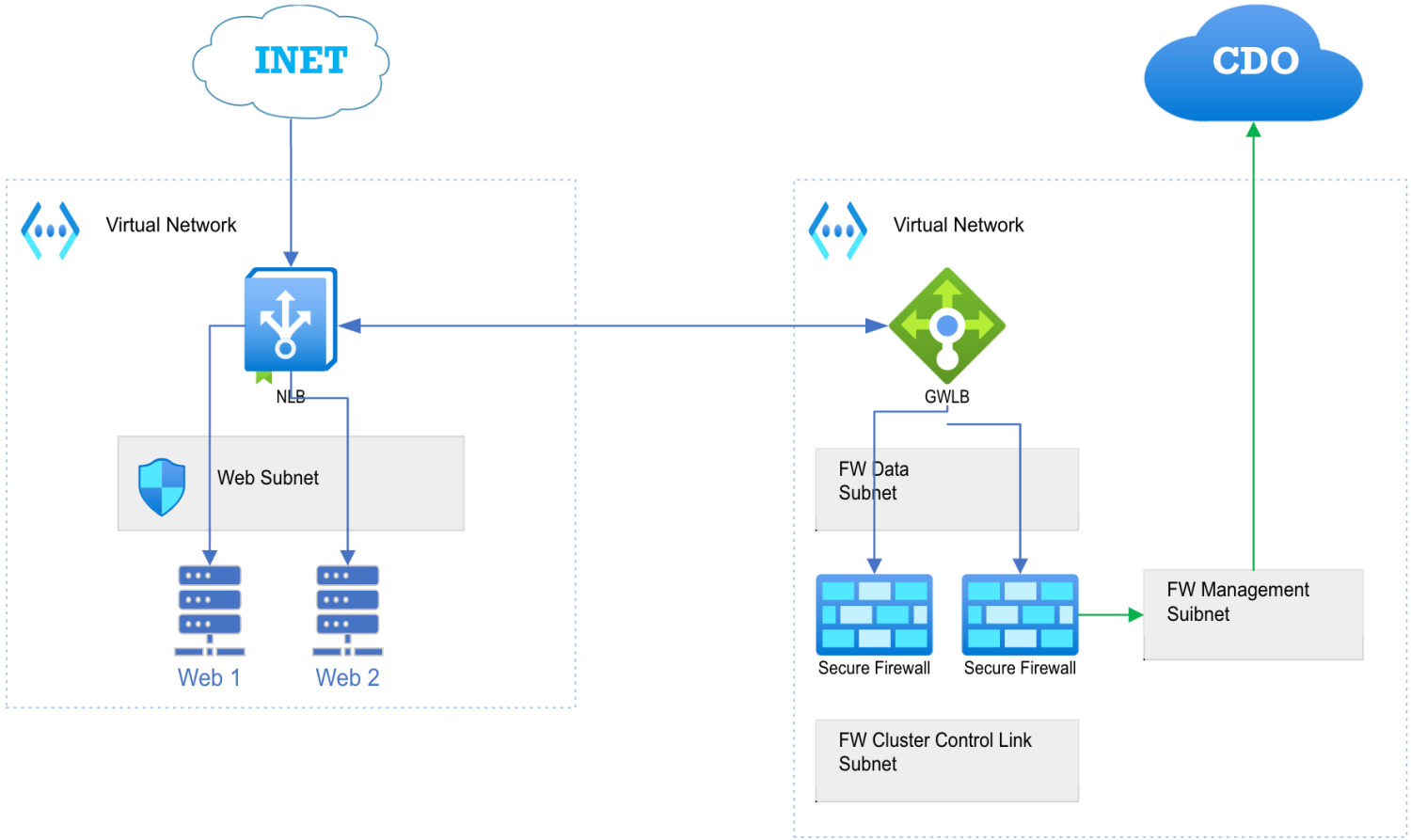

Defense Orchestrator

**GitHub Repo - https://github.com/emcnicholas/Azure_TF_GWLB.git**



cisco Live!

# The Architecture



# Before diving in headfirst

# Let's talk terminology

cisco Live!

# Ansible – Imperative & Mutable


- Shallow Learning Curve
- Agentless
- Supports Every Device from Every Vendor
- Modular
- Open-Source Community and Vendor Commercial Support
- Common Toolset with server / DevOps / infra teams

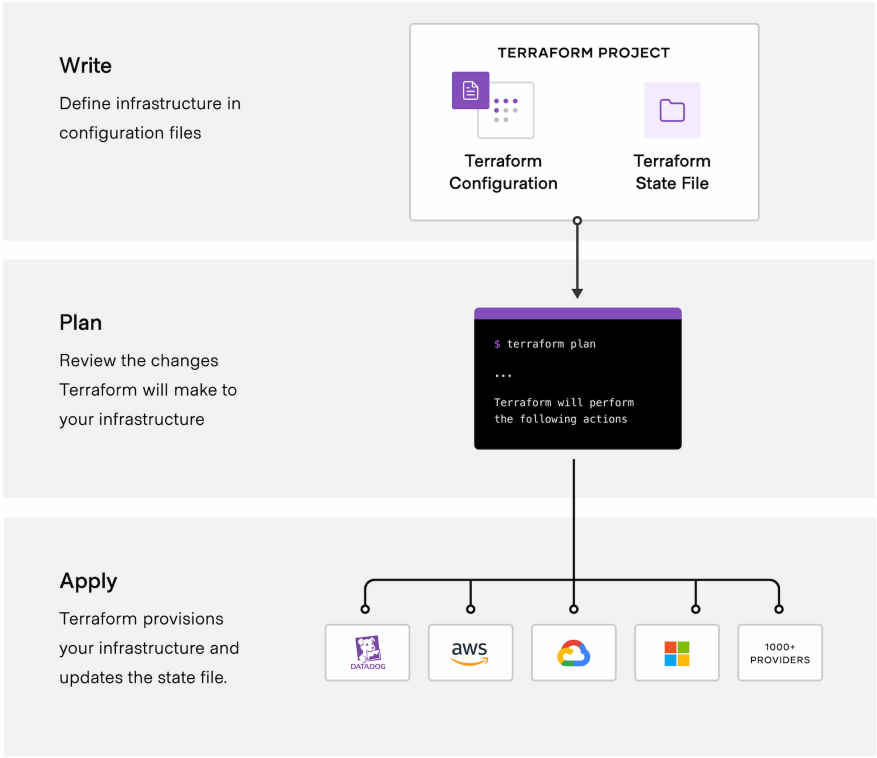
# Terraform – Declarative & Immutable

- Developed by HashiCorp 

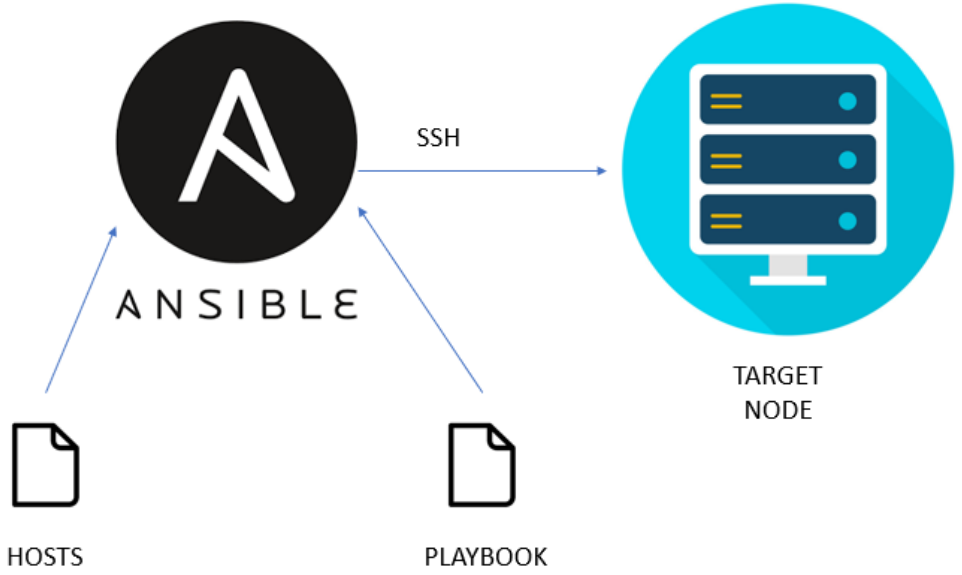
- Initial release in July 2014
- Designed to be a full Infrastructure as Code (IaC) management tool for Cloud Infrastructure Provisioning
- Completely written in Go, creating a single binary file 

- Fully declarative leveraging HashiCorp Configuration Language (HCL)

