cisco

You make **possible**

Automating ACI with Ansible

Thomas Renzy – Technical Leader CX
@ThomasRenzy

BRKACI-1619

Barcelona | January 27-31, 2020

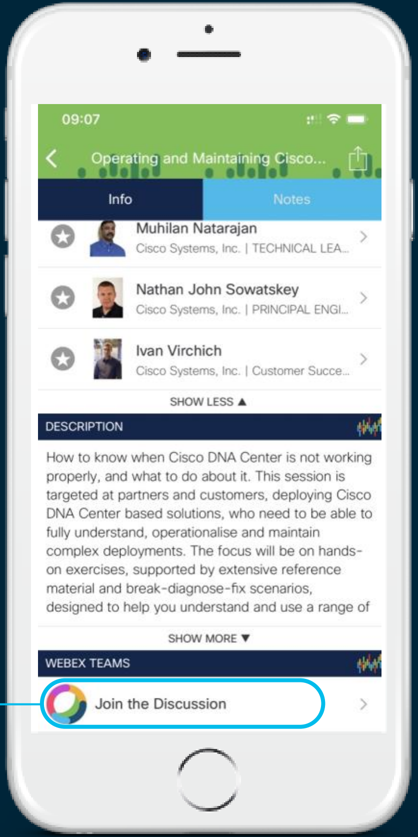
Cisco Webex Teams

Questions?

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

How

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



Muhilan Natarajan
Cisco Systems, Inc. | TECHNICAL LEA...

Nathan John Sowatskey
Cisco Systems, Inc. | PRINCIPAL ENGI...

Ivan Virchich
Cisco Systems, Inc. | Customer Succe...

How to know when Cisco DNA Center is not working properly, and what to do about it. This session is targeted at partners and customers, deploying Cisco DNA Center based solutions, who need to be able to fully understand, operationalise and maintain complex deployments. The focus will be on hands-on exercises, supported by extensive reference material and break-diagnose-fix scenarios, designed to help you understand and use a range of

Agenda

- Introduction – What is Automation?
- Overview of Ansible
- Automating ACI with Playbooks
- Signature Based Authentication
- A Three Tier Application
- ACI REST module

What is Automation?

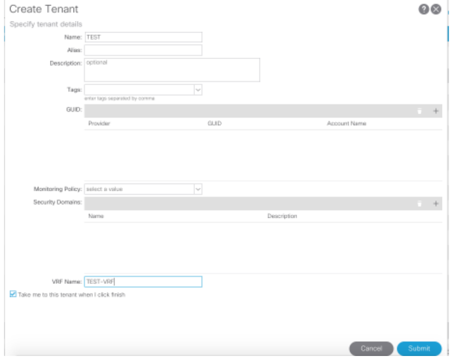
(Why ACI Automation?)

What is automation?

- Exists to make repeatable things easier
- Uses tools to create process and instructions
  - Replaces manual work
- Benefits – speed, efficiency, $$$

Why automation with ACI?

- GUI Point-and-click for configuration – one at a time
- Repetitive Tasks
- Does not scale when deploying large configurations
- ACI APIC provides robust API
    - Automation tools can leverage



Create Tenant

Specify tenant details

Name: TEST

Alias:

Description: optional

Tags: enter tags separated by comma

GUID:

Provider	GUID	Account Name

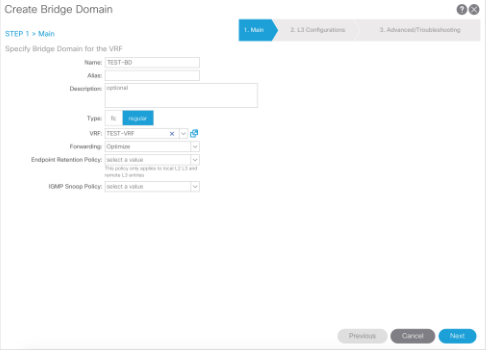
Monitoring Policy: select a value

Security Domains:

Name	Description

VRF Name: TEST-VRF

☑ Take me to this tenant when I click finish



Create Bridge Domain

STEP 1 > Main

Specify Bridge Domain for the VRF

Name: TEST-BD

Alias:

Description: optional

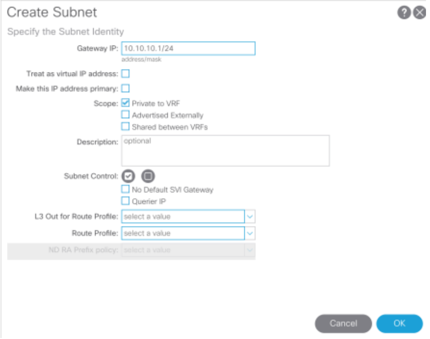
Type: fc [✓ regular]

VRF: TEST-VRF

Forwarding: Optimize

Endpoint Retention Policy: select a value
This policy only applies to local L2 L3 and remote L3 entries

IGMP Snoop Policy: select a value



Create Subnet

Specify the Subnet Identity

Gateway IP: 10.10.10.1/24
address/mask

Treat as virtual IP address: □

Make this IP address primary: □

Scope: ☑ Private to VRF
□ Advertised Externally
□ Shared between VRFs

Description: optional

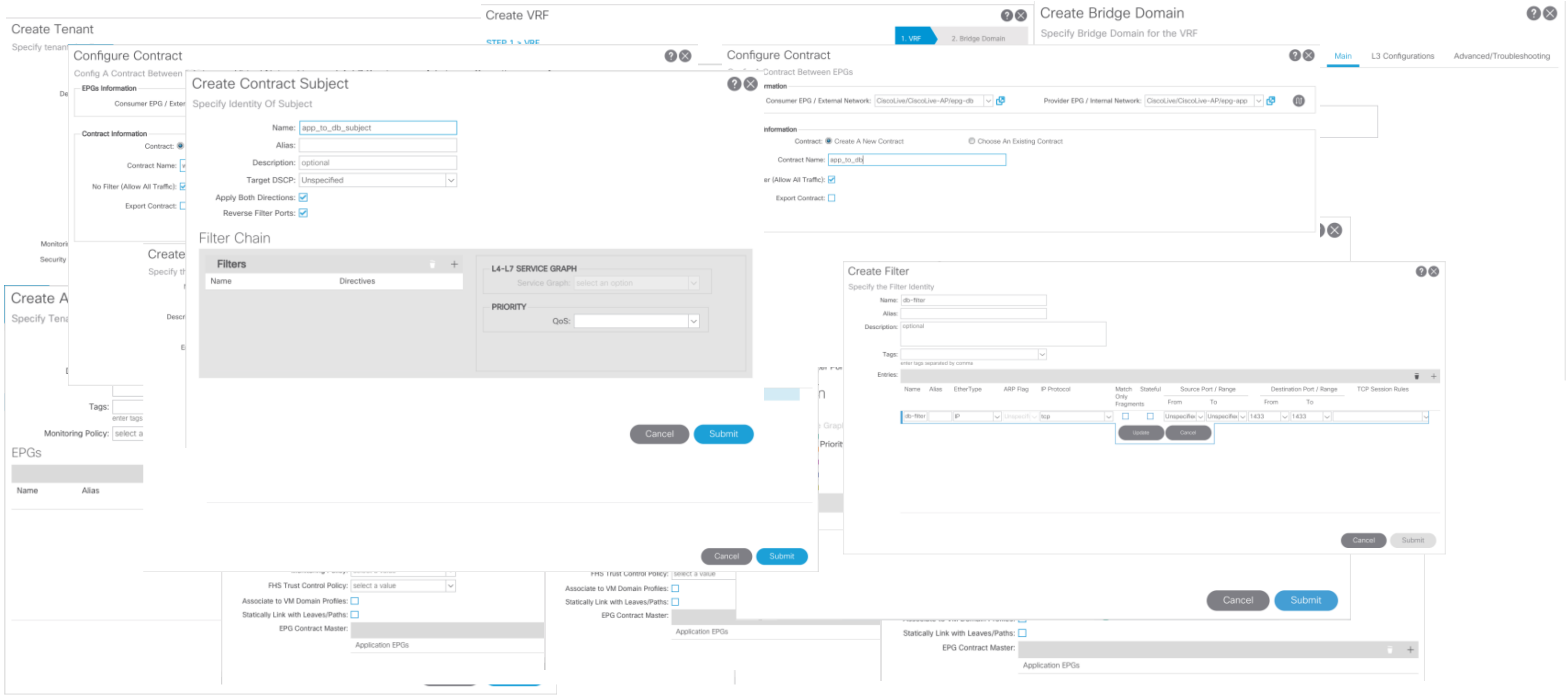
Subnet Control:
□ No Default SVI Gateway
□ Querier IP

L3 Out for Route Profile: select a value

Route Profile: select a value

ND RA Prefix policy: select a value

Deploy Three Tier Application – APIC GUI



Create Tenant

Specify tenant

Create VRF

STEP 1 > VRF

1. VRF — 2. Bridge Domain

Create Bridge Domain

Specify Bridge Domain for the VRF

Configure Contract

Config A Contract Between EPGs

EPGs Information

Consumer EPG / External Network: CiscoLive/CiscoLive-AP/epg-db

Provider EPG / Internal Network: CiscoLive/CiscoLive-AP/epg-app

Contract Information

Contract: ● Create A New Contract ○ Choose An Existing Contract

Contract Name: app_to_db

No Filter (Allow All Traffic): ☑

Export Contract: □

Create Contract Subject

Specify Identity Of Subject

Name: app_to_db_subject

Alias:

Description: optional

Target DSCP: Unspecified

Apply Both Directions: ☑

Reverse Filter Ports: ☑

Filter Chain

Filters	
Name	Directives

L4-L7 SERVICE GRAPH

Service Graph: select an option

PRIORITY

QoS:

Create Filter

Specify the Filter Identity

Name: db-filter

Alias:

Description: optional

Tags:

enter tags separated by comma

Entries:

Name	Alias	EtherType	ARP Flag	IP Protocol	Match Only Fragments	Stateful	Source Port / Range From	Source Port / Range To	Destination Port / Range From	Destination Port / Range To	TCP Session Rules
db-filter		IP	Unspecifi	tcp	□	□	Unspecified	Unspecified	1433	1433	

Create Application EPG

Specify Tenant

Tags: enter tags

Monitoring Policy: select a

EPGs

Name	Alias

FHS Trust Control Policy: select a value

Associate to VM Domain Profiles: □

Statically Link with Leaves/Paths: □

EPG Contract Master:

Application EPGs

Overview of Ansible

Inventory, Playbooks, and Modules

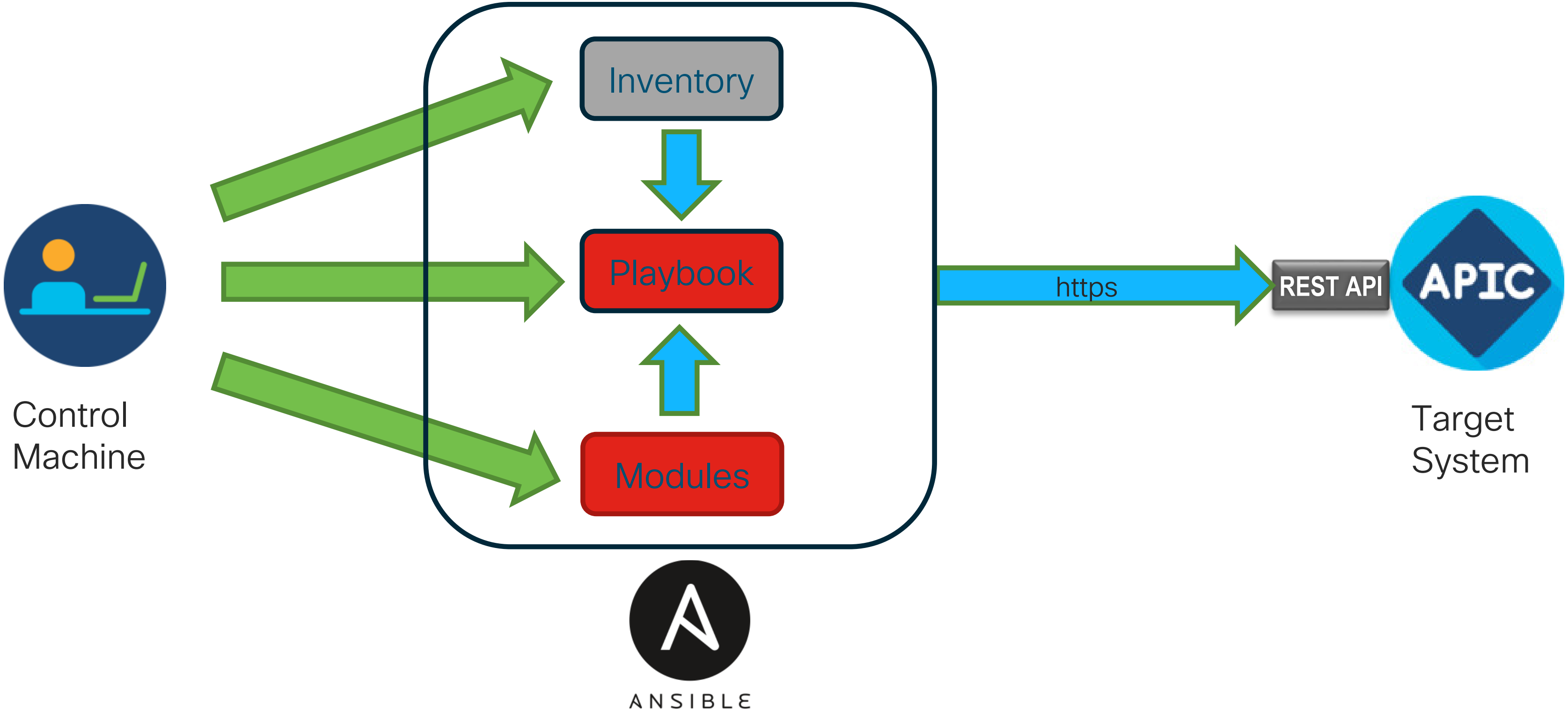
What is Ansible?

- Open Source
- Automation, Configuration & Orchestration
- Version 2.9
  - 2.8 & 2.7 also available
  - ACI support - 2.4
- Supported on UNIX/Linux
  - Windows Subsystem for Linux
- Can manage different systems
  - ACI, IOS, NX-OS, IOS-XR

What is Ansible?

- Agentless
  - Push Model
- Idempotent
- YAML based
  - Easily Readable
- APIC REST API interface
  - Same as GUI
- Requires no programming skills
  - Python is helpful – not required

What makes up Ansible?



Example ACI Ansible Inventory

YAML inventory file

```
all:
  hosts:
    Fabric1:
      inventory_hostname: 10.50.62.1
      username: admin
      password: cisco
    Fabric2:
      inventory_hostname: 10.51.92.1
      username: admin
      password: cisco
```

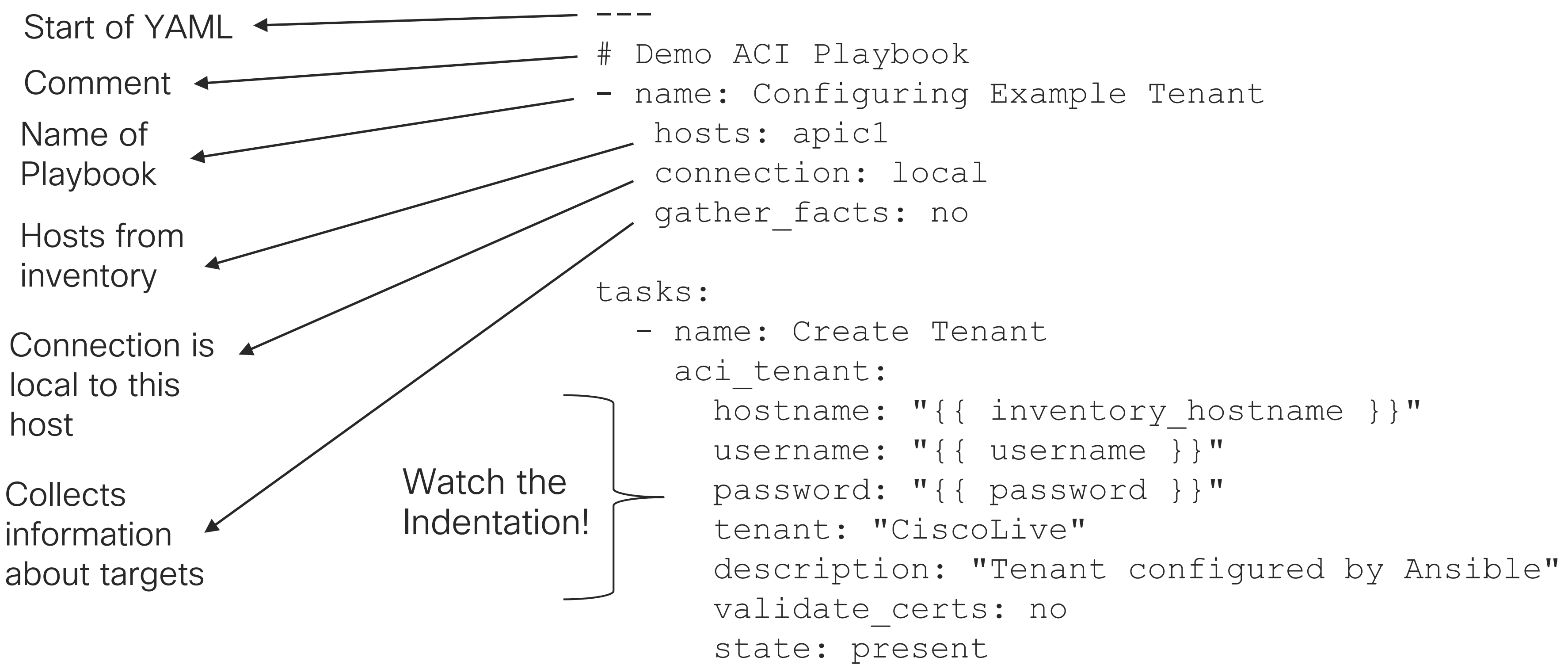
INI inventory file

```
[Fabric1]
la-apic1 username=admin password=cisco

[Fabric2]
ny-apic1 username=admin password=cisco
```

Ansible Playbooks Breakdown

- Contains a list of plays
  - Series of tasks to be performed on target systems
- Tasks are executed in order
- Built on YAML
- Proper Indentation is required
- “---” exists at the start of every playbook
- Apply specific roles to targets

Ansible Playbook breakdown



Start of YAML

Comment

Name of Playbook

Hosts from inventory

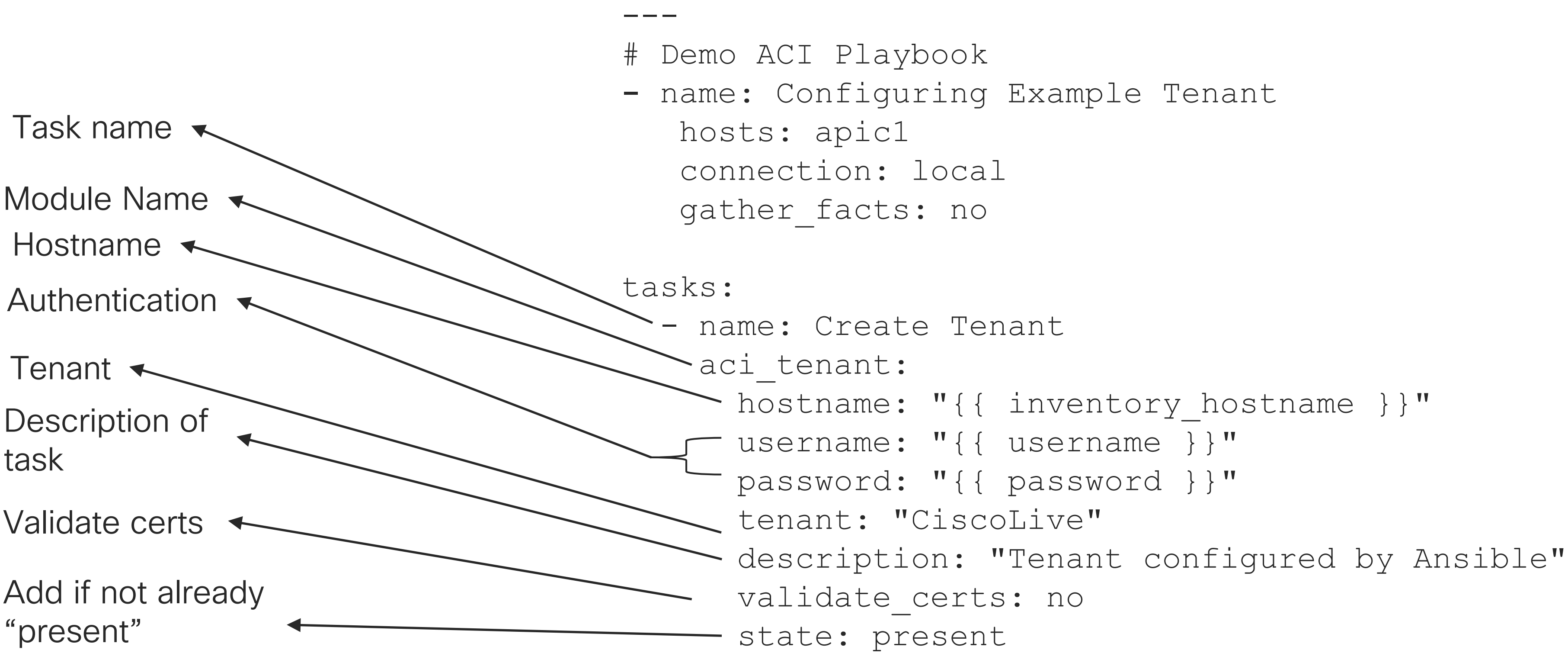
Connection is local to this host

Collects information about targets

Watch the Indentation!

```
---
# Demo ACI Playbook
- name: Configuring Example Tenant
  hosts: apic1
  connection: local
  gather_facts: no

tasks:
 - name: Create Tenant
  aci_tenant:
   hostname: "{{ inventory_hostname }}"
   username: "{{ username }}"
   password: "{{ password }}"
   tenant: "CiscoLive"
   description: "Tenant configured by Ansible"
   validate_certs: no
   state: present
```

