cisco Live!

Let’s go

#CiscoLive

The bridge to possible

Real-World Automation in Multidomain IBN Networks

Jeremy Bowman
@ibnsrevenge
BRKOPS-3028

Cisco Webex App

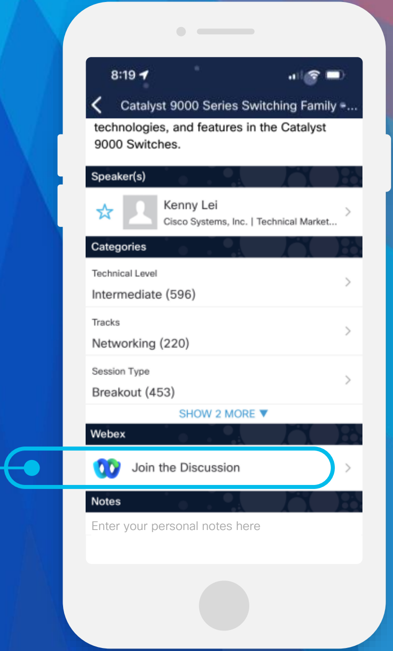
Questions?

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 June 9, 2023.


8:19
Catalyst 9000 Series Switching Family ...
technologies, and features in the Catalyst 9000 Switches.
Speaker(s)
Kenny Lei
Cisco Systems, Inc. | Technical Market...
Categories
Technical Level
Intermediate (596)
Tracks
Networking (220)
Session Type
Breakout (453)
SHOW 2 MORE
Webex
Join the Discussion
Notes
Enter your personal notes here

https://ciscolive.ciscoevents.com/ciscolivebot/#BRKOPS-3028

cisco Live!

Who are you?

Jeremy Bowman

Sr. Delivery Architect

Cisco CX

8+ Years @ Cisco

CCIE #51241 (R/S, Security)

CCDE #2018::16

Specialized in: Full Enterprise IBN with Security and Automation

@ibnsrevenge

jdb1@cisco.com

“Everyone knew it was impossible, until a fool who didn’t know came along and did it.”

Albert Einstein

Agenda

- What Are Multidomain IBN Networks
- How Can Automation Help
- Automation Troubles
- Simple Use Case: Device Password Management
- More Complicated: CI/CD Template Management
- Complicated: New Client Segmentation
- Conclusion

Multidomain Networks

Multidomain IBN Networks

SD-Access
Cisco DNA Center

SD-WAN
Cisco vManage

ACI
Cisco APIC

Campus and IoT

Users & Devices

- Identify and onboard everything
- Authenticate and authorize access

+

Branch/WAN

Hybrid Cloud

- Deliver great application experience
- Secure internet and cloud access

+

Data Center and Cloud

Data & Applications

- Automate resources and workloads
- Prevent data breaches

Characteristics

- Unique ‘controller’ for each domain
  - vManage, APIC, DNAC, Meraki Cloud
- Different network architectures
  - OMP Route-Reflector Control Plane, IPsec Data Plane
  - COOP, MP-BGP eVPN, VXLAN
  - LISP, VXLAN, Cisco TrustSec
- Different API approaches
  - Even login/token differs

Automation

Automation and Orchestration

- Automation
  - Performing an action on a single device without human intervention.
  - Would EEM qualify?
  - What about the same one change on multiple devices?

- Orchestration
  - Performing various unique automation changes in a coordinated way to achieve a desired state.
  - Domain One and Domain Two should work together

Enterprise Concerns When Moving to IBN

- Monitoring
  - Enterprise tools utilize older practices
  - Streaming Telemetry
  - Monitoring versus Observability

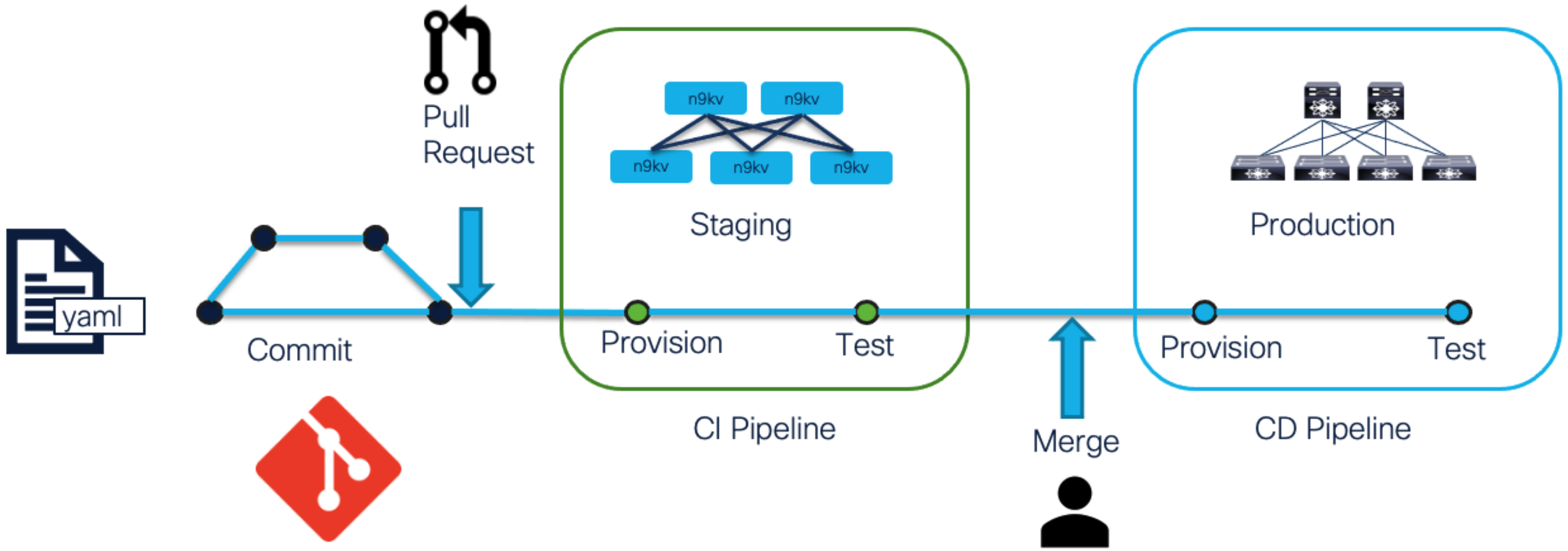
- Management
  - Enterprise tools written with CLI in mind
  - Domain controllers use UI

CI/CD and IaC

- Continuous Integration/Continuous Delivery
  - Configurations centrally stored in a repository
  - Production environment same as test environment
  - Validated testing

- Infrastructure as Code
  - State is maintained via templates, YAML
  - Is reproducible

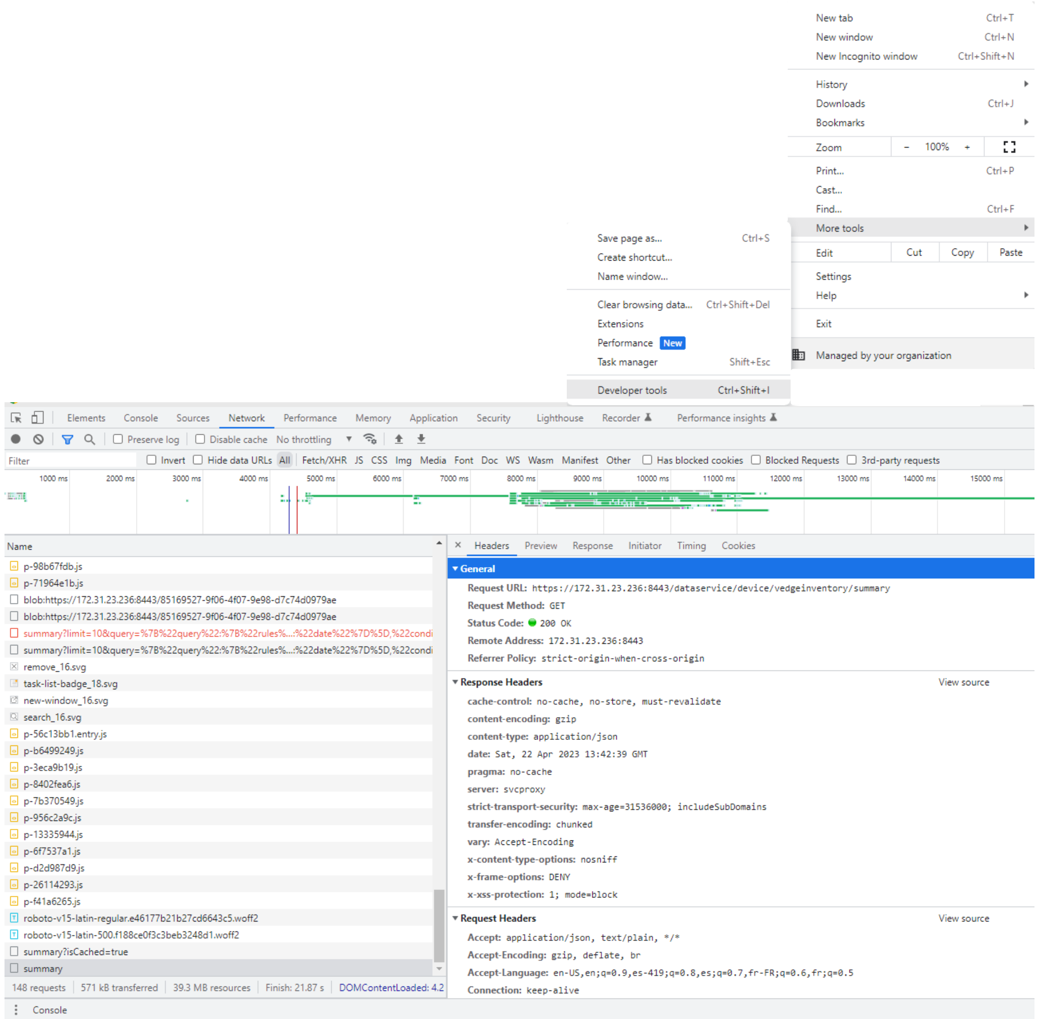
CI/CD and IaC


yaml
Commit
Pull Request
n9kv
n9kv
n9kv
n9kv
n9kv
Staging
Provision
Test
CI Pipeline
Merge
Production
Provision
Test
CD Pipeline

Automation Troubles

Automation Concerns

- How do I get started?
  - May be overwhelming at first, but not impossible.
  - The UI is the API.
  - Best Friends:
    - Chrome Inspection
    - Postman
    - curl

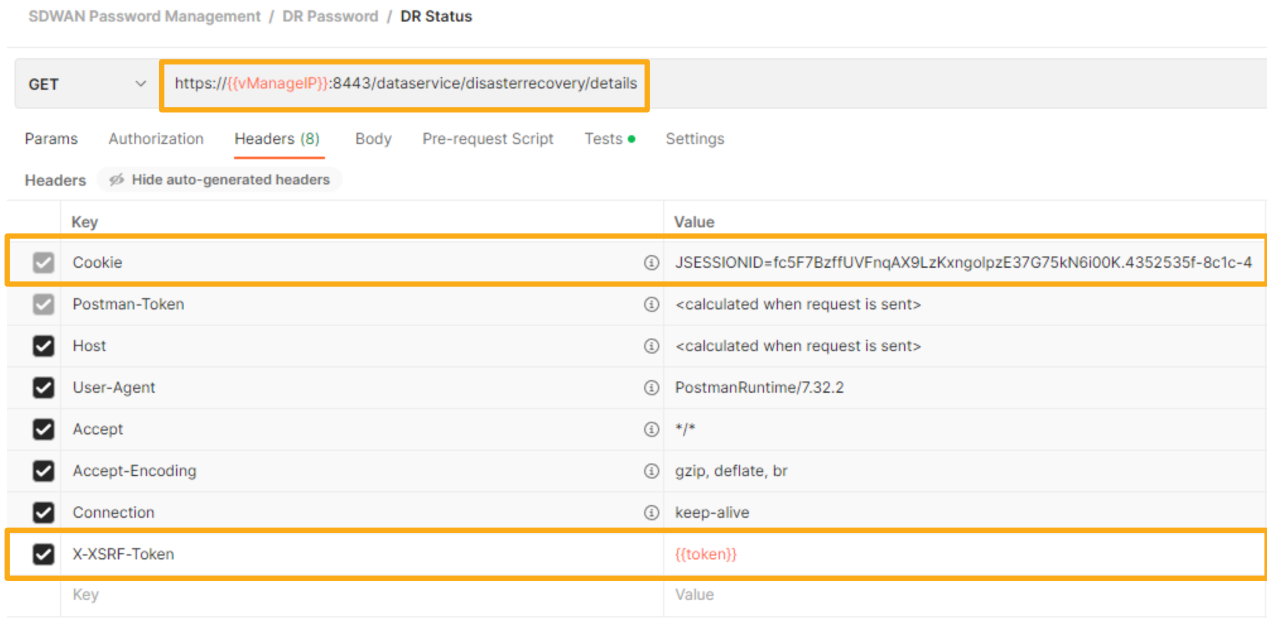


```
Name
p-98b67fdb.js
p-71964e1b.js
blob:https://172.31.23.236:8443/85169527-9f06-4f07-9e98-d7c74d0979ae
blob:https://172.31.23.236:8443/85169527-9f06-4f07-9e98-d7c74d0979ae
summary?limit=10&query=%7B%22query%22:%7B%22rules%...:%22date%22%7D%5D,%22condi
summary?limit=10&query=%7B%22query%22:%7B%22rules%...:%22date%22%7D%5D,%22condi
remove_16.svg
task-list-badge_18.svg
new-window_16.svg
search_16.svg
p-56c13bb1.entry.js
p-b6499249.js
p-3eca9b19.js
p-8402fea6.js
p-7b370549.js
p-956c2a9c.js
p-13335944.js
p-6f7537a1.js
p-d2d987d9.js
p-26114293.js
p-f41a6265.js
roboto-v15-latin-regular.e46177b21b27cd6643c5.woff2
roboto-v15-latin-500.f188ce0f3c3beb3248d1.woff2
summary?isCached=true
summary

148 requests | 571 kB transferred | 39.3 MB resources | Finish: 21.87 s | DOMContentLoaded: 4.2
```

```
▼ General
  Request URL: https://172.31.23.236:8443/dataservice/device/vedgeinventory/summary
  Request Method: GET
  Status Code: ● 200 OK
  Remote Address: 172.31.23.236:8443
  Referrer Policy: strict-origin-when-cross-origin
▼ Response Headers                                              View source
  cache-control: no-cache, no-store, must-revalidate
  content-encoding: gzip
  content-type: application/json
  date: Sat, 22 Apr 2023 13:42:39 GMT
  pragma: no-cache
  server: svcproxy
  strict-transport-security: max-age=31536000; includeSubDomains
  transfer-encoding: chunked
  vary: Accept-Encoding
  x-content-type-options: nosniff
  x-frame-options: DENY
  x-xss-protection: 1; mode=block
▼ Request Headers                                               View source
  Accept: application/json, text/plain, */*
  Accept-Encoding: gzip, deflate, br
  Accept-Language: en-US,en;q=0.9,es-419;q=0.8,es;q=0.7,fr-FR;q=0.6,fr;q=0.5
  Connection: keep-alive
```

