cisco Live!

ALL IN

#CiscoLive

The bridge to possible

Infrastructure as Code (IaC) of EVPN with DCNM and Terraform

Faisal Chaudhry, Distinguished Engineer
Lei Tian, Solution Architecture

DEVNET-3011

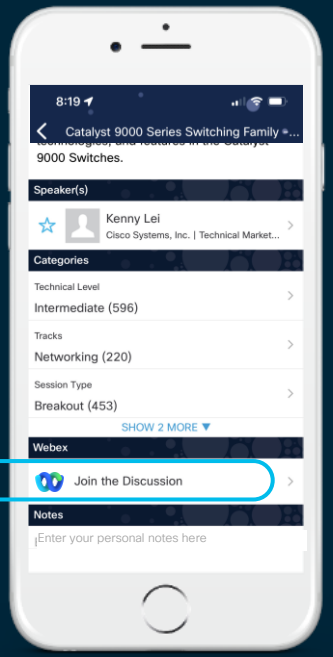
Cisco Webex App

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



https://ciscolive.ciscoevents.com/ciscolivebot/#LTRDCN-2765

Webex spaces will be moderated by the speaker until June 17, 2022.

Agenda

- Introduction to Infrastructure as Code (IaC)
- Overview of EVPN Fabric, DCNM and Terraform
- Terraform and Intersight Service for Terraform (IST)
- Deployment Scenarios
- See it in action (Demo)

What is Infrastructure as Code (IaC)?

Infrastructure as Code (IaC)

Provisioning and **management** of Infrastructure through **code** instead of using manual processes

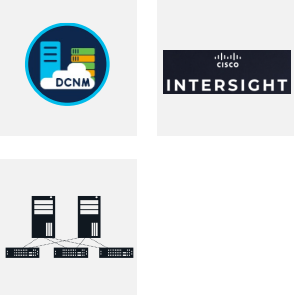
Automated **Provisioning**

Convert Manual Tasks into **reusable code**

Utilize **Software Development** practices

Version control, Automated testing, CI/CD pipelines

Infrastructure as Code (IaC) – session focus



Provisioning and **management** of Infrastructure through **code** instead of using manual processes

Automated **Provisioning**

Convert Manual Tasks into **reusable code**

Utilize **Software Development** practices

Terraform

GitHub GitLab

Version control, Automated testing, CI/CD pipelines

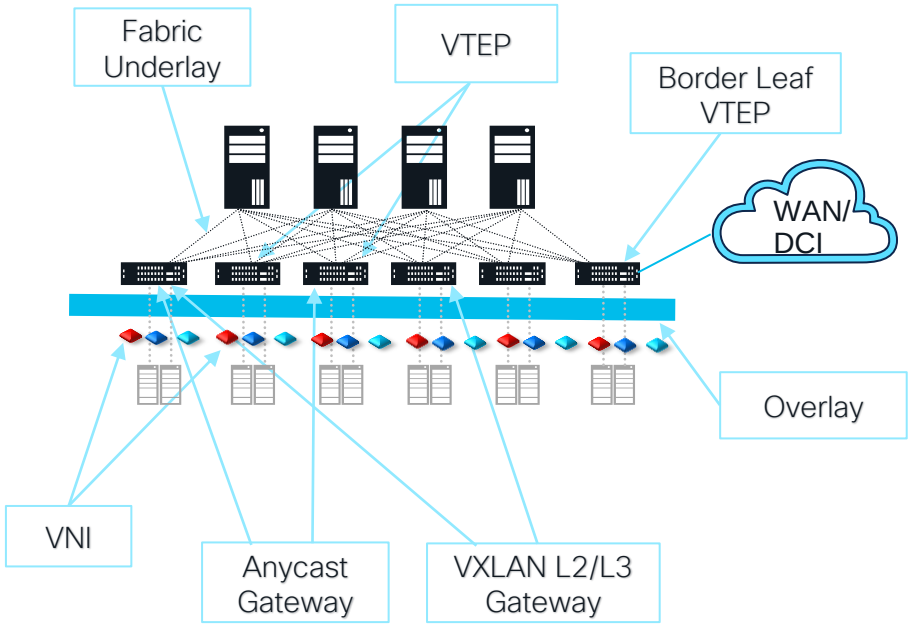
Overview of EVPN (VXLAN Fabric), DCNM & Intersight

VXLAN BGP EVPN

Fabric in Data Center

provides Layer2/Layer3 with Anycast Gateway using Cisco NX-OS

- VTEP – Virtual Tunnel Endpoint: Hardware or software element at the edge for VXLAN encapsulation
- VNI – Virtual Network Instance: A logical network instance for layer 2 broadcast domain
- VNID – Virtual Network Identifier: 24 bits segment ID
- DAG – Distributed Anycast Gateway: Default gateway function across al leaf nodes
- VXLAN L2 Gateway: gateway translate VLAN to VXLAN and VXLAN to VLAN in same BD
- VXLAN L3 Gateway: gateway translate VXLAN to VXLAN or VXLAN to VLAN in different BD



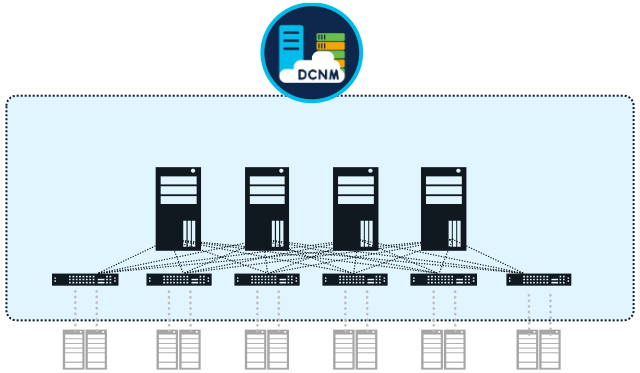
DCNM

Data Center Network Management (DCNM)

provides **Automation**, **Visibility**, **Monitoring** of **VXLAN-EVPN Fabric** for Cisco NX-OS