Ansible Playbook breakdown

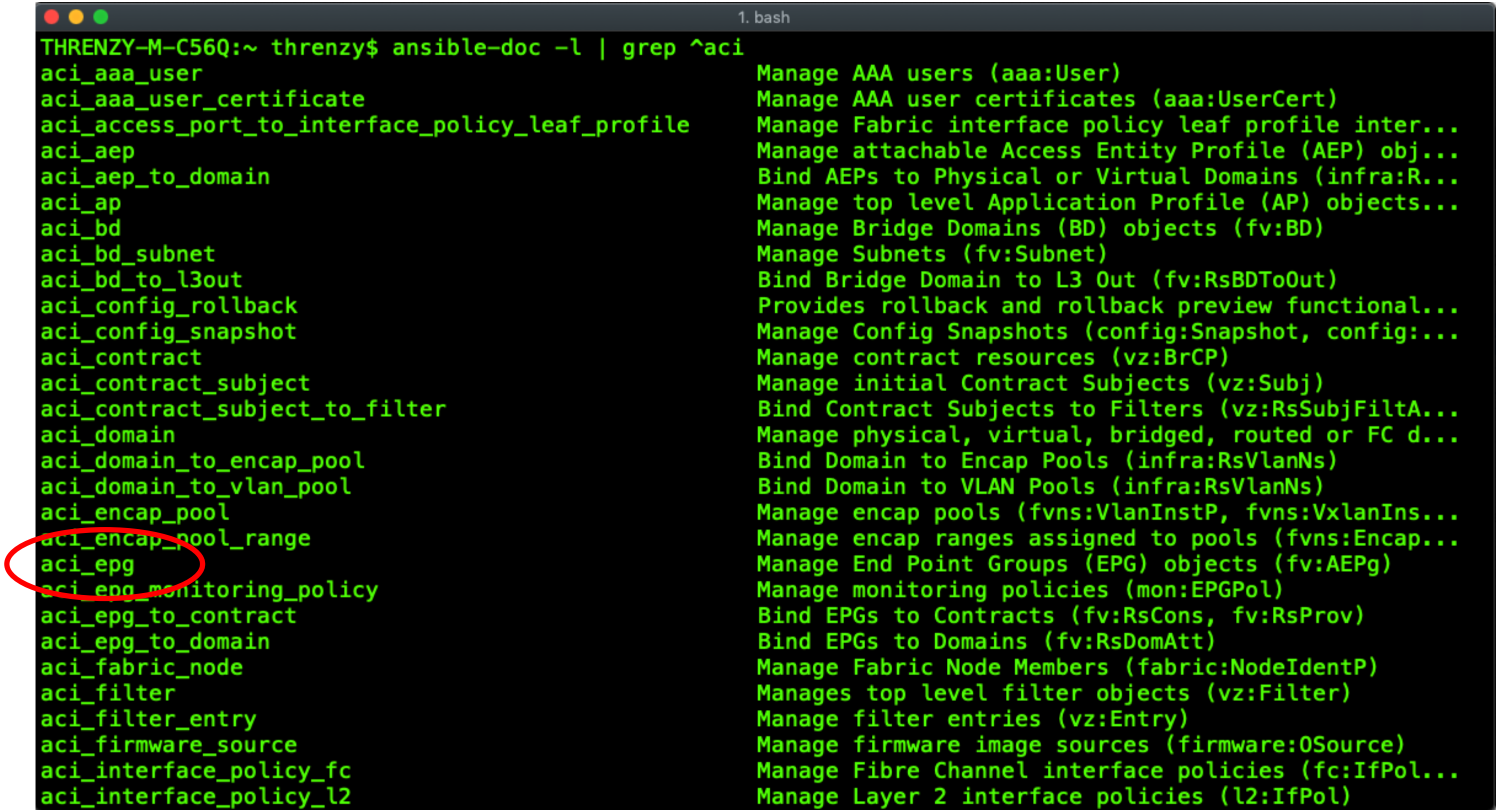


```
---
# Demo ACI Playbook
- name: Configuring Example Tenant
  hosts: apic1
  connection: local
  gather_facts: no

tasks:
 - name: Create Tenant
  aci_tenant:
   hostname: "{{ inventory_hostname }}"
   username: "{{ username }}"
   password: "{{ password }}"
   tenant: "CiscoLive"
   description: "Tenant configured by Ansible"
   validate_certs: no
   state: present
```

Ansible ACI Modules

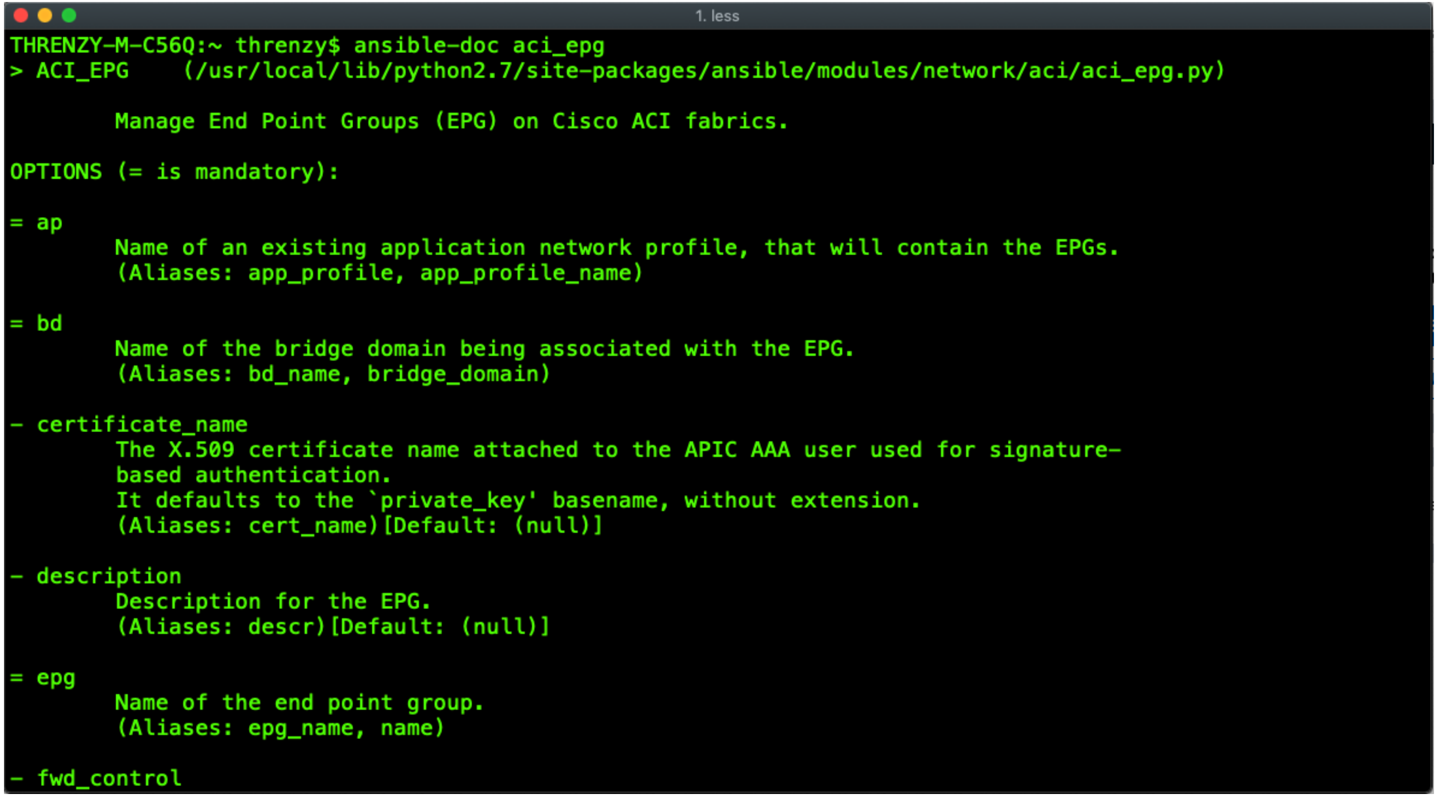
- Perform specific tasks (Create Tenant/VRF/BD)
- Already installed when you install Ansible
- Written in Python
  - Can develop your own modules
- 60+ ACI modules as of 2.9
  - 30+ Multisite Orchestrator Modules
- To see all Ansible Modules – ansible-doc -l
  - ACI specific ones – ansible-doc -l | grep ^aci