Automation Concerns

- What about enterprise security?
  - No hardcoded passwords
  - Uses TLS
  - OWASP followed



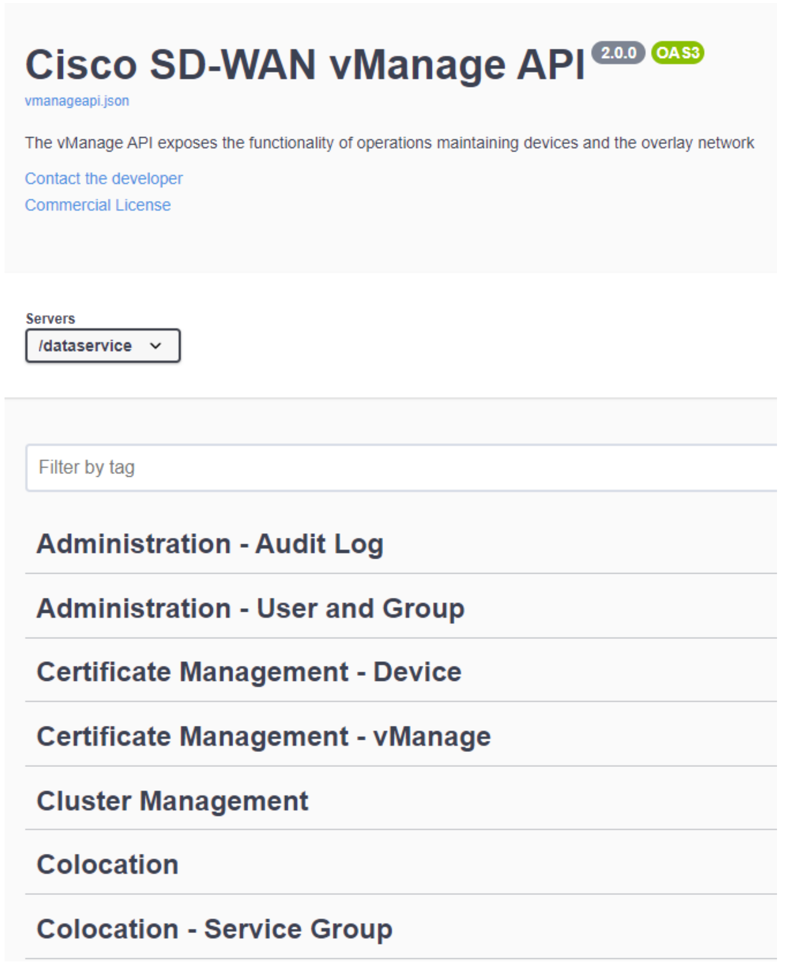
SDWAN Password Management / DR Password / **DR Status**

GET https://{{vManageIP}}:8443/dataservice/disasterrecovery/details

Key	Value
Cookie	JSESSIONID=fc5F7BzffUVFnqAX9LzKxngolpzE37G75kN6i00K.4352535f-8c1c-4
Postman-Token	<calculated when request is sent>
Host	<calculated when request is sent>
User-Agent	PostmanRuntime/7.32.2
Accept	*/*
Accept-Encoding	gzip, deflate, br
Connection	keep-alive
X-XSRF-Token	{{token}}
Key	Value

Automation Concerns

- Where can I find documentation and examples?
  - DevNet – https://developer.cisco.com/docs
  - DevNet Blogs – https://blogs.cisco.com/developer
  - DevNet Learning Labs – https://developer.cisco.com/learning/labs
  - Swagger to try it out (lab) – https://{vManage}:8443/apidocs



Cisco SD-WAN vManage API 2.0.0 OAS3

vmanageapi.json

The vManage API exposes the functionality of operations maintaining devices and the overlay network

Contact the developer

Commercial License

Servers

/dataservice

Filter by tag

Administration - Audit Log

Administration - User and Group

Certificate Management - Device

Certificate Management - vManage

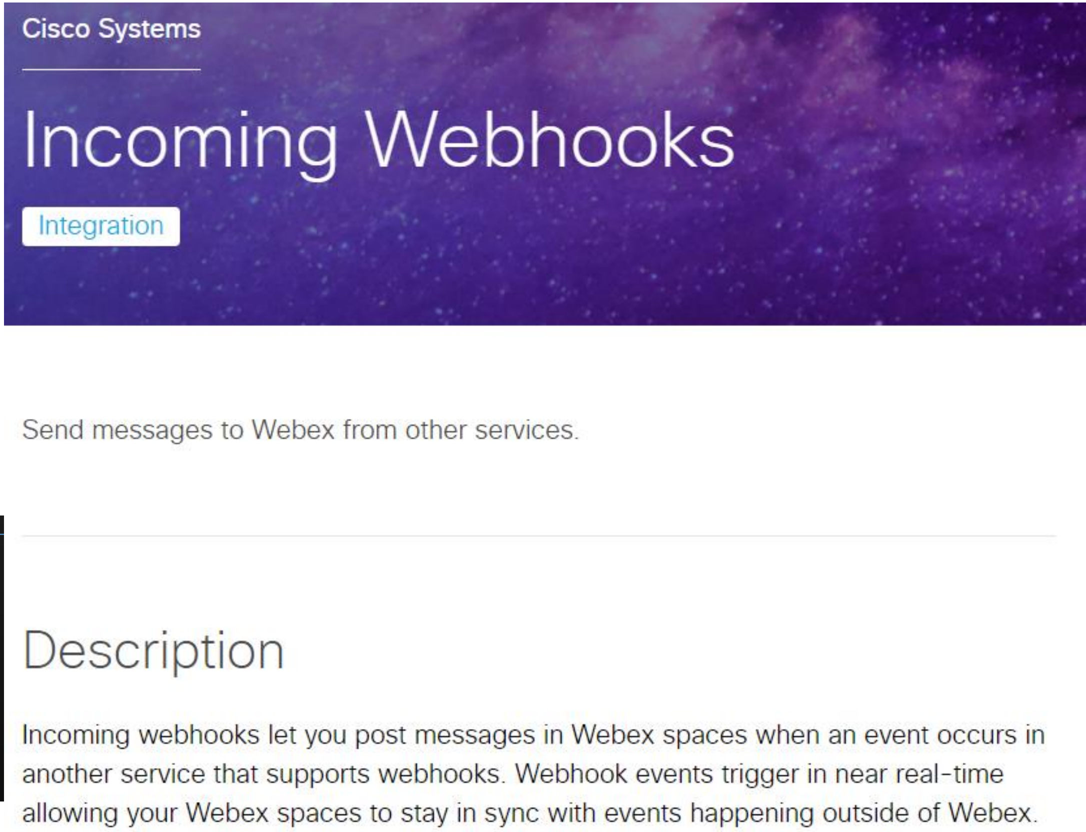
Cluster Management

Colocation

Colocation - Service Group

Automation Concerns

- What about results?
  - WebEx has APIs too!
  - Post results as a markdown message in WebEx Teams
  - Incoming Webhooks
  - Not a bot
  - Cloud based – TLS


Cisco Systems

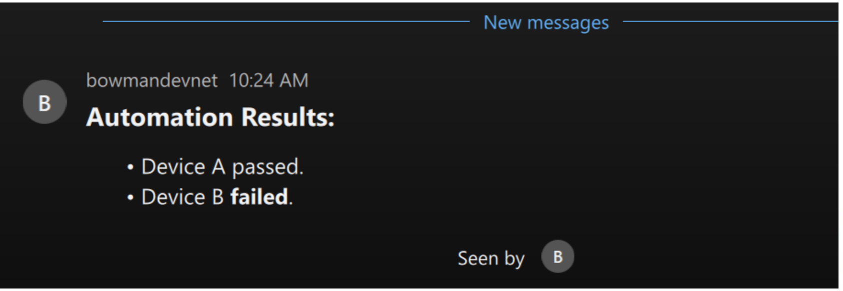
Incoming Webhooks

Integration

Send messages to Webex from other services.

Description

Incoming webhooks let you post messages in Webex spaces when an event occurs in another service that supports webhooks. Webhook events trigger in near real-time allowing your Webex spaces to stay in sync with events happening outside of Webex.


New messages

B bowmandevnet 10:24 AM

Automation Results:

- Device A passed.
- Device B **failed**.

Seen by B

Device Password Management

Use Case

- Multiple Domains
  - SDWAN
  - SDA

- Security Requirements
  - Last Resort Password (local admin user) must change every 90 days.
  - Hundreds of SDWAN routers with many device templates.
  - Hundreds of SDA fabric devices.
  - Passwords managed via 3rd party tool.
  - Same password or different password per device or domain?

Solution Breakdown – One Piece at a Time

- Password Management Tool
  - Third party tool
  - Limited access – SecOps only
  - Manages passwords and updates on device schedule.

- Supported Options
  - SSH to device
  - HTTPS to ‘controller’
  - Python scripting

Solution Breakdown – One Piece at a Time

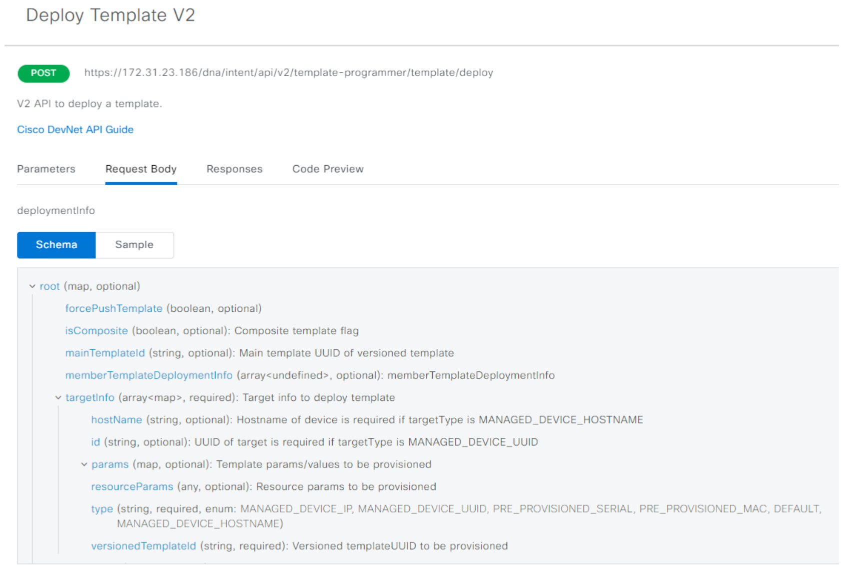
- SDWAN
  - Passwords are variables in templates.
  - No other variables are changing.

- High Level API Workflow
  - vManage Login and Token
  - Determine template attached to device(s)
  - Export the template CSV (list of dictionaries)
  - Update CSV and push to vManage

Solution Breakdown – One Piece at a Time

- SDA
  - Passwords are inherited from DNAC site design hierarchy.
  - Additional users can be managed via CLI templates.

- High Level API Workflow
  - DNAC Login and Token
  - Obtain password template ID
  - Deploy template with updated password


Deploy Template V2

POST https://172.31.23.186/dna/intent/api/v2/template-programmer/template/deploy

V2 API to deploy a template.