Day Zero

Bootstrap/POAP, Underlay deploy & Provision (IP Addr, Routing, …) Template based Provisioning



Servers & Virtual Machines

Day 1 / 2

Overlay, API based provisioning, Monitor, Operate

Intersight

Cisco Intersight

SaaS services to bring tools, infrastructure & apps together

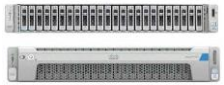
Intersight Service for Terraform (IST)

On-Prem automation using Terraform Cloud, Ease of use (Agent, compliance)

Cloud Orchestrator

Orchestration of Infrastructure & workloads (low-code workflow design)

Infrastructure Services



UCS, Hyperflex

Kubernetes Service (IKS)

k8s Management & Monitoring

Workload Optimization

Covered in Demo

Terraform & Intersight Service for Terraform (IST)

What is Terraform?

Delivers IaC

Infrastructure Management

Write	Plan	Apply
Write infrastructure as code using declarative configuration language (HCL)	Perform dry run Check if Execution Plan matches your expectations	Apply changes to reach the desired state

What is Terraform? Under the hood

Config

Code /
Desired State
(HCL `.tf` File)

State

terraform.tstate

TF keeps track of
infra/environment
Current State

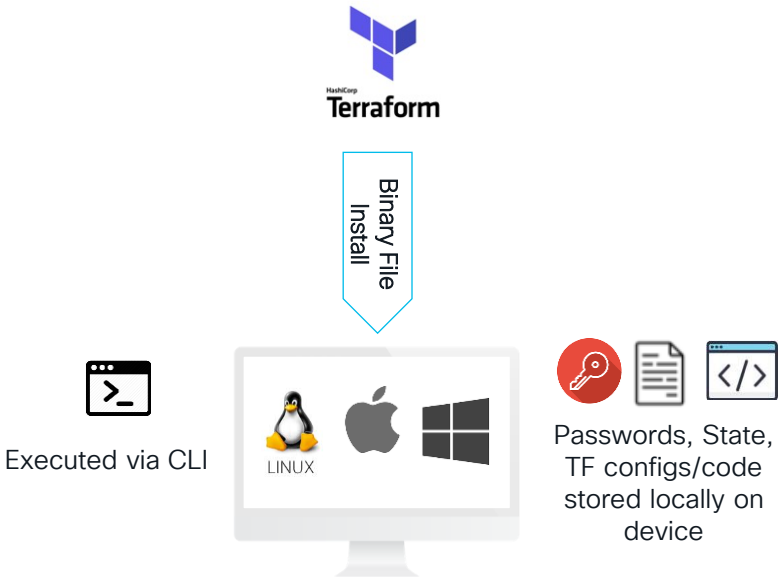
Built dynamicaly &
stored as JSON file

PROVIDER

Plug-ins = Providers
allows Terraform to
interact with
infrastructure using
their APIs

How do you use Terraform?

LOCAL INSTALL



Open Source (OSS)

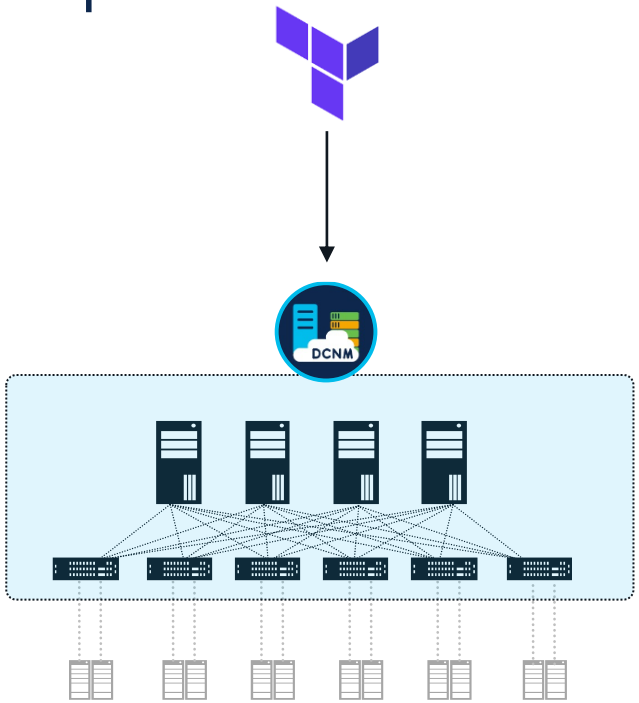
CLOUD DELIVERED

HashiCorp Terraform Cloud

- Passwords/Secrets, RBAC, API, Organization, Workspaces …
- Centralized – share the Terraform State
- Can store configs in a Version Control System (VCS) – github, gitlab, bitbucket
- Paid version: users > 5, different levels of permissions, enforce policies
- Business tier allows private infrastructure (via Terraform Cloud Agents), SSO, …

https://app.terraform.io/app

What are the steps to use Terraform Open Source with DCNM and NX-OS Fabric to provision a VRF?



Terraform – how to write declarative state?

Config File

- HashiCorp Config Language (HCL) syntax
- Which Provider (plugin) to install
- What infrastructure to create
- Filename extension of `.tf`

main.tf

```
terraform {
  required_providers {
    dcnm = {
      source = "CiscoDevNet/dcnm"
    }
  }
}

provider "dcnm" {
  username = "admin"
  password = "password"
  url      = "https://dcnm-domain.com"
  insecure = true
  proxy_url = "https://proxy_server:proxy_port"
  platform = "dcnm"
}
```

Provider "**dcnm**" in Public Terraform registry. Published within namespace of "**CiscoDevNet**"

Additional details such as URL, credentials etc. required for "**dcnm**" provider

Terraform – how to write declarative state?

Config File

terraform init

- Downloads and install provider from the registry upon execution of "`terraform init`" command
- 1000+ Providers

Terminal

```
$ terraform init

Initializing the backend...

Initializing provider plugins...
- Finding latest version of ciscodevnet/dcnm...
- Installing ciscodevnet/dcnm v1.2.2...
- Installed ciscodevnet/dcnm v1.2.2 (signed by a HashiCorp partner, key ID
433649E2C56309DE)

Partner and community providers are signed by their developers.
If you'd like to know more about provider signing, you can read about it here:
https://www.terraform.io/docs/cli/plugins/signing.html

Terraform has created a lock file .terraform.lock.hcl to record the provider
selections it made above. Include this file in your version control repository
so that Terraform can guarantee to make the same selections by default when
you run "terraform init" in the future.

Terraform has been successfully initialized!

You may now begin working with Terraform. Try running "terraform plan" to see
any changes that are required for your infrastructure. All Terraform commands
should now work.

$
```

So, **`provider`** is now installed.

But what element in config file (HCL) declares the desired infra?

Answer:

`resource`

Terraform – how to get desired state?

Config File

terraform init

main.tf

```
terraform { … }
provider "dcnm" { … }

resource "dcnm_vrf" "vrf_db1" {
  fabric_name = "myfabric"
  name        = "db1"
  vlan_name   = "vlan_db1"
  deploy      = true
  attachments {
    serial_number = "9VQPZ1V31O1"
    attach        = true
  }
  attachments {
    serial_number = "9414O6IO5Y0"
    attach        = true
  }
}
```

Resource "**dcnm_vrf**" creates a VRF on Fabric via DCNM

Terraform Resources – DCNM

Reference

https://registry.terraform.io/providers/CiscoDevNet/dcnm/latest/docs

dcnm provider

Resources

- dcnm_interface
- dcnm_inventory
- dcnm_network
- dcnm_policy
- dcnm_rest
- dcnm_route_peering
- dcnm_service_node
- dcnm_service_policy
- dcnm_template
- dcnm_vrf

dcnm_vrf

Manages DCNM VRF

Example Usage

ON THIS PAGE

- Example Usage
- Argument Reference
- Attribute Reference
- Importing

Report an issue

Argument Reference

- `name` - (Required) Name of Object VRF.
- `fabric_name` - (Required) Fabric name under which VRF should be created.
- `segment_id` - (Optional) VRF-Segment id. This field is auto-calculated if not provided. However while creating multiple VRFs in the same plan use this field to reserve the VRF id to avoid any conflicts due to concurrent execution.

Terraform Resources – Intersight

Reference

https://registry.terraform.io/providers/CiscoDevNet/intersight/latest/docs

intersight provider

- aaa
- access
- adapter
- appliance
- asset
- bios
- boot
- bulk
- capability
- certificatemanagement
- chassis
- cloud

kubernetes

Resources

- intersight_kubernetes_aci_cni_apic
- intersight_kubernetes_aci_cni_profile
- intersight_kubernetes_aci_cni_tenant_cluster_allocation
- intersight_kubernetes_addon_definition
- intersight_kubernetes_addon_policy
- intersight_kubernetes_addon_repository
- intersight_kubernetes_baremetal_node_profile
- intersight_kubernetes_cluster
- intersight_kubernetes_cluster_addon_profile

workflow

Resources

- intersight_workflow_ansible_batch_executor
- intersight_workflow_batch_api_executor
- intersight_workflow_custom_data_type_definition
- intersight_workflow_error_response_handler
- intersight_workflow_power_shell_batch_api_executor
- intersight_workflow_rollback_workflow
- intersight_workflow_service_item_action_definition
- intersight_workflow_service_item

Terraform – how to get desired state?

Config File

`terraform init`

`terraform plan`

- Perform dry run based upon your config
- Read current state of infra, compares the proposed config to current state, and proposes changes to infra (e.g., nothing exists, or against the desired state)
- Source of desired state is **`.tf`** configuration file

Terminal

```
$ terraform plan
Refreshing state...
Terraform used the selected providers to generate the following execution plan.
Resource actions are indicated with the following symbols:
  + create

Terraform will perform the following actions:
  # dcnm_vrf.vrf_db1 will be created
  + resource "dcnm_vrf" "vrf_db1" {
      + advertise_default_route = "true"
      + deploy                  = true
      + fabric_name             = "myfabric"
      + id                      = (known after apply)
      + vlan_id                 = (known after apply)
      + vlan_name               = "vlan_db1"
…
      + attachments {
          + attach            = true
          + serial_number     = "9414O6IO5Y0"
…
        }
      + attachments { … }
    }

Plan: 1 to add, 0 to change, 0 to destroy.
```

Terraform – how to get desired state?

Config File

terraform init

terraform plan

terraform apply

- Starts with **plan** (what needs to be done) and then executes (apply) the changes
- Stores the of state of managed infra in **terraform.tfstate** (JSON file)
- View state: "terraform state list", "terraform state show {ID}"

Terminal

```
$ terraform apply
Refreshing state...
Terraform used the selected providers to generate the following execution plan.
Resource actions are indicated with the following symbols:
  + create

Terraform will perform the following actions:

  # dcnm_vrf.vrf_db1 will be created
  + resource "dcnm_vrf" "vrf_db1" {
…
Plan: 1 to add, 0 to change, 0 to destroy.

Do you want to perform these actions?
  Terraform will perform the actions described above.
  Only 'yes' will be accepted to approve.

  Enter a value: yes
dcnm_vrf.vrf_db1: Creating...
dcnm_vrf.vrf_db1: Creation complete after 23s [id=db1]

Apply complete! Resources: 1 added, 0 changed, 0 destroyed.

$
```

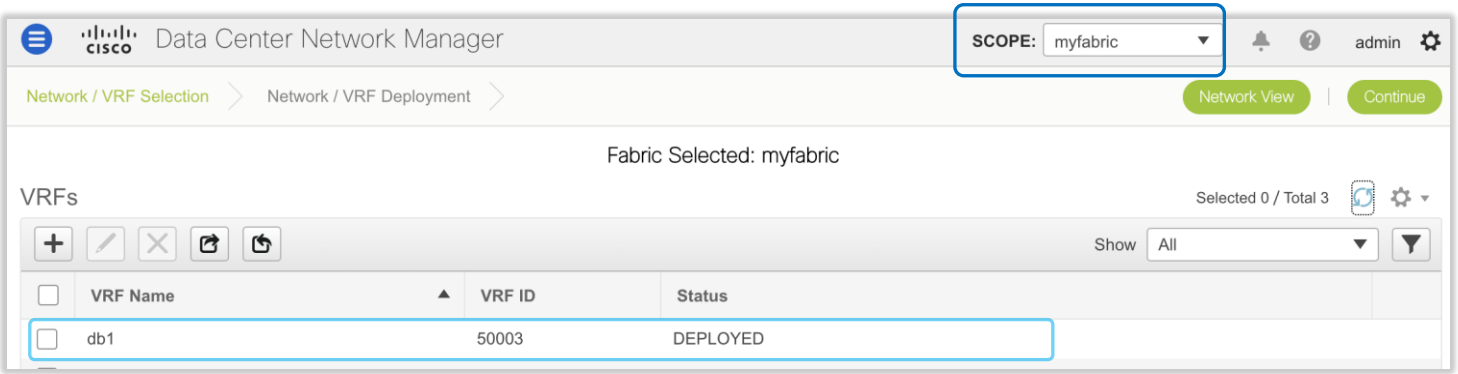
Terraform – how to get desired state?

Config File

`terraform init`

`terraform plan`

`terraform apply`



Terminal

```
Leaf-2# show vrf
VRF-Name                            VRF-ID State    Reason
db1                                      9 Up       --
default                                  1 Up       --
management                               2 Up       --
Leaf-2#
```

Terraform State

Reference

Use "`terraform state`" commands

```
FCHAUDHR-M-C1AR:15_dcnm_local fchaudhr$ terraform state list
dcnm_vrf.vrf_db1
FCHAUDHR-M-C1AR:15_dcnm_local fchaudhr$
FCHAUDHR-M-C1AR:15_dcnm_local fchaudhr$ terraform state show dcnm_vrf.vrf_db1
# dcnm_vrf.vrf_db1:
resource "dcnm_vrf" "vrf_db1" {
    advertise_default_route = "true"
    advertise_host_route    = "false"
    deploy                  = true
    deploy_timeout          = 300
    extension_template      = "Default_VRF_Extension_Universal"
    fabric_name             = "myfabric"
    id                      = "db1"
    ipv6_link_local_flag    = "true"
    max_bgp_path            = 1
    max_ibgp_path           = 2
    mtu                     = 9216
    name                    = "db1"
    rp_external_flag        = "false"
    segment_id              = "50002"
    static_default_route    = "true"
    tag                     = "12345"
    template                = "Default_VRF_Universal"
    trm_bgw_msite_flag      = "false"
    trm_enable              = "false"
    vlan_id                 = 2002
    vlan_name               = "vlan_db1"

    attachments {
        attach        = true
        loopback_id   = 0
        serial_number = "941406I05Y0"
        vlan_id       = 0
    }
    attachments {
        attach        = true
        loopback_id   = 0
        serial_number = "9VQPZ1V3101"
        vlan_id       = 0
    }
}
```

JSON file

{} terraform.tfstate ×

{} terraform.tfstate > [ ] resources > {} 0


```
1  {
2    "version": 4,
3    "terraform_version": "1.1.6",
4    "serial": 18,
5    "lineage": "129d4553-1132-ba19-b514-fe9d38a431b9",
6    "outputs": {},
7    "resources": [
8      {
9        "mode": "managed",
10       "type": "dcnm_vrf",
11       "name": "vrf_db1",
12       "provider": "provider[\"registry.terraform.io/ciscodevnet/dcnm\"]",
13       "instances": [
14         {
15           "schema_version": 0,
16           "attributes": {
17             "advertise_default_route": "true",
18             "advertise_host_route": "false",
19             "attachments": [
20               {
```

Terraform – how to get desired state?

Reference

Config File

terraform init

terraform plan

terraform apply

terraform destroy

- Delete or Decommission
- Runs through a Plan and then executes the cleanup

Terminal

```
$ terraform destroy --target dcnm_vrf.vrf_db1
dcnm_vrf.vrf_db1: Refreshing state... [id=db1]

Terraform used the selected providers to generate the following execution plan.
Resource actions are indicated with the following
symbols:
  - destroy

Terraform will perform the following actions:

  # dcnm_vrf.vrf_db1 will be destroyed
  - resource "dcnm_vrf" "vrf_db1" {
…
Plan: 0 to add, 0 to change, 1 to destroy.
…
Do you really want to destroy all resources?
  Terraform will destroy all your managed infrastructure, as shown above.
  There is no undo. Only 'yes' will be accepted to confirm.

  Enter a value: yes

dcnm_vrf.vrf_db1: Destroying... [id=db1]
dcnm_vrf.vrf_db1: Still destroying... [id=db1, 10s elapsed]
dcnm_vrf.vrf_db1: Destruction complete after 16s
…
Destroy complete! Resources: 1 destroyed.
```

So, in terraform **`resources`** can be used for managing new infrastructure

But how to read or utilize existing infra components?