# Terraform to Deploy



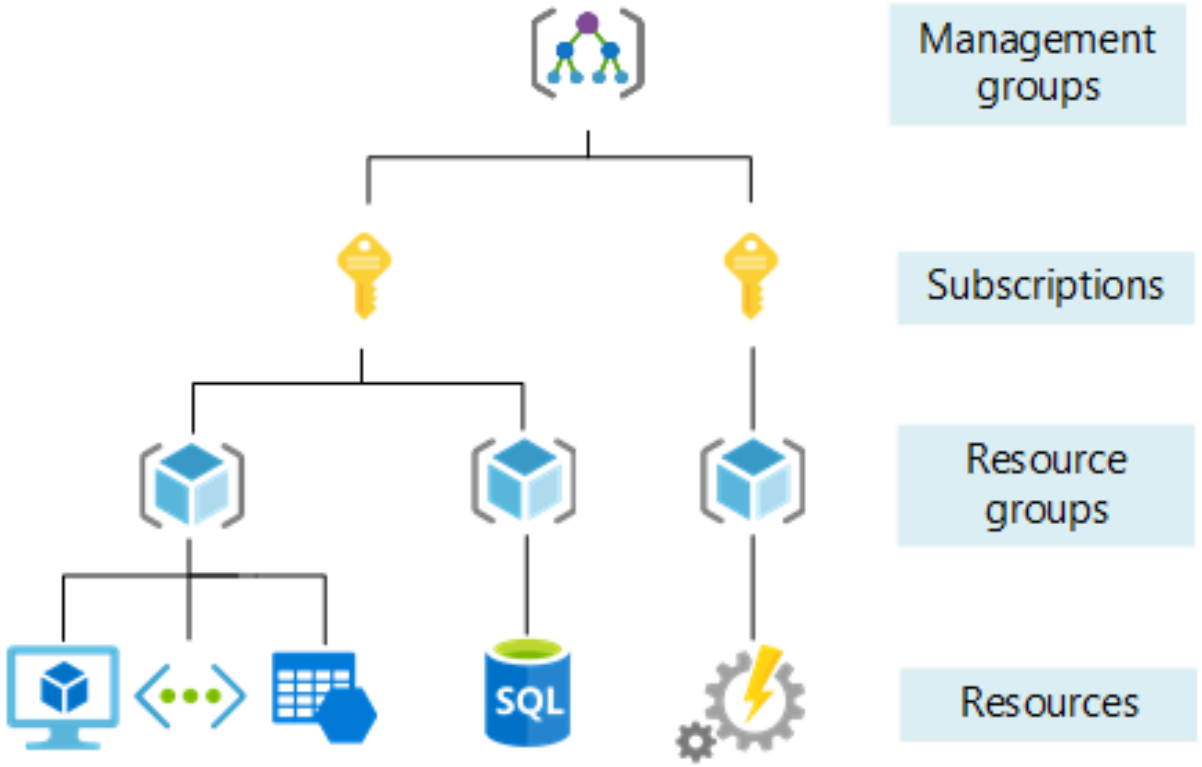

# Ansible to Configure



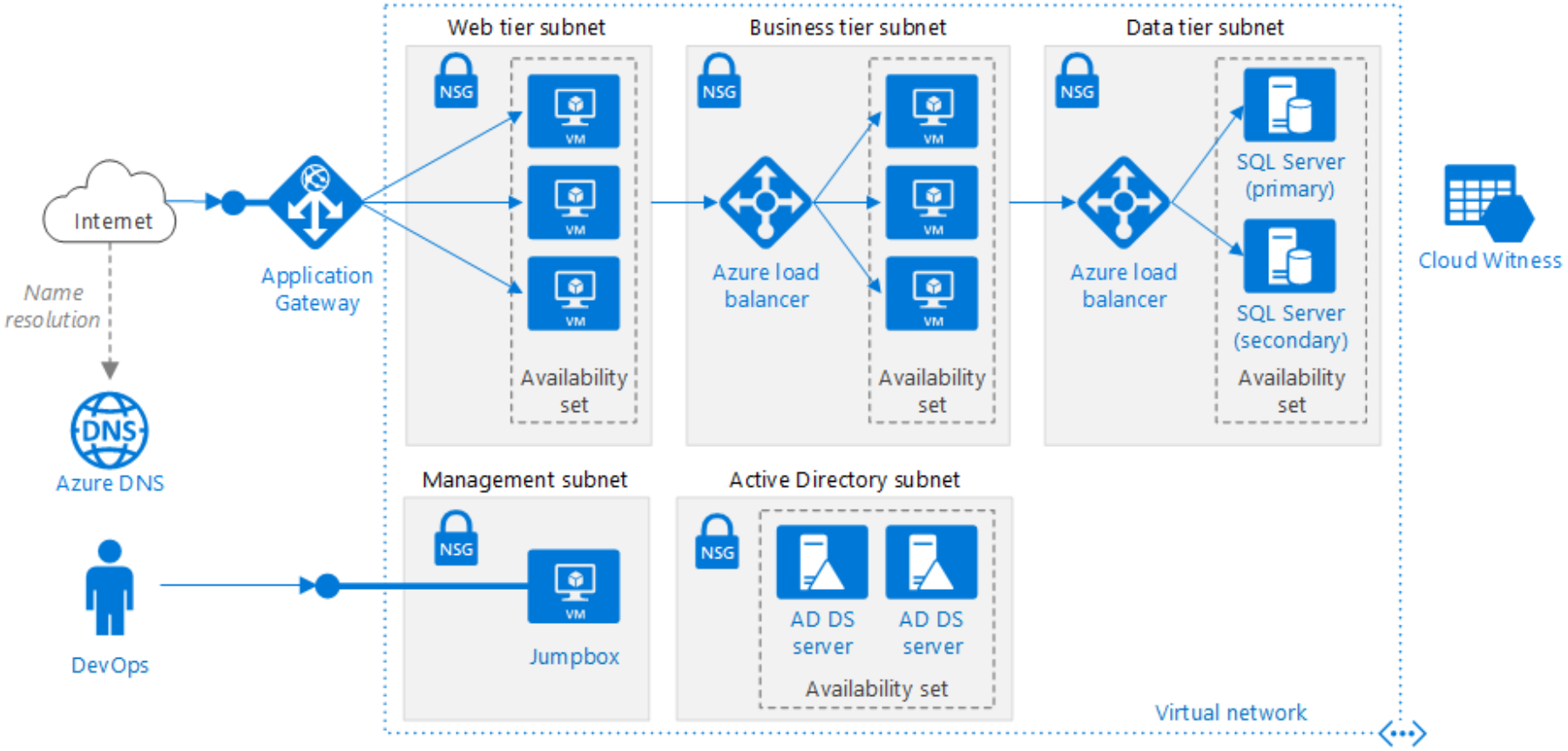
# Azure

cisco Live!

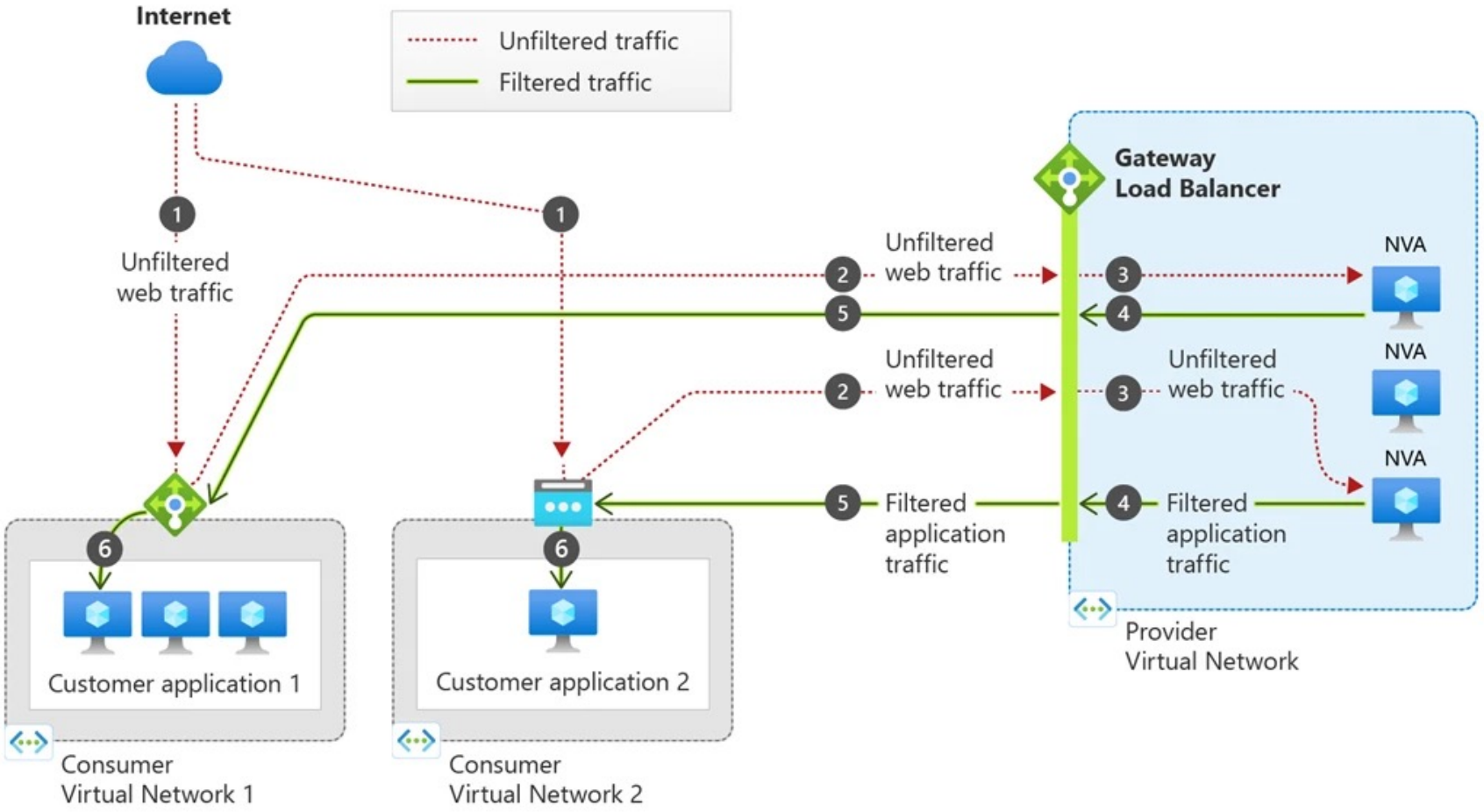
# Azure Resources



# Azure Resources



# Azure Gateway Load Balancer



# Terraform

cisco Live!

# Providers

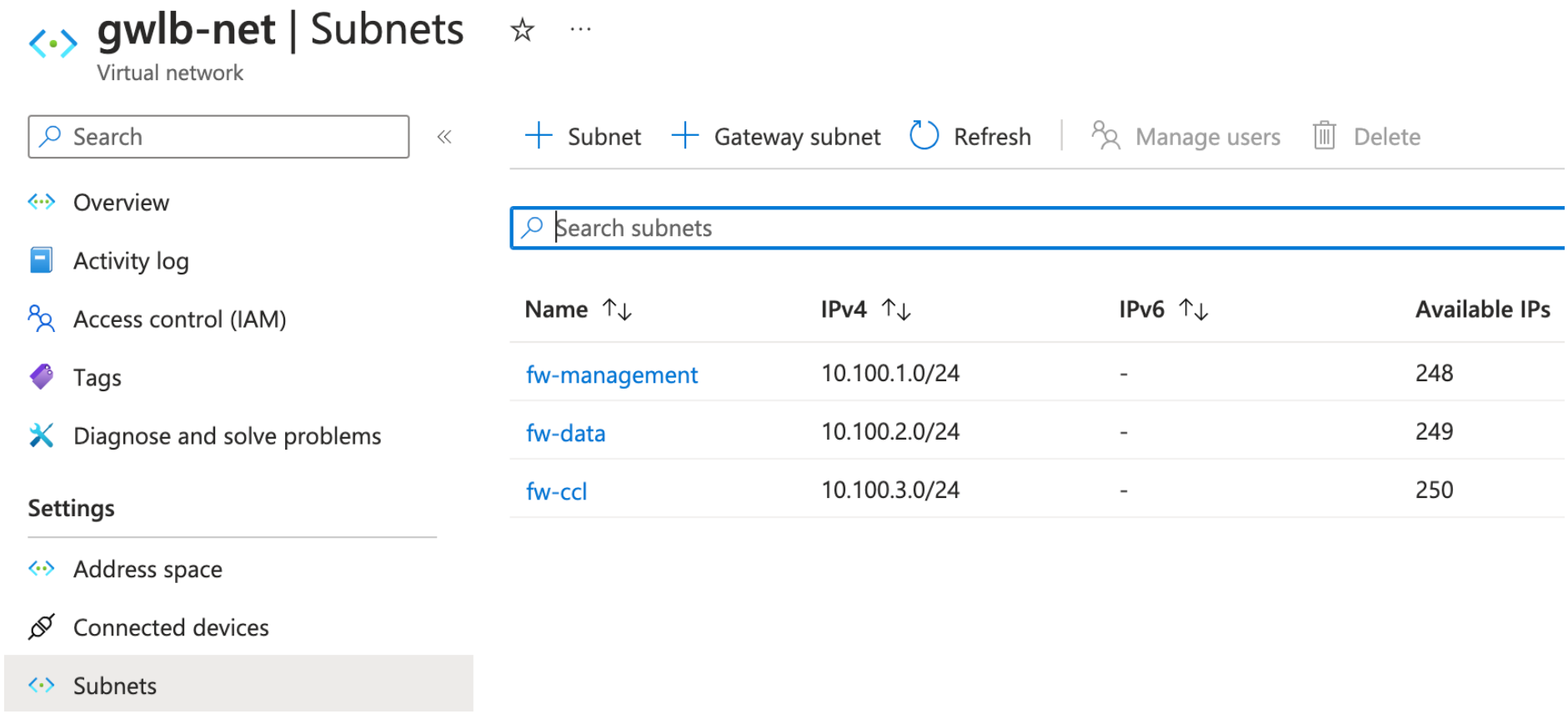
- Microsoft Azure
  - https://registry.terraform.io/providers/hashicorp/azurerm/latest/docs
- Cisco FMC
  - https://registry.terraform.io/providers/CiscoDevNet/fmc/latest/docs
- Cisco CDO
  - https://registry.terraform.io/providers/CiscoDevNet/cdo/latest/docs

```
terraform {
  required_providers {
    azurerm = {
      source = "hashicorp/azurerm"
      version = "3.89.0"
    }
    fmc = {
      source = "CiscoDevNet/fmc"
      version = ">=1.4.5"
    }
    cdo = {
      source = "CiscoDevNet/cdo"
      version = ">=0.7.2, <1.0.0"
    }
  }
}

provider "azurerm" {
  features {}
}

provider "fmc" {
  is_cdfmc  = true
  cdo_token = var.cdo_token
  fmc_host  = var.cdFMC
  cdfmc_domain_uuid = var.cdfmc_domain_uuid
}

provider "cdo" {
  base_url  = var.cdo_base_url
  api_token = var.cdo_token
}
```

