ARISTA

Design Guide

Campus Network and Security

Version 1.0

Table of contents

Table of contents

Introduction

A campus network is a multi-tiered infrastructure designed to ensure robust connectivity, comprehensive security, and scalable performance across an organization's environment. This infrastructure is composed of several essential services:

Wired Network services: Encompassing traditional three-tiered L2/L3 architectures as well as modern leaf-spine architectures featuring EVPN-VXLAN overlays, the wired network facilitates connections for users,IoT, and OT devices, enabling communication among themselves and access to services within datacenters, cloud environments, and the public internet.

Wireless Network services: Consisting of WiFi Access Points (APs) and wireless termination devices like switches or controllers to manage wireless traffic in a distributed Enterprise environment and enable roaming and mobility of wireless endpoints.

Security Services: Including adaptive Network Access Control (NAC), firewalls for zone based macrosegmentation, microsegmentation services, and threat detection solutions all working together in a multilayer approach safeguarding data and users.

Management and Orchestration services: Essential for the configuration and operational management of the aforementioned services, this component offers various interfaces such as Command Line Interface (CLI), Graphical User Interface (GUI), and Application Programming Interfaces (APIs) to enable automated workflows.

Wired Network Service

The architecture of wired campus networks has evolved significantly over recent decades. Initially, a traditional three-tiered model, comprising Access, Distribution, and Core layers, was prevalent. This was primarily a Layer 2 VLAN-based design, with Layer 3 switching concentrated at the Core layers, and Spanning Tree Protocol (STP) was utilized for loop prevention. Subsequent iterations shifted Layer 3 switching to the Distribution layer, aiming to reduce the STP domain. However, this design exhibited scalability, performance, and operational limitations, hindering its applicability in modern campus network environments. The addition of network segments necessitated reconfiguration of the Distribution and Core layers. Furthermore, increasing east-west traffic patterns, driven by device-to-device communication, Internet of Things (IoT), and edge computing, resulted in traffic hairpinning through the Distribution and Core layers, causing performance bottlenecks and inefficient bandwidth utilization. Failures of switches at the Core or Distribution layers could have catastrophic consequences. Although STP prevented loops by blocking redundant links, it also diminished available network bandwidth. Reconvergence events further exacerbated performance degradation by rendering links temporarily unavailable. Mobility, a critical requirement for modern users and IoT devices, posed considerable challenges within the three-tiered design, necessitating manual VLAN adjustments across multiple network layers. Consequently, contemporary campus network deployments are transitioning toward Leaf-Spine and Ethernet VPN-Virtual Extensible Local Area Network (EVPN-VXLAN) architectures to address these aforementioned challenges.

Eliminates STP: Uses Equal-Cost Multi-Path (ECMP) for optimized traffic distribution.

Improved Scalability: Easily expands by adding more Leaf switches without redesigning the entire network.

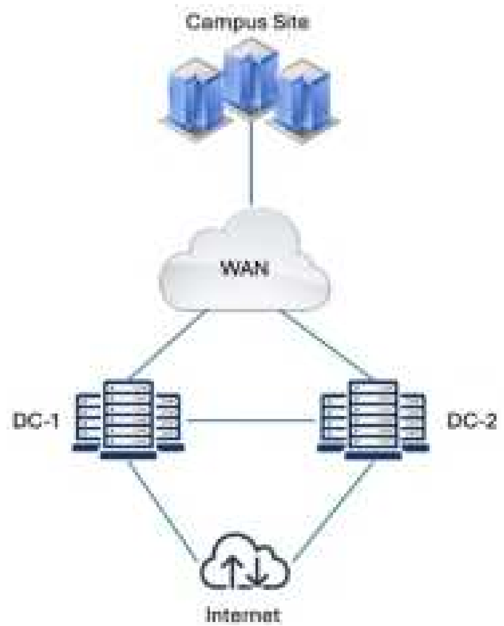
EVPN-VXLAN Overlays: Simplify the configuration of VLANs/subnets and VRFs segmentation stretched across multiple locations to enhance mobility and security for users and IOT devices

Improved Performance: Offers consistent low-latency, high-bandwidth paths between endpoints.

Improved Redundancy: Multiple active paths between devices improve fault tolerance.