Answer:
`data source`

Terraform Data Sources – DCNM

Reference

https://registry.terraform.io/providers/CiscoDevNet/dcnm/latest/docs

dcnm provider

- Resources
- Data Sources
  - dcnm_interface
  - dcnm_inventory
  - dcnm_network
  - dcnm_policy
  - dcnm_route_peering
  - dcnm_service_node
  - dcnm_service_policy
  - dcnm_template
  - dcnm_vrf

dcnm_inventory

Data source for DCNM inventory module

Example Usage

ON THIS PAGE

- Example Usage
- Argument Reference
- Attribute Reference

Report an issue

Argument Reference

- `fabric_name` - (Required) fabric name under which inventory should be created.
- `switch_name` - (Required) name of switch.

How to get data from Data Sources?

Reference

main.tf

```
data "dcnm_inventory" "leaf"
{
  for_each     = toset(var.switches)
## Iterate each switch defined in list
variable “swiches”
  fabric_name = var.fabric
  switch_name = each.value
}

output "fabric"
{
  value = data.dcnm_inventory.leaf
}
```

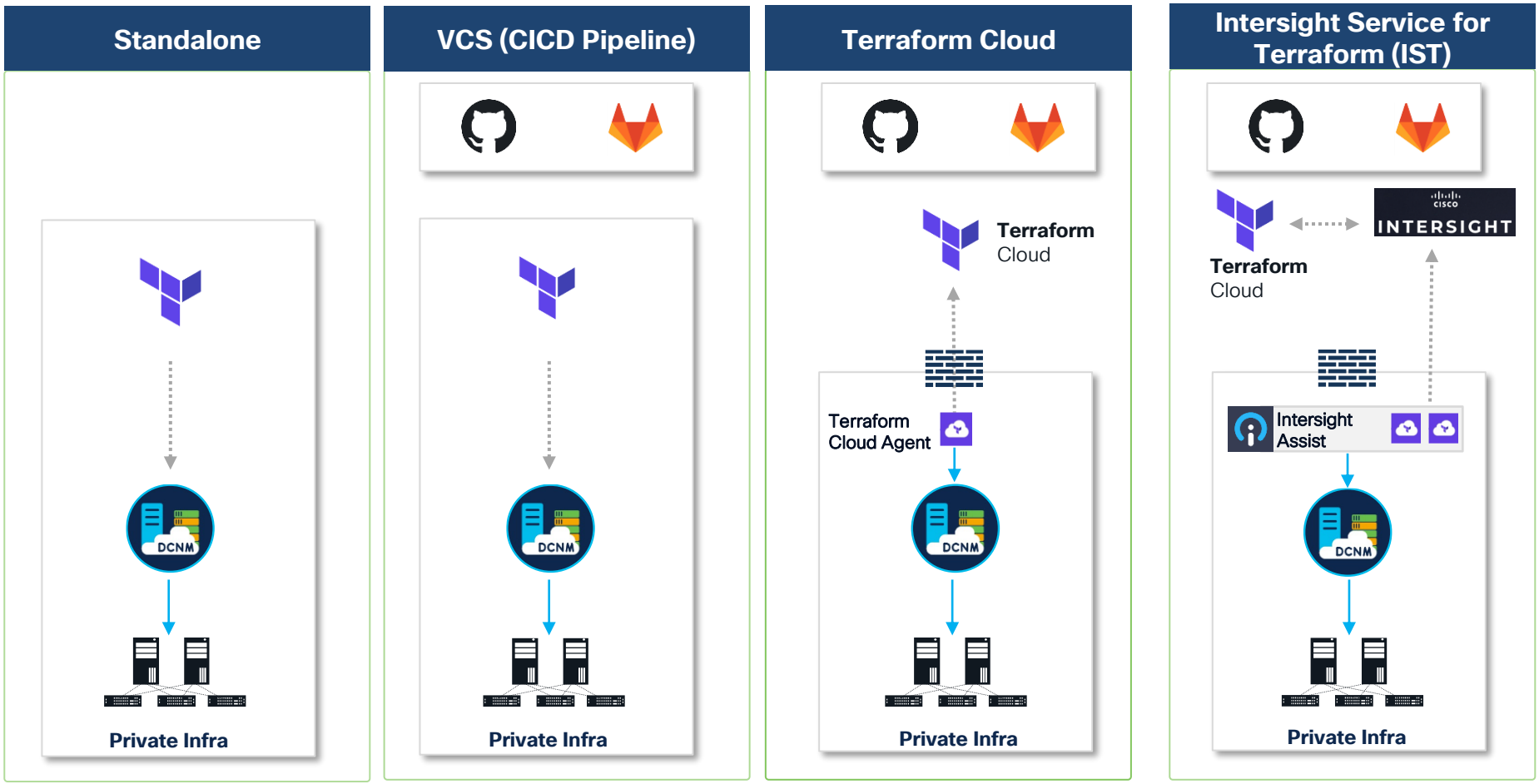
Output from terraform apply

```
Outputs:

fabric = {
  "Leaf-2" = {
    "deploy" = true
    "fabric_name" = "myfabric"
    "id" = "198.18.4.203"
    "ip" = "198.18.4.203"
    "mode" = "Normal"
    "model" = "N9K-9000v"
    "role" = "leaf"
    "serial_number" = "9VQPZ1V31O1"
    "switch_db_id" = "173150"
    "switch_name" = "Leaf-2"
  }
  "Leaf-3" = {
    "deploy" = true
    "fabric_name" = "myfabric"
    "id" = "198.18.4.201"
    "ip" = "198.18.4.201"
    "mode" = "Normal"
    "model" = "N9K-9000v"
    "role" = "leaf"
    "serial_number" = "9414O6IO5Y0"
    "switch_db_id" = "176670"
    "switch_name" = "Leaf-3"
  }
}
```

Deployment scenarios

Deployment scenarios



Terraform Cloud & DCNM

A workspace contains everything used by Terraform to manage a “infra”

Workspace

- VCS
- Credential/Secrets
- vars
- State
- Agent Pools
- ....

Terraform Cloud

password	SENSITIVE	*Sensitive - write only*
username	HCL	"admin"

Triggered via UI
#sv-TC9KDYnAhLEKWwSj | CLUS_TERRAFORM triggered from Terraform | #run-CdqQZgt3rMDARj9 | eba377d 2 days ago

Triggered via UI
#sv-HSLpsVGSFGbRniFG | CLUS_TERRAFORM triggered from Terraform | #run-R9JkvWy3EA2eLUKw | eba377d 2 days ago

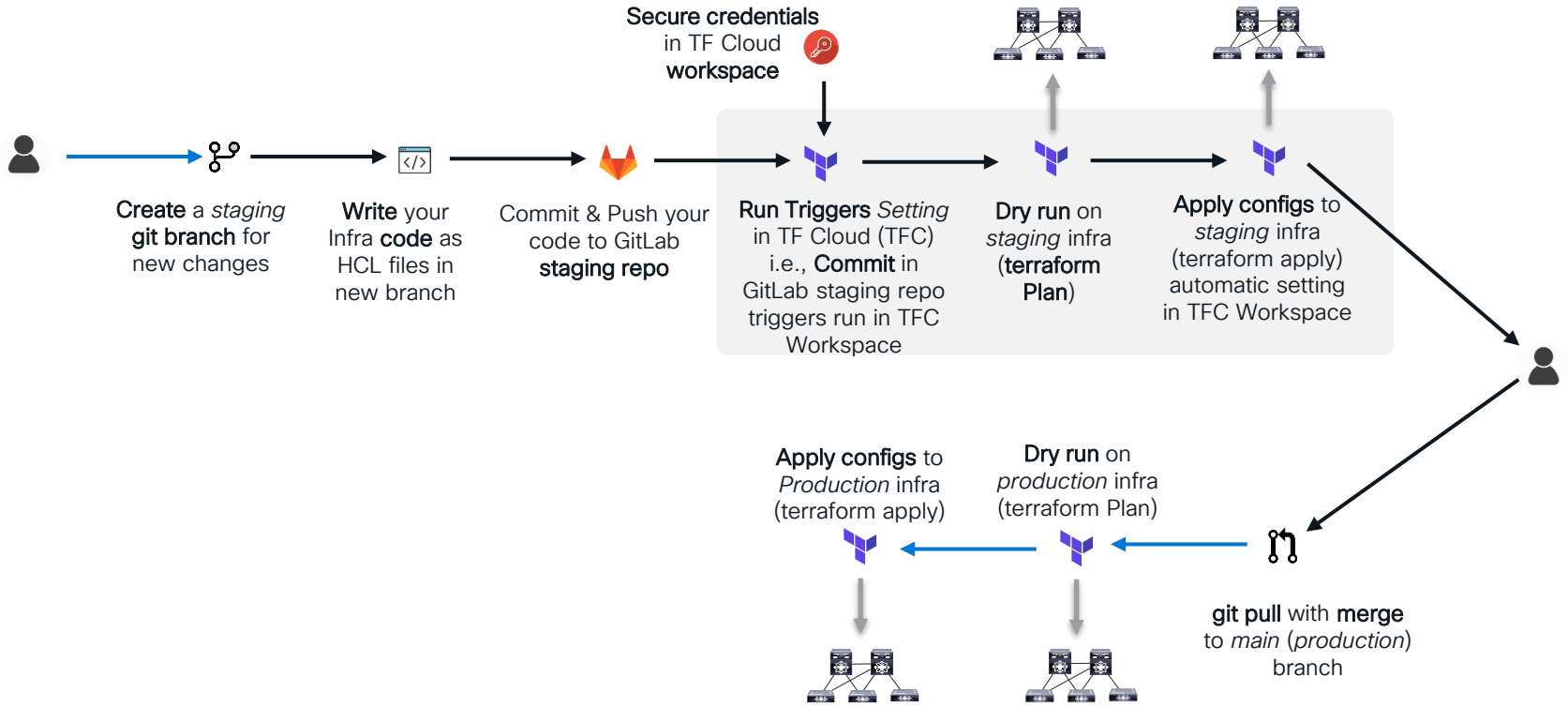
DCNM

Terrafrom Cloud Agents

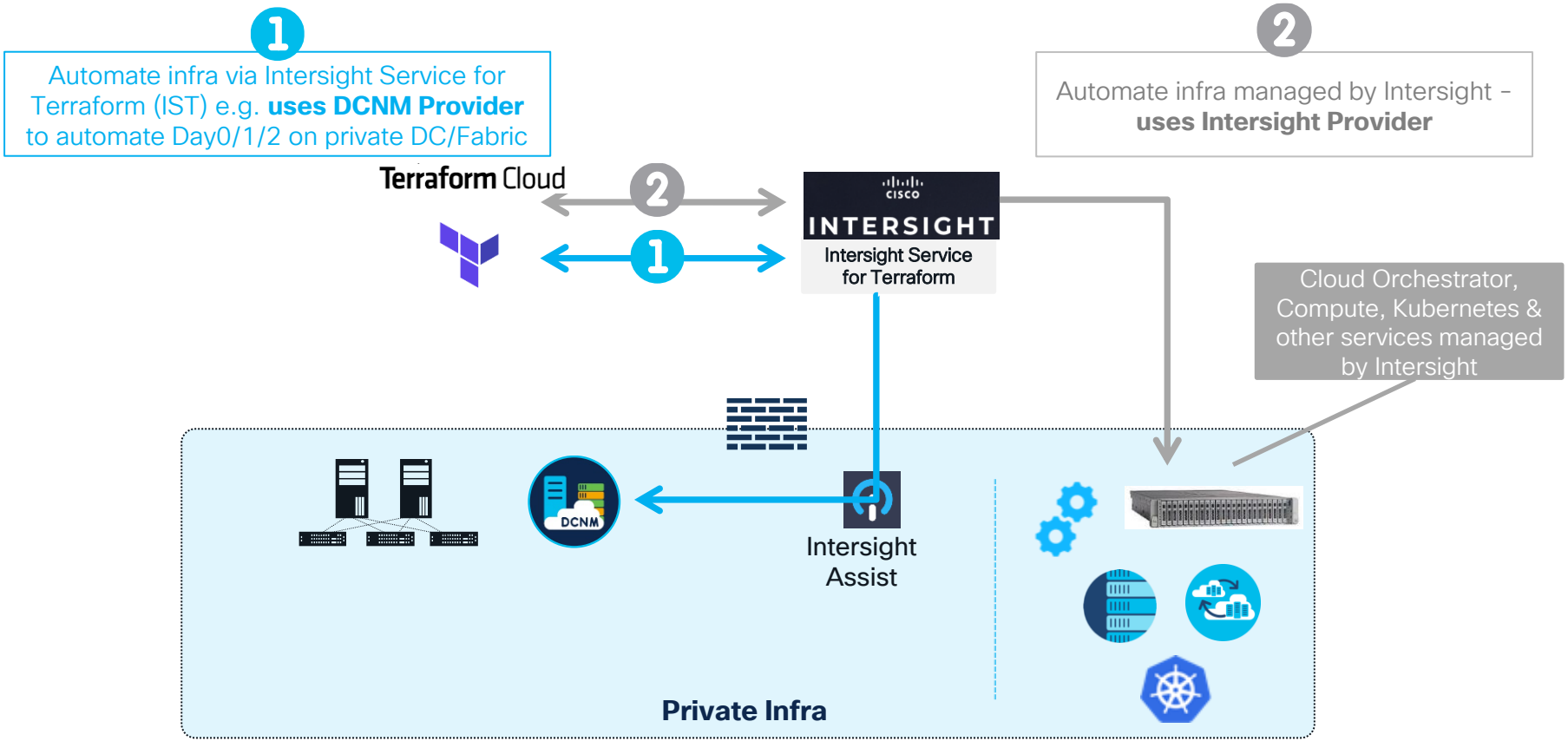
Private Infra

Terraform Cloud – Workflow

Reference

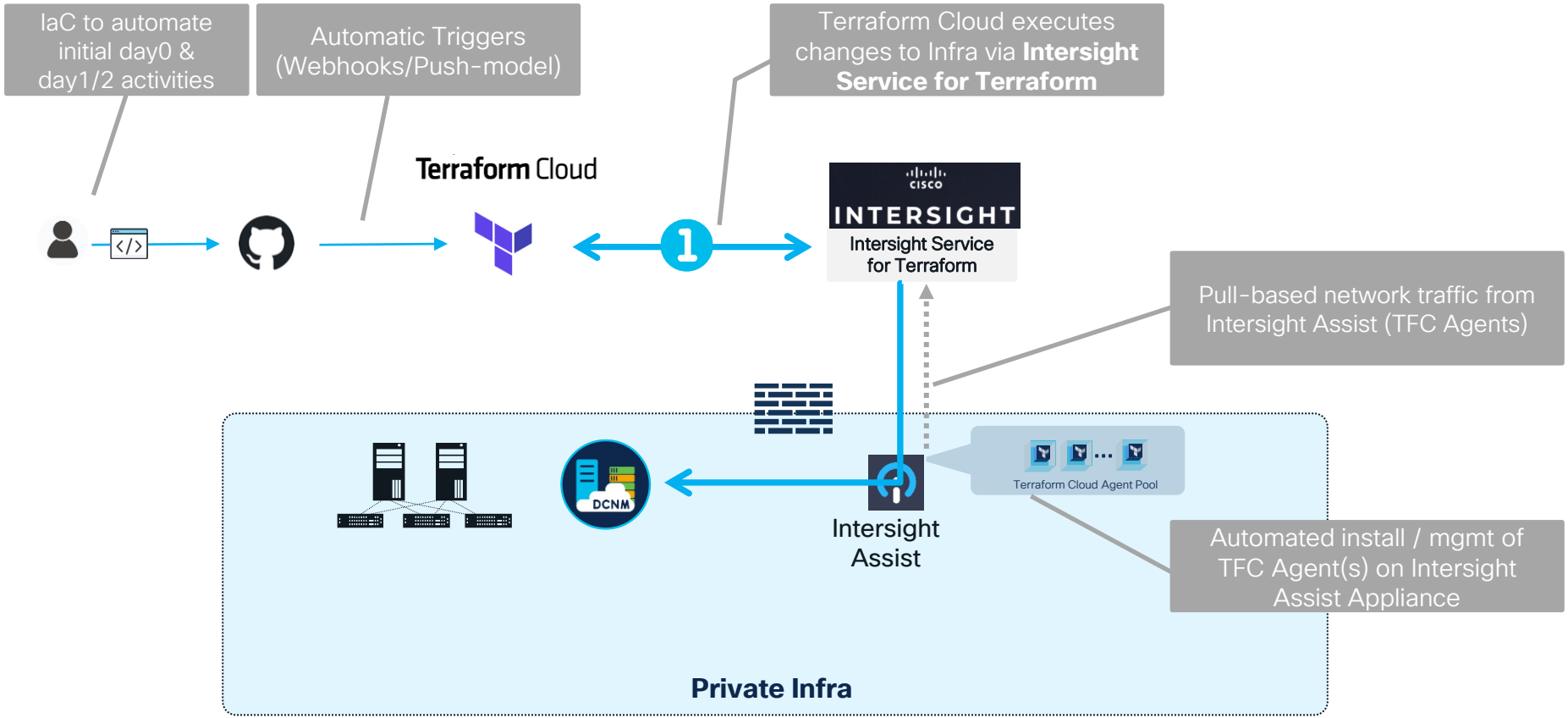


Terraform Cloud & Intersight



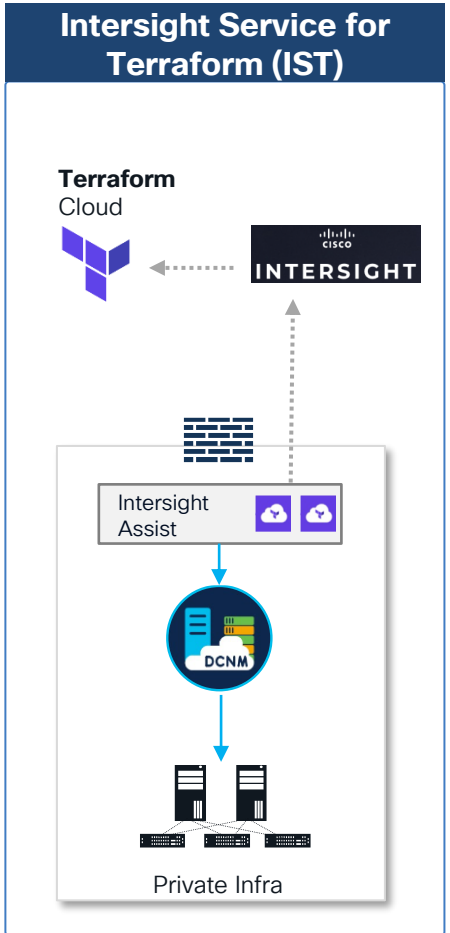
Terraform Cloud & Intersight – Workflow

Reference



Terraform Cloud & Intersight – Workflow

Reference



- Intersight Service for Terraform (IST) responsible for the end-to-end lifecycle of Terraform Cloud Agents besides Compute, Orchestration, MultiCloud workload Optimization, IKS …
- Intersight Assist automates Terraform Cloud (TFC) Agent on-boarding process:
  - Fetch the agent pool selected by user
  - Create an agent token using TFC API
  - Spin off a Terraform Agent in the customers datacenter with above token and the agent should register with TFC Business
- Pool of licenses for TFC Agent – assigned via Intersight
- "*Managed Host*" list (IP, repo URL) that Agent communicate & execute IaC
- TFC Agent: pull-based, so no inbound connectivity is required
  - Agent polls via Intersight and carry out execution work locally
  - Admin configures the TFC Workspace to target/use specific organization's agents (once installed)

Show me in action!

Demo – Terraform Cloud & Intersight

Github repo for demo and code:
https://github.com/ciscolivelab/DEVNET-3011

Technical Session Surveys

- Attendees who fill out a minimum of four session surveys and the overall event survey will get Cisco Live branded socks!
- Attendees will also earn 100 points in the Cisco Live Game for every survey completed.
- These points help you get on the leaderboard and increase your chances of winning daily and grand prizes.

Cisco Learning and Certifications

From technology training and team development to Cisco certifications and learning plans, let us help you empower your business and career. www.cisco.com/go/certs

Pay for Learning with Cisco Learning Credits

(CLCs) are prepaid training vouchers redeemed directly with Cisco.

Learn

Cisco U.
IT learning hub that guides teams and learners toward their goals

Cisco Digital Learning
Subscription-based product, technology, and certification training

Cisco Modeling Labs
Network simulation platform for design, testing, and troubleshooting

Cisco Learning Network
Resource community portal for certifications and learning

Train

Cisco Training Bootcamps
Intensive team & individual automation and technology training programs

Cisco Learning Partner Program
Authorized training partners supporting Cisco technology and career certifications

Cisco Instructor-led and Virtual Instructor-led training
Accelerated curriculum of product, technology, and certification courses

Certify

Cisco Certifications and Specialist Certifications
Award-winning certification program empowers students and IT Professionals to advance their technical careers

Cisco Guided Study Groups
180-day certification prep program with learning and support

Cisco Continuing Education Program
Recertification training options for Cisco certified individuals

Here at the event? Visit us at **The Learning and Certifications lounge at the World of Solutions**

Continue your education

- Visit the Cisco Showcase for related demos
- Book your one-on-one Meet the Engineer meeting
- Attend the interactive education with DevNet, Capture the Flag, and Walk-in Labs
- Visit the On-Demand Library for more sessions at www.CiscoLive.com/on-demand

Thank you

cisco Live!

ALL IN

#CiscoLive