Automating Arista Network Fabric

What is Arista Validated Design (AVD)?

An extensible data model that defines Arista's Unified Cloud Network (UCN) architecture as "code"

Benefits

- Automatic generation of documentation and validation tests
- Foundation for Infrastructure-as-Code
- Faster time to production
- Reduced risk of configuration error
- Consistent global configuration changes across the network

How

```yaml
# Fabric/Host variables
underlay_routing_protocol: EBGP
bgp_as: 65001
```

```yaml
# Structured configuration
router_bgp:
  as: 65001
  address_family_ipv4:
    peer_groups:
      UNDERLAY-PEERS:
        active: true
```

```
{# eos - Router BGP #}
{% if router_bgp.as is arista.avd.defi
!
router bgp {{ router_bgp.as }}
```

```
# EOS CLI
router bgp 65001
   address-family ipv4
      neighbor UNDERLAY-PEERS activate
```

Automated documentation

POINT-TO-POINT LINKS NODE ALLOCATION

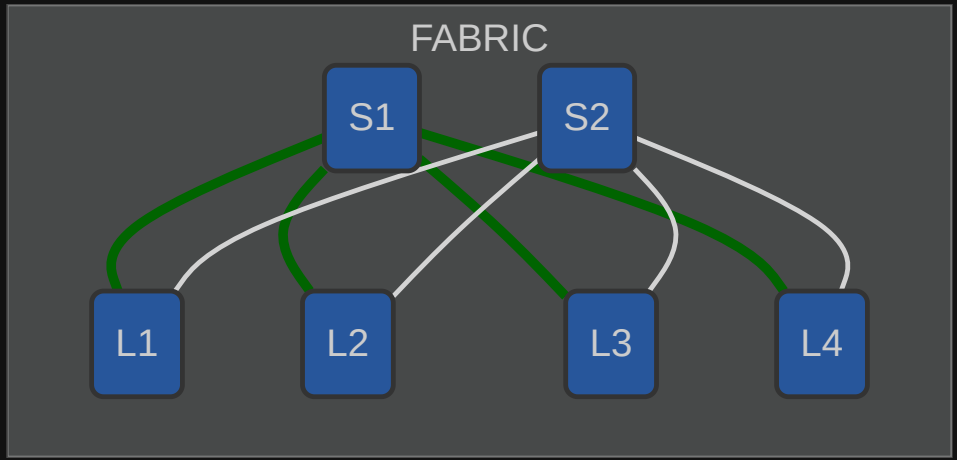
Node	Node Interface	Node IP Address	Peer Node	Peer Interface	Peer IP Address
dc1-leaf1a	Ethernet1	10.255.255.1/31	dc1-spine1	Ethernet1	10.255.255.0/31
dc1-leaf1a	Ethernet2	10.255.255.3/31	dc1-spine2	Ethernet1	10.255.255.2/31
dc1-leaf1b	Ethernet1	10.255.255.5/31	dc1-spine1	Ethernet2	10.255.255.4/31
dc1-leaf1b	Ethernet2	10.255.255.7/31	dc1-spine2	Ethernet2	10.255.255.6/31

Group variables

Fabric wide definitions | Topology

```yaml
# FABRIC.yml
underlay_routing_protocol: EBGP
overlay_routing_protocol: EBGP

local_users:
  ansible:
    privilege: 15
    role: network-admin
  admin:
    privilege: 15
    role: network-admin
```