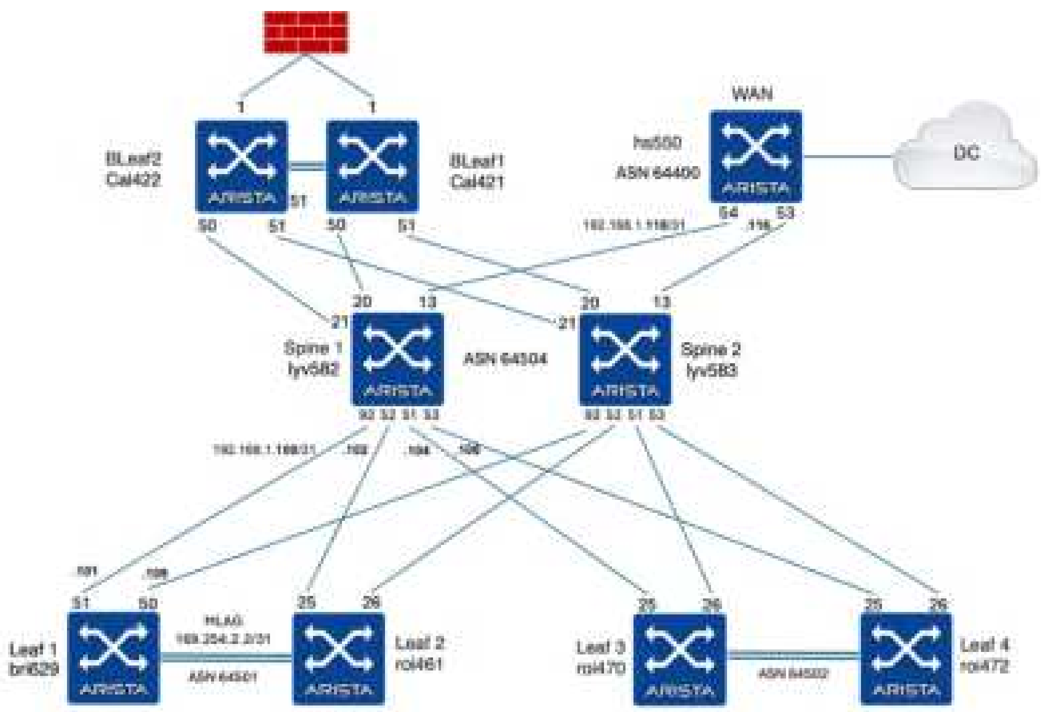
This guide details the design and implementation of a Campus fabric based on a leaf-spine architecture featuring an EVPN-VXLAN overlay.

The sample design also includes redundant datacenters and multiple campus sites, interconnected via a wide area network. The datacenters host essential internal services for campus users and IoT devices. Internet connectivity is centralized and accessible through the datacenters.



This document describes how to build wired, wireless, and integrate security services for a single campus site.

The wired service is based on a leaf-spine network topology, with four Campus leaf switches and two spine switches. The spine switches also connect to the WAN for DC internal services and Internet connectivity. The border leaves connect to service devices such as firewalls.