Ansible ACI Modules



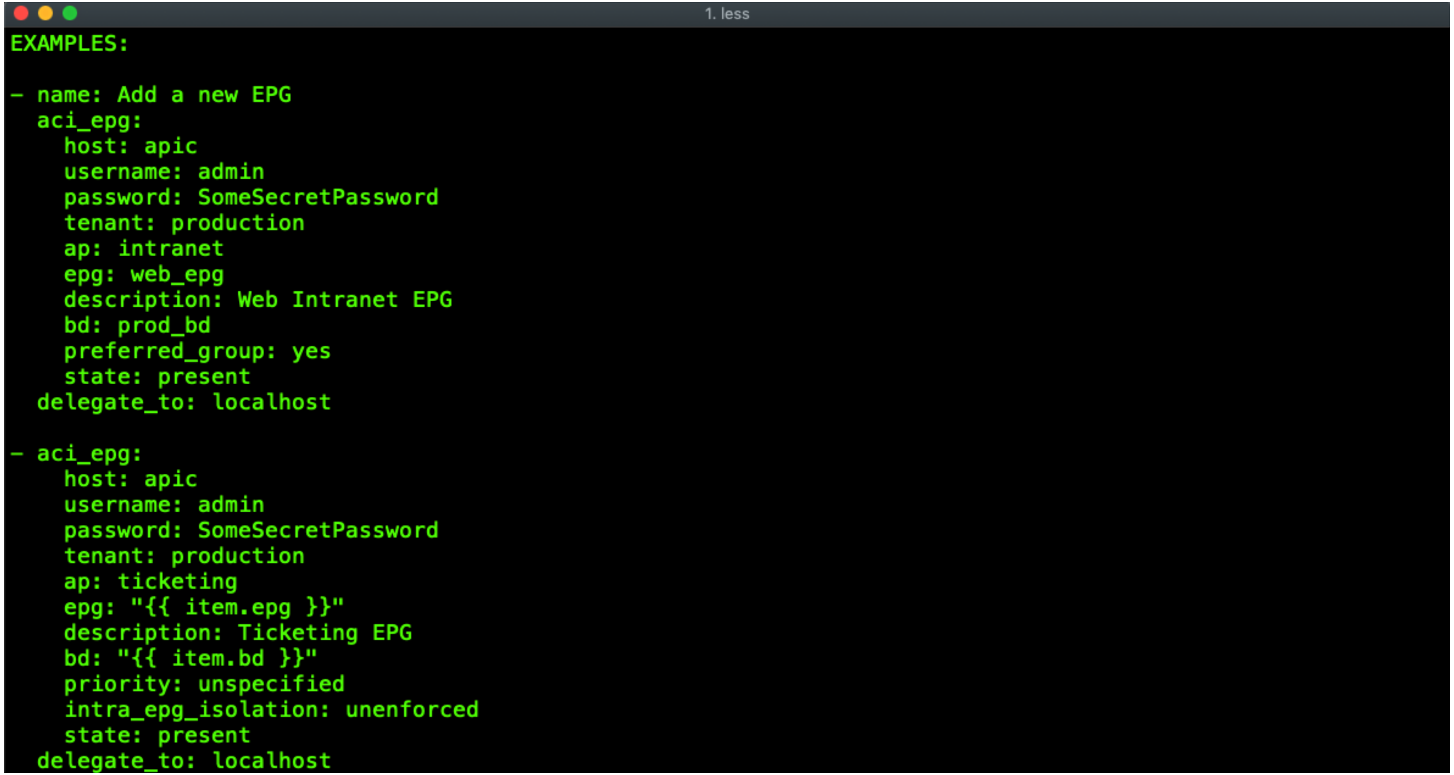
```
THRENZY-M-C56Q:~ threnzy$ ansible-doc -l | grep ^aci
aci_aaa_user                                         Manage AAA users (aaa:User)
aci_aaa_user_certificate                             Manage AAA user certificates (aaa:UserCert)
aci_access_port_to_interface_policy_leaf_profile     Manage Fabric interface policy leaf profile inter...
aci_aep                                              Manage attachable Access Entity Profile (AEP) obj...
aci_aep_to_domain                                    Bind AEPs to Physical or Virtual Domains (infra:R...
aci_ap                                               Manage top level Application Profile (AP) objects...
aci_bd                                               Manage Bridge Domains (BD) objects (fv:BD)
aci_bd_subnet                                        Manage Subnets (fv:Subnet)
aci_bd_to_l3out                                      Bind Bridge Domain to L3 Out (fv:RsBDToOut)
aci_config_rollback                                  Provides rollback and rollback preview functional...
aci_config_snapshot                                  Manage Config Snapshots (config:Snapshot, config:...
aci_contract                                         Manage contract resources (vz:BrCP)
aci_contract_subject                                 Manage initial Contract Subjects (vz:Subj)
aci_contract_subject_to_filter                       Bind Contract Subjects to Filters (vz:RsSubjFiltA...
aci_domain                                           Manage physical, virtual, bridged, routed or FC d...
aci_domain_to_encap_pool                             Bind Domain to Encap Pools (infra:RsVlanNs)
aci_domain_to_vlan_pool                              Bind Domain to VLAN Pools (infra:RsVlanNs)
aci_encap_pool                                       Manage encap pools (fvns:VlanInstP, fvns:VxlanIns...
aci_encap_pool_range                                 Manage encap ranges assigned to pools (fvns:Encap...
aci_epg                                              Manage End Point Groups (EPG) objects (fv:AEPg)
aci_epg_monitoring_policy                            Manage monitoring policies (mon:EPGPol)
aci_epg_to_contract                                  Bind EPGs to Contracts (fv:RsCons, fv:RsProv)
aci_epg_to_domain                                    Bind EPGs to Domains (fv:RsDomAtt)
aci_fabric_node                                      Manage Fabric Node Members (fabric:NodeIdentP)
aci_filter                                           Manages top level filter objects (vz:Filter)
aci_filter_entry                                     Manage filter entries (vz:Entry)
aci_firmware_source                                  Manage firmware image sources (firmware:OSource)
aci_interface_policy_fc                              Manage Fibre Channel interface policies (fc:IfPol...
aci_interface_policy_l2                              Manage Layer 2 interface policies (l2:IfPol)
```

Ansible ACI Modules



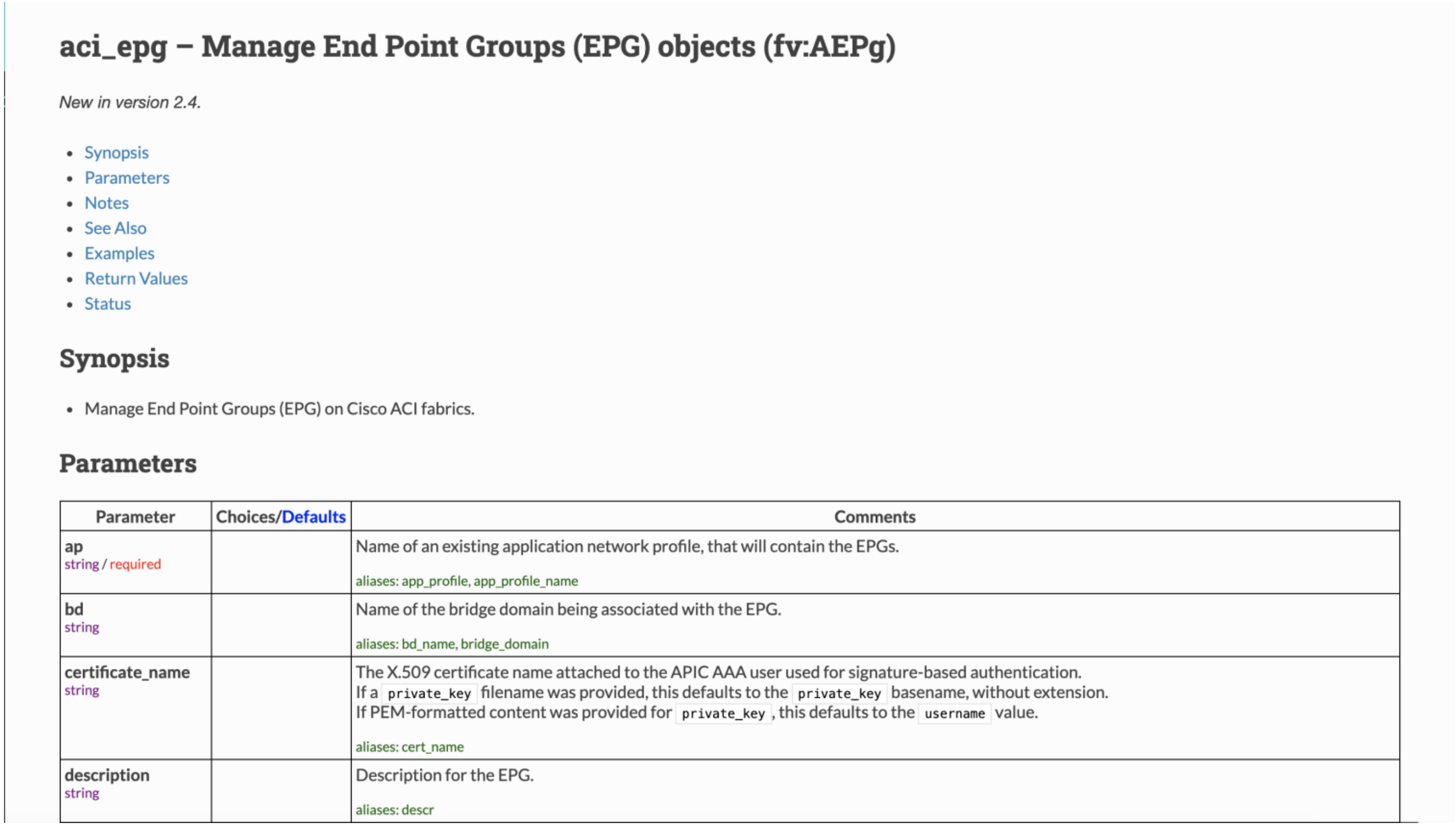
```
THRENZY-M-C56Q:~ threnzy$ ansible-doc aci_epg
> ACI_EPG    (/usr/local/lib/python2.7/site-packages/ansible/modules/network/aci/aci_epg.py)

        Manage End Point Groups (EPG) on Cisco ACI fabrics.

OPTIONS (= is mandatory):

= ap
        Name of an existing application network profile, that will contain the EPGs.
        (Aliases: app_profile, app_profile_name)

= bd
        Name of the bridge domain being associated with the EPG.
        (Aliases: bd_name, bridge_domain)

- certificate_name
        The X.509 certificate name attached to the APIC AAA user used for signature-
        based authentication.
        It defaults to the `private_key' basename, without extension.
        (Aliases: cert_name)[Default: (null)]

- description
        Description for the EPG.
        (Aliases: descr)[Default: (null)]

= epg
        Name of the end point group.
        (Aliases: epg_name, name)

- fwd_control
```

Ansible ACI Modules



```
EXAMPLES:

- name: Add a new EPG
  aci_epg:
    host: apic
    username: admin
    password: SomeSecretPassword
    tenant: production
    ap: intranet
    epg: web_epg
    description: Web Intranet EPG
    bd: prod_bd
    preferred_group: yes
    state: present
  delegate_to: localhost

- aci_epg:
    host: apic
    username: admin
    password: SomeSecretPassword
    tenant: production
    ap: ticketing
    epg: "{{ item.epg }}"
    description: Ticketing EPG
    bd: "{{ item.bd }}"
    priority: unspecified
    intra_epg_isolation: unenforced
    state: present
  delegate_to: localhost
```

Ansible ACI Modules



aci_epg – Manage End Point Groups (EPG) objects (fv:AEPg)

New in version 2.4.