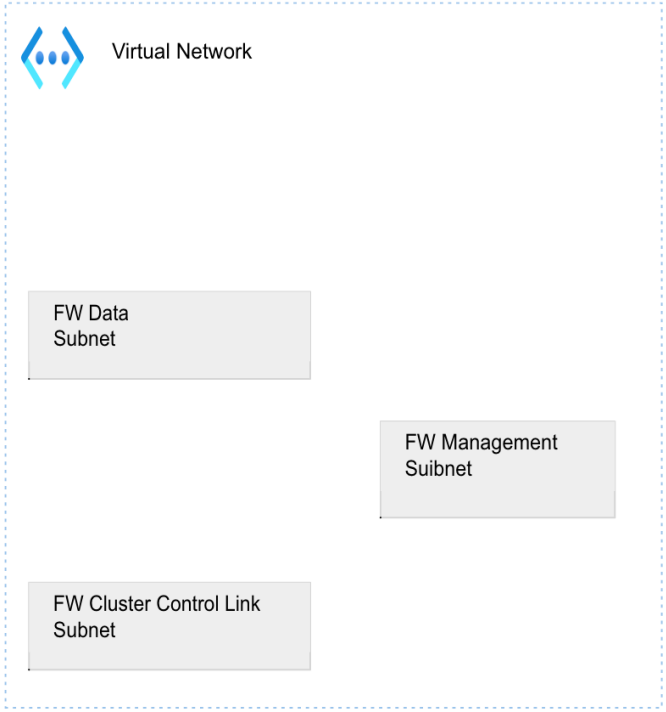
# GWLB Virtual Network and Subnets

```
resource "azurerm_virtual_network" "gwlb" {
  name                = "gwlb-net"
  address_space       = ["10.100.0.0/16"]
  resource_group_name = azurerm_resource_group.gwlb.name
  location            = azurerm_resource_group.gwlb.location
}

resource "azurerm_subnet" "fw_management" {
  name                 = "fw-management"
  resource_group_name  = azurerm_resource_group.gwlb.name
  virtual_network_name = azurerm_virtual_network.gwlb.name
  address_prefixes     = [cidrsubnet(azurerm_virtual_network.gwlb.address_space[0], 8, 1)]
}

resource "azurerm_subnet" "fw_data" {
  name                 = "fw-data"
  resource_group_name  = azurerm_resource_group.gwlb.name
  virtual_network_name = azurerm_virtual_network.gwlb.name
  address_prefixes     = [cidrsubnet(azurerm_virtual_network.gwlb.address_space[0], 8, 2)]
}

resource "azurerm_subnet" "fw_ccl" {
  name                 = "fw-ccl"
  resource_group_name  = azurerm_resource_group.gwlb.name
  virtual_network_name = azurerm_virtual_network.gwlb.name
  address_prefixes     = [cidrsubnet(azurerm_virtual_network.gwlb.address_space[0], 8, 3)]
}
```

# GWLB Virtual Network and Subnets



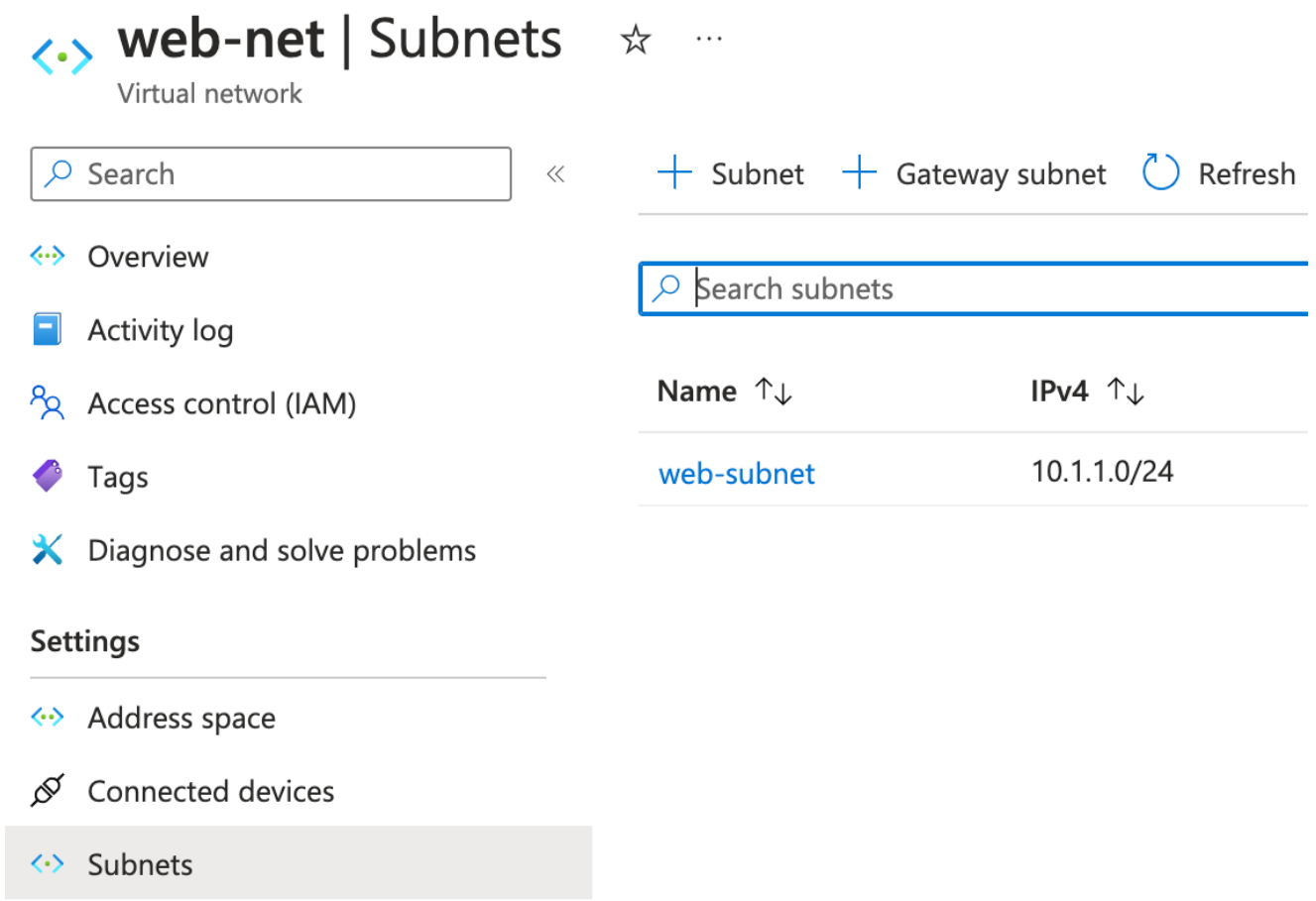
# GWLB Virtual Network and Subnets



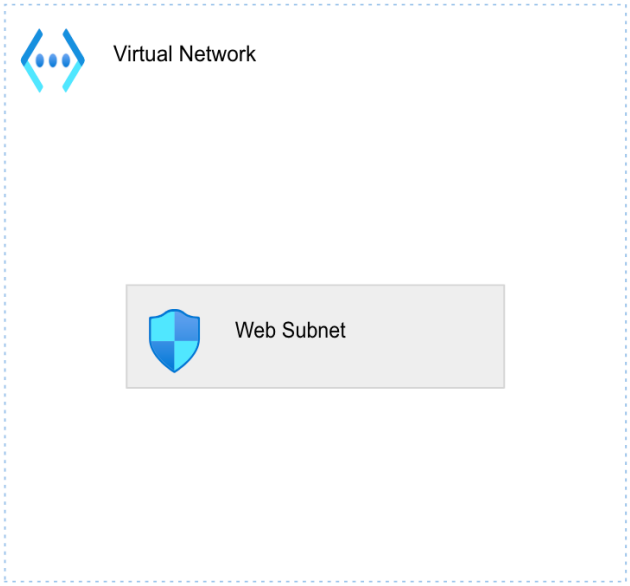
# Web App Virtual Network and Subnets

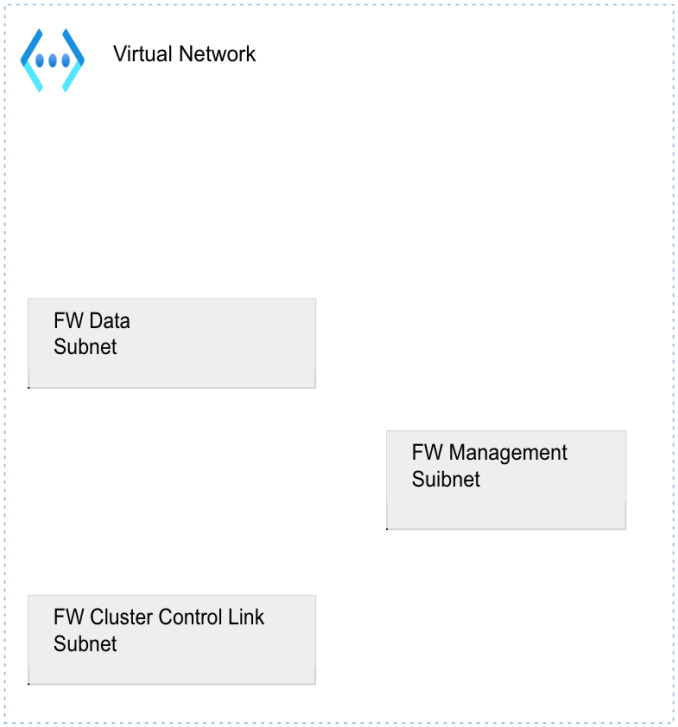
```
resource "azurerm_virtual_network" "web" {
  name                = "web-net"
  address_space       = ["10.1.0.0/16"]
  resource_group_name = azurerm_resource_group.gwlb.name
  location            = azurerm_resource_group.gwlb.location
}
resource "azurerm_subnet" "web" {
  name                 = "web-subnet"
  resource_group_name  = azurerm_resource_group.gwlb.name
  virtual_network_name = azurerm_virtual_network.web.name
  address_prefixes     = [cidrsubnet(azurerm_virtual_network.web.address_space[0], 8, 1)]
}
```

# Web App Virtual Network and Subnets



# Web App Virtual Network and Subnets

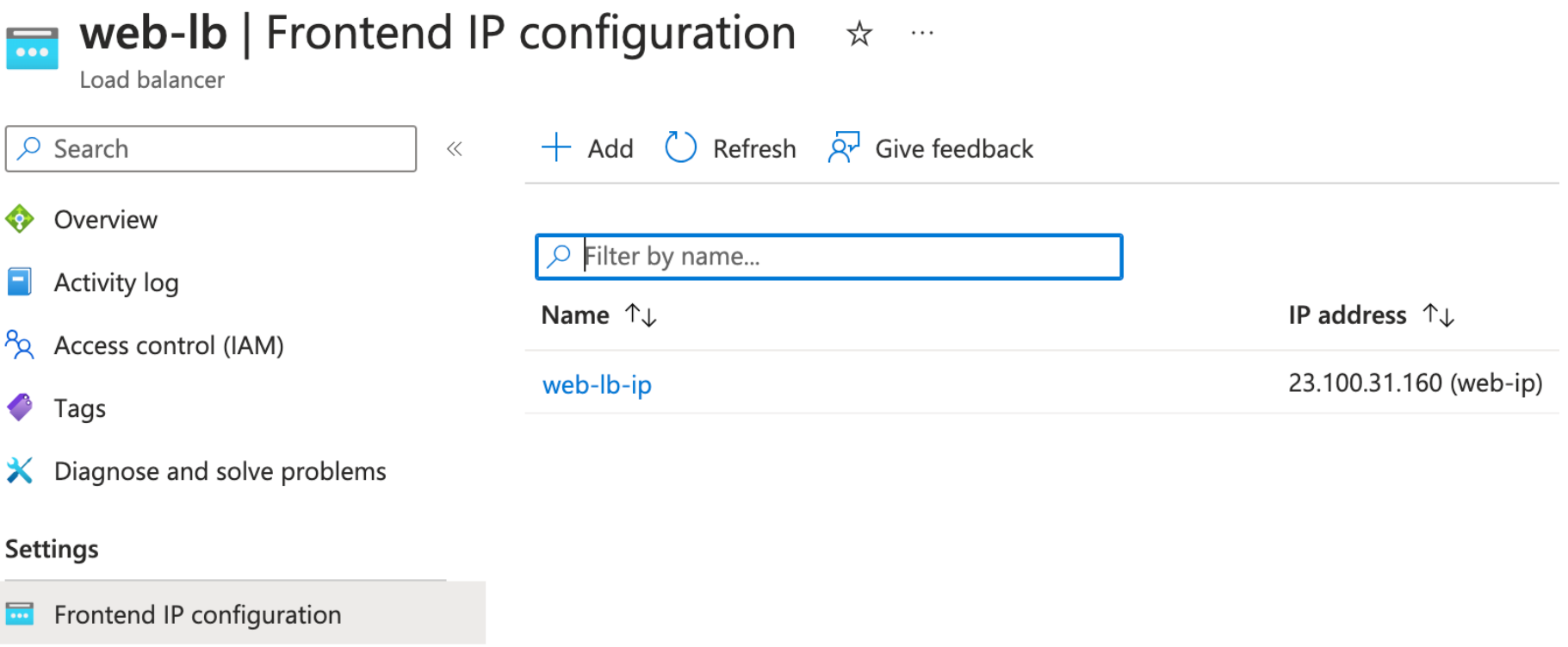


# Web Facing Load Balancer in Web VNET

```
resource "azurerm_public_ip" "web_lb" {
  name                = "web-ip"
  location            = azurerm_resource_group.gwlb.location
  resource_group_name = azurerm_resource_group.gwlb.name
  allocation_method   = "Static"
  sku                 = "Standard"
}

resource "azurerm_lb" "web" {
  name                = "web-lb"
  location            = azurerm_resource_group.gwlb.location
  resource_group_name = azurerm_resource_group.gwlb.name
  sku                 = "Standard"

  frontend_ip_configuration {
    name                 = "web-lb-ip"
    public_ip_address_id = azurerm_public_ip.web_lb.id
    gateway_load_balancer_frontend_ip_configuration_id = azurerm_lb.fw.frontend_ip_configuration[0].id
  }
```