The initial step involves constructing the IP network , which functions as the underlay for our EVPN/VXLAN overlay Campus fabric carrying all the services. Below, we provide configuration examples for 1 Leaf, 1 Spine, 1 Border Leaf and WAN device within the config blocks. The leaf switches are configured to be MLAG pairs. We are using a /31 subnet to configure IP addresses on the point to point links between the switches. EBGP is configured between the leaf and spine and between spine and WAN devices to set up the underlay and exchange routing information. IBGP is used between the leaf switches.

```
Leaf1 bri629
bri629.09:52:28#show running-config
!
interface Ethernet50
   description lyv583-E92
   mtu 9214
   error-correction encoding fire-code
   no switchport
   ip address 192.168.1.109/31
!
interface Ethernet51
   description lyv582-E92
   mtu 9214
   error-correction encoding reed-solomon
   no switchport
   ip address 192.168.1.101/31
!
interface Ethernet52
   description roi461-E27
   channel-group 2000 mode active
!
interface Loopback0
   description router-id
   ip address 10.135.1.1/32

!
vlan 4094
   name MLAG_PEER
   trunk group MLAG
!
interface Port-Channel2000
   description MLAG_PEER
   switchport mode trunk
   switchport trunk group MLAG
!
interface Vlan4094
   description SVI_MLAG
   mtu 9214
   ip address 169.254.2.2/31
!
mlag configuration
   domain-id C-MLAG1
   local-interface Vlan4094
   peer-address 169.254.2.3
   peer-link Port-Channel2000
   reload-delay mlag 360
   reload-delay non-mlag 300


!
router bgp 64501
   router-id 10.135.1.1
   distance bgp 20 200 200
   maximum-paths 2
   neighbor MLAG-UNDERLAY peer group
   neighbor MLAG-UNDERLAY remote-as 64501
   neighbor MLAG-UNDERLAY description MLAG-PEER
   neighbor MLAG-UNDERLAY maximum-routes 0
   neighbor SPINE-UNDERLAY peer group
   neighbor SPINE-UNDERLAY remote-as 64504
   neighbor SPINE-UNDERLAY send-community
   neighbor SPINE-UNDERLAY maximum-routes 0
   neighbor 169.254.2.3 peer group MLAG-UNDERLAY
   neighbor 192.168.1.100 peer group SPINE-UNDERLAY
   neighbor 192.168.1.108 peer group SPINE-UNDERLAY
   redistribute connected
   !

   address-family ipv4
      neighbor MLAG-UNDERLAY activate
      neighbor SPINE-UNDERLAY activate
   !
   ip routing
```

```
Spine1 lyv582

!
interface Ethernet5/1
   description roi470-E25
   mtu 9214
   speed forced 25gfull
   no switchport
   ip address 192.168.1.104/31
!
interface Ethernet5/2
   description roi461-E25
   mtu 9214
   no switchport
   ip address 192.168.1.102/31
!
interface Ethernet5/3
   description roi472-E25
   mtu 9214
   no switchport
   ip address 192.168.1.106/31
!
interface Ethernet9/2
   description bri629-E51
   mtu 9214
   speed forced 25gfull
   error-correction encoding reed-solomon
   no switchport
   ip address 192.168.1.100/31
!
interface Ethernet13/1
   description hs550-E53/1
   mtu 9214
   speed forced 100gfull
   no switchport
   ip address 192.168.1.117/31
!
interface Ethernet20/1
   description cal421.50/1
   mtu 9214
   no switchport
   ip address 192.168.1.124/31
!


interface Loopback0
   description router-id
   ip address 10.135.1.7/32
!
vlan 4094
   name MLAG_PEER
   trunk group MLAG
!
interface Port-Channel2000
   description MLAG_PEER
   switchport mode trunk
   switchport trunk group MLAG
!
interface Vlan4094
   description SVI_MLAG
   mtu 9214
   ip address 169.254.2.0/31
!
mlag configuration
   domain-id C-MLAG0
   local-interface Vlan4094
   peer-address 169.254.2.1
   peer-link Port-Channel2000
   reload-delay mlag 360
   reload-delay non-mlag 300

!
```

```
router bgp 64504
   router-id 10.135.1.7
   distance bgp 20 200 200
   maximum-paths 2
   bgp bestpath d-path
   neighbor MLAG-UNDERLAY peer group
   neighbor MLAG-UNDERLAY remote-as 64504
   neighbor MLAG-UNDERLAY description MLAG-PEER
   neighbor MLAG-UNDERLAY maximum-routes 0
   neighbor UNDERLAY peer group
   neighbor UNDERLAY send-community
   neighbor UNDERLAY maximum-routes 0

   neighbor 169.254.2.1 peer group MLAG-UNDERLAY
   neighbor 192.168.1.101 peer group UNDERLAY
   neighbor 192.168.1.101 remote-as 64501
   neighbor 192.168.1.101 description Leaf-1
   neighbor 192.168.1.103 peer group UNDERLAY
   neighbor 192.168.1.103 remote-as 64501
   neighbor 192.168.1.103 description Leaf-2
   neighbor 192.168.1.105 peer group UNDERLAY
   neighbor 192.168.1.105 remote-as 64502
   neighbor 192.168.1.105 description Leaf-3
   neighbor 192.168.1.107 peer group UNDERLAY
   neighbor 192.168.1.107 remote-as 64502
   neighbor 192.168.1.107 description Leaf-4
   neighbor 192.168.1.116 peer group UNDERLAY
   neighbor 192.168.1.116 remote-as 64400
   neighbor 192.168.1.116 description Core
   redistribute connected
   !
      address-family ipv4
      neighbor MLAG-UNDERLAY activate
      neighbor UNDERLAY activate
   !
   ip routing
```

```
WAN h550
!
interface Ethernet53/1
   description lyv583-E46/1
   mtu 9214
   no switchport
   ip address 192.168.1.116/31
!
interface Ethernet54/1
   description lyv582-E46/1
   mtu 9214
   no switchport
   ip address 192.168.1.118/31
!
interface Loopback0
   description router-id
   ip address 10.134.1.1/32
!
ip routing
!
router bgp 64400
   router-id 10.134.1.1
   update wait-for-convergence
   update wait-install
   distance bgp 20 200 200
   maximum-paths 2
   bgp bestpath d-path
   neighbor UNDERLAY peer group
   neighbor UNDERLAY send-community
   neighbor UNDERLAY maximum-routes 0
   neighbor 192.168.1.22 peer group UNDERLAY
   neighbor 192.168.1.22 remote-as 64104
   neighbor 192.168.1.22 description Leaf4A
   neighbor 192.168.1.24 peer group UNDERLAY
   neighbor 192.168.1.24 remote-as 64104
```

```
   neighbor 192.168.1.24 description DC-Leaf4B
   neighbor 192.168.1.34 peer group UNDERLAY
   neighbor 192.168.1.34 remote-as 64204
   neighbor 192.168.1.34 description DC-Leaf4
   neighbor 192.168.1.117 peer group UNDERLAY
   neighbor 192.168.1.117 remote-as 64504
   neighbor 192.168.1.117 description CSpine1
   neighbor 192.168.1.119 peer group UNDERLAY
   neighbor 192.168.1.119 remote-as 64504
   neighbor 192.168.1.119 description CSpine2
   redistribute connected
   !
   address-family ipv4
      neighbor UNDERLAY activate
!
```

```
BLeaf1 cal421

!
vlan 1000,1501,2051-2500
!
vrf instance Campus
!
interface Ethernet1
   description to-FW
   speed forced 10000full
   switchport trunk allowed vlan 2405-2410
   switchport mode trunk
   switchport source-interface tx
!
interface Ethernet50/1
   description lyv582-20/1
   mtu 9214
   no switchport
   ip address 192.168.1.125/31
!
interface Ethernet51/1
   description lyv583-20/1
   mtu 9214
   no switchport
   ip address 192.168.1.127/31
!
interface Vlan2405
   description CampusFW
   mtu 9214
   vrf Campus
   ip address virtual 10.240.5.2/24
!
interface Vlan2406
   description CampusFW-ZT
   mtu 9214
   vrf Cmps-FWZT
   ip address virtual 10.240.6.2/24
!

!
ip routing
ip routing vrf Campus
!
!
router bgp 64505
   router-id 10.135.1.5
   distance bgp 20 200 200
   maximum-paths 2
   neighbor SPINE-UNDERLAY peer group
   neighbor SPINE-UNDERLAY remote-as 64504
   neighbor SPINE-UNDERLAY send-community
   neighbor SPINE-UNDERLAY maximum-routes 0
   neighbor 192.168.1.124 peer group SPINE-UNDERLAY
   neighbor 192.168.1.124 description Spine-1
   neighbor 192.168.1.126 peer group SPINE-UNDERLAY
```

```
   neighbor 192.168.1.126 description Spine-2
   redistribute connected
   !
   !
   address-family ipv4
      neighbor SPINE-UNDERLAY activate
   !
```