Cisco DevNet API Guide

```
root (map, optional)
    forcePushTemplate (boolean, optional)
    isComposite (boolean, optional): Composite template flag
    mainTemplateId (string, optional): Main template UUID of versioned template
    memberTemplateDeploymentInfo (array<undefined>, optional): memberTemplateDeploymentInfo
    targetInfo (array<map>, required): Target info to deploy template
        hostName (string, optional): Hostname of device is required if targetType is MANAGED_DEVICE_HOSTNAME
        id (string, optional): UUID of target is required if targetType is MANAGED_DEVICE_UUID
        params (map, optional): Template params/values to be provisioned
        resourceParams (any, optional): Resource params to be provisioned
        type (string, required, enum: MANAGED_DEVICE_IP, MANAGED_DEVICE_UUID, PRE_PROVISIONED_SERIAL, PRE_PROVISIONED_MAC, DEFAULT,
            MANAGED_DEVICE_HOSTNAME)
        versionedTemplateId (string, required): Versioned templateUUID to be provisioned
```

Final Solution

- Password Management Tool Initiated
  - Selects device for update.
  - Determines domain for the device selected
    - SDWAN
    - SDA
  - Generates a new random password
  - Uses API calls based on domain workflow
  - Validates new password after modifying AAA order to prefer local over TACACS
  - Restores AAA order or preference

Final Solution – SDWAN

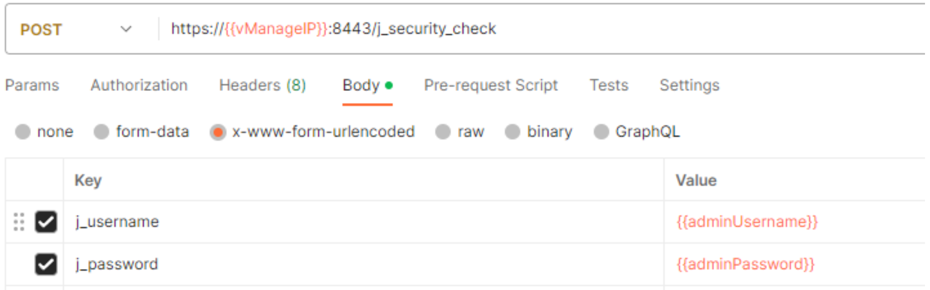
- Login
  - Two steps: Cookie and Token

/j_security_check

returns a cookie

/dataservice/client/token

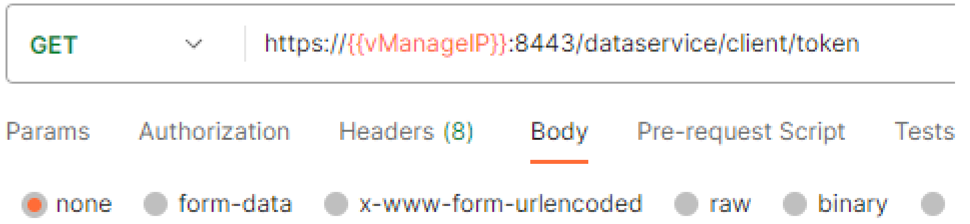
returns the token in the bod



POST https://{{vManageIP}}:8443/j_security_check

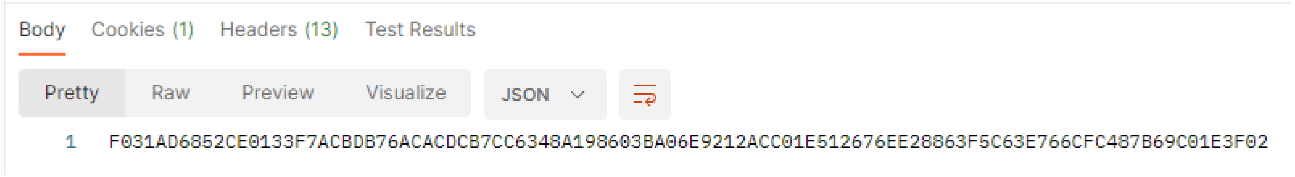
none | form-data | [✓ x-www-form-urlencoded] | raw | binary | GraphQL

	Key	Value
✓	j_username	{{adminUsername}}
✓	j_password	{{adminPassword}}



GET https://{{vManageIP}}:8443/dataservice/client/token

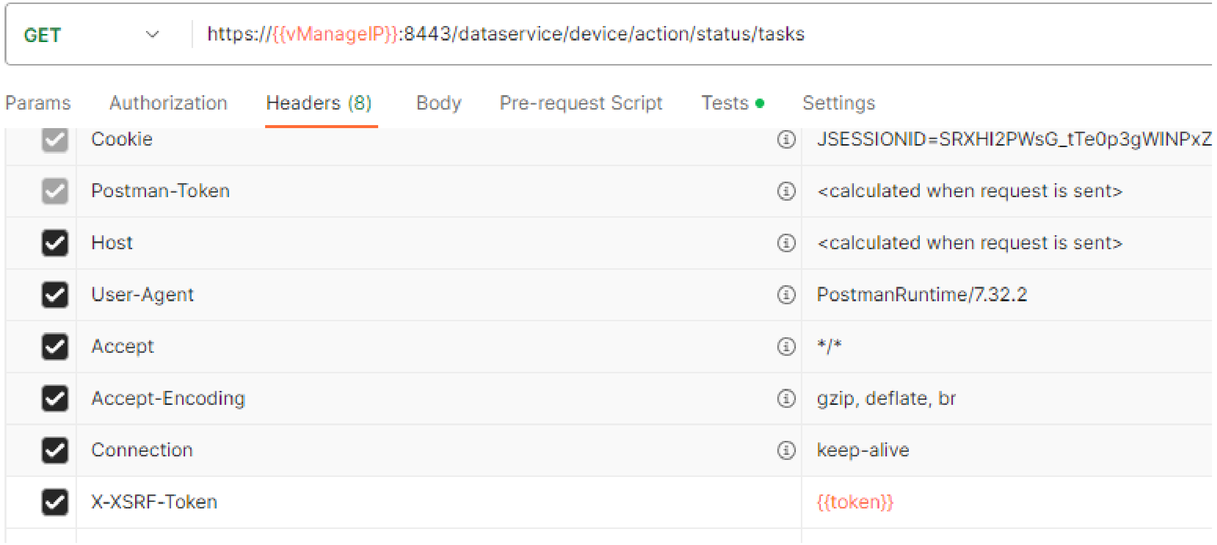
[✓ none] | form-data | x-www-form-urlencoded | raw | binary



```
F031AD6852CE0133F7ACBDB76ACACDCB7CC6348A198603BA06E9212ACC01E512676EE28863F5C63E766CFC487B69C01E3F02
```

Final Solution – SDWAN

- Subsequent Calls
  - Cookie and token provided in header



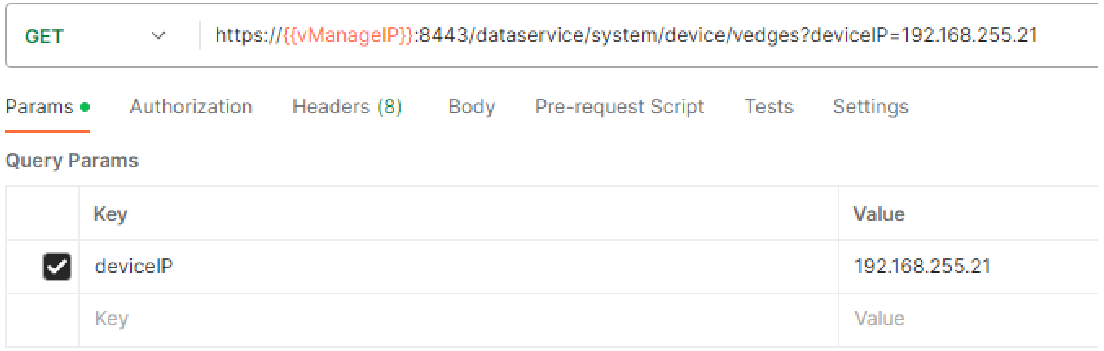
GET https://{{vManageIP}}:8443/dataservice/device/action/status/tasks

Params | Authorization | Headers (8) | Body | Pre-request Script | Tests | Settings

✓	Cookie	JSESSIONID=SRXHI2PWsG_tTe0p3gWINPxZ
✓	Postman-Token	<calculated when request is sent>
✓	Host	<calculated when request is sent>
✓	User-Agent	PostmanRuntime/7.32.2
✓	Accept	*/*
✓	Accept-Encoding	gzip, deflate, br
✓	Connection	keep-alive
✓	X-XSRF-Token	{{token}}

Final Solution – SDWAN

- Identify Template ID Attached to Target Device
- Identify Chassis Number of Device



GET https://{{vManageIP}}:8443/dataservice/system/device/vedges?deviceIP=192.168.255.21

Params ● | Authorization | Headers (8) | Body | Pre-request Script | Tests | Settings

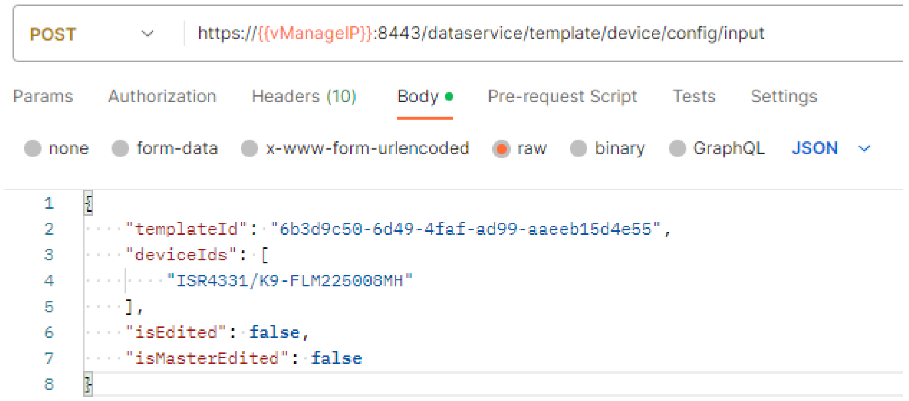
Query Params

	Key	Value
✓	deviceIP	192.168.255.21
	Key	Value

```
{
  "data": [
    {
      "deviceIP": "192.168.255.21",
      "chasisNumber": "ISR4331/K9-
FLM225008MH",
      "site-id": "3001",
      "host-name": "SOME_HOSTNAME",
      "availableVersions": [
        "17.06.03a.0.3"
      ],
      "template": "SOME_TEMPLATE",
      "templateId": "6b3d9c50-6d49-4faf-
ad99-aaeeb15d4e55"
    }
  ]
}
```

Final Solution – SDWAN

- Use Information to Get Current Variable Values



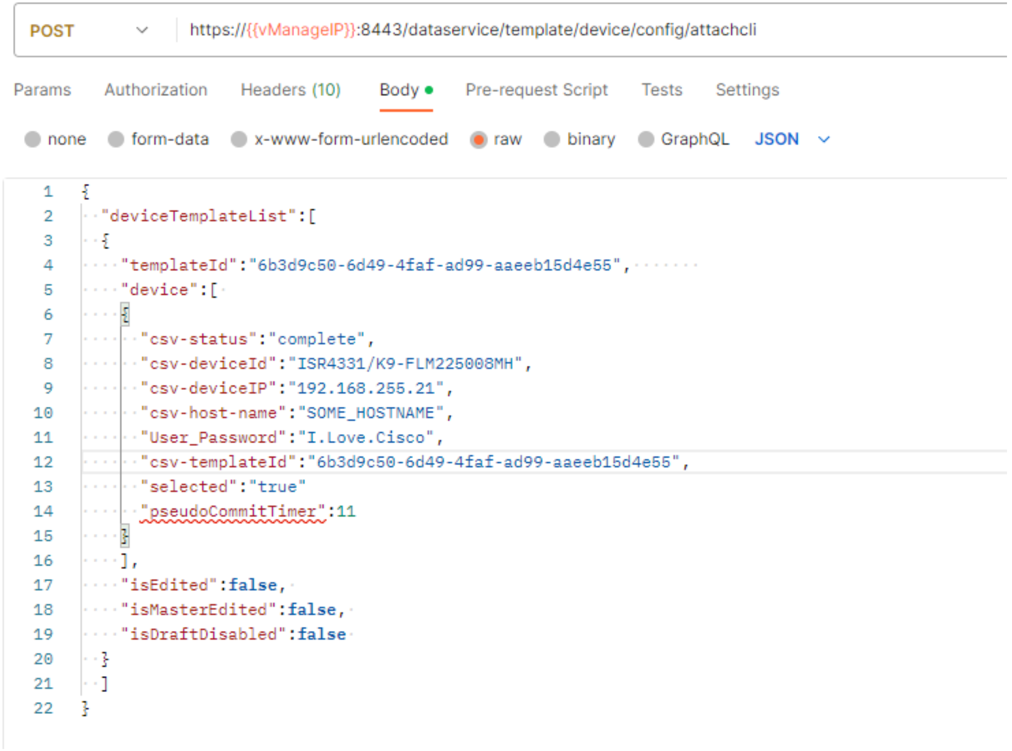
```
https://{{vManageIP}}:8443/dataservice/template/device/config/input
```

```
{
    "templateId": "6b3d9c50-6d49-4faf-ad99-aaeeb15d4e55",
    "deviceIds": [
        "ISR4331/K9-FLM225008MH"
    ],
    "isEdited": false,
    "isMasterEdited": false
}
```

```
{
  "data": [
    {
      "csv-status": "complete",
      "csv-deviceId": "ISR4331/K9-FLM225008MH",
      "csv-deviceIP": "192.168.255.21",
      "csv-host-name": "SOME_HOSTNAME",
      "User_Password": "cisco.123"
    }
  ]
}
```