- Synopsis
- Parameters
- Notes
- See Also
- Examples
- Return Values
- Status

Synopsis

- Manage End Point Groups (EPG) on Cisco ACI fabrics.

Parameters

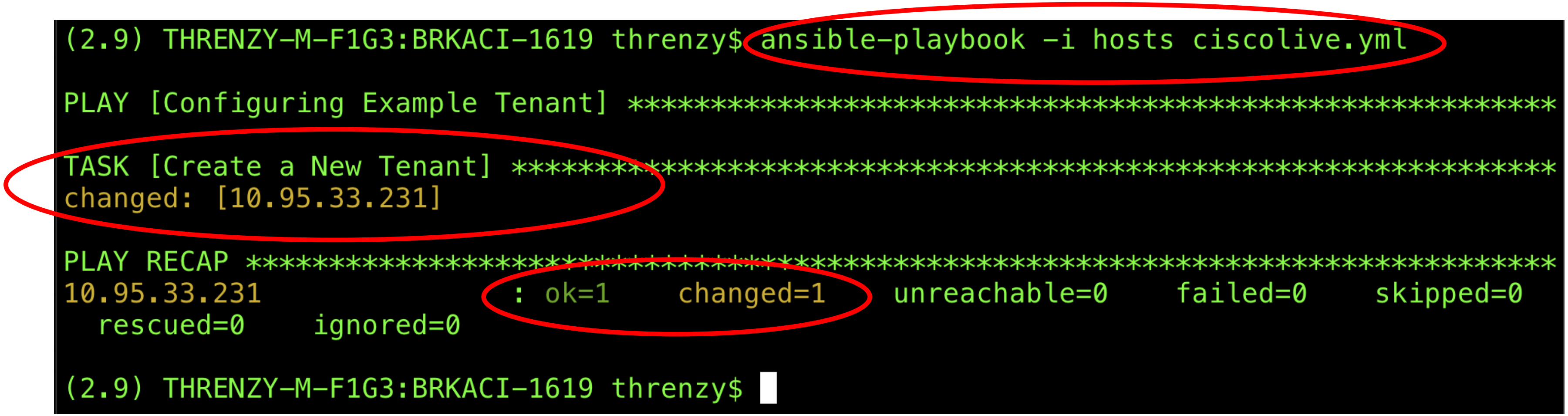
Parameter	Choices/Defaults	Comments
ap string / required		Name of an existing application network profile, that will contain the EPGs. aliases: app_profile, app_profile_name
bd string		Name of the bridge domain being associated with the EPG. aliases: bd_name, bridge_domain
certificate_name string		The X.509 certificate name attached to the APIC AAA user used for signature-based authentication. If a `private_key` filename was provided, this defaults to the `private_key` basename, without extension. If PEM-formatted content was provided for `private_key`, this defaults to the `username` value. aliases: cert_name
description string		Description for the EPG. aliases: descr

Automating ACI with Playbooks

Running an ACI Playbook

- Ansible command
  - Good for running single commands – ad-hoc
  - **`ansible 10.15.20.101 --user=admin --ask-pass -a "uptime"`**
- Command to run our playbooks
  - **`ansible-playbook –i {inventory file} {Playbook file}`**
  - **`ansible-playbook –i hosts ciscolive.yml`**
- Check mode(--check)
  - Run through playbook without making changes
  - **`ansible-playbook –i hosts tenant.yml --check`**

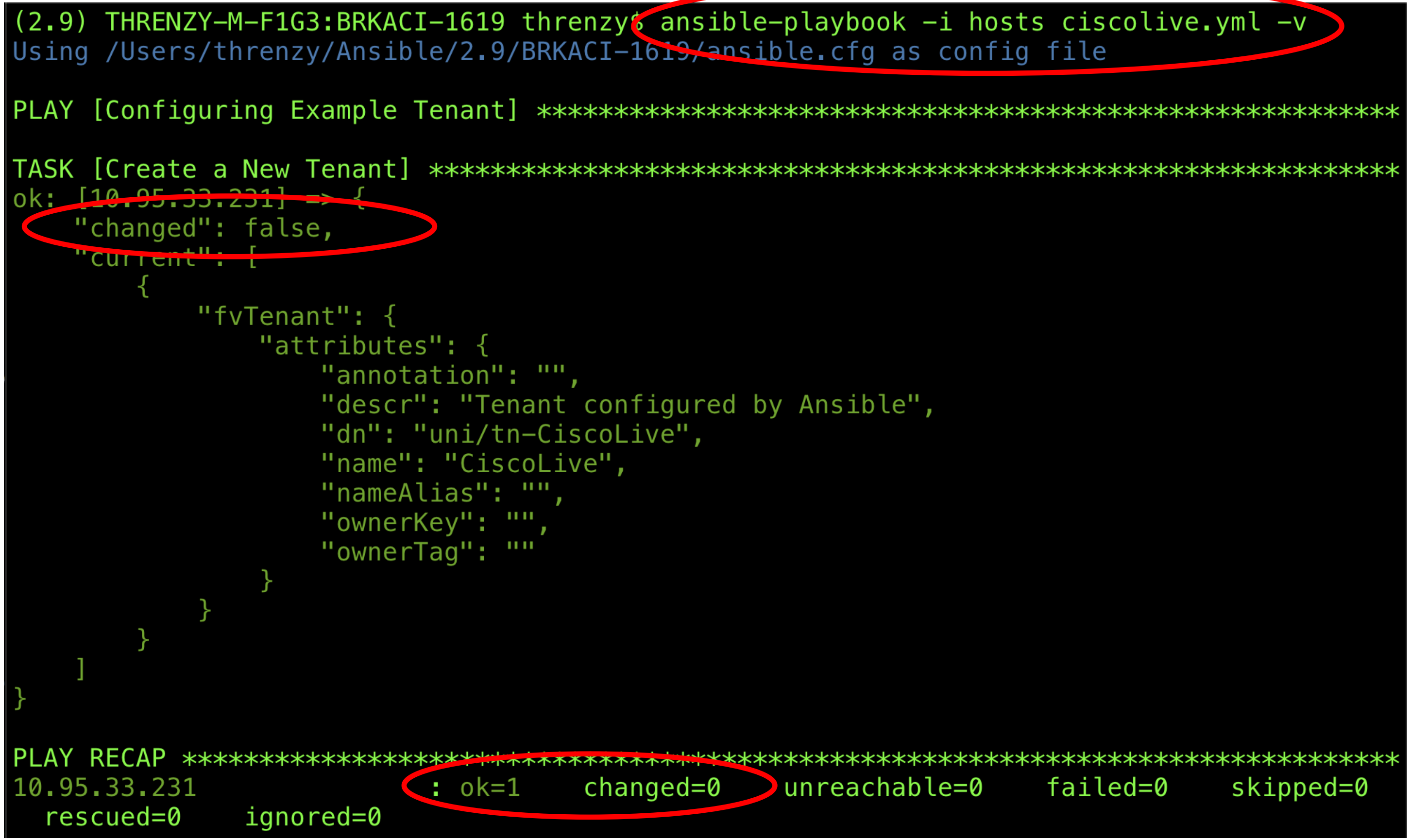
Running our Tenant Playbook



```
(2.9) THRENZY-M-F1G3:BRKACI-1619 threnzy$ ansible-playbook -i hosts ciscolive.yml

PLAY [Configuring Example Tenant] *********************************************

TASK [Create a New Tenant] ****************************************************
changed: [10.95.33.231]

PLAY RECAP ********************************************************************
10.95.33.231               : ok=1    changed=1    unreachable=0    failed=0    skipped=0
   rescued=0    ignored=0

(2.9) THRENZY-M-F1G3:BRKACI-1619 threnzy$ 
```

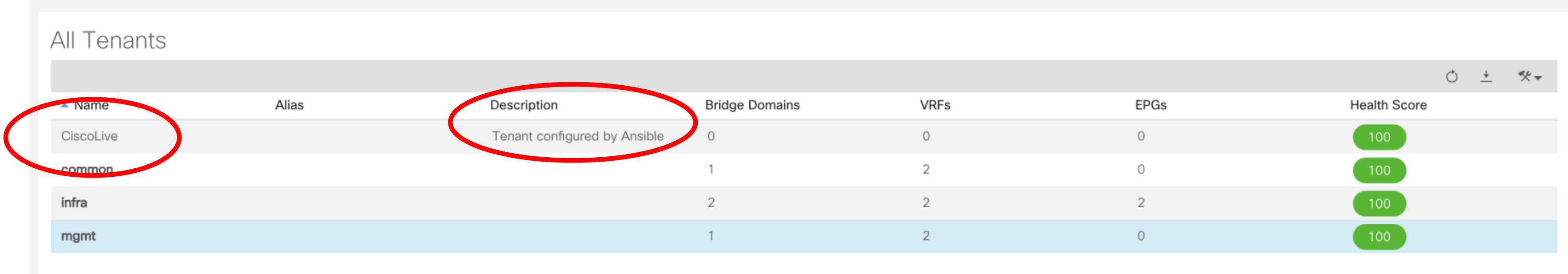
- Runs through each task.
- Let’s you know how many tasks were OK, changed, failed, etc.
- To see more output use “-v”, “-vvv”, or “-vvvv”

Tenant Playbook with verbose output



```
(2.9) THRENZY-M-F1G3:BRKACI-1619 threnzy$ ansible-playbook -i hosts ciscolive.yml -v
Using /Users/threnzy/Ansible/2.9/BRKACI-1619/ansible.cfg as config file

PLAY [Configuring Example Tenant] *********************************************************

TASK [Create a New Tenant] ****************************************************************
ok: [10.95.33.231] => {
    "changed": false,
    "current": [
        {
            "fvTenant": {
                "attributes": {
                    "annotation": "",
                    "descr": "Tenant configured by Ansible",
                    "dn": "uni/tn-CiscoLive",
                    "name": "CiscoLive",
                    "nameAlias": "",
                    "ownerKey": "",
                    "ownerTag": ""
                }
            }
        }
    ]
}

PLAY RECAP ********************************************************************************
10.95.33.231               : ok=1    changed=0    unreachable=0    failed=0    skipped=0
  rescued=0    ignored=0
```