With the IP fabric established and BGP configured in the underlay, the subsequent phase involves implementing the EVPN overlay between the leaf and spine nodes, as well as between the spine and core routers. This will be followed by the integration of VXLAN configurations alongside the VLANs designated for wired and wireless endpoint connectivity.

```
Leaf
!
vlan 1000,1501,2051-2500
!
interface Loopback1
   description vxlan-source-intf
   ip address 10.135.2.1/32
!
vrf instance Campus

!
interface Vxlan1
   vxlan source-interface Loopback0
   vxlan virtual-router encapsulation mac-address mlag-system-id
   vxlan udp-port 4789
   vxlan vlan 1000,2051-2500 vni 11000,12051-12500
   vxlan vrf Campus vni 555888
   vxlan mlag source-interface Loopback1
!
ip routing vrf Campus

router bgp 64501
   router-id 10.135.1.1
   distance bgp 20 200 200
   maximum-paths 2
   neighbor EVPN-OVERLAY peer group
   neighbor EVPN-OVERLAY remote-as 64504
   neighbor EVPN-OVERLAY update-source Loopback0
   neighbor EVPN-OVERLAY ebgp-multihop
   neighbor EVPN-OVERLAY send-community
   neighbor EVPN-OVERLAY maximum-routes 0
   neighbor 10.135.1.7 peer group EVPN-OVERLAY
   neighbor 10.135.1.7 description Spine-1
   neighbor 10.135.1.8 peer group EVPN-OVERLAY
   neighbor 10.135.1.8 description Spine-2
   redistribute connected
   !
   vlan 1000
      rd 1.135.1.1:64501
      route-target both 1000:1000
      redistribute learned
      redistribute dot1x
   !
   vlan-aware-bundle C-VLANS
      rd 10.135.1.1:64501
      route-target both 2051:2400
      redistribute learned
      vlan 2051-2400
   !
   vlan-aware-bundle W-VLANS
      rd 100.135.1.1:64501
      route-target both 2401:2500
      redistribute learned
      vlan 2401-2500
   !
   address-family evpn
      neighbor EVPN-OVERLAY activate
```

```
      no neighbor SPINE-UNDERLAY activate

   !
   vrf Campus
      rd 10.135.1.1:64501
      route-target import evpn 5:5
      route-target import evpn 10:10
      route-target export evpn 5:5
      redistribute connected
!
```

```
Spine
!
vlan 1000,1501,2051-2500
!
vrf instance Campus

!
interface Loopback1
   description vxlan-source-intf
   ip address 10.135.2.4/32
!
interface Vxlan1
   vxlan source-interface Loopback1
   vxlan virtual-router encapsulation mac-address mlag-system-id
   vxlan udp-port 4789
   vxlan vlan 1000,2051-2500 vni 11000,12051-12500
   vxlan vrf Campus vni 555888

!
ip routing vrf Campus
!

router bgp 64504
   router-id 10.135.1.7
   distance bgp 20 200 200
   maximum-paths 2
   bgp bestpath d-path
   neighbor EVPN-OVERLAY peer group
   neighbor EVPN-OVERLAY update-source Loopback0
   neighbor EVPN-OVERLAY ebgp-multihop
   neighbor EVPN-OVERLAY send-community
   neighbor EVPN-OVERLAY maximum-routes 0
   neighbor GW-EVPN-OVERLAY peer group
   neighbor GW-EVPN-OVERLAY remote-as 64400
   neighbor GW-EVPN-OVERLAY update-source Loopback0
   neighbor GW-EVPN-OVERLAY ebgp-multihop
   neighbor GW-EVPN-OVERLAY send-community extended
   neighbor GW-EVPN-OVERLAY maximum-routes 0
   neighbor 10.134.1.1 peer group GW-EVPN-OVERLAY
   neighbor 10.134.1.1 description Core
   neighbor 10.135.1.1 peer group EVPN-OVERLAY
   neighbor 10.135.1.1 remote-as 64501
   neighbor 10.135.1.1 description Leaf-1
   neighbor 10.135.1.2 peer group EVPN-OVERLAY
   neighbor 10.135.1.2 remote-as 64501
   neighbor 10.135.1.2 description Leaf-2
   neighbor 10.135.1.3 peer group EVPN-OVERLAY
   neighbor 10.135.1.3 remote-as 64502
   neighbor 10.135.1.3 description Leaf-3
   neighbor 10.135.1.4 peer group EVPN-OVERLAY
   neighbor 10.135.1.4 remote-as 64502
   neighbor 10.135.1.4 description Leaf-4
   !
   !
   vlan 1000
      rd evpn domain all 1.135.1.7:64504
      route-target both 1000:1000
      route-target import export evpn domain remote 10000:10000
      redistribute learned
   !
   vlan-aware-bundle C-VLANS
```

```
      rd 10.135.1.7:64504
      route-target both 2051:2400
      redistribute learned
      vlan 2051-2400
   !
   vlan-aware-bundle W-VLANS
      rd 100.135.1.7:64504
      route-target both 2401:2500
      redistribute learned
      vlan 2401-2500
   !
   address-family evpn
      neighbor EVPN-OVERLAY activate
      neighbor GW-EVPN-OVERLAY activate
      neighbor GW-EVPN-OVERLAY domain remote
      neighbor IXIA-Overlay activate
      domain identifier 5:5
      neighbor default next-hop-self received-evpn-routes route-type ip-prefix inter-domain
   !
   address-family ipv4
      neighbor MLAG-UNDERLAY activate
      neighbor UNDERLAY activate
   !
   vrf Campus
      rd 10.135.1.7:64504
      route-target import evpn 5:5
      route-target import evpn 10:10
      route-target export evpn 5:5
      redistribute connected
   !
```