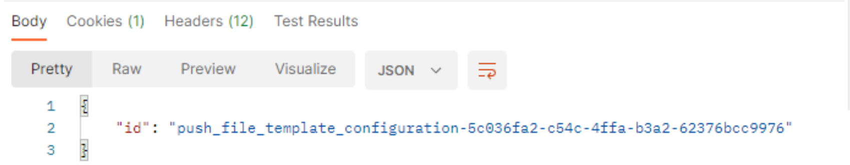
Final Solution – SDWAN

- POST variables back to vManage with new password.
- Returns a Task ID



POST https://{{vManageIP}}:8443/dataservice/template/device/config/attachcli

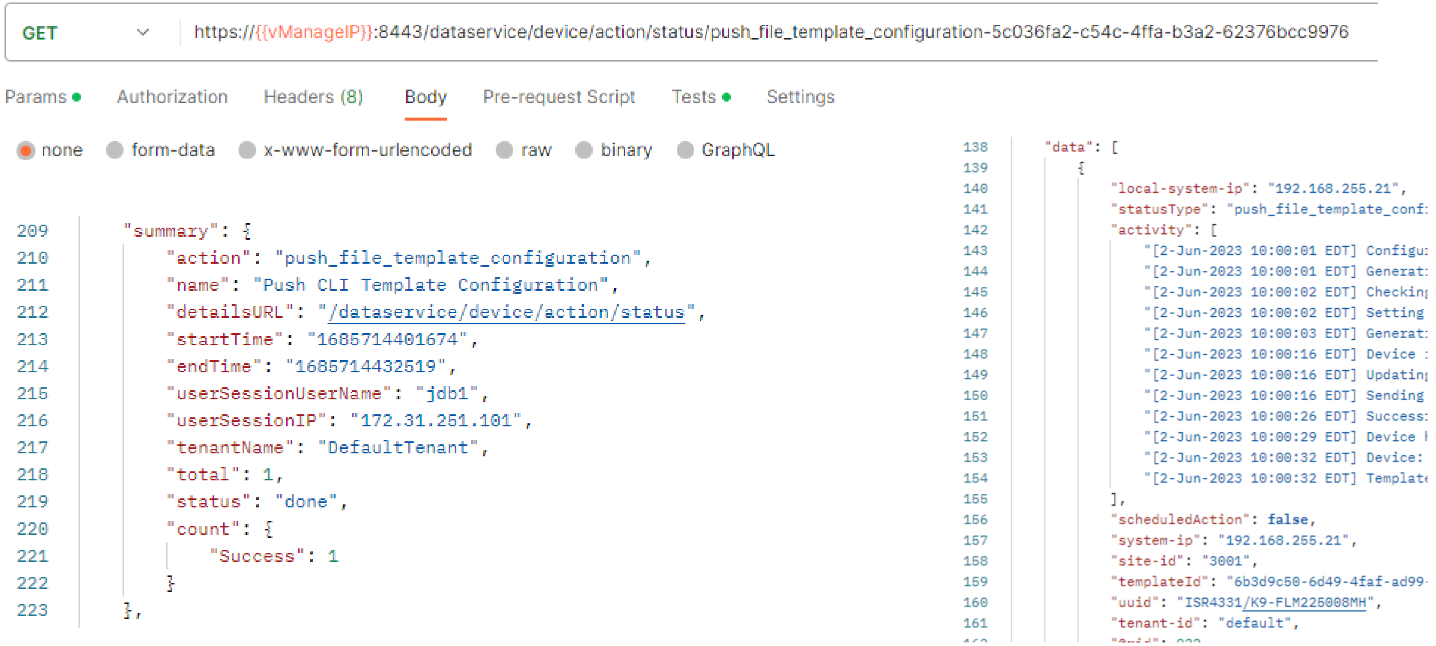
```
{
  "deviceTemplateList":[
  {
    "templateId":"6b3d9c50-6d49-4faf-ad99-aaeeb15d4e55",
    "device":[
    {
      "csv-status":"complete",
      "csv-deviceId":"ISR4331/K9-FLM225008MH",
      "csv-deviceIP":"192.168.255.21",
      "csv-host-name":"SOME_HOSTNAME",
      "User_Password":"I.Love.Cisco",
      "csv-templateId":"6b3d9c50-6d49-4faf-ad99-aaeeb15d4e55",
      "selected":"true"
      "pseudoCommitTimer":11
    }
    ],
    "isEdited":false,
    "isMasterEdited":false,
    "isDraftDisabled":false
  }
  ]
}
```



```
{
    "id": "push_file_template_configuration-5c036fa2-c54c-4ffa-b3a2-62376bcc9976"
}
```

Final Solution – SDWAN

- Monitor Task Status



GET https://{{vManageIP}}:8443/dataservice/device/action/status/push_file_template_configuration-5c036fa2-c54c-4ffa-b3a2-62376bcc9976

Params ● Authorization Headers (8) Body Pre-request Script Tests ● Settings

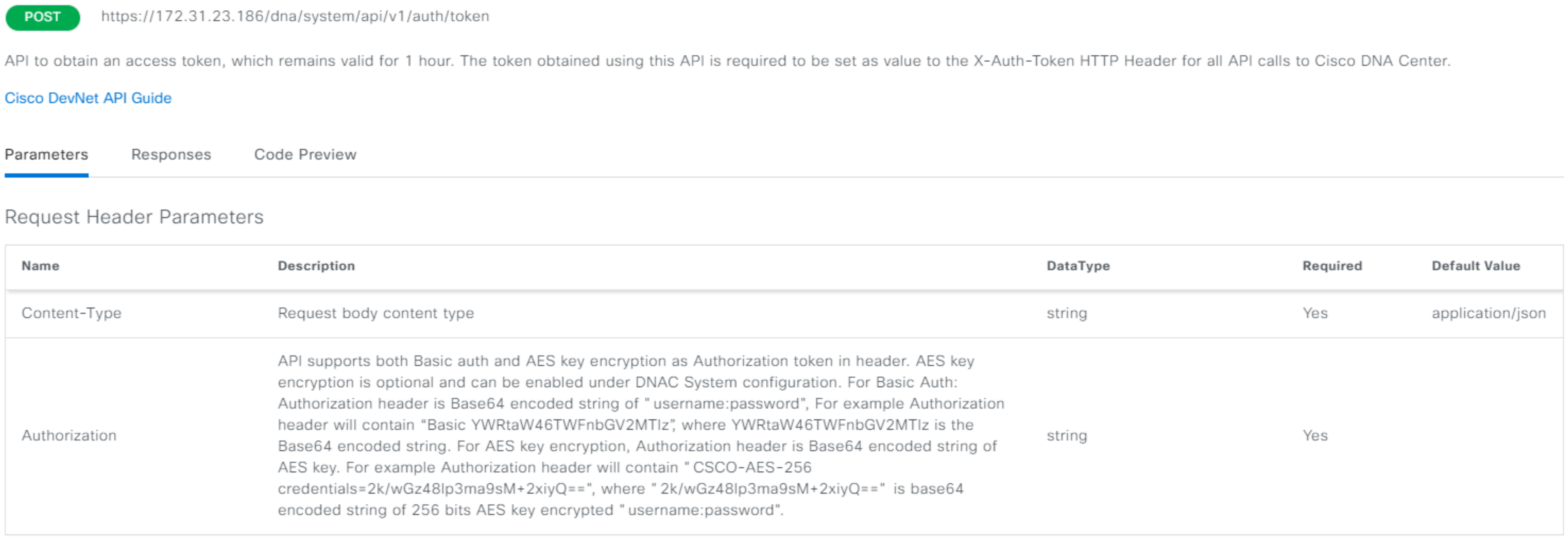
[✓ none] form-data x-www-form-urlencoded raw binary GraphQL

```
    "summary": {
        "action": "push_file_template_configuration",
        "name": "Push CLI Template Configuration",
        "detailsURL": "/dataservice/device/action/status",
        "startTime": "1685714401674",
        "endTime": "1685714432519",
        "userSessionUserName": "jdb1",
        "userSessionIP": "172.31.251.101",
        "tenantName": "DefaultTenant",
        "total": 1,
        "status": "done",
        "count": {
            "Success": 1
        }
    },
```

```
    "data": [
        {
            "local-system-ip": "192.168.255.21",
            "statusType": "push_file_template_conf:
            "activity": [
                "[2-Jun-2023 10:00:01 EDT] Configu:
                "[2-Jun-2023 10:00:01 EDT] Generat:
                "[2-Jun-2023 10:00:02 EDT] Checkin
                "[2-Jun-2023 10:00:02 EDT] Setting
                "[2-Jun-2023 10:00:03 EDT] Generat:
                "[2-Jun-2023 10:00:16 EDT] Device :
                "[2-Jun-2023 10:00:16 EDT] Updatin
                "[2-Jun-2023 10:00:16 EDT] Sending
                "[2-Jun-2023 10:00:26 EDT] Success:
                "[2-Jun-2023 10:00:29 EDT] Device
                "[2-Jun-2023 10:00:32 EDT] Device:
                "[2-Jun-2023 10:00:32 EDT] Templat
            ],
            "scheduledAction": false,
            "system-ip": "192.168.255.21",
            "site-id": "3001",
            "templateId": "6b3d9c50-6d49-4faf-ad99
            "uuid": "ISR4331/K9-FLM225008MH",
            "tenant-id": "default",
```

Final Solution – SDA

- Login – Returns a Token to be used in header as X-Auth-Token



POST https://172.31.23.186/dna/system/api/v1/auth/token

API to obtain an access token, which remains valid for 1 hour. The token obtained using this API is required to be set as value to the X-Auth-Token HTTP Header for all API calls to Cisco DNA Center.

Cisco DevNet API Guide

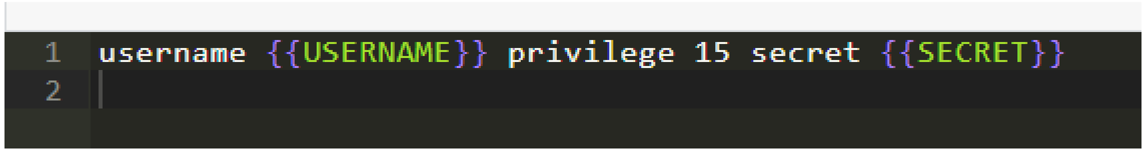
Parameters Responses Code Preview

Request Header Parameters

Name	Description	DataType	Required	Default Value
Content-Type	Request body content type	string	Yes	application/json
Authorization	API supports both Basic auth and AES key encryption as Authorization token in header. AES key encryption is optional and can be enabled under DNAC System configuration. For Basic Auth: Authorization header is Base64 encoded string of " username:password", For example Authorization header will contain "Basic YWRtaW46TWFnbGV2MTIz", where YWRtaW46TWFnbGV2MTIz is the Base64 encoded string. For AES key encryption, Authorization header is Base64 encoded string of AES key. For example Authorization header will contain " CSCO-AES-256 credentials=2k/wGz48lp3ma9sM+2xiyQ==", where " 2k/wGz48lp3ma9sM+2xiyQ==" is base64 encoded string of 256 bits AES key encrypted " username:password".	string	Yes	

Final Solution – SDA

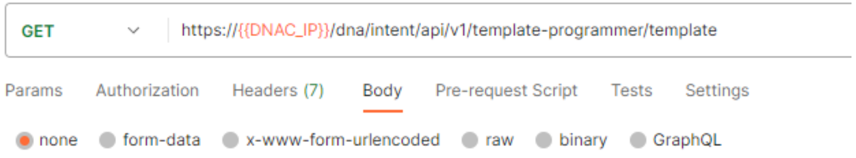
- CLI Template Created that configures the user on the device.

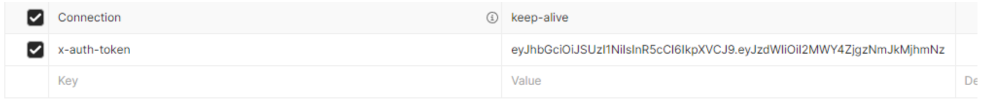


```
username {{USERNAME}} privilege 15 secret {{SECRET}}
```

Final Solution – SDA

- Get template to deploy.


GET https://{{DNAC_IP}}/dna/intent/api/v1/template-programmer/template

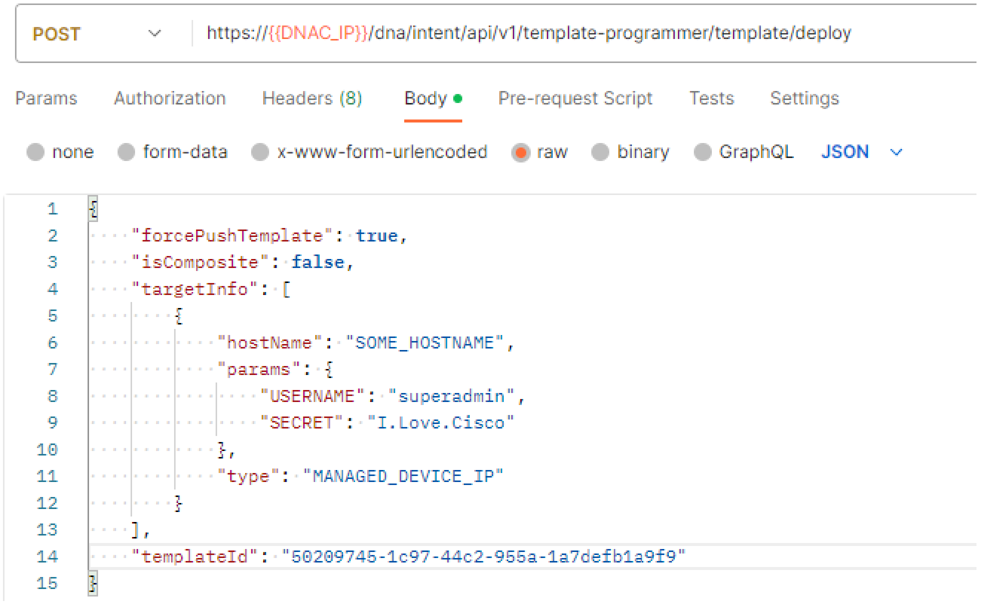


✓	Connection	keep-alive
✓	x-auth-token	eyJhbGciOiJSUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiI2MWY4ZjgzNmJkMjhmNz
	Key	Value

```
[
  {
    "name": “MySuperTemplate",
    "projectId": "0223b225-59b1-430a-95ef-4a548cf8d7aa",
    "templateId": "50209745-1c97-44c2-955a-1a7defb1a9f9",
    "versionsInfo": [
      {
        "id": "8860eed6-c039-4364-9aec-e4b00daaba01",
        "description": "",
        "author": "SYSTEM",
        "version": "1",
        "versionComment": "ImportedTemplate",
        "versionTime": 1646235247766
      }
    ]
  }
]
```

Final Solution – SDA

- Deploy template to target device.



POST https://{{DNAC_IP}}/dna/intent/api/v1/template-programmer/template/deploy

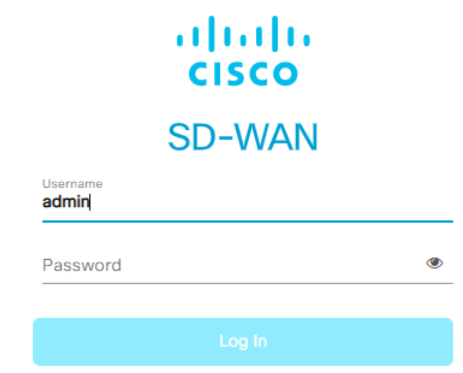
Params | Authorization | Headers (8) | Body ● | Pre-request Script | Tests | Settings

none | form-data | x-www-form-urlencoded | [✓ raw] | binary | GraphQL | JSON

```
{
    "forcePushTemplate": true,
    "isComposite": false,
    "targetInfo": [
        {
            "hostName": "SOME_HOSTNAME",
            "params": {
                "USERNAME": "superadmin",
                "SECRET": "I.Love.Cisco"
            },
            "type": "MANAGED_DEVICE_IP"
        }
    ],
    "templateId": "50209745-1c97-44c2-955a-1a7defb1a9f9"
}
```

Issues and Hiccups

- Static passwords for vManage/DNAC login
  - AAA service account for tool
- What about the ‘controllers’?
  - vManage API for vManage
  - DNAC via SSH for UI user and maglev



CI/CD Template Management

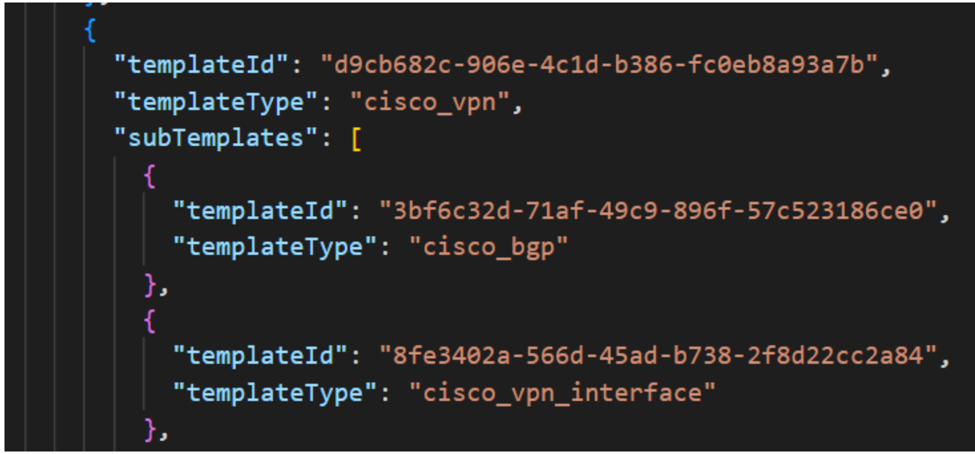
Use Case

- Client Environments
  - SDWAN Dev
  - SDWAN QA
  - SDWAN Prod
- Template Requirements
  - Dev environment is for development and experimentation of new templates.
  - QA environment for testing validation of a version. Must match dev version.
  - QA templates ‘promoted’ to Prod. Must be exact match.