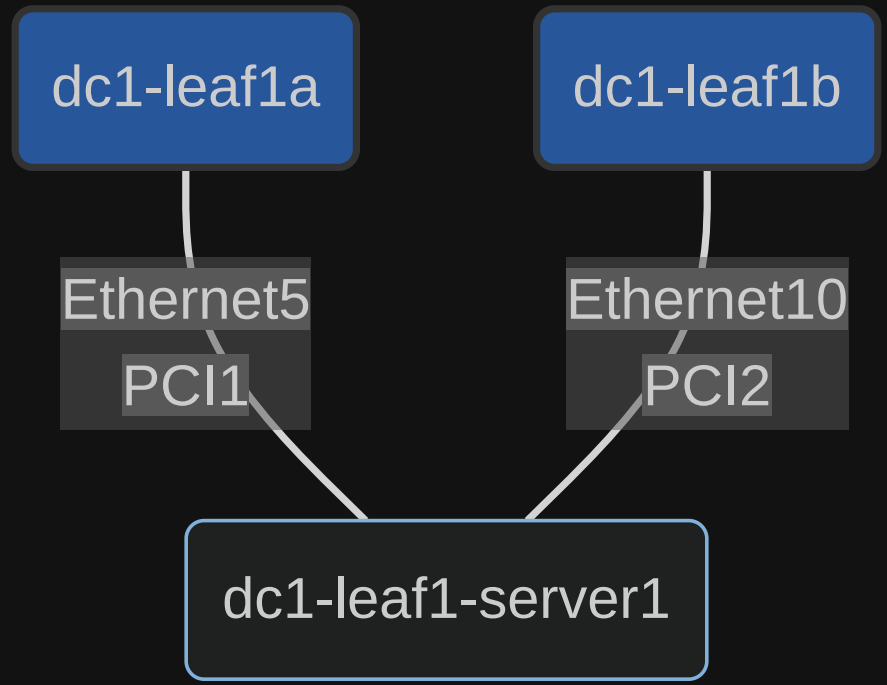
Network services

- Tenants
- L3 & L2 services

```yaml
---
# NETWORK_SERVICES.yml
tenants:
  TENANT1:
    mac_vrf_vni_base: 10000
    vrfs:
      VRF10:
        vrf_vni: 10
        svis:
          "11":
            name: VRF10_VLAN11
            enabled: true
            ip_address_virtual: 10.10.11.1/24
    l2vlans:
      "3401":
        name: L2_VLAN3401
      "3402":
        name: L2_VLAN3402
```

Connected endpoints | Topology

```yaml
---
# CONNECTED_ENDPOINTS.yml
servers:
  dc1-leaf1-server1:
    adapters:
    - type: server
      server_ports: [ PCI1, PCI2 ]
      switch_ports: [ Ethernet5, Ethernet10 ]
      switches: [ dc1-leaf1a, dc1-leaf1b ]
      vlans: 11-12,21-22
      native_vlan: 4092
      mode: trunk
      spanning_tree_portfast: edge
      port_channel:
        description: PortChannel dc1-leaf1-server1
        mode: active
```



Ansible AVD Collection

KEYBOARD SHORTCUTS

`right` / `space`	next animation or slide
`left` / `shift` `space`	previous animation or slide
`up`	previous slide
`down`	next slide

Code

```typescript
interface User {
  id: number
  firstName: string
  lastName: string
  role: string
}

function updateUser(id: number, update: User) {
  const user = getUser(id)
  const newUser = { ...user, ...update }
  saveUser(id, newUser)
}
```

```yaml
---
all:
  children:
    cv_servers:
      hosts:
        cv_atd1:
          ansible_host: 192.168.0.5
          ansible_user: arista
          ansible_password: # update password with "Lab Cre
          cv_collection: v3
```

Themes

Slidev comes with powerful theming support. Themes can provide styles, layouts, components, or even configurations for tools. Switching between themes by just **one edit** in your frontmatter:

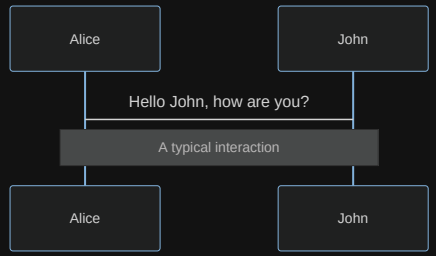
```
---
theme: default
---
```

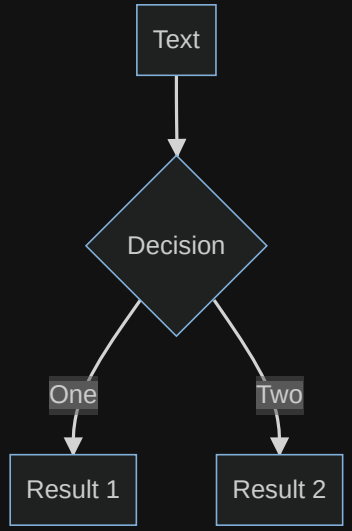
```
---
theme: seriph
---
```

Read more about How to use a theme and check out the Awesome Themes Gallery.

Diagrams

You can create diagrams / graphs from textual descriptions, directly in your Markdown.





Thank you

Documentation · GitHub · Showcases