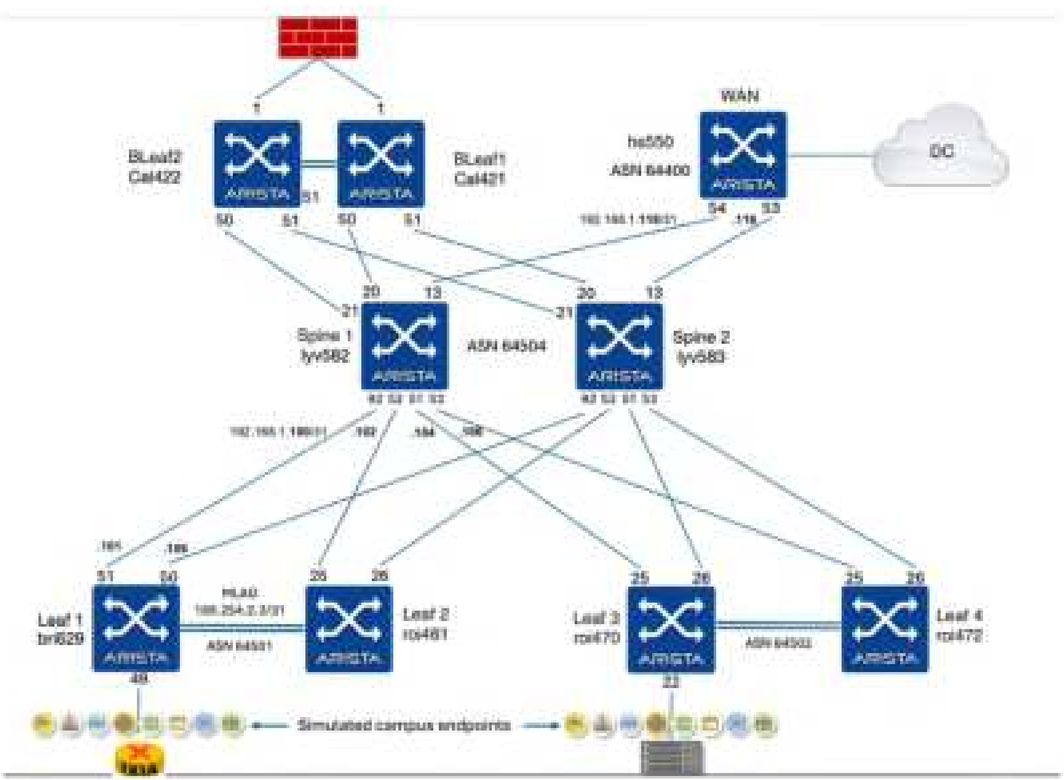
WAN

```
!
router bgp 64400
   neighbor GW-EVPN-OVERLAY peer group
   neighbor GW-EVPN-OVERLAY update-source Loopback0
   neighbor GW-EVPN-OVERLAY ebgp-multihop
   neighbor GW-EVPN-OVERLAY send-community extended
   neighbor GW-EVPN-OVERLAY maximum-routes 0
   neighbor 10.131.1.7 peer group GW-EVPN-OVERLAY
   neighbor 10.131.1.7 remote-as 64104
   neighbor 10.131.1.7 description DC-1-Leaf4A
   neighbor 10.131.1.8 peer group GW-EVPN-OVERLAY
   neighbor 10.131.1.8 remote-as 64104
   neighbor 10.131.1.8 description DC-1-Leaf4B
   neighbor 10.135.1.7 peer group GW-EVPN-OVERLAY
   neighbor 10.135.1.7 remote-as 64504
   neighbor 10.135.1.7 description CSpine1
   neighbor 10.135.1.8 peer group GW-EVPN-OVERLAY
   neighbor 10.135.1.8 remote-as 64504
   neighbor 10.135.1.8 description CSpine2

   !
   address-family evpn
      bgp next-hop-unchanged
      neighbor GW-EVPN-OVERLAY activate
      neighbor GW-EVPN-OVERLAY domain remote
      domain identifier 3:3
   !
```