Solution Breakdown – One Piece at a Time

- Template Location
  - Three vManage deployments
  - Naming conventions
  - Device templates composed of feature templates

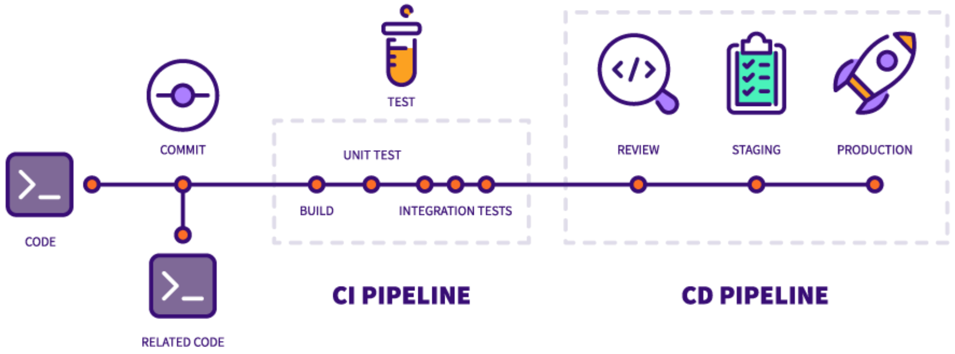
- Device Templates
  - Data structures
  - Feature template IDs are unique



```
    {
      "templateId": "d9cb682c-906e-4c1d-b386-fc0eb8a93a7b",
      "templateType": "cisco_vpn",
      "subTemplates": [
        {
          "templateId": "3bf6c32d-71af-49c9-896f-57c523186ce0",
          "templateType": "cisco_bgp"
        },
        {
          "templateId": "8fe3402a-566d-45ad-b738-2f8d22cc2a84",
          "templateType": "cisco_vpn_interface"
        },
```

Solution Breakdown – One Piece at a Time

- Template Management Tool
  - Promotion of device template
    - Requires exact feature templates
    - Same names and versions
  - Remove templates from Prod
    - If not matched in QA
- Workflow
  - Git repository for Dev
  - GitLab runner deploys to QA
  - Approval deploys to Prod



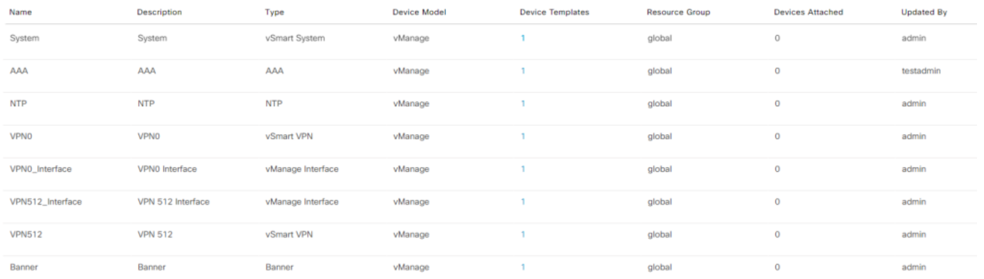
Final Solution

- Workflow initiated by developers.
  - New templates/versions are created in Dev vManage
  - Candidate template commit to Git repository
    - Data structure includes required Dev feature templates
  - GitLab workflow provisions versioned templates on QA vManage
  - QA testing and validation is performed.
    - If template is approved, GitLab continues
    - If template fails, removed from QA and notifications
  - GitLab provisions to Prod exact replica of Dev and QA version

Issues and Hiccups

- Unique feature template IDs on Dev
  - Different from QA
  - Different from Prod
  - Script used to marry IDs to names and update

- Who created the template?
  - All QA and Prod templates only created by the runner's user
  - All others are removed



Name	Description	Type	Device Model	Device Templates	Resource Group	Devices Attached	Updated By
System	System	vSmart System	vManage	1	global	0	admin
AAA	AAA	AAA	vManage	1	global	0	testadmin
NTP	NTP	NTP	vManage	1	global	0	admin
VPN0	VPN0	vSmart VPN	vManage	1	global	0	admin
VPN0_Interface	VPN0 Interface	vManage Interface	vManage	1	global	0	admin
VPN512_Interface	VPN 512 Interface	vManage Interface	vManage	1	global	0	admin
VPN512	VPN 512	vSmart VPN	vManage	1	global	0	admin
Banner	Banner	Banner	vManage	1	global	0	admin

New Client Segmentation

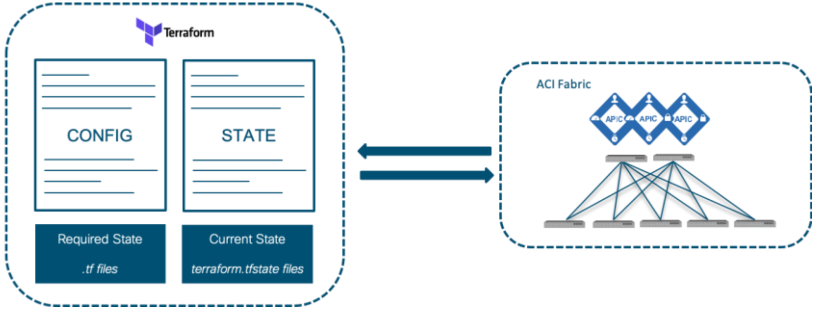
Use Case

- Multiple Domains
  - SDWAN
  - ACI

- Business Requirements
  - Managed call center services.
  - Each client must be segmented from all others.
  - New client onboarding requires configurations on many devices in many locations.
  - ACI provides segmented services. Segmentation is maintained to remote locations through SDWAN.

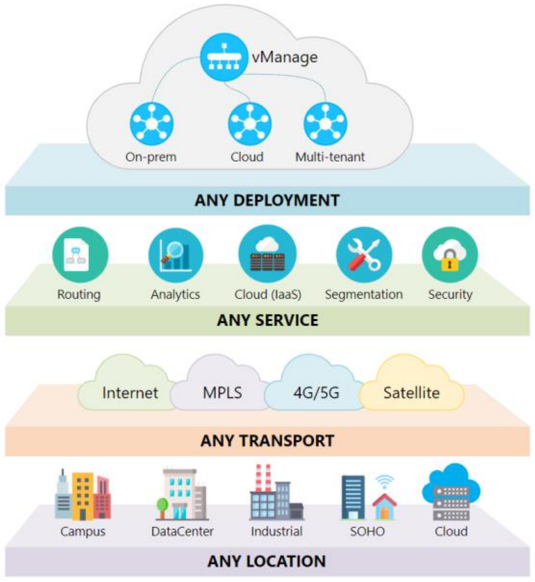
Solution Breakdown – One Piece at a Time

- ACI
  - Create new tenant
    - New IP pools
    - New bridge domains
    - New L3Out handoffs
- Workflow
  - Leverage Terraform
  - New client plan folder from template
  - Naming convention includes client name for uniqueness
  - Variable values different per client, rest of plan is consistent.



Solution Breakdown – One Piece at a Time

- SDWAN
  - Different clients exist at different sites
  - Different sites have
    - Different combinations of clients
    - Different amount of clients
- Workflow
  - vManage API Login/Token
  - Create client Service VPN feature template
  - Identify template for a site
  - Uprev template with additional service VPN
  - Provision with additional client data



Final Solution

- Workflow initiated by python script.
  - ACI client folder created from templates
  - Commit into Git repository
- GitLab Runner performs
  - Terraform init, plan, apply for ACI updates
  - ACI client validation
  - Deploy services to VMware environment
  - Provision DC and Remote cEdge updates
  - End to end network validation

Final Solution - ACI

- Client Folder File Structure
  - client1
    - main.tf Complete Terraform file for one ACI tenant.
    - variables.tf Variables specific to the tenant.
    - sdwan.csv CSV of the DC and remote site IP addressing required.

Final Solution - ACI

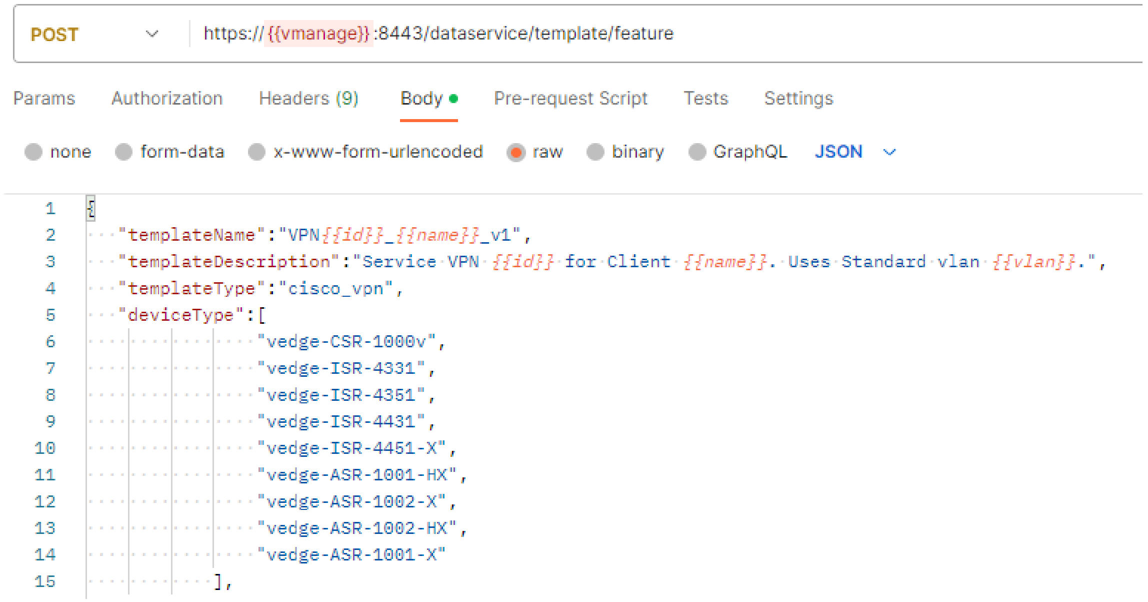
```
# Bridge Domains
resource "aci_bridge_domain" "bds" {
  for_each                         = var.bds
  name                             = each.key
  tenant_dn                        = aci_tenant.tenant1.id
  relation_fv_rs_ctx               = aci_vrf.vrf1.id
  relation_fv_rs_bd_to_out  = [for key, value in var.epgs :
          data.aci_l3_outside.shared_l3_out.id if value.external_access == true &&
          value.bd == each.key]
}

# Bridge Domains Subnets
resource "aci_subnet" "subnets" {
  for_each      = { for key, value in var.epgs : key => value }
  parent_dn    = aci_bridge_domain.bds[var.epgs[each.key].bd].id
  ip                = var.bds[var.epgs[each.key].bd].ip
  scope          = each.value.external_access ? ["public", "shared"] : ["private"]
}
```

```
# Bridge Domains and Subnets
variable “bds" {
  default = {
    "192.168.100.0_24" = {
      ip = "192.168.100.1/24"
    },
    "192.168.101.0_24" = {
      ip = "192.168.101.1/24"
    },
    "192.168.102.0_24" = {
      ip = "192.168.102.1/24"
    }
  }
}
```

https://github.com/datacenter/Terraform-recipes-for-ACI

Final Solution - SDWAN

- Clone base Service VPN Template



POST https://{{vmanage}}:8443/dataservice/template/feature

```
{
  "templateName":"VPN{{id}}_{{name}}_v1",
  "templateDescription":"Service VPN {{id}} for Client {{name}}. Uses Standard vlan {{vlan}}.",
  "templateType":"cisco_vpn",
  "deviceType":[
                "vedge-CSR-1000v",
                "vedge-ISR-4331",
                "vedge-ISR-4351",
                "vedge-ISR-4431",
                "vedge-ISR-4451-X",
                "vedge-ASR-1001-HX",
                "vedge-ASR-1002-X",
                "vedge-ASR-1002-HX",
                "vedge-ASR-1001-X"
            ],
```

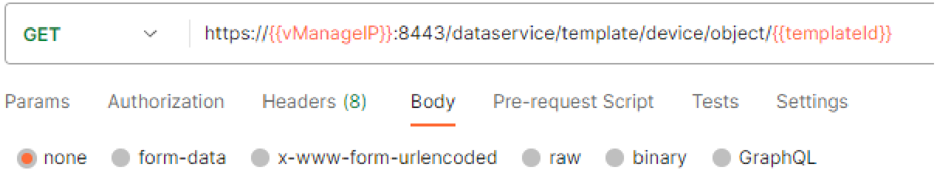
Final Solution - SDWAN

- Repeat these steps at each location requiring the new VPN
  - Information is part of the CSV file for programmatic execution