# Web Facing Load Balancer in Web VNET

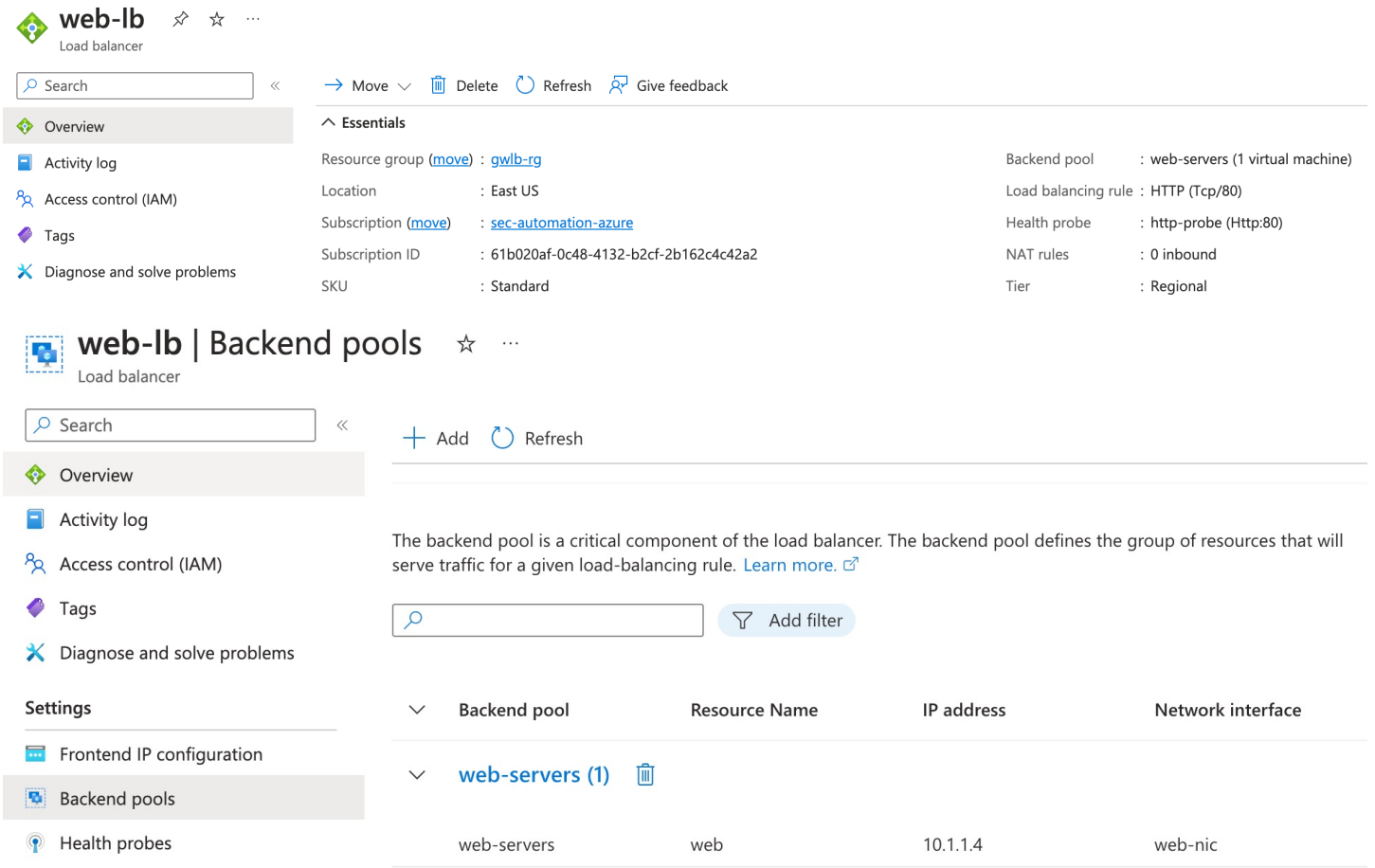


# Web Facing Load Balancer Configuration

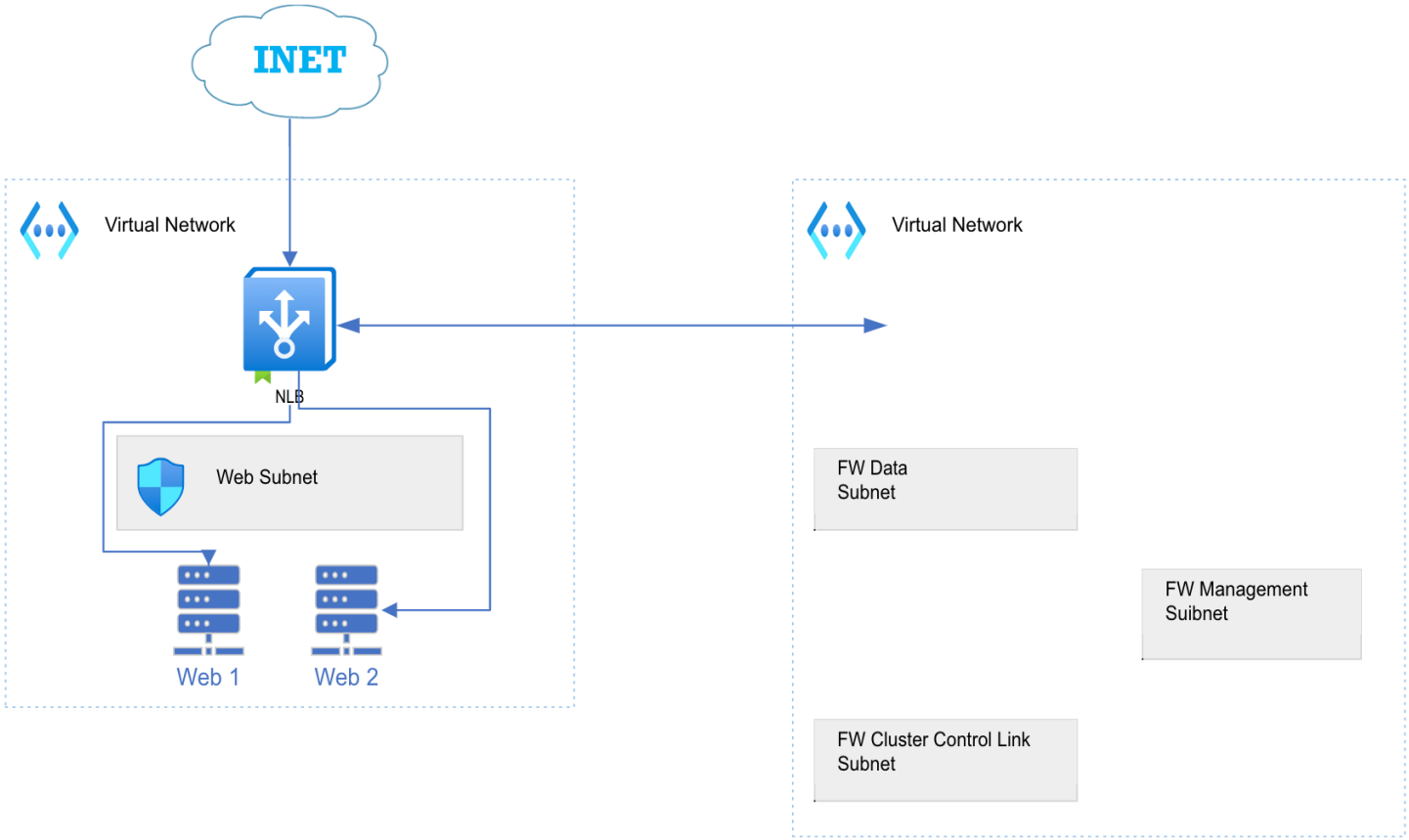
```
resource "azurerm_lb_backend_address_pool" "web" {
  loadbalancer_id = azurerm_lb.web.id
  name            = "web-servers"
}

#...

resource "azurerm_network_interface_backend_address_pool_association" "web" {
  network_interface_id    = azurerm_network_interface.web.id
  ip_configuration_name   = "web-nic-ip"
  backend_address_pool_id = azurerm_lb_backend_address_pool.web.id
}

resource "azurerm_lb_probe" "http_probe" {
  loadbalancer_id = azurerm_lb.web.id
  name            = "http-probe"
  protocol        = "Http"
  request_path    = "/"
  port            = 80
}
```

```
resource "azurerm_lb_rule" "web" {
  loadbalancer_id                = azurerm_lb.web.id
  name                           = "HTTP"
  protocol                       = "Tcp"
  frontend_port                  = 80
  backend_port                   = 80
  frontend_ip_configuration_name = "web-lb-ip"
  backend_address_pool_ids       = [azurerm_lb_backend_address_pool.web.id]
  probe_id                       = azurerm_lb_probe.http_probe.id
  disable_outbound_snat = true
}

resource "azurerm_lb_outbound_rule" "web" {
  name                    = "web-outbound"
  loadbalancer_id         = azurerm_lb.web.id
  protocol                = "All"
  backend_address_pool_id = azurerm_lb_backend_address_pool.web.id
  allocated_outbound_ports =512

  frontend_ip_configuration {
    name = "web-lb-ip"
  }
}
```