Verifying the APIC



All Tenants

Name	Alias	Description	Bridge Domains	VRFs	EPGs	Health Score
CiscoLive		Tenant configured by Ansible	0	0	0	100
common			1	2	0	100
infra			2	2	2	100
mgmt			1	2	0	100

Signature-Based Authentication

A Note about Authentication

- Authentication using username/password
  - Not very secure
- Large playbooks with lots of tasks can fail
  - Especially with iteration
- Can cause sessions to get throttled
  - NGINX throttling – ACI 3.1
- Workarounds
  - Disable APIC session throttling
  - Add pause in tasks
  - Signature-based authentication***

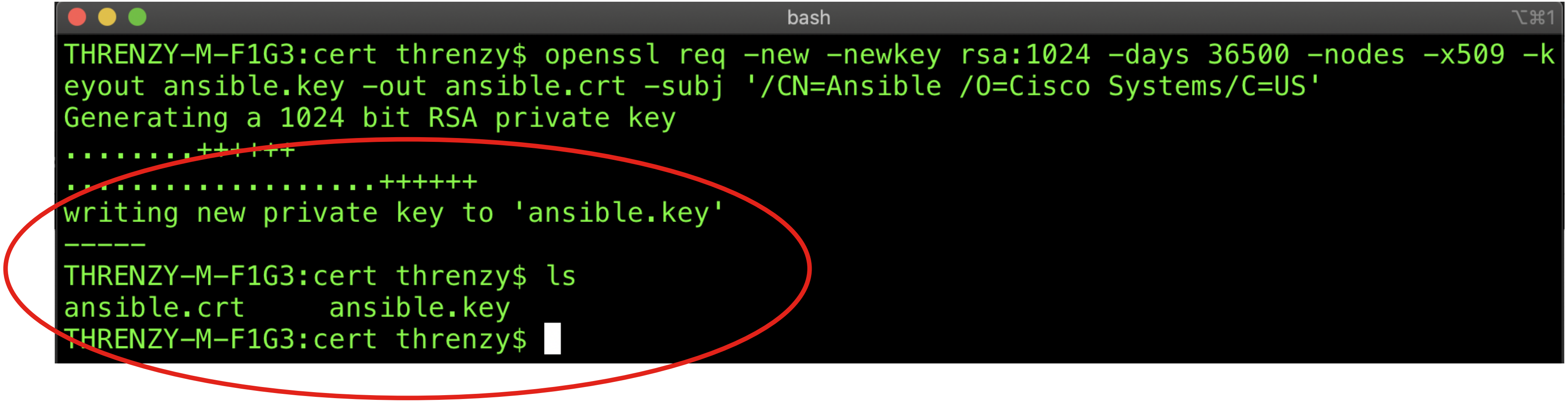
Signature-based Authentication

- Available as of 2.5
- Generate certificate using **`openssl`**
- Create a local user on APIC
  - Ansible Module - **`aci_aaa_user`**
- Push Certificate up to APIC
  - Ansible Module - **`aci_aaa_user_certificate`**
- Modify your tasks to leverage new Key
  - Replace username/password - private_key: keyname.key

Generate Self Signed Certificate

- Use “openssl” to generate your cert

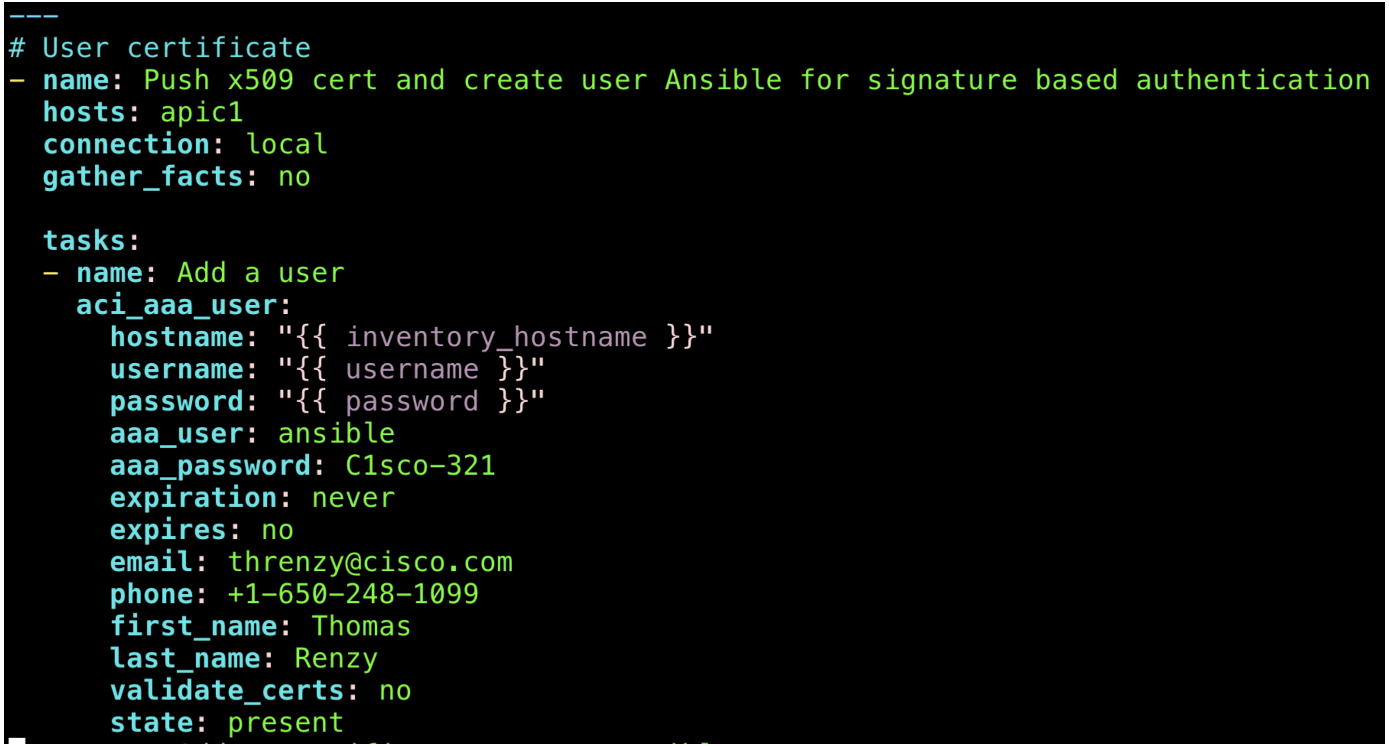
```
openssl req -new -newkey rsa:1024 -days 36500 -nodes -x509 -keyout admin.key -out admin.crt -
subj '/CN=Admin/O=Your Company/C=US'
```



```
THRENZY-M-F1G3:cert threnzy$ openssl req -new -newkey rsa:1024 -days 36500 -nodes -x509 -k
eyout ansible.key -out ansible.crt -subj '/CN=Ansible /O=Cisco Systems/C=US'
Generating a 1024 bit RSA private key
........++++++
...................++++++
writing new private key to 'ansible.key'
-----
THRENZY-M-F1G3:cert threnzy$ ls
ansible.crt      ansible.key
THRENZY-M-F1G3:cert threnzy$ █
```

Automate Local User Creation

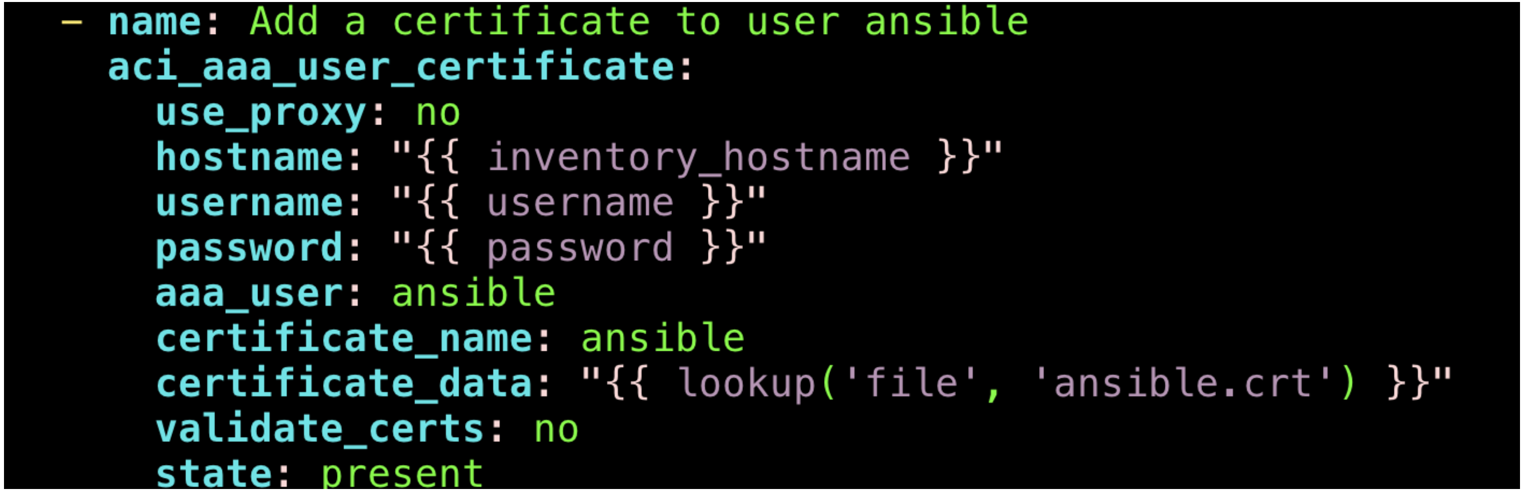
- Create a local user using `aci_aaa_user` module



```
---
# User certificate
- name: Push x509 cert and create user Ansible for signature based authentication
  hosts: apic1
  connection: local
  gather_facts: no

  tasks:
  - name: Add a user
    aci_aaa_user:
      hostname: "{{ inventory_hostname }}"
      username: "{{ username }}"
      password: "{{ password }}"
      aaa_user: ansible
      aaa_password: C1sco-321
      expiration: never
      expires: no
      email: threnzy@cisco.com
      phone: +1-650-248-1099
      first_name: Thomas
      last_name: Renzy
      validate_certs: no
      state: present
```