- Identify the current template attached at the site
- Obtain JSON of template definition
- Update JSON to add the new VPN template (and buildout)
- POST new device template to vManage
- Attach device(s) to the new template

Final Solution - SDWAN

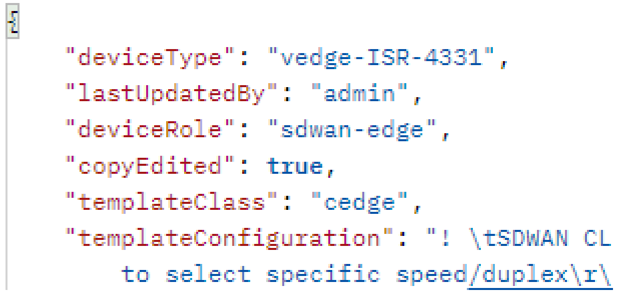
- GET Current Device Template JSON



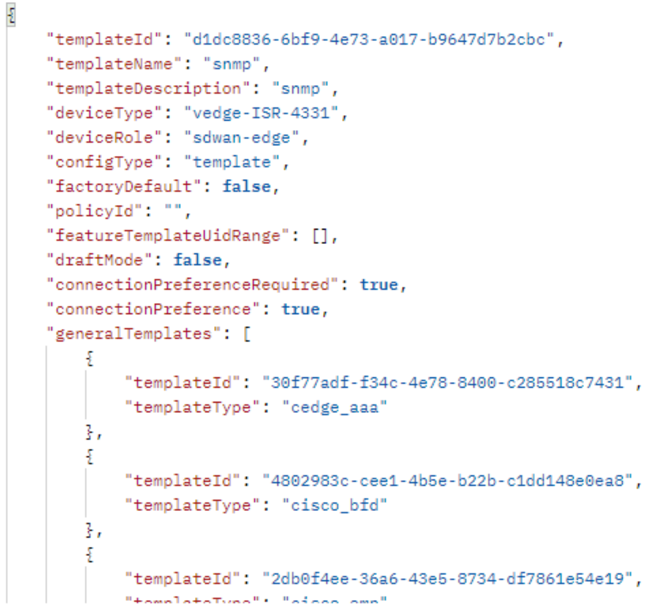
```
GET  https://{{vManageIP}}:8443/dataservice/template/device/object/{{templateId}}
```

Params Authorization Headers (8) Body Pre-request Script Tests Settings

● none form-data x-www-form-urlencoded raw binary GraphQL



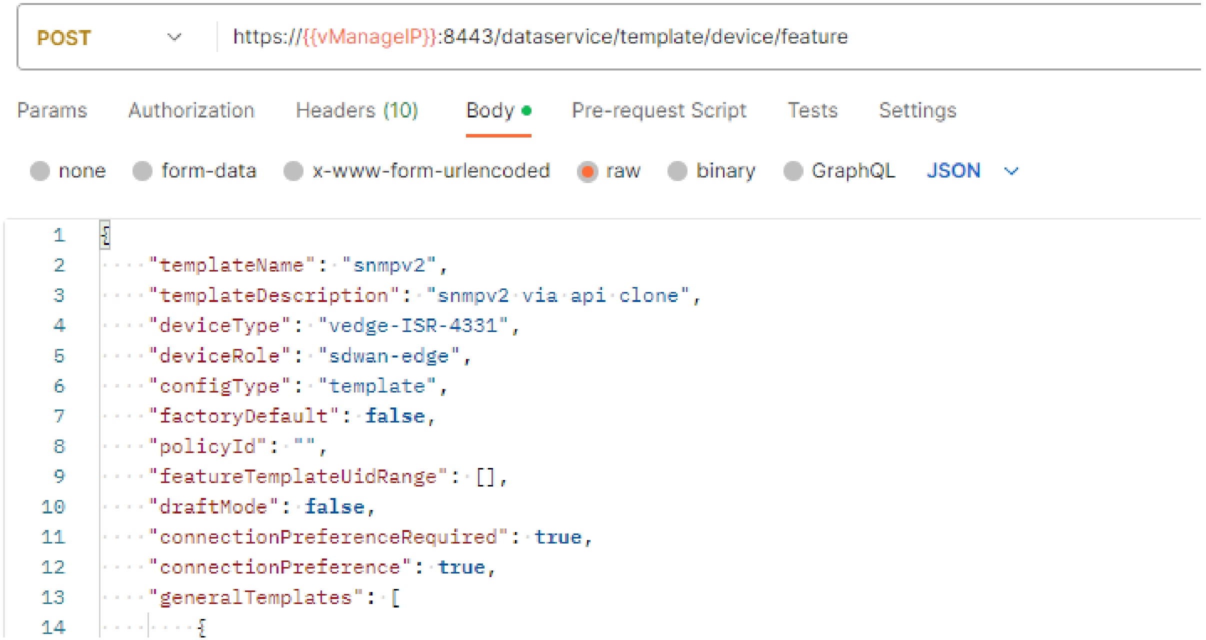
```
{
    "deviceType": "vedge-ISR-4331",
    "lastUpdatedBy": "admin",
    "deviceRole": "sdwan-edge",
    "copyEdited": true,
    "templateClass": "cedge",
    "templateConfiguration": "! \tSDWAN CL
        to select specific speed/duplex\r\
```



```
{
    "templateId": "d1dc8836-6bf9-4e73-a017-b9647d7b2cbc",
    "templateName": "snmp",
    "templateDescription": "snmp",
    "deviceType": "vedge-ISR-4331",
    "deviceRole": "sdwan-edge",
    "configType": "template",
    "factoryDefault": false,
    "policyId": "",
    "featureTemplateUidRange": [],
    "draftMode": false,
    "connectionPreferenceRequired": true,
    "connectionPreference": true,
    "generalTemplates": [
        {
            "templateId": "30f77adf-f34c-4e78-8400-c285518c7431",
            "templateType": "cedge_aaa"
        },
        {
            "templateId": "4802983c-cee1-4b5e-b22b-c1dd148e0ea8",
            "templateType": "cisco_bfd"
        },
        {
            "templateId": "2db0f4ee-36a6-43e5-8734-df7861e54e19",
```

Final Solution - SDWAN

- POST the New, Updated Template Structure Back – Returns a new

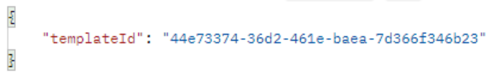


POST https://{{vManageIP}}:8443/dataservice/template/device/feature

Params | Authorization | Headers (10) | Body ● | Pre-request Script | Tests | Settings

none | form-data | x-www-form-urlencoded | raw (selected) | binary | GraphQL | JSON

```
{
    "templateName": "snmpv2",
    "templateDescription": "snmpv2 via api clone",
    "deviceType": "vedge-ISR-4331",
    "deviceRole": "sdwan-edge",
    "configType": "template",
    "factoryDefault": false,
    "policyId": "",
    "featureTemplateUidRange": [],
    "draftMode": false,
    "connectionPreferenceRequired": true,
    "connectionPreference": true,
    "generalTemplates": [
        {
```

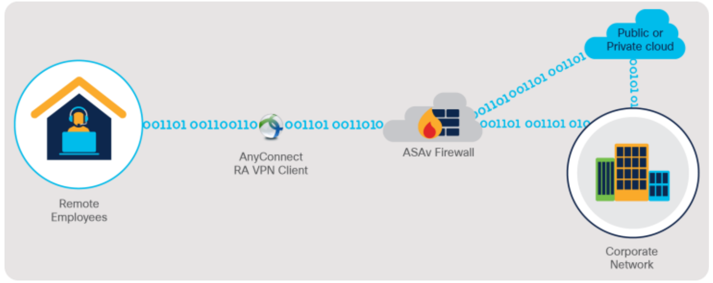


```
{
  "templateId": "44e73374-36d2-461e-baea-7d366f346b23"
}
```

Issues and Hiccups

- Additional domains
  - ASAv deployments and configurations
  - Non-ACI Nexus platforms in DCs

- Client customizations
  - Standardization is your friend
  - Support for client specific configurations on ASAv



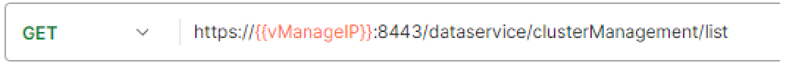
“Everyone knew it was impossible, until a fool who didn’t know came along and did it.”

Albert Einstein

vManage HA/DR

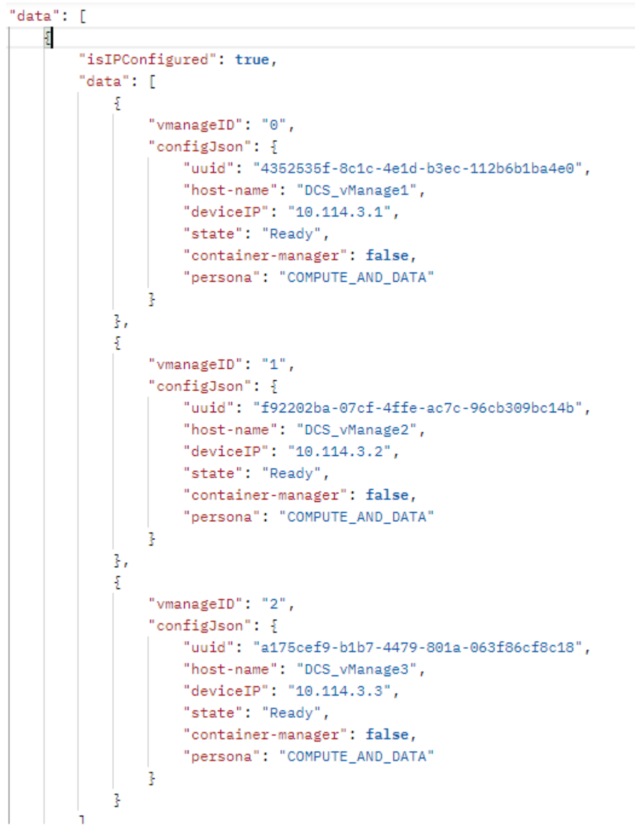
Disclaimer

- Using these vManage APIs incorrectly will break your HA cluster.



GET https://{{vManageIP}}:8443/dataservice/clusterManagement/list

- vmanageID to deviceIP mapping MUST be maintained in all API calls.

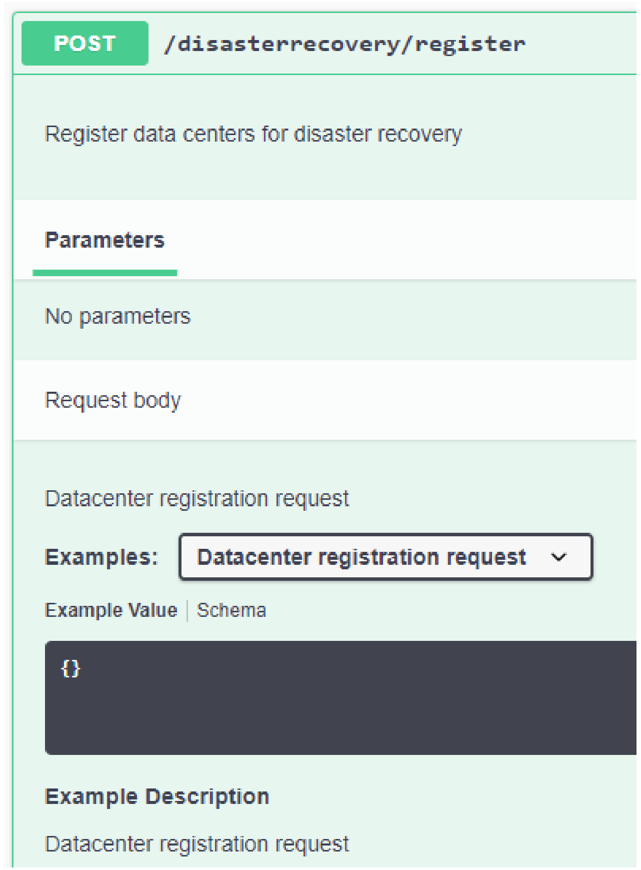


```
"data": [
    {
        "isIPConfigured": true,
        "data": [
            {
                "vmanageID": "0",
                "configJson": {
                    "uuid": "4352535f-8c1c-4e1d-b3ec-112b6b1ba4e0",
                    "host-name": "DCS_vManage1",
                    "deviceIP": "10.114.3.1",
                    "state": "Ready",
                    "container-manager": false,
                    "persona": "COMPUTE_AND_DATA"
                }
            },
            {
                "vmanageID": "1",
                "configJson": {
                    "uuid": "f92202ba-07cf-4ffe-ac7c-96cb309bc14b",
                    "host-name": "DCS_vManage2",
                    "deviceIP": "10.114.3.2",
                    "state": "Ready",
                    "container-manager": false,
                    "persona": "COMPUTE_AND_DATA"
                }
            },
            {
                "vmanageID": "2",
                "configJson": {
                    "uuid": "a175cef9-b1b7-4479-801a-063f86cf8c18",
                    "host-name": "DCS_vManage3",
                    "deviceIP": "10.114.3.3",
                    "state": "Ready",
                    "container-manager": false,
                    "persona": "COMPUTE_AND_DATA"
                }
            }
        ]
```

Use Case

- HA and DR cluster passwords must be updated.
  - Exist in ISE/TACACS server.
  - Allows full netadmin role.

- Note: Documentation of the payloads of HA/DR API calls is incomplete.