# Web Facing Load Balancer Configuration



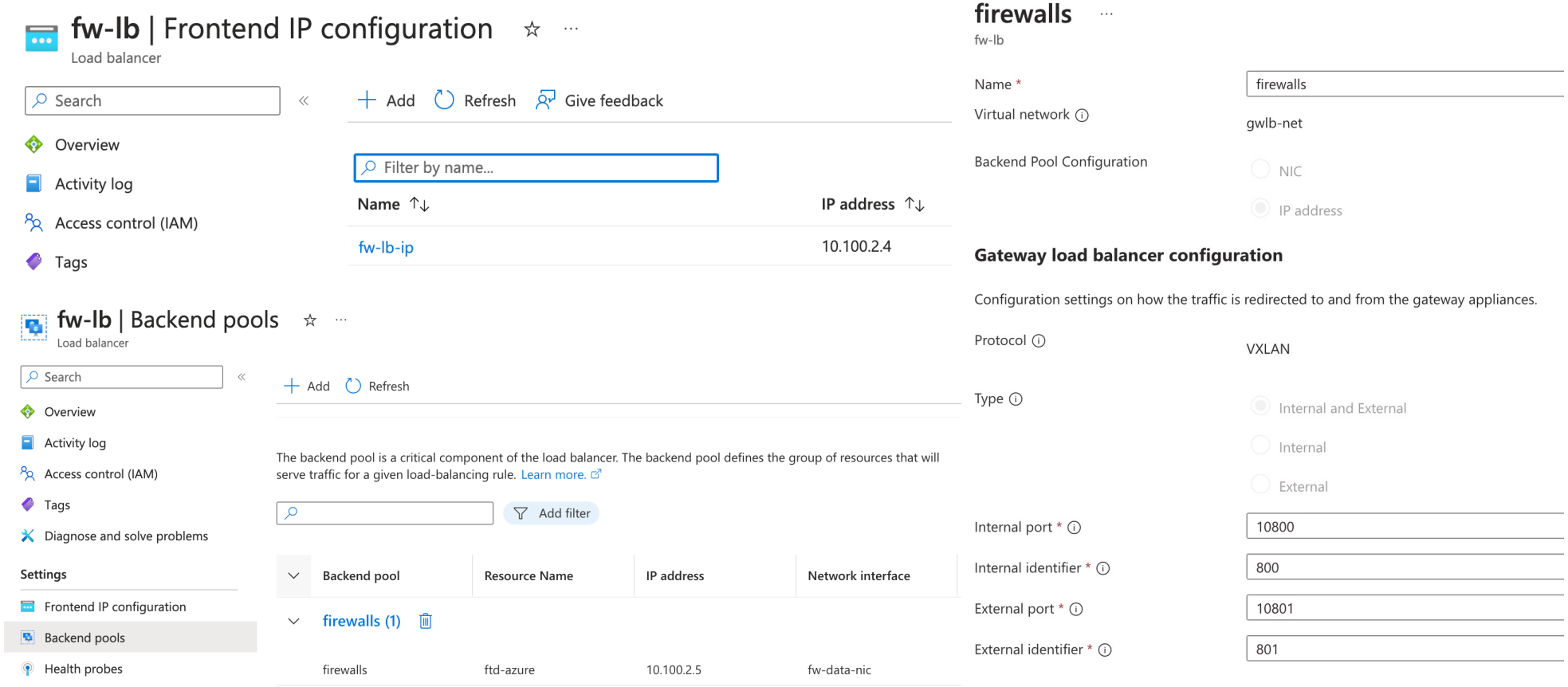
# Web Facing Load Balancer Configuration



# Gateway Load Balancer Configuration

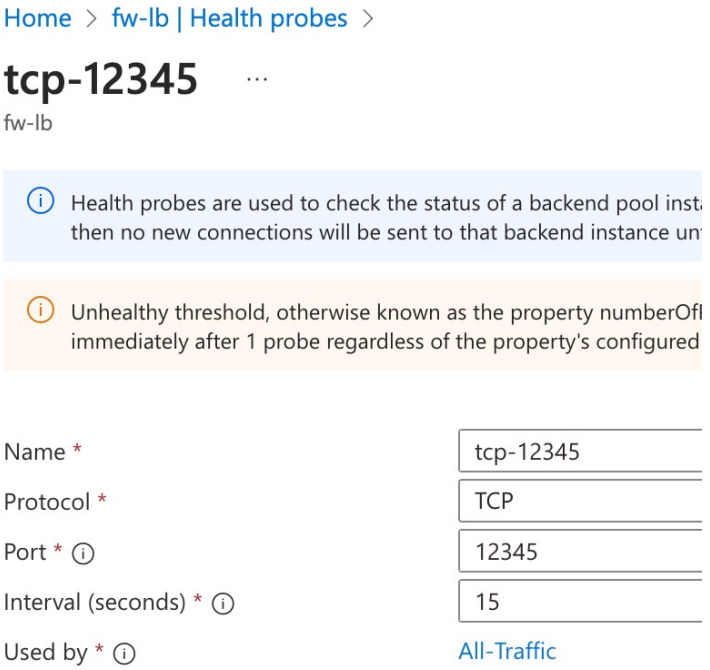
```
resource "azurerm_lb" "fw" {
  name                = "fw-lb"
  location            = azurerm_resource_group.gwlb.location
  resource_group_name = azurerm_resource_group.gwlb.name
  sku                 = "Gateway"

  frontend_ip_configuration {
    name      = "fw-lb-ip"
    subnet_id = azurerm_subnet.fw_data.id
  }
}

resource "azurerm_lb_backend_address_pool" "fw" {
  loadbalancer_id = azurerm_lb.fw.id
  name            = "firewalls"
  tunnel_interface {
    identifier = 800
    type       = "Internal"
    protocol   = "VXLAN"
    port       = 10800
  }

  tunnel_interface {
    identifier = 801
    type       = "External"
    protocol   = "VXLAN"
    port       = 10801
  }
}
```

# Gateway Load Balancer Configuration

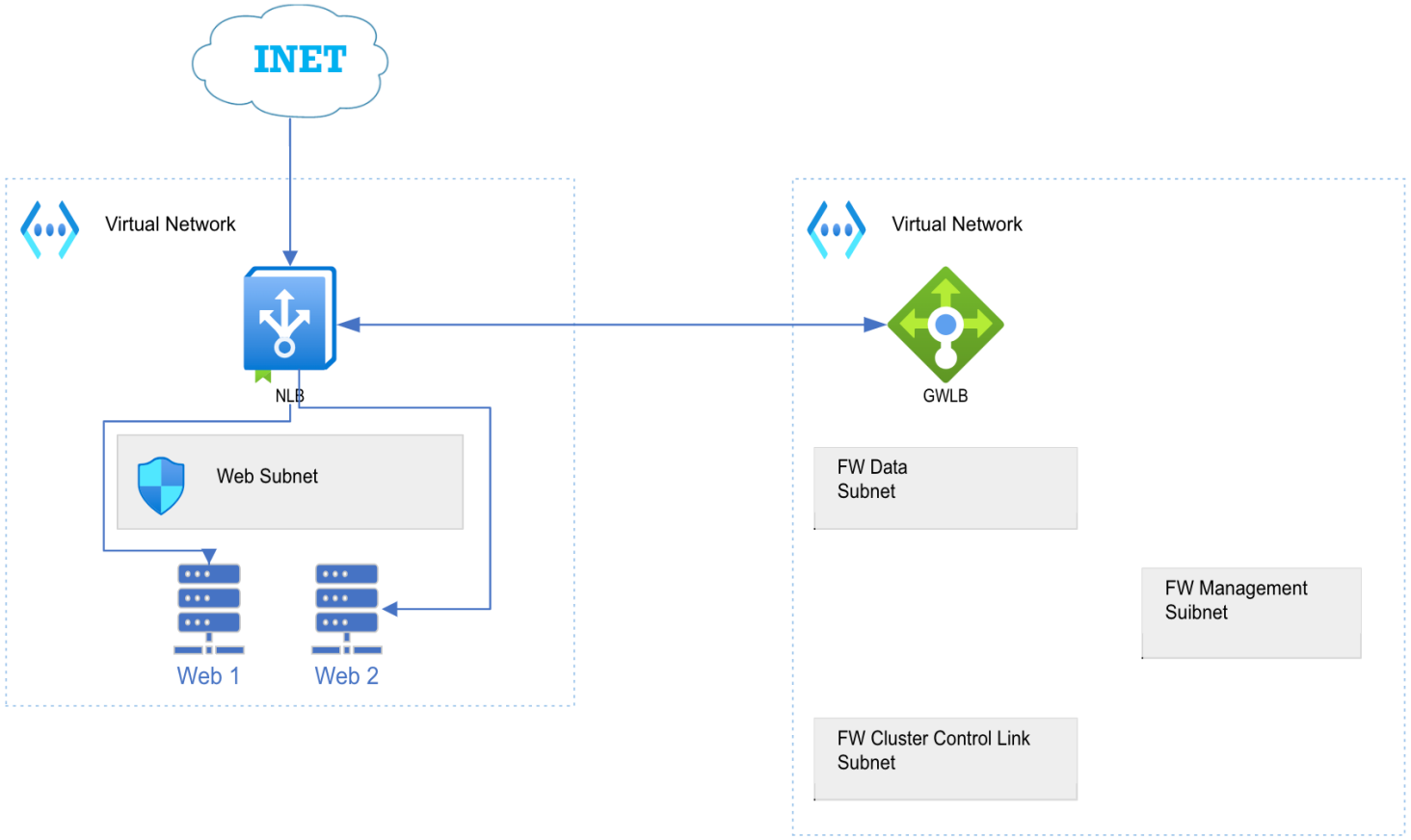


# Gateway Load Balancer Configuration

```
resource "azurerm_lb_backend_address_pool_address" "fw" {
  name                    = "fw-lb-pool"
  backend_address_pool_id = azurerm_lb_backend_address_pool.fw.id
  virtual_network_id      = azurerm_virtual_network.gwlb.id
  ip_address              = azurerm_network_interface.fw_data.ip_configuration[0].private_ip_address
}

resource "azurerm_lb_probe" "tcp_12345" {
  loadbalancer_id = azurerm_lb.fw.id
  name            = "tcp-12345"
  protocol        = "Tcp"
  port            = 12345
}

resource "azurerm_lb_rule" "gwlb" {
  loadbalancer_id                = azurerm_lb.fw.id
  name                           = "All-Traffic"
  protocol                       = "All"
  frontend_ip_configuration_name = "fw-lb-ip"
  frontend_port                  = 0
  backend_port                   = 0
  load_distribution              = "SourceIP"
  backend_address_pool_ids       = [azurerm_lb_backend_address_pool.fw.id]
  probe_id                       = azurerm_lb_probe.tcp_12345.id
}
```