Add new Certificate to new Local User

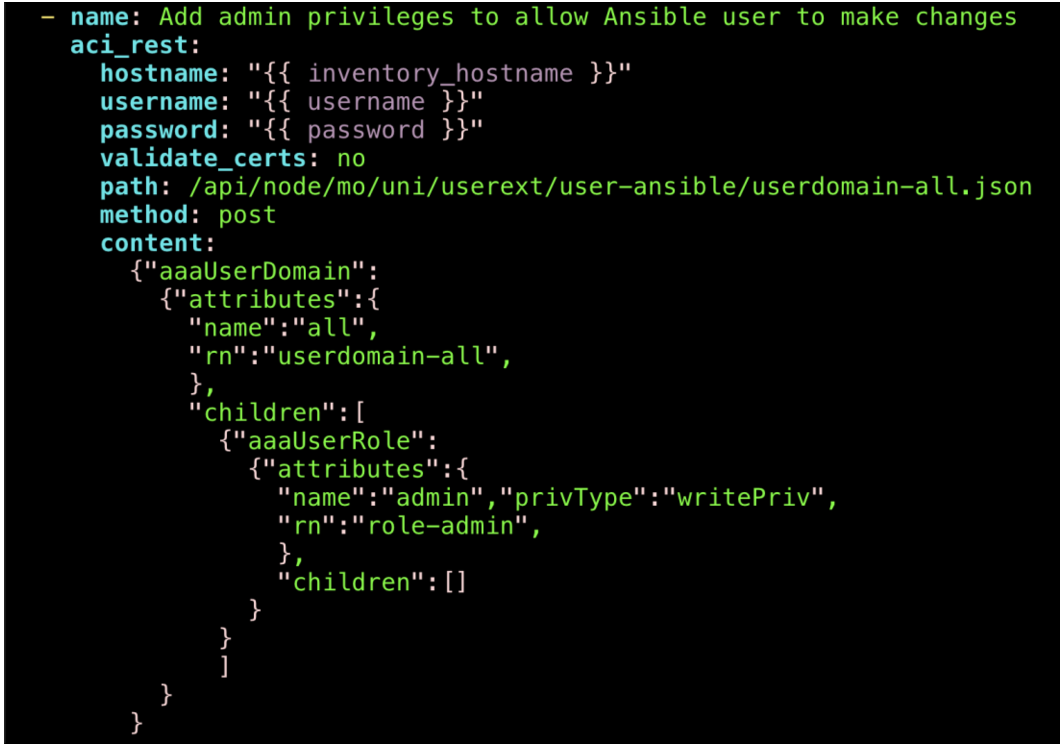
- Can copy Cert to use **`aci_aaa_user_certificate`** Module



```
- name: Add a certificate to user ansible
  aci_aaa_user_certificate:
    use_proxy: no
    hostname: "{{ inventory_hostname }}"
    username: "{{ username }}"
    password: "{{ password }}"
    aaa_user: ansible
    certificate_name: ansible
    certificate_data: "{{ lookup('file', 'ansible.crt') }}"
    validate_certs: no
    state: present
```

Assign proper privileges to Local User

- Leverages the aci_rest module. – More later



```
- name: Add admin privileges to allow Ansible user to make changes
  aci_rest:
   hostname: "{{ inventory_hostname }}"
   username: "{{ username }}"
   password: "{{ password }}"
   validate_certs: no
   path: /api/node/mo/uni/userext/user-ansible/userdomain-all.json
   method: post
   content:
    {"aaaUserDomain":
     {"attributes":{
      "name":"all",
      "rn":"userdomain-all",
      },
      "children":[
       {"aaaUserRole":
        {"attributes":{
         "name":"admin","privType":"writePriv",
         "rn":"role-admin",
         },
         "children":[]
        }
       }
       ]
     }
    }
```

Demo – Deploy Signature-Based Authentication

Updated Tenant Playbook

```
---
# Demo ACI Playbook
- name: Configuring Example Tenant
   hosts: apic1
   connection: local
   gather_facts: no

tasks:
  - name: Create Tenant
    aci_tenant:
      hostname: "{{ inventory_hostname }}"
      username: ansible
      private_key: ansible.key
      tenant: "CiscoLive"
      description: "Tenant configured by Ansible"
      validate_certs: no
      state: present
```

Finally…

IF YOU DON'T IMPLEMENT SIGNATURE BASED AUTHENTICATION
NSTRUCTO
YOU'RE GONNA HAVE A BAD TIME
imgflip.com

More complex Playbook – A Three Tier Application

A Sample Three Tier Application in Ansible

- We want to do the following:
  - Create a new Tenant – Ansible
  - New VRF – ansible-VRF
  - New BD – ansible-BD
  - Application Profile – ansible-AP
  - 3 EPGs
    - Web, App, DB
- 2 Contracts (and associated subjects/filters)
    - web_to_app – Communication between Web EPG and App EPG on http (tcp 80)
    - app_to_db - Communication between App EPG and DB EPG on sql (tcp 1433)

Variables in Three Tier Application

- Use of variables in Ansible
  - Can be used to substitute values in playbooks
  - Leverages jinja2 templating – “{{ Variable Value }}”
  - Defined in inventory, playbook, external
    - Variables have precedence

```
vars:
    mytenant: ciscolive
…
tenant: "{{ mytenant }}"
```

Variables in Three Tier Application

```
vars:
     tenant: Ansible
     vrf: ansible-VRF
     bd:
       name: ansible-BD
       ip: 10.255.255.1
       mask: 24
     app_profile: ansible-AP
     http_filter: http_ans
     http_filter_entry: http_ans_entry
     web_to_app_contract: web_to_app
     web_to_app_contract_subject: web_to_app_subject
     db_filter: db_ans_entry
     db_filter_entry: db_ans_entry
     app_to_db_contract: db_to_app
     app_to_db_contract_subject: app_to_db_subject
     epg1: web
     epg2: app
     epg3: db
```

Loops (iteration) with loop

- Repeat a task multiple times
  - Suppose you need to create 3 or more EPGs
  - Tedious to write out 3 or more additional tasks
  - with_items: Also a method

```
aci_epg:
  …
  epg: "{{ item.epg }}"
loop:
  - epg: "{{ epg1 }}"
  - epg: "{{ epg2 }}"
  - epg: "{{ epg3 }}"
```

Modules used in Three-Tier Application

- aci_tenant
- aci_vrf
- aci_bd
- aci_bd_subnet
- aci_ap
- aci_epg

- aci_contract
- aci_filter
- aci_filter_entry
- aci_epg_to_contract
- aci_contract_subject
- aci_contract_subject_to_filter

Demo – Deploy a Three Tier Application

The Ansible ACI REST Module

Ansible ACI Modules, XML and JSON

- Ansible is a great solution to automate ACI tasks
- ACI modules can do most common configurations
- Lots modules as of 2.9
    - Modules added to every version
    - Modules aren’t 1-to-1 with all ACI features
- What if you are already using XML and JSON?

ACI REST Module (aci_rest)

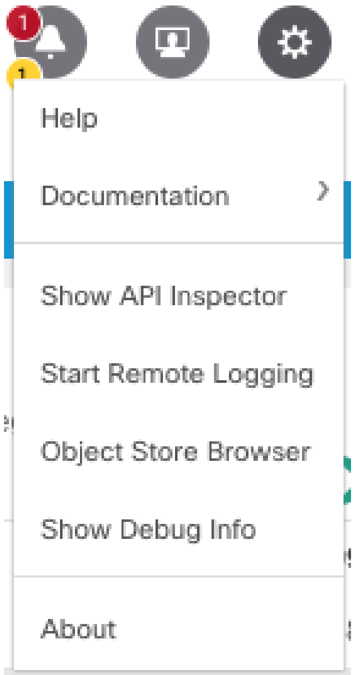
- Direct access and management to APIC REST API
- Can use JSON, XML, and even YAML
- Can POST, DELETE, GET
    - Similar to what you can do in POSTMAN
- Variables work with this as well
- Can grab GUI configurations through
    - API Inspector
    - Download JSON/XML configuration

Example aci_rest module task

```
tasks:
 - name: Add admin privileges to allow Ansible user to make changes
    aci_rest:
      hostname: "{{ inventory_hostname }}"
      username: "{{ username }}"
      password: "{{ password }}"
      validate_certs: no
      path: /api/node/mo/uni/userext/user-ansible/userdomain-all.json
      method: post
      content:
        {"aaaUserDomain":
          {"attributes":{
            "name":"all",
            "rn":"userdomain-all",
            },
            "children":[
              {"aaaUserRole":
                {"attributes":{
                  "name":"admin","privType":"writePriv",
                  "rn":"role-admin",
                  },
                  "children":[]
                }
              }
              ]
          }
        }
```

Configuration Example with aci_rest

- Set my COOP policy to strict
  - Enables authenticated MD5 only
- End Point Loop Protection
  - Specified how frequent MAC moves are handled
- Global Enforce Subnet Check
  - Limit IP learning at the VRF level
- Currently no Ansible modules
- Grabbed from API Inspector



COOP Policy with ACI REST module

```
path: /api/node/mo/uni/fabric/pol-default.json
        method: post
        content: |
          {
            "coopPol": {
              "attributes":{
                "type":"strict",
                "dn":"uni/fabric/pol-default"
                }
              }
            }
```

Enforce Subnet Check with ACI REST module

```
path: /api/node/mo/uni/infra/settings.json
        method: post
        content: |
          {
            "infraSetPol": {
              "attributes": {
                "enforceSubnetCheck":"true",
                "dn":"uni/infra/settings"
                }
             }
           }
```

End Point Loop Protection with ACI REST module

```
path: /api/node/mo/uni/infra/epLoopProtectP-default.json
        method: post
        content: |
          {
            "epLoopProtectP": {
              "attributes": {
                "action": "",
                "adminSt": "enabled",
                "loopDetectIntvl": "60",
                "loopDetectMult": "4",
                "dn":"uni/infra/epLoopProtectP-default"
              }
            }
          }
```