Bleaf

```
!
interface Vxlan1
   vxlan source-interface Loopback0
   vxlan virtual-router encapsulation mac-address mlag-system-id
   vxlan udp-port 4789
   vxlan vlan 1000,2051-2500 vni 11000,12051-12500
```

```
   vxlan vrf Campus vni 555888
   vxlan vrf Cmps-FWZT vni 555666
!
router bgp 64505
   router-id 10.135.1.5
   distance bgp 20 200 200
   maximum-paths 2
   neighbor EVPN-OVERLAY peer group
   neighbor EVPN-OVERLAY remote-as 64504
   neighbor EVPN-OVERLAY update-source Loopback0
   neighbor EVPN-OVERLAY ebgp-multihop
   neighbor EVPN-OVERLAY send-community
   neighbor EVPN-OVERLAY maximum-routes 0
   neighbor 10.135.1.7 peer group EVPN-OVERLAY
   neighbor 10.135.1.7 description Spine-1
   neighbor 10.135.1.8 peer group EVPN-OVERLAY
   neighbor 10.135.1.8 description Spine-2
   !
   vlan-aware-bundle C-VLANS
      rd 10.135.1.5:64505
      route-target both 2051:2400
      redistribute learned
      vlan 2051-2400
   !
   vlan-aware-bundle W-VLANS
      rd 100.135.1.5:64505
      route-target both 2401:2500
      redistribute learned
      vlan 2401-2500
   !
   address-family evpn
      neighbor EVPN-OVERLAY activate
   !
   vrf Campus
      rd 10.135.1.5:64505
      route-target import evpn 5:5
      route-target import evpn 10:10
      route-target export evpn 5:5
      redistribute connected
   !
```



The wired endpoints within our environment are simulated with VMs and a Traffic generator (Ixia). The virtualized server has a connection to leaf switch roi470, while the Ixia port connects to leaf switch bri629. Additionally, we will implement SVI configurations for all VLANs across all leaf switches to facilitate a distributed gateway.

```
Leaf

bri629

!
interface Ethernet49
   description ixs342-2
   switchport mode trunk
!
!
interface Vlan1501
   ip address virtual 192.168.151.1/24
!
interface Vlan2064
   description Ixia-External-Services
   vrf Campus
   ip address virtual 90.90.64.1/20
!
interface Vlan2080
   description Ixia-DGroup1
   vrf Campus
   ip address virtual 10.80.0.1/16
!
interface Vlan2081
   description Ixia-DGroup3
   vrf Campus
   ip address virtual 10.81.0.1/16
!
interface Vlan2082
   description Ixia-DGroup4
   vrf Campus
   ip address virtual 10.82.0.1/16
!
interface Vlan2083
   description Ixia-DGroup5
   vrf Campus
   ip address virtual 10.83.0.1/16
!
interface Vlan2084
   description Ixia-DGroup6
   vrf Campus
   ip address virtual 10.84.0.1/16
!
interface Vlan2090
   description Ixia-DGroup2-Users
   vrf Campus
   ip address virtual 10.90.0.1/16
!
interface Vlan2099
   description Ixia-Internal-Services
   vrf Campus
   ip address virtual 10.99.0.1/24
!
interface Vlan2403
   description esxiVM-1-4
   mtu 9214
   vrf Campus
   ip address virtual 10.243.0.1/19
!
interface Vlan2404
   description esxiVM-5-8
   mtu 9214
   vrf Campus
   ip address virtual 10.244.0.1/19
!

roi470
```

```
!
interface Ethernet22
   description poc-srv-51.vmnic1
   switchport mode trunk

!
interface Vlan1501
   ip address virtual 192.168.151.1/24
!
interface Vlan2064
   description Ixia-External-Services
   vrf Campus
   ip address virtual 90.90.64.1/20
!
interface Vlan2080
   description Ixia-DGroup1
   vrf Campus
   ip address virtual 10.80.0.1/16
!
interface Vlan2081
   description Ixia-DGroup3
   vrf Campus
   ip address virtual 10.81.0.1/16
!
interface Vlan2082
   description Ixia-DGroup4
   vrf Campus
   ip address virtual 10.82.0.1/16
!
interface Vlan2083
   description Ixia-DGroup5
   vrf Campus
   ip address virtual 10.83.0.1/16
!
interface Vlan2084
   description Ixia-DGroup6
   vrf Campus
   ip address virtual 10.84.0.1/16
!
interface Vlan2090
   description Ixia-DGroup2-Users
   vrf Campus
   ip address virtual 10.90.0.1/16
!
interface Vlan2099
   description Ixia-Internal-Services
   vrf Campus
   ip address virtual 10.99.0.1/24
!
interface Vlan2403
   description esxiVM-1-4
   mtu 9214
   vrf Campus
   ip address virtual 10.243.0.1/19
!
interface Vlan2404
   description esxiVM-5-8
   mtu 9214
   vrf Campus
   ip address virtual 10.244.0.1/19
!
```