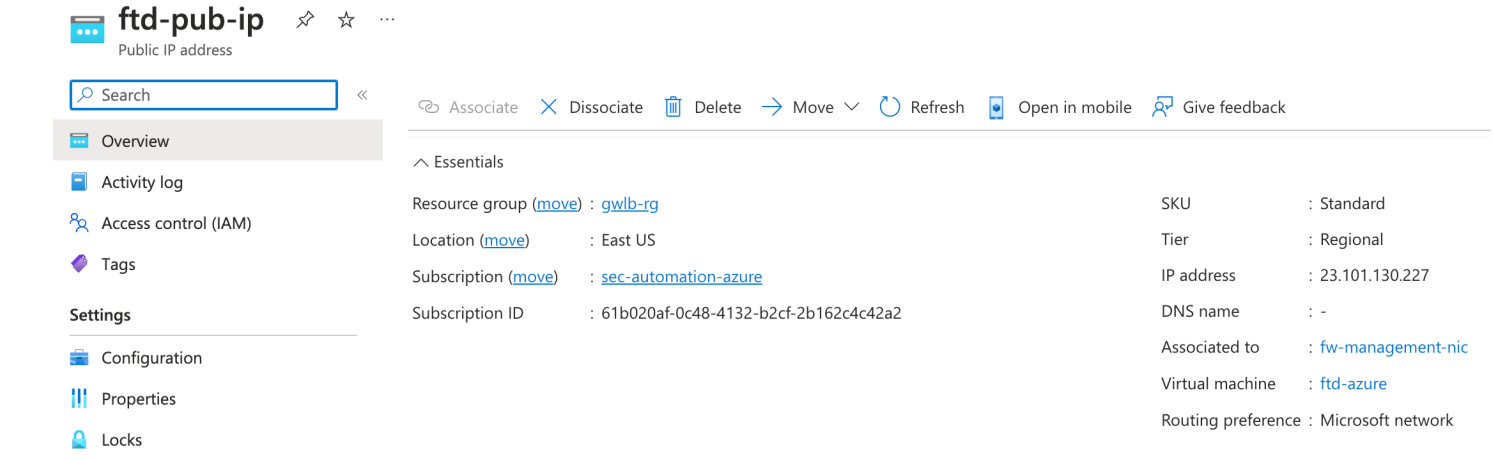
# Gateway Load Balancer Configuration

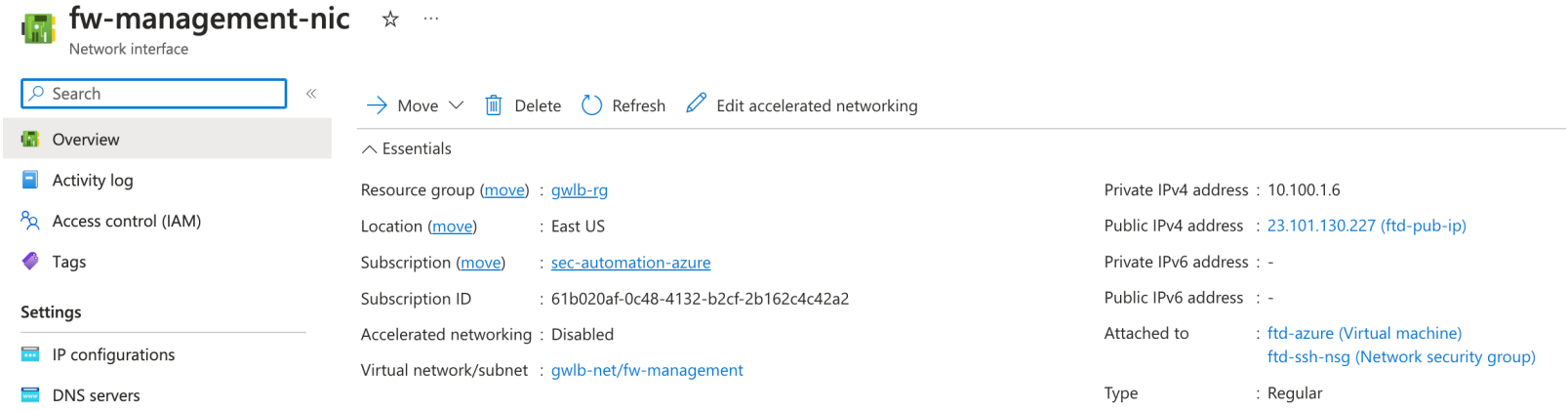


# Cisco Secure Firewall (aka FTD)

```
resource "azurerm_public_ip" "ftd-pub-ip" {
  name                = "ftd-pub-ip"
  location            = azurerm_resource_group.gwlb.location
  resource_group_name = azurerm_resource_group.gwlb.name
  allocation_method   = "Static"
  sku                 = "Standard"
}

resource "azurerm_network_interface" "fw_management" {
  name                = "fw-management-nic"
  location            = azurerm_resource_group.gwlb.location
  resource_group_name = azurerm_resource_group.gwlb.name

  ip_configuration {
    name                          = "fw-management-nic-ip"
    subnet_id                     = azurerm_subnet.fw_management.id
    private_ip_address_allocation = "Dynamic"
    public_ip_address_id          = azurerm_public_ip.ftd-pub-ip.id
  }
}
```

# Cisco Secure Firewall (aka FTD)

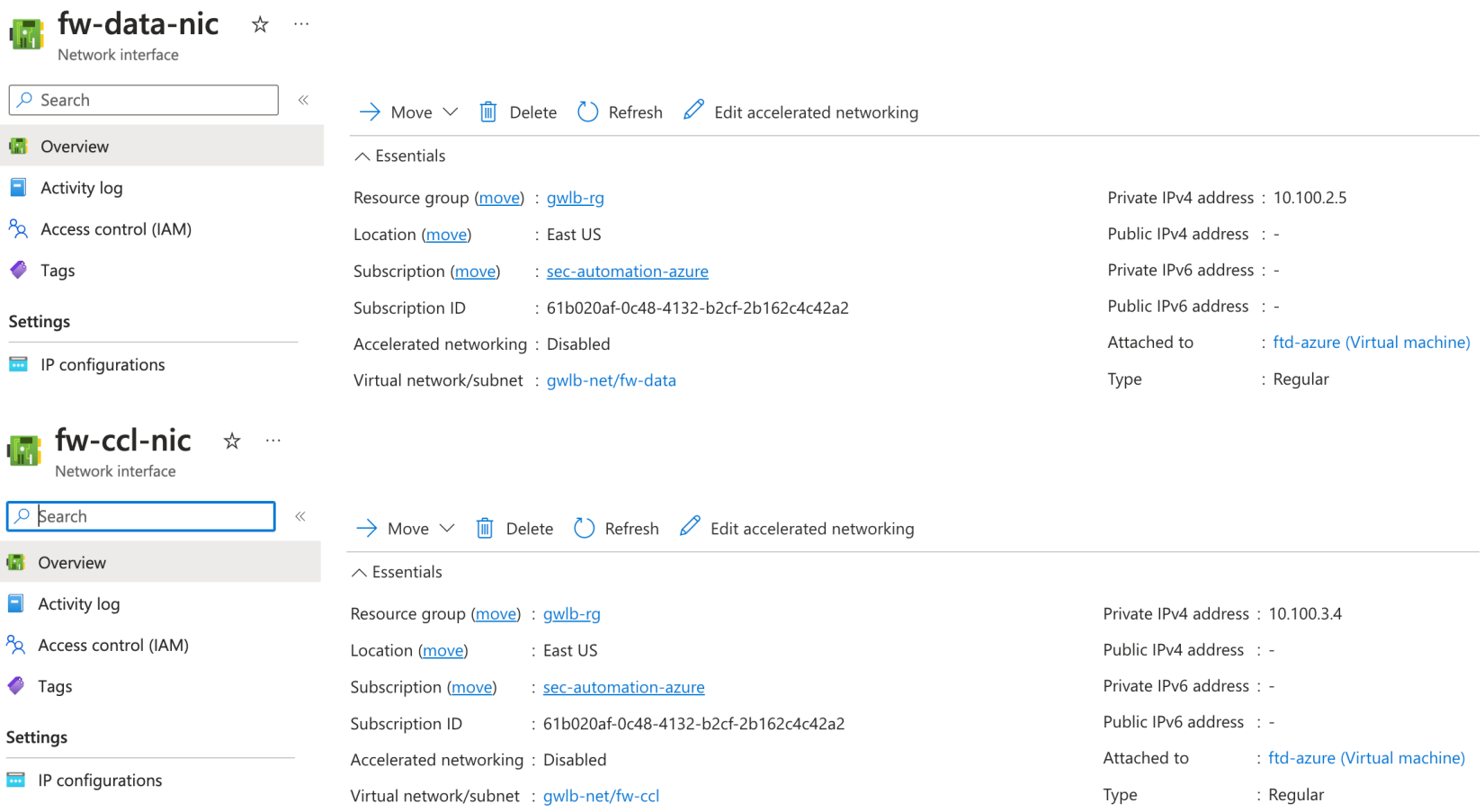


# Cisco Secure Firewall (aka FTD)

```
resource "azurerm_network_interface" "fw_data" {
  name                = "fw-data-nic"
  location            = azurerm_resource_group.gwlb.location
  resource_group_name = azurerm_resource_group.gwlb.name

  ip_configuration {
    name                          = "fw-data-nic-ip"
    subnet_id                     = azurerm_subnet.fw_data.id
    private_ip_address_allocation = "Dynamic"
  }
}

resource "azurerm_network_interface" "fw_ccl" {
  name                = "fw-ccl-nic"
  location            = azurerm_resource_group.gwlb.location
  resource_group_name = azurerm_resource_group.gwlb.name

  ip_configuration {
    name                          = "fw-cl-nic-ip"
    subnet_id                     = azurerm_subnet.fw_ccl.id
    private_ip_address_allocation = "Dynamic"
  }
}
```

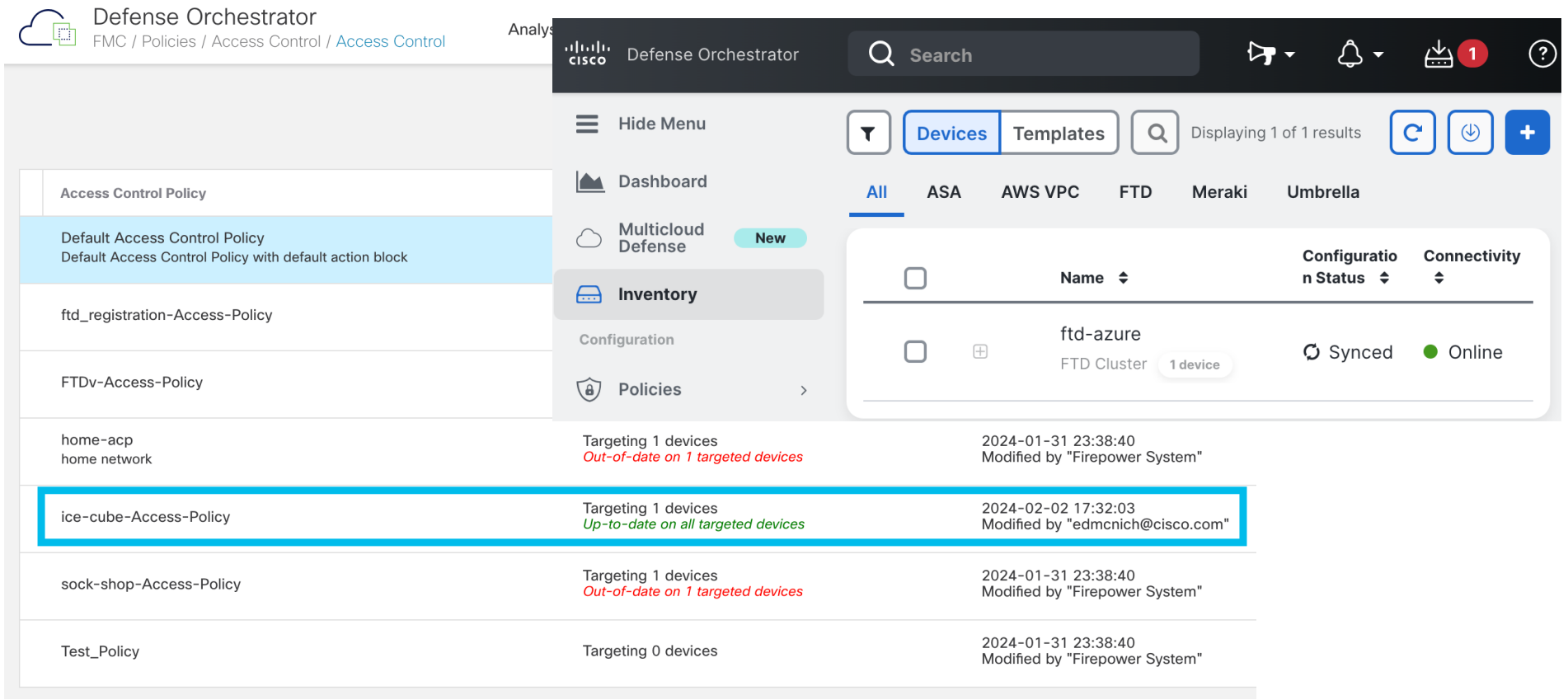
# Cisco Secure Firewall (aka FTD)



# Cisco Defense Orchestrator

```
# Create default Access Control Policy
resource "fmc_access_policies" "access_policy" {
  name           = "${var.name}-Access-Policy"
  default_action = "block"
}

resource "cdo_ftd_device" "ftd" {
  access_policy_name = fmc_access_policies.access_policy.name
  licenses           = ["BASE","MALWARE","URLFilter","THREAT"]
  name               = "ftd-azure"
  virtual            = true
  performance_tier   = "FTDv30"
}
```