Demo - Configuration with aci_rest

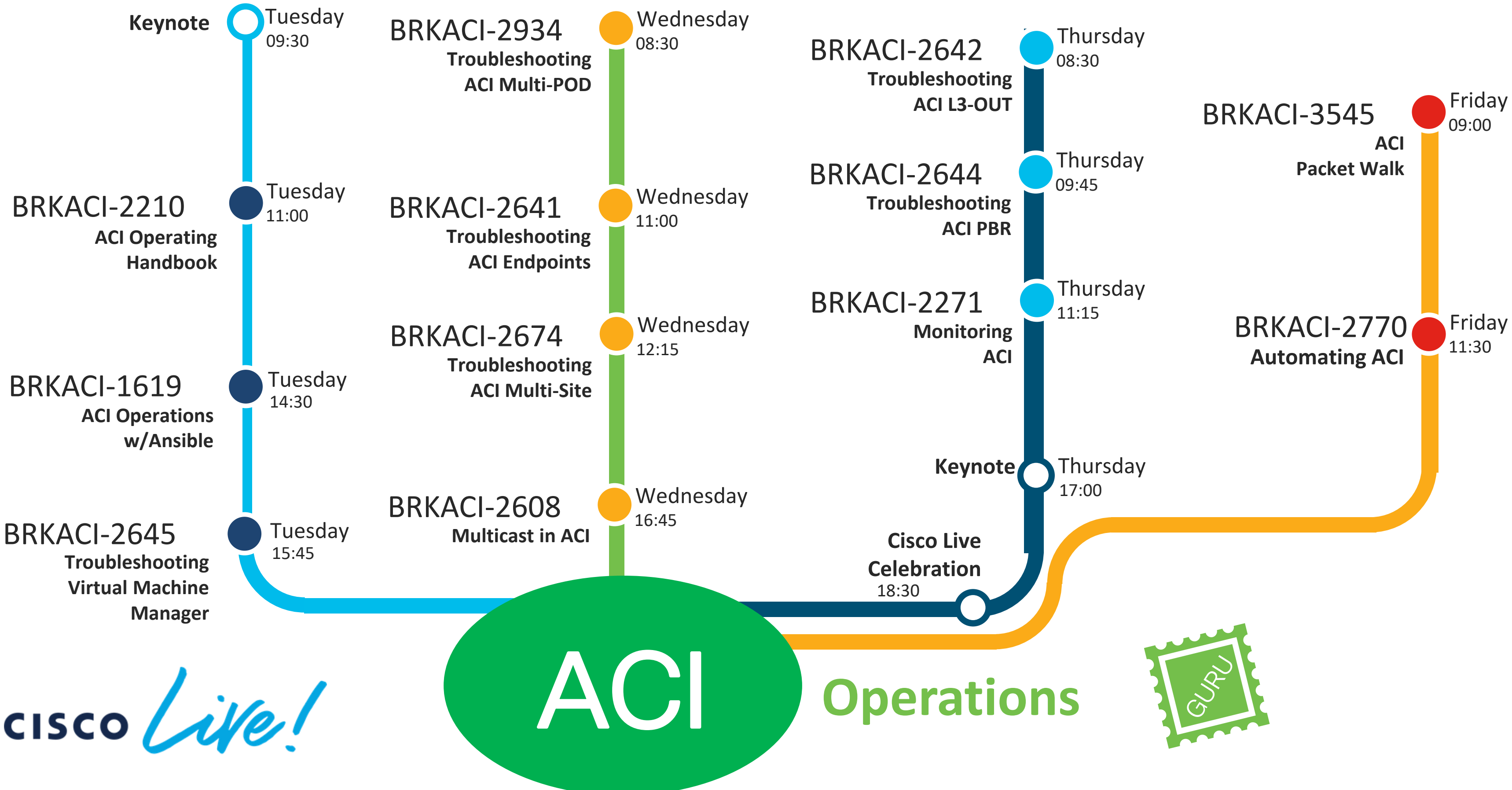
Summary

Benefits of Automating ACI with Ansible

- Automate repeatable tasks
- Saves time, efficient
- Ease of writing/reading inventory/playbooks
- No special programming skills needed
- Small learning curve
- Modules pre-built with most common tasks
- aci_rest module for leveraging JSON/XML
    - Can build tasks/plays not covered by a module

Hands on sessions

LABACI-1013 – Intro to Automating ACI with Ansible
LABACI-1001 – Introduction to the APIC
LABACI-1011 – Intro to Programming ACI with Python
LABACI-1007 – ACI Infrastructure as code with Terraform
LABACI-2148 - ACI Monitoring, Stats, Analytics and Notifications…
LABDCN-1258 – Network Automation with Ansible (NX-OS)
DEVWKS-2232 – Automate your ACI Multisite with APIs



Keynote Tuesday 09:30

BRKACI-2210 ACI Operating Handbook Tuesday 11:00

BRKACI-1619 ACI Operations w/Ansible Tuesday 14:30

BRKACI-2645 Troubleshooting Virtual Machine Manager Tuesday 15:45

BRKACI-2934 Troubleshooting ACI Multi-POD Wednesday 08:30

BRKACI-2641 Troubleshooting ACI Endpoints Wednesday 11:00

BRKACI-2674 Troubleshooting ACI Multi-Site Wednesday 12:15

BRKACI-2608 Multicast in ACI Wednesday 16:45

BRKACI-2642 Troubleshooting ACI L3-OUT Thursday 08:30

BRKACI-2644 Troubleshooting ACI PBR Thursday 09:45

BRKACI-2271 Monitoring ACI Thursday 11:15

Keynote Thursday 17:00

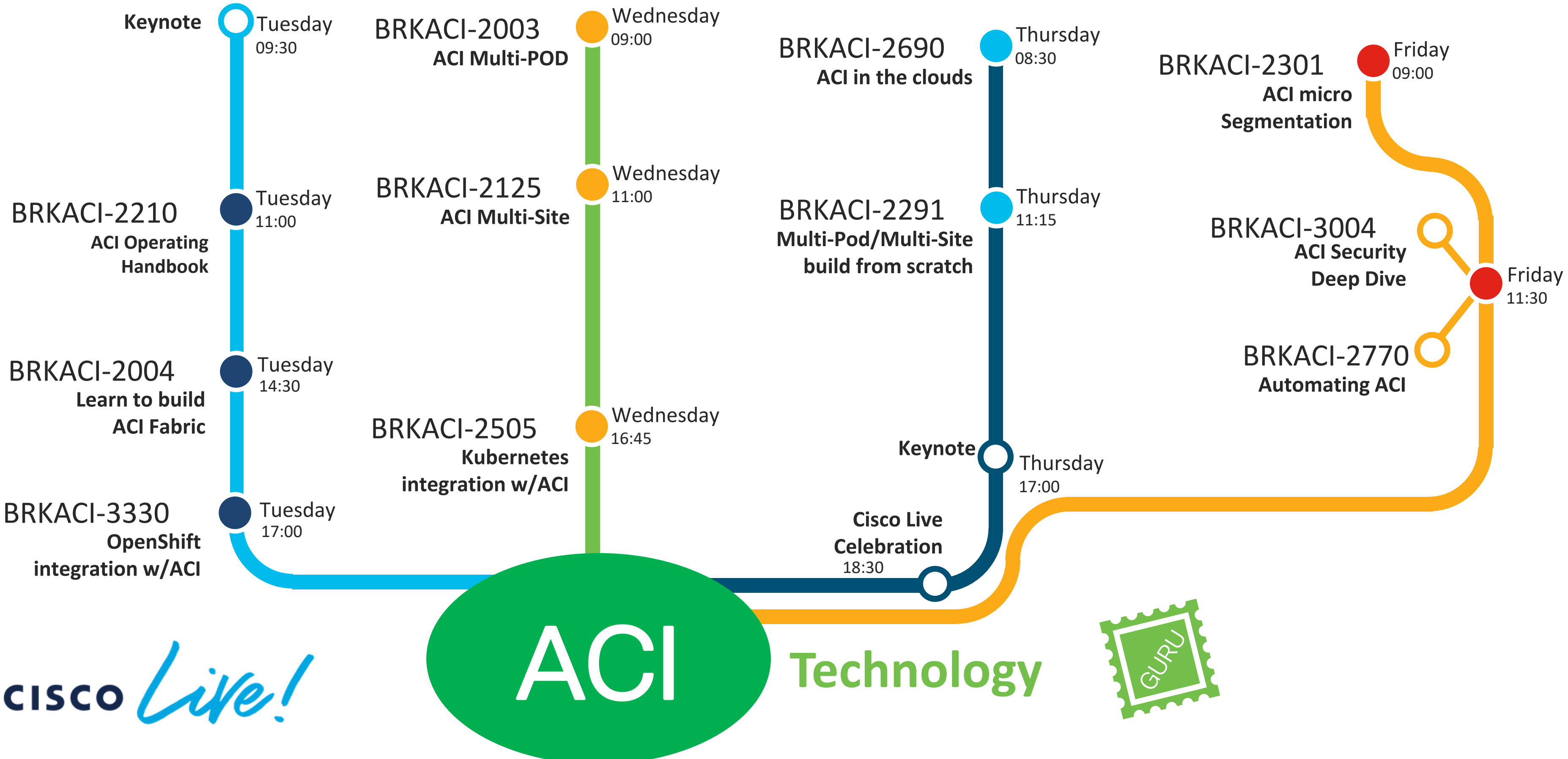
Cisco Live Celebration 18:30

BRKACI-3545 ACI Packet Walk Friday 09:00

BRKACI-2770 Automating ACI Friday 11:30

ACI

Operations



Keynote
Tuesday
09:30

BRKACI-2210
ACI Operating Handbook
Tuesday
11:00

BRKACI-2004
Learn to build ACI Fabric
Tuesday
14:30

BRKACI-3330
OpenShift integration w/ACI
Tuesday
17:00

BRKACI-2003
ACI Multi-POD
Wednesday
09:00

BRKACI-2125
ACI Multi-Site
Wednesday
11:00

BRKACI-2505
Kubernetes integration w/ACI
Wednesday
16:45

BRKACI-2690
ACI in the clouds
Thursday
08:30

BRKACI-2291
Multi-Pod/Multi-Site build from scratch
Thursday
11:15

Keynote
Thursday
17:00

Cisco Live Celebration
18:30

BRKACI-2301
ACI micro Segmentation
Friday
09:00

BRKACI-3004
ACI Security Deep Dive

BRKACI-2770
Automating ACI

Friday
11:30

ACI

Technology

GURU

cisco Live!

References

Ansible Documentation

http://docs.ansible.com/

Ansible ACI Documentation

https://docs.ansible.com/ansible/devel/scenario_guides/guide_aci.html

Ansible ACI Modules

http://docs.ansible.com/ansible/devel/modules/list_of_network_modules.html#aci

Ansible Variables (and precedence)

https://docs.ansible.com/ansible/latest/user_guide/playbooks_variables.html

Complete your online session survey

- Please complete your session survey after each session. Your feedback is very important.
- Complete a minimum of 4 session surveys and the Overall Conference survey (starting on Thursday) to receive your Cisco Live t-shirt.
- All surveys can be taken in the Cisco Events Mobile App or by logging in to the Content Catalog on ciscolive.com/emea.

Cisco Live sessions will be available for viewing on demand after the event at ciscolive.com.

Continue your education

Demos in the Cisco Showcase

Walk-In Labs

Meet the Engineer 1:1 meetings

Related sessions

Thank you

You make **possible**