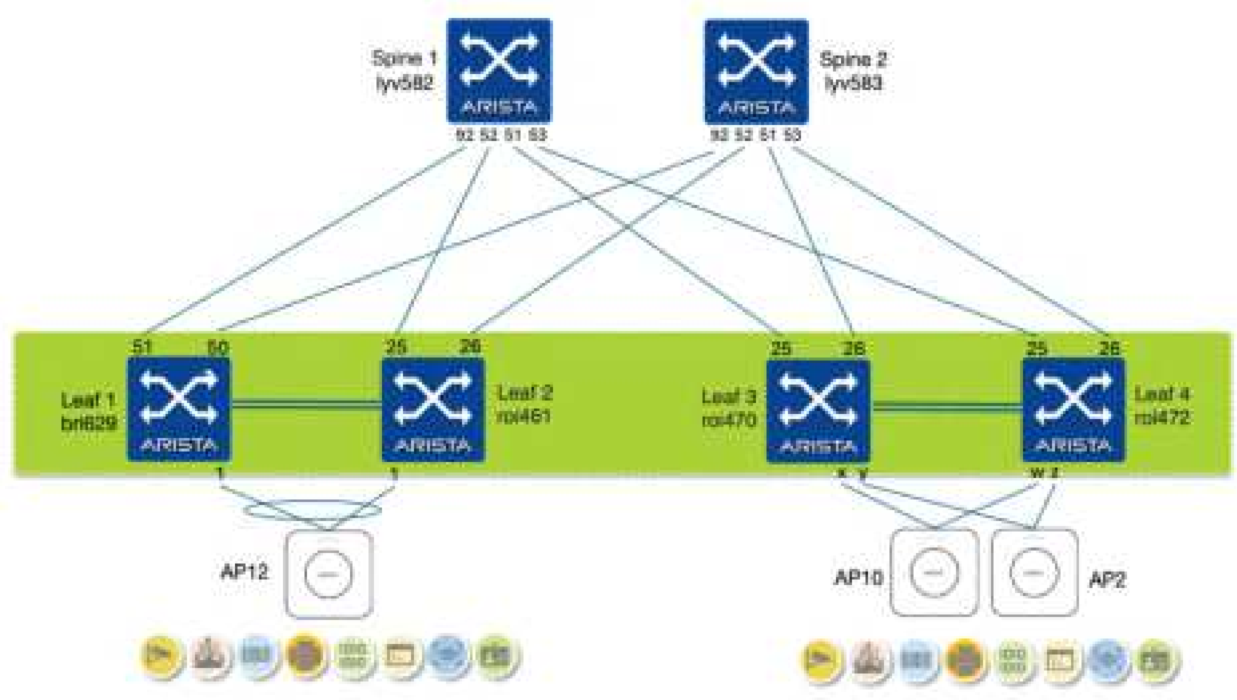
At this point we have a fully functional EVPN-VXLAN fabric that provides connectivity to the wired endpoints. Next we will onboard wireless endpoints

Wireless Network Service

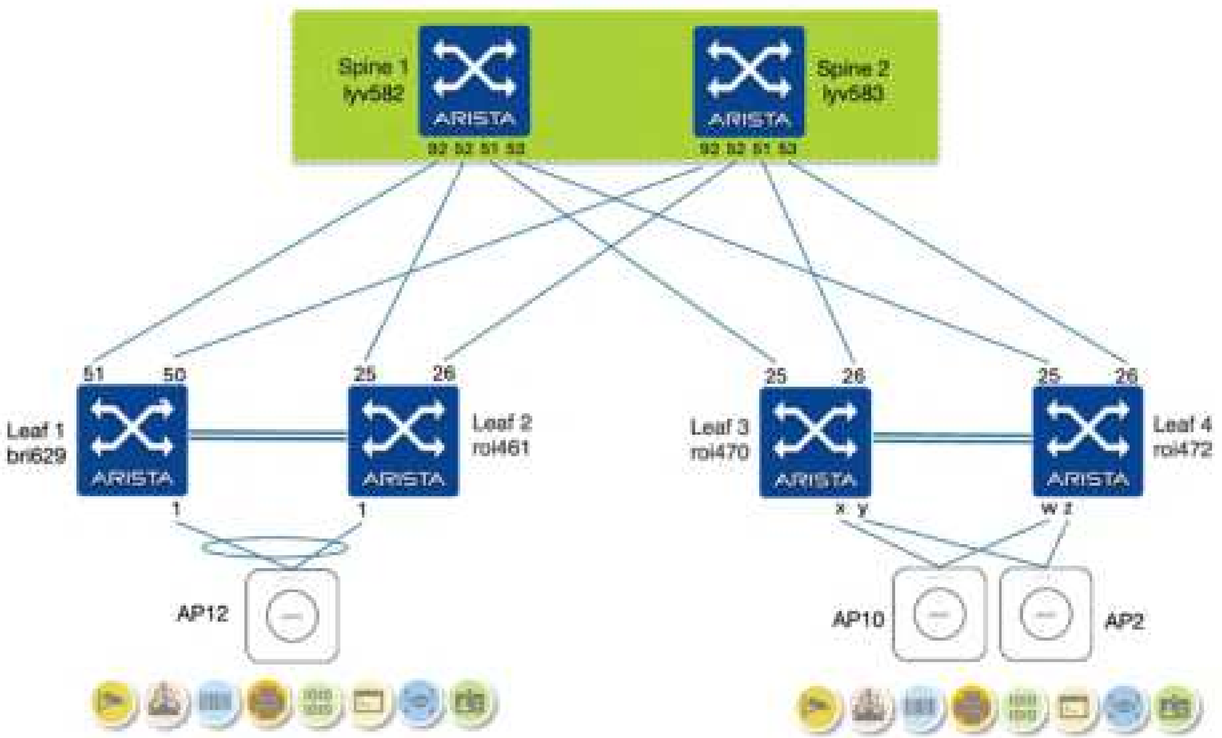
The wireless endpoint can be onboarded onto the wired network using the wireless network service. Endpoints connect to the Access Points (AP) using the SSID advertised by the APs, which are connected to the wired network. With Arista Wi-Fi the wireless traffic handoff from the APs to the wired network happens in a two ways:

- Bridged mode
- Tunneled mode

In bridged mode, the AP has an 802.1Q trunk to the leaf switch it connects to; the trunk includes all the vlan tags assigned to the wireless endpoints. A vlan handoff takes place for the wireless traffic, and as a result, the leaf switch can perform local switching for the wireless traffic (or route the traffic between the wireless vlans if it's the gateway). However, each leaf switch will need to have all the wireless vlans defined.



In Tunneled mode, a VXLAN tunnel is created between the AP and a centralized switch, usually at the Spine/Aggregation layer. This means that the Campus leaf switches connected to the AP only need to participate in underlay connectivity for the tunnel and do not need the wireless VLANs. This approach consolidates the wireless VLANs at the Campus Spine layer, simplifying the Campus leaf configuration.



Our deployment example will use bridged mode and define the wireless VLANs on all leaf switches.

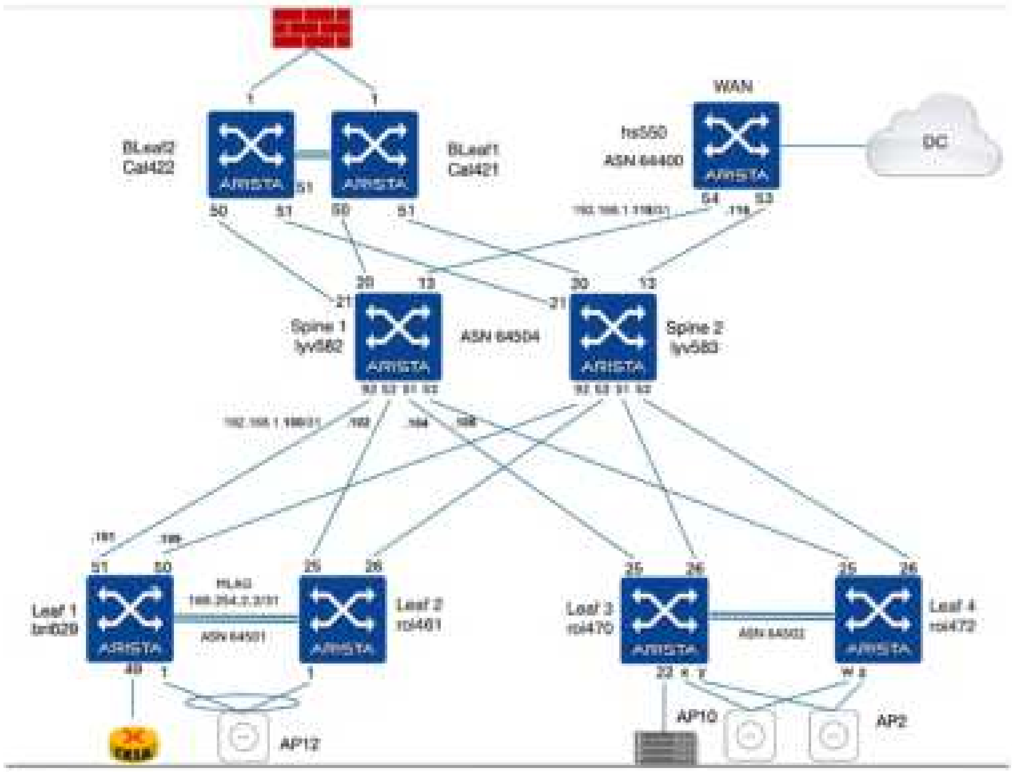
To ensure there is no local switching happening at the access point and all traffic is sent to the leaf switches, we need to configure Layer 2 Traffic Inspection and Filtering (L2TIF) for each SSID on the access point.

Access Point Configuration

In our network architecture, we've chosen to use CloudVision CUE as the management platform for our Access Points (APs). Before these APs can be effectively managed by CloudVision, they need to be assigned a management IP address. This process begins as soon as the APs are powered on and they start sending out untagged DHCP requests in search of an IP address.

As we've established earlier in our design, we're operating in bridged mode. This necessitates the configuration of the interface connecting the AP to the switch as an 801.1q trunk port. To accommodate the untagged DHCP requests from the APs, we need to designate a native VLAN (VLAN 1000 in this example) on the trunk port. It's crucial to ensure that VLAN 1000 has a route to a DHCP server to fulfill these DHCP requests. In our specific deployment scenario, the DHCP server resides in a virtual machine hosted in the datacenter.

Therefore, we need to configure our network to ensure that DHCP requests originating from the APs on VLAN 1000 can reach the DHCP server in the datacenter. Once these configurations are in place, the APs should be able to obtain IP addresses via DHCP, allowing them to be onboarded and managed by