# Cisco Defense Orchestrator



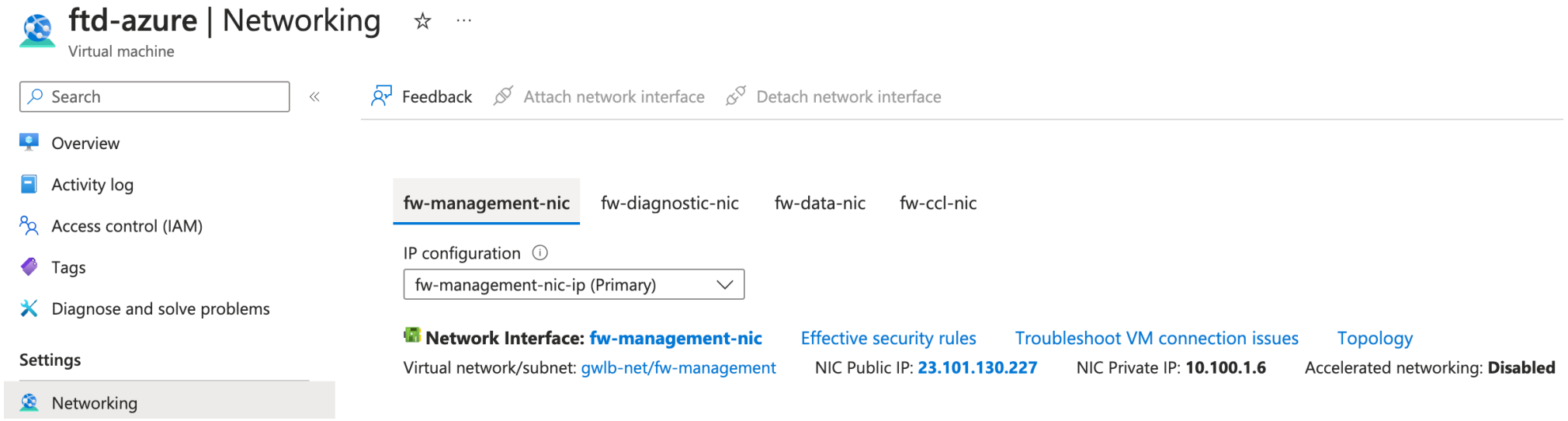
# FTD Instance Configuration

```
resource "azurerm_linux_virtual_machine" "ftd" {
  name                = "ftd-azure"
  computer_name       = "ftd-azure"
  location            = azurerm_resource_group.gwlb.location
  resource_group_name = azurerm_resource_group.gwlb.name
  network_interface_ids = [
    azurerm_network_interface.fw_management.id,
    azurerm_network_interface.fw_diagnostic.id,
    azurerm_network_interface.fw_data.id,
    azurerm_network_interface.fw_ccl.id
  ]
  size                            = "Standard_D3_v2"
  admin_username                  = "azadmin"
  admin_password                  = var.admin_password
  disable_password_authentication = false
```

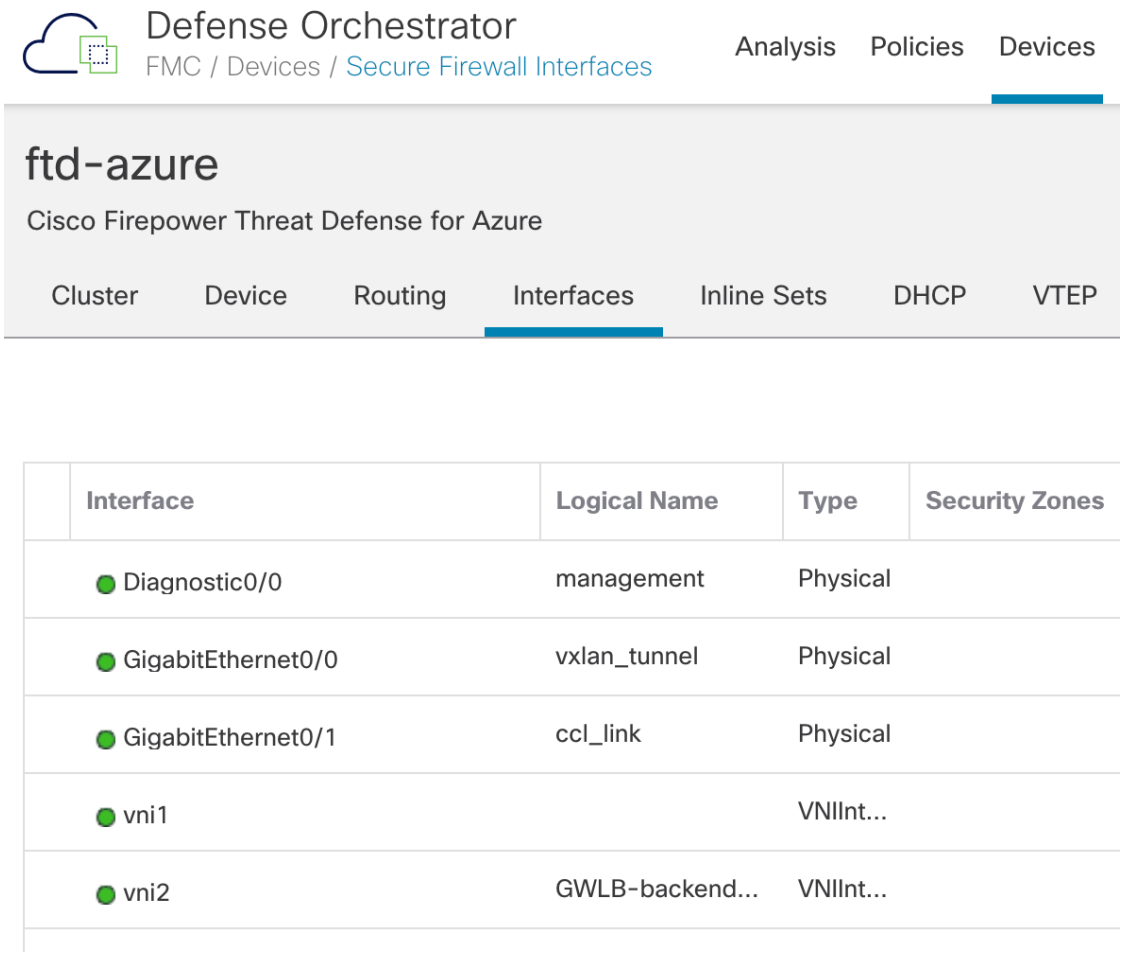
# FTD Instance Configuration

```
custom_data = base64encode(jsonencode(
  {
    "AdminPassword": var.admin_password,
    "Hostname": "ftd-azure",
    "FirewallMode": "Routed",
    "ManageLocally": "No",
    "FmcIp":"${var.cdFMC}",
    "FmcRegKey":"${cdo_ftd_device.ftd.reg_key}",
    "FmcNatId":"${cdo_ftd_device.ftd.nat_id}",
    "Cluster": {
      "CclSubnetRange": "${cidrhost(azurerm_subnet.fw_ccl.address_prefixes[0],1)} ${cidrhost(azurerm_subnet.fw_ccl.address_prefixes[0],32)}",
      "ClusterGroupName": "ftd-azure",
      "HealthProbePort": "12345",
      "GatewayLoadBalancerIP": "${azurerm_lb.fw.frontend_ip_configuration[0].private_ip_address}",
      "EncapsulationType": "vxlan",
      "InternalPort": "10800",
      "ExternalPort": "10801",
      "InternalSegId": "800",
      "ExternalSegId": "801"
    }
```

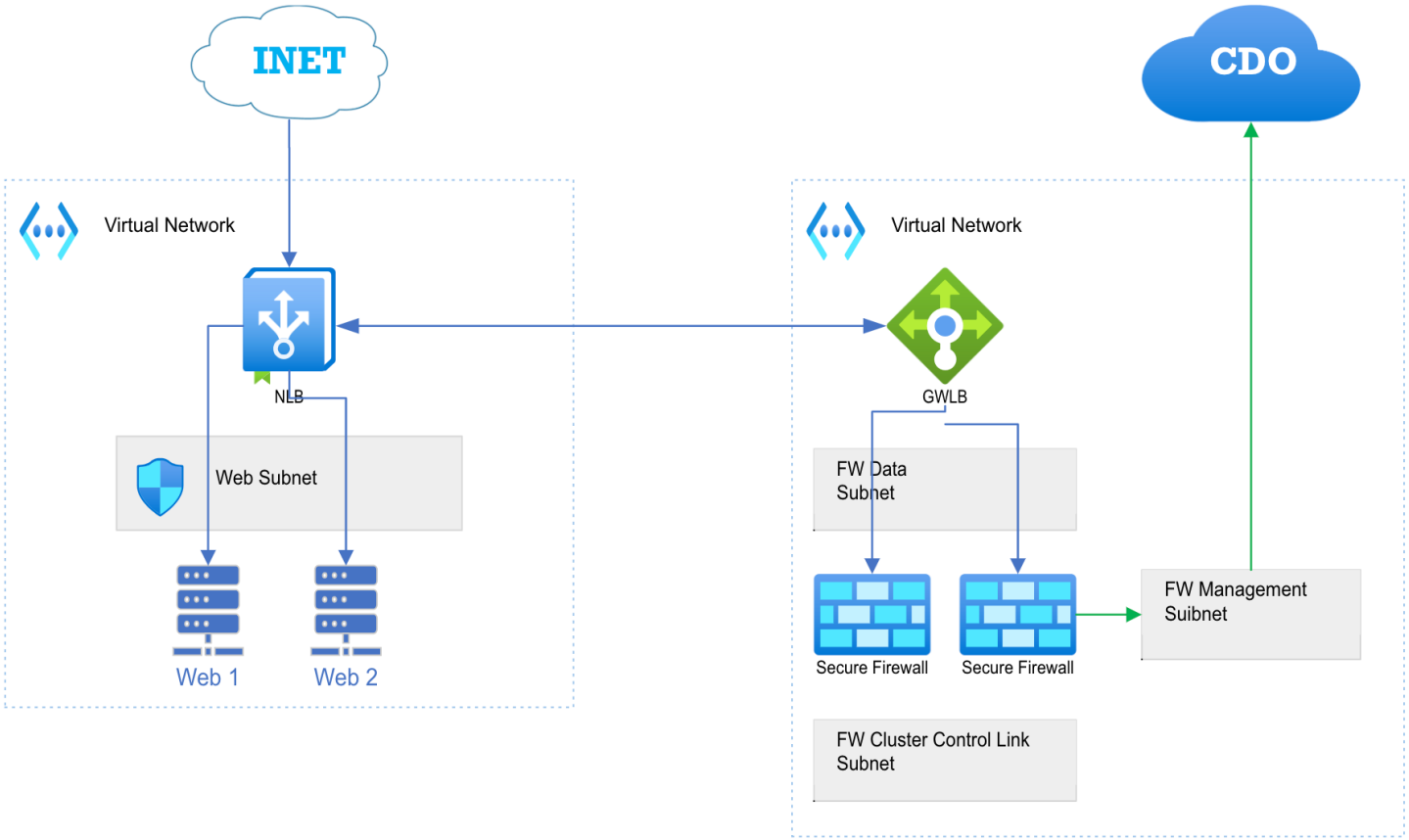
# FTD Instance Configuration



# FTD Instance Configuration



# FTD Instance Configuration

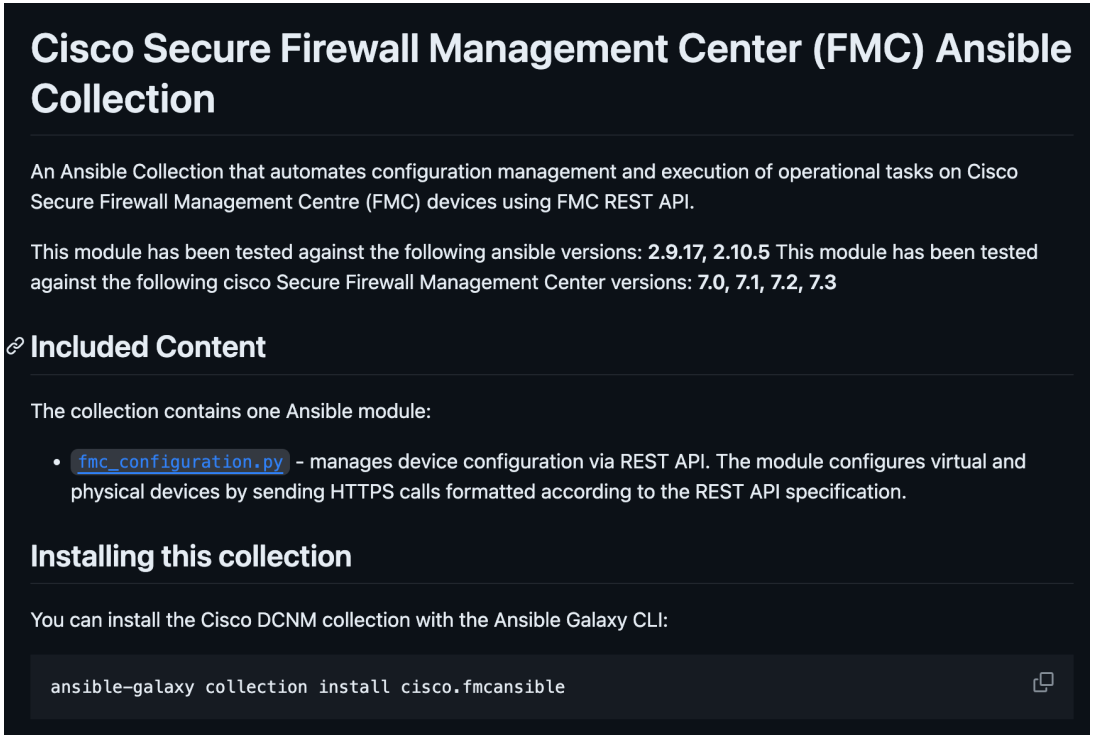


# Ansible

cisco Live!