POST /disasterrecovery/register

Register data centers for disaster recovery

Parameters

No parameters

Request body

Datacenter registration request

Examples: Datacenter registration request

Example Value | Schema

```
{}
```

Example Description

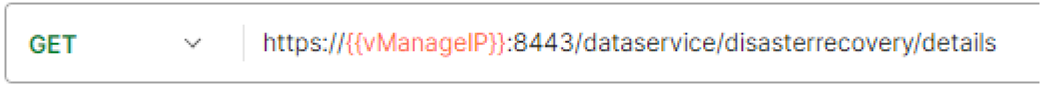
Datacenter registration request

Solution Breakdown – One Piece at a Time

- Disable Disaster Recovery
  - Pause DR Replication
  - Deregister DR Devices
- Edit HA Cluster Configuration
- Enable Disaster Recovery

Final Solution

- Track DR Replication Status



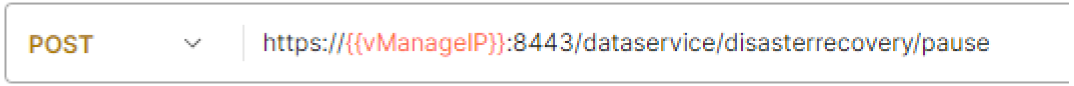
```
GET    https://{{vManageIP}}:8443/dataservice/disasterrecovery/details
```

```
{
  "replicationDetails": [
    {
      "lastReplicated": 1682007190157,
      "exportDuration": "45 secs",
      "exportSize": "7.189 MB",
      "replicationStatus": "Success"
    }
  ]
}
```

```
{
    "replicationDetails": []
}
```

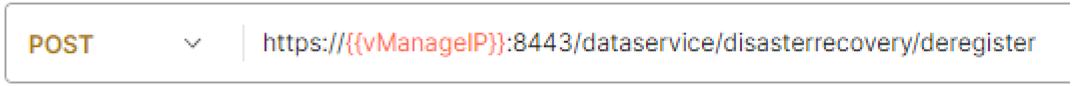
Final Solution

- Pause Disaster Recovery Replication



```
POST   https://{{vManageIP}}:8443/dataservice/disasterrecovery/pause
```

- Deregister Disaster Recovery



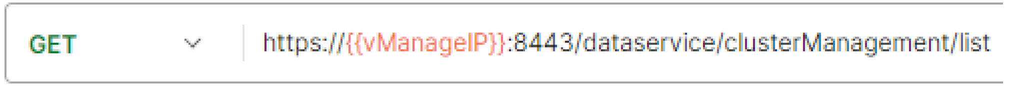
```
POST   https://{{vManageIP}}:8443/dataservice/disasterrecovery/deregister
```

Response: {"id":"15fcf8fe-e3d1-4d73-8ff7-92906691b183"}

Track the status of the Task ID. It will take 10 or more minutes to complete.

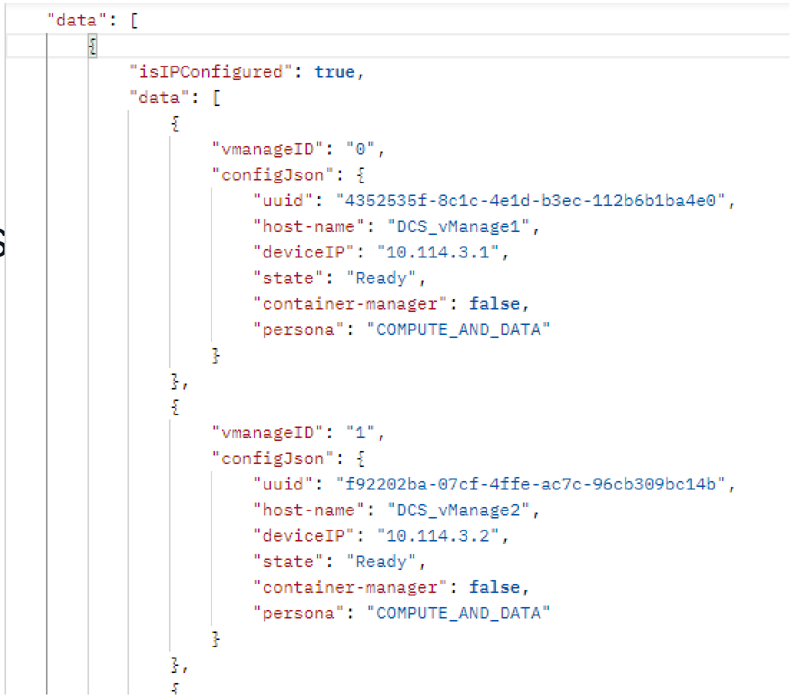
Final Solution

- Get the HA cluster list.



GET https://{{vManageIP}}:8443/dataservice/clusterManagement/list

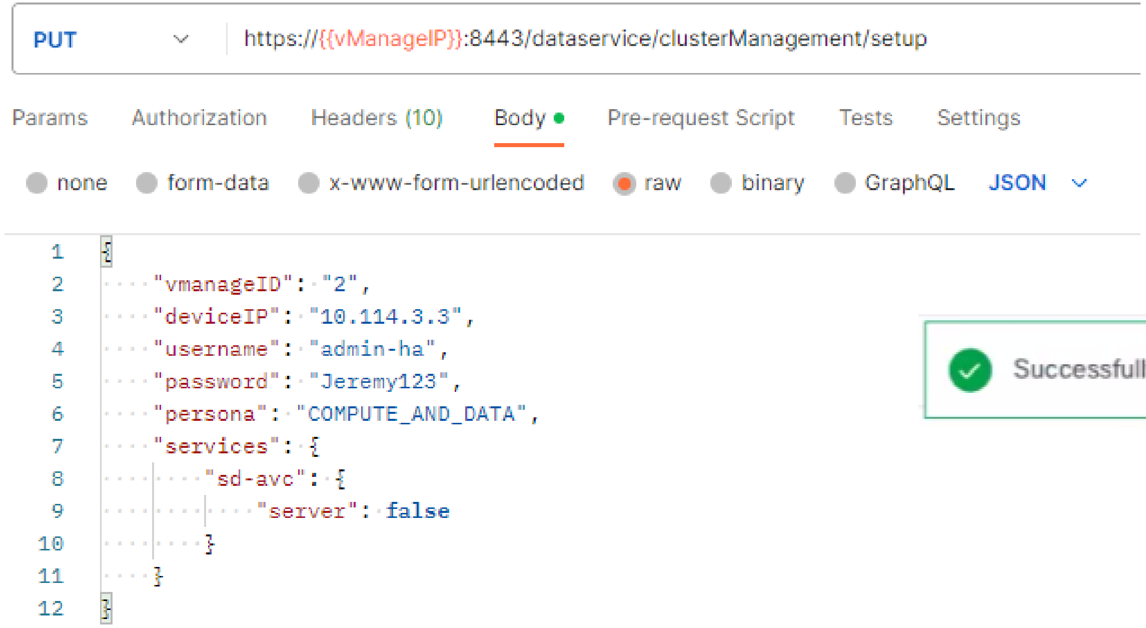
- Repeat these steps for both HA clusters



```
"data": [
    {
        "isIPConfigured": true,
        "data": [
            {
                "vmanageID": "0",
                "configJson": {
                    "uuid": "4352535f-8c1c-4e1d-b3ec-112b6b1ba4e0",
                    "host-name": "DCS_vManage1",
                    "deviceIP": "10.114.3.1",
                    "state": "Ready",
                    "container-manager": false,
                    "persona": "COMPUTE_AND_DATA"
                }
            },
            {
                "vmanageID": "1",
                "configJson": {
                    "uuid": "f92202ba-07cf-4ffe-ac7c-96cb309bc14b",
                    "host-name": "DCS_vManage2",
                    "deviceIP": "10.114.3.2",
                    "state": "Ready",
                    "container-manager": false,
                    "persona": "COMPUTE_AND_DATA"
                }
            },
            {
```

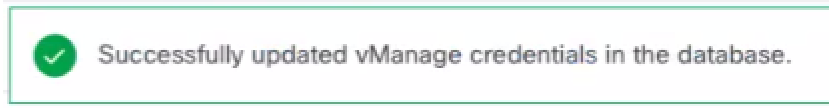
Final Solution

- Change the HA cluster password. (not a list, called for each



PUT https://{{vManageIP}}:8443/dataservice/clusterManagement/setup

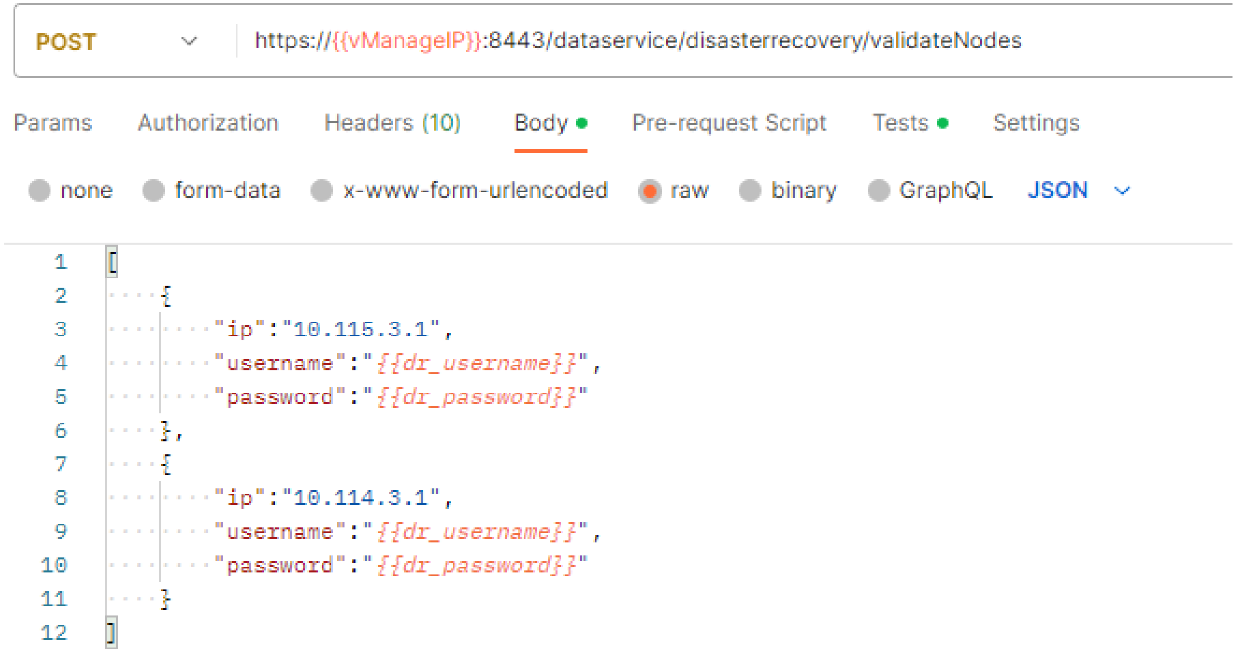
```
{
    "vmanageID": "2",
    "deviceIP": "10.114.3.3",
    "username": "admin-ha",
    "password": "Jeremy123",
    "persona": "COMPUTE_AND_DATA",
    "services": {
        "sd-avc": {
            "server": false
        }
    }
}
```



Successfully updated vManage credentials in the database.

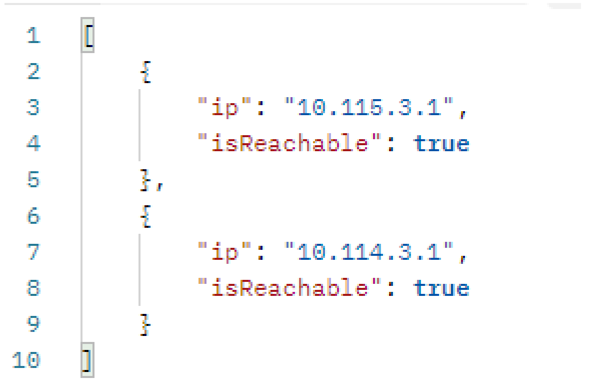
Final Solution

- Validate the DR cluster members with new credentials.



POST https://{{vManageIP}}:8443/dataservice/disasterrecovery/validateNodes

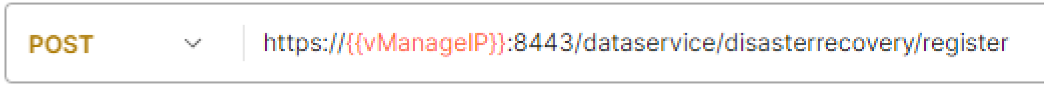
```
[
    {
        "ip":"10.115.3.1",
        "username":"{{dr_username}}",
        "password":"{{dr_password}}"
    },
    {
        "ip":"10.114.3.1",
        "username":"{{dr_username}}",
        "password":"{{dr_password}}"
    }
]
```



```
[
    {
        "ip": "10.115.3.1",
        "isReachable": true
    },
    {
        "ip": "10.114.3.1",
        "isReachable": true
    }
]
```

Final Solution

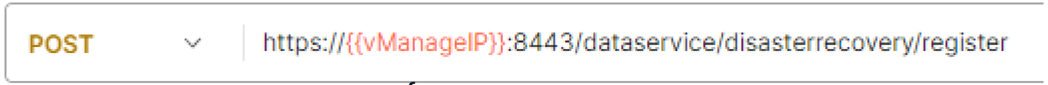
- Recreate the DR cluster with new credentials.



```
POST    https://{{vManageIP}}:8443/dataservice/disasterrecovery/register
```

- Body data structure on following pages.

Final Solution


POST https://{{vManageIP}}:8443/dataservice/disasterrecovery/register

```
{
    "dataCenters": [
        {
            "name":"DC1",
            "nmsPersonality":"nms_user",
            "dcPersonality":"primary",
            "mgmtIPAddress":"10.114.3.1",
            "username":"{{dr_username}}",
            "password":"{{dr_password}}"
        },
        {
            "name":"DC2",
            "nmsPersonality":"nms_user",
            "dcPersonality":"secondary",
            "mgmtIPAddress":"10.115.3.1",
            "username":"{{dr_username}}",
            "password":"{{dr_password}}"
        }
    ],
```

Final Solution

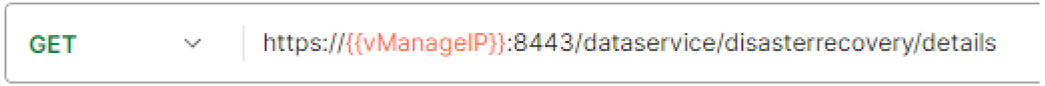
```
"disasterRecoverySettings":
      {
         "delayThreshold":"2",
         "startTime":"12:00am",
         "interval":"30"
      },
   "vbonds":[
      {
         "name":"",
         "ip":"10.114.4.1",
         "username":"{{adminUsername}}",
         "password":"{{adminPassword}}"
      },
      {
         "name":"",
         "ip":"10.115.4.3",
         "username":"{{adminUsername}}",
         "password":"{{adminPassword}}"
      }
   ]
}
```

Final Solution

- Response returns the Task ID.

- Monitor the Task ID. Completion will take 10 minutes.

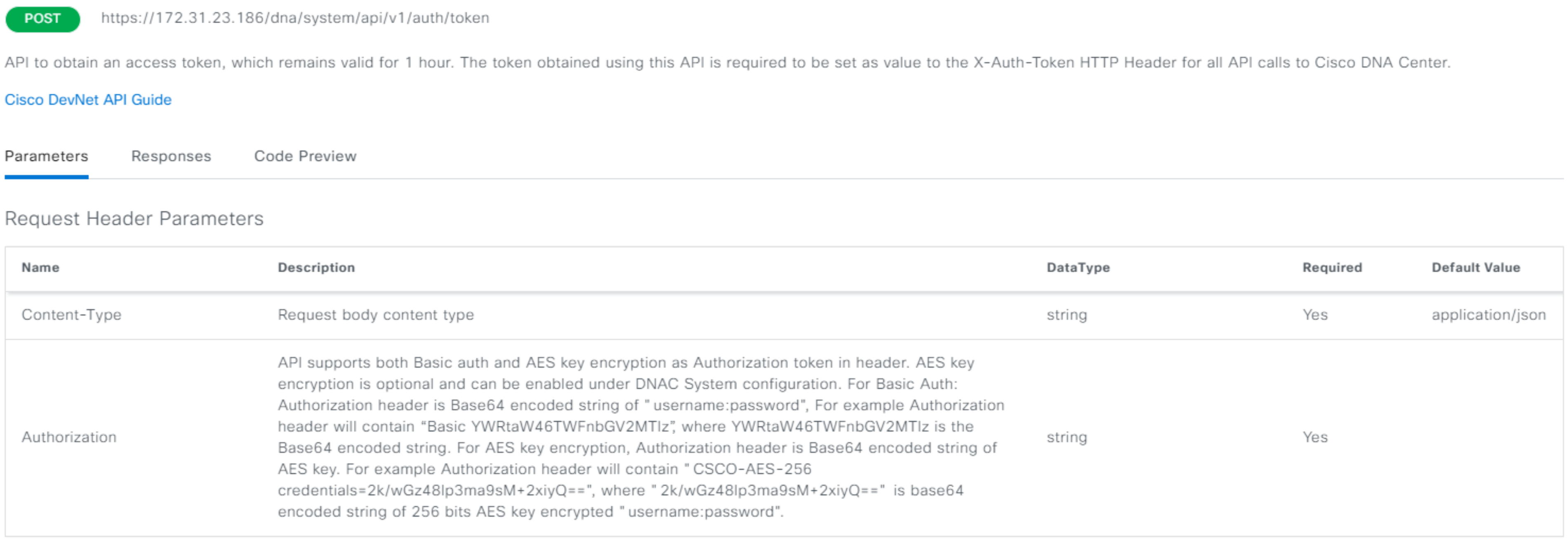
- Repeat the DR Replication Status API.



```
GET  https://{{vManageIP}}:8443/dataservice/disasterrecovery/details
```

Unusual APIs

DNAC API Authentication



POST https://172.31.23.186/dna/system/api/v1/auth/token

API to obtain an access token, which remains valid for 1 hour. The token obtained using this API is required to be set as value to the X-Auth-Token HTTP Header for all API calls to Cisco DNA Center.

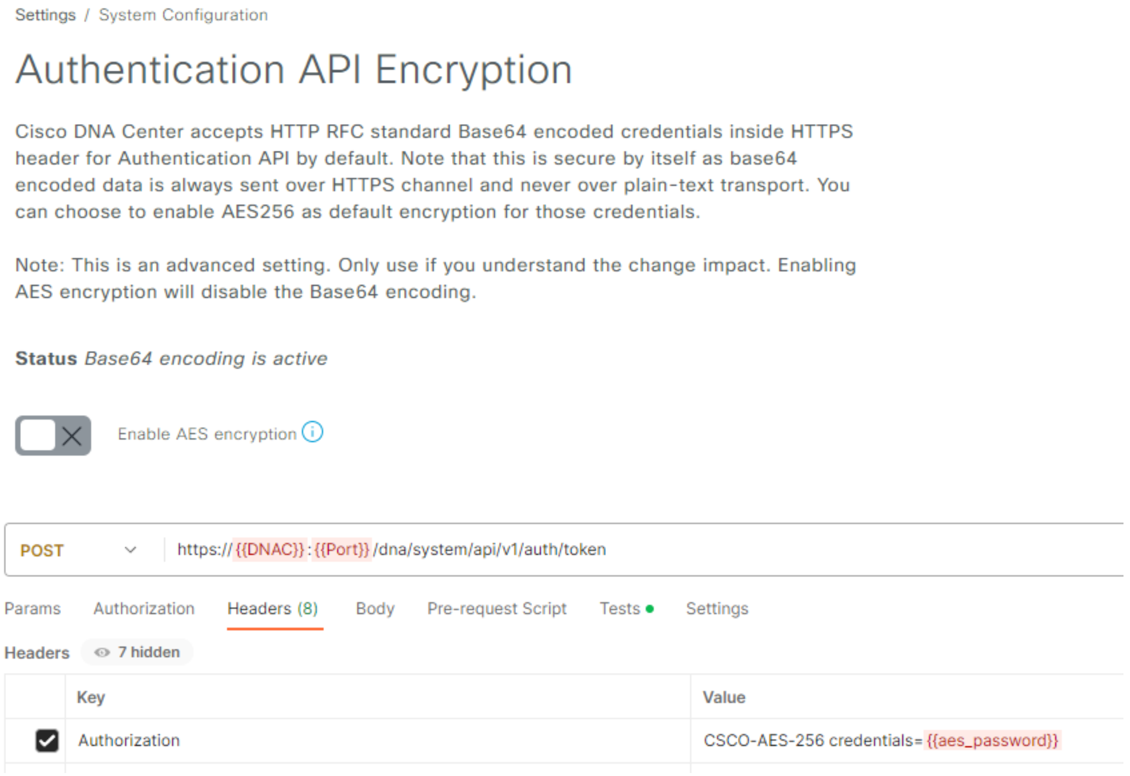
Cisco DevNet API Guide

Parameters Responses Code Preview

Request Header Parameters

Name	Description	DataType	Required	Default Value
Content-Type	Request body content type	string	Yes	application/json
Authorization	API supports both Basic auth and AES key encryption as Authorization token in header. AES key encryption is optional and can be enabled under DNAC System configuration. For Basic Auth: Authorization header is Base64 encoded string of " username:password", For example Authorization header will contain "Basic YWRtaW46TWFnbGV2MTIz", where YWRtaW46TWFnbGV2MTIz is the Base64 encoded string. For AES key encryption, Authorization header is Base64 encoded string of AES key. For example Authorization header will contain " CSCO-AES-256 credentials=2k/wGz48lp3ma9sM+2xiyQ==", where " 2k/wGz48lp3ma9sM+2xiyQ==" is base64 encoded string of 256 bits AES key encrypted " username:password".	string	Yes	

DNAC API Authentication



Settings / System Configuration

Authentication API Encryption

Cisco DNA Center accepts HTTP RFC standard Base64 encoded credentials inside HTTPS header for Authentication API by default. Note that this is secure by itself as base64 encoded data is always sent over HTTPS channel and never over plain-text transport. You can choose to enable AES256 as default encryption for those credentials.

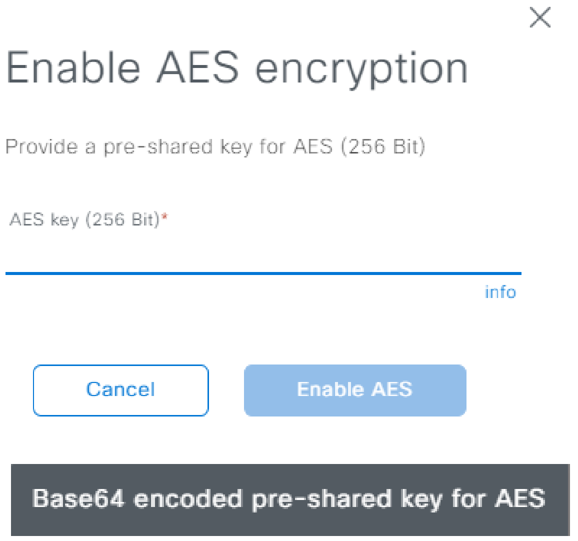
Note: This is an advanced setting. Only use if you understand the change impact. Enabling AES encryption will disable the Base64 encoding.

Status *Base64 encoding is active*

Enable AES encryption

POST https://{{DNAC}}:{{Port}}/dna/system/api/v1/auth/token

	Key	Value
✓	Authorization	CSCO-AES-256 credentials={{aes_password}}



Enable AES encryption

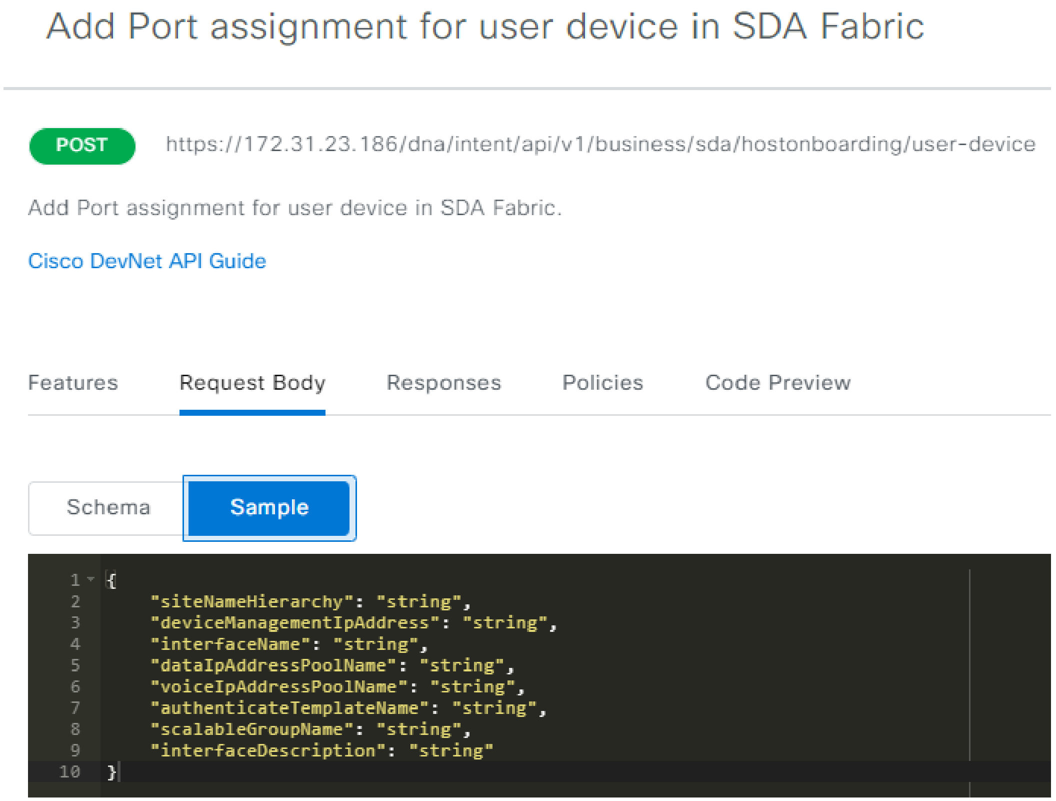
Provide a pre-shared key for AES (256 Bit)

AES key (256 Bit)*

Base64 encoded pre-shared key for AES

SDA Fabric Edge Static Port Assignment

- Only configures one interface.
- Each call requires 40-60 seconds for DNAC to process. (per switch)
- Interface list to be supported.



https://172.31.23.186/dna/intent/api/v1/business/sda/hostonboarding/user-device

Add Port assignment for user device in SDA Fabric.

```
{
    "siteNameHierarchy": "string",
    "deviceManagementIpAddress": "string",
    "interfaceName": "string",
    "dataIpAddressPoolName": "string",
    "voiceIpAddressPoolName": "string",
    "authenticateTemplateName": "string",
    "scalableGroupName": "string",
    "interfaceDescription": "string"
}
```

Q&A

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 socks** (while supplies last)!

Attendees will also earn 100 points in the **Cisco Live Challenge** for every survey completed.

These points help you get on the leaderboard and increase your chances of winning daily and grand prizes

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

The bridge to possible

Thank you

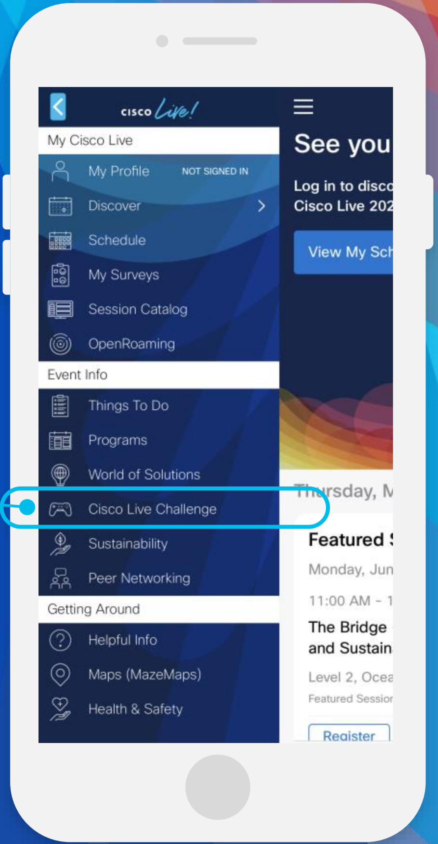
Cisco Live **Challenge**

Gamify your Cisco Live experience!

Get points for attending this session!

How:

1. Open the Cisco Events App.
2. Click on 'Cisco Live Challenge' in the side menu.
3. Click on View Your Badges at the top.
4. Click the + at the bottom of the screen and scan the QR code:



cisco Live!

Let's go

#CiscoLive