# Ansible Collections

- Cisco FMC
  - https://github.com/CiscoDevNet/FMCAnsibleCisco
- Cisco CDO
  - https://github.com/CiscoDevNet/ansible-cisco-cdo/tree/main/docs



## Cisco Secure Firewall Management Center (FMC) Ansible Collection

An Ansible Collection that automates configuration management and execution of operational tasks on Cisco Secure Firewall Management Centre (FMC) devices using FMC REST API.

This module has been tested against the following ansible versions: **2.9.17, 2.10.5** This module has been tested against the following cisco Secure Firewall Management Center versions: **7.0, 7.1, 7.2, 7.3**

### Included Content

The collection contains one Ansible module:

- `fmc_configuration.py` - manages device configuration via REST API. The module configures virtual and physical devices by sending HTTPS calls formatted according to the REST API specification.

### Installing this collection

You can install the Cisco DCNM collection with the Ansible Galaxy CLI:

```
ansible-galaxy collection install cisco.fmcansible
```

# Create Inventory File from Terraform Resources

```
resource "local_file" "host_file" {
    content     = <<-EOT
    ---
    all:
      hosts:
        cdFMC:
          ansible_host: ${var.cdFMC}
          ansible_network_os: cisco.fmcansible.fmc
          ansible_httpapi_port: 443
          ansible_httpapi_use_ssl: True
          ansible_httpapi_validate_certs: True
          web_lb_public_ip: ${azurerm_public_ip.web_lb.ip_address}
    EOT
    filename = "${path.module}/hosts.yaml"
}
```

# Inventory File Created (YAML)

```yaml
---
all:
  hosts:
    cdFMC:
      ansible_host: cisco-edmcnich.app.us.cdo.cisco.com
      ansible_network_os: cisco.fmcansible.fmc
      ansible_httpapi_port: 443
      ansible_httpapi_use_ssl: True
      ansible_httpapi_validate_certs: True
      web_lb_public_ip: 23.100.31.160
```

# Fetch FMC Domain and FTD Devices

```yaml
# Fetch Data from FMC
- name: Get Domain UUID
  cisco.fmcansible.fmc_configuration:
    operation: getAllDomain
    register_as: domain
- name: Get Devices
  cisco.fmcansible.fmc_configuration:
    operation: getAllDevice
    path_params:
      domainUUID: '{{ domain[0].uuid }}'
    filters:
      name: ftd-azure
    register_as: device_list
  until: device_list is defined
  retries: 60
  delay: 5
```

# Fetch HTTP Object and Access Policy

```yaml
- name: Get Port Object HTTP
  cisco.fmcansible.fmc_configuration:
    operation: getAllProtocolPortObject
    path_params:
      domainUUID: '{{ domain[0].uuid }}'
    filters:
      name: HTTP
    register_as: http
```

```yaml
- name: Get Access Policy
  cisco.fmcansible.fmc_configuration:
    operation: getAllAccessPolicy
    path_params:
      domainUUID: '{{ domain[0].uuid }}'
    filters:
      name: ftd-azure-Access-Policy
    register_as: accesspolicy
```

# Create Host Object

```yaml
- name: Web Load Balancer
  cisco.fmcansible.fmc_configuration:
    operation: upsertHostObject
    data:
      name: web_lb_public_ip
      value: '{{ web_lb_public_ip }}'
      type: Host
    path_params:
      domainUUID: '{{ domain[0].uuid }}'
    register_as: web_lb_public_ip
```

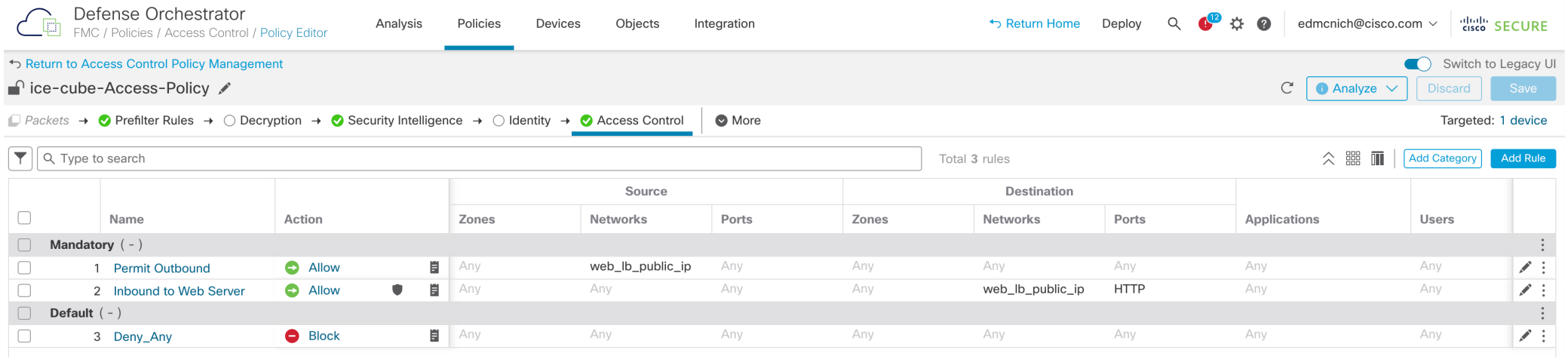
# Outbound Rule

```yaml
- name: Access Rule 1
  cisco.fmcansible.fmc_configuration:
    operation: upsertAccessRule
    data:
      name: Permit Outbound
      type: accessrule
      action: ALLOW
      section: mandatory
      enabled: true
      sendEventsToFMC: true
      logBegin: true
      logEnd: true
      sourceNetworks:
        objects:
          - id: '{{ web_lb_public_ip.id }}'
            name: '{{ web_lb_public.ip.name }}'
            type: '{{ web_lb_public_ip.type }}'
      newComments:
        - 'Outbound Traffic'
    path_params:
      section: 'Mandatory'
      containerUUID: '{{ accesspolicy[0].id }}'
      domainUUID: '{{ domain[0].uuid }}'
    register_as: accessrule1
```

# Inbound Rule

```yaml
- name: Access Rule 2
  cisco.fmcansible.fmc_configuration:
    operation: upsertAccessRule
    data:
      name: Access to Web Server
      type: accessrule
      action: ALLOW
      section: mandatory
      enabled: true
      sendEventsToFMC: true
      logBegin: true
      logEnd: true
      destinationNetworks:
        objects:
          - id: '{{ web_lb_public_ip.id }}'
            name: '{{ web_lb_public.ip.name }}'
            type: '{{ web_lb_public_ip.type }}'
      destinationPorts:
        objects:
          - id: '{{ HTTP.id }}'
            name: '{{ HTTP.name }}'
            type: '{{ HTTP.type }}'
      newComments:
        - 'Web Server'
    path_params:
      section: 'Mandatory'
      containerUUID: '{{ accesspolicy[0].id }}'
      domainUUID: '{{ domain[0].uuid }}'
    register_as: accessrule2
```

# Access Policy



# Conclusion

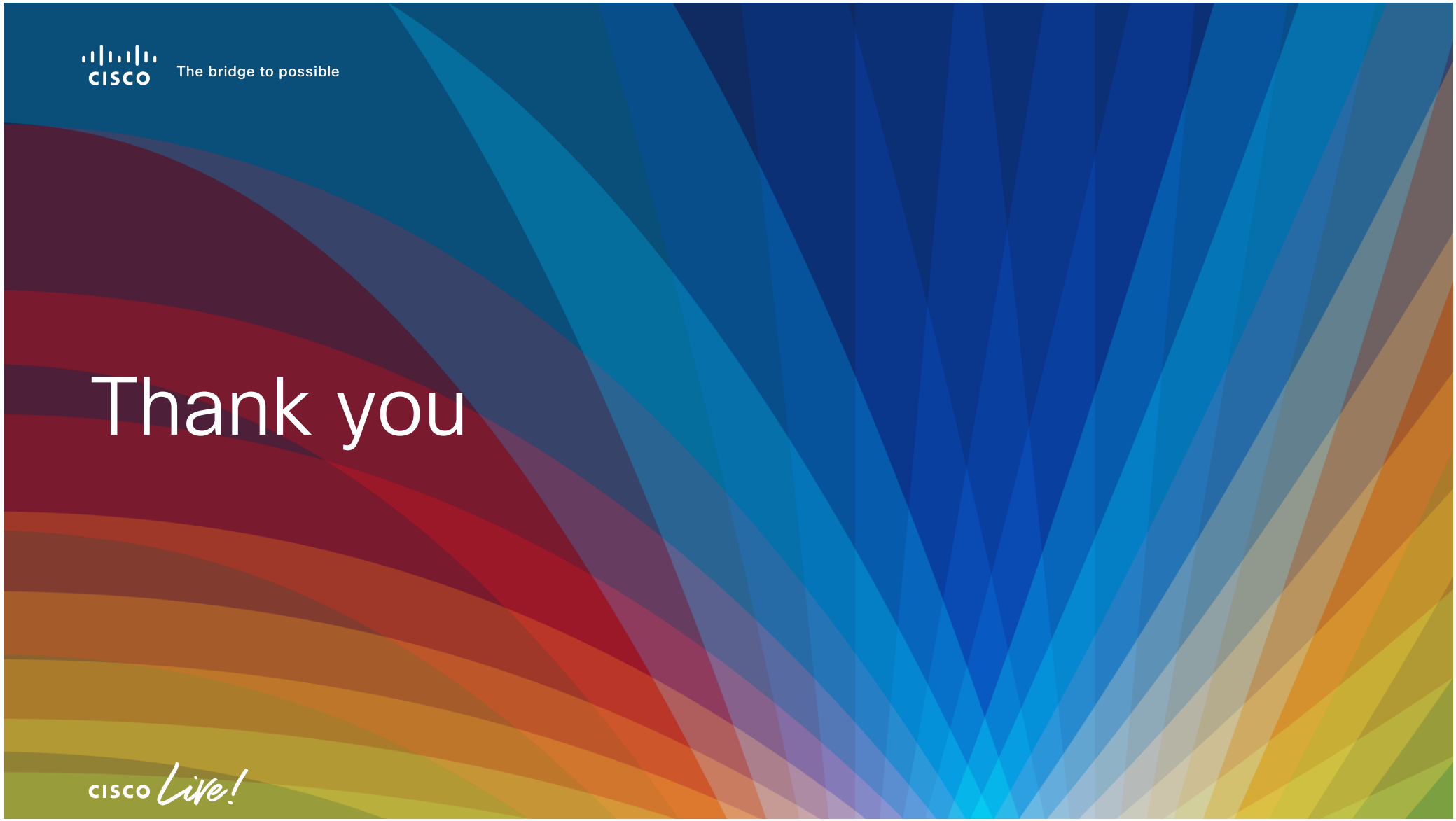
# Conclusion

- Provisioned Cisco Secure Firewall into Microsoft Azure using Terraform
  - Deployed the Firewall cluster into a Service VNET behind an Azure Gateway Load Balancer
  - Deployed web application into Web App VNET behind a Standard Load Balancer
  - Linked the Standard Load Balancer to the Gateway Load Balancer to service chain Ingress/Egress for the Web App to the Secure Firewall Cluster
- Configured the Secure Firewall Policy using Ansible
  - Created dynamic object using Terraform resources
  - Created Policy rules to allow access to and from the Web App

# Fill out your session surveys!

Attendees who fill out a minimum of four session surveys and the overall event survey will get **Cisco Live-branded t-shirt** (while supplies last)!

All surveys can be taken in the Cisco Events Mobile App or by logging in to the Session Catalog and clicking the "Attendee Dashboard" at https://www.ciscolive.com/emea/learn/sessions/session-catalog.html


cisco
The bridge to possible
Thank you
cisco Live!


cisco Live!
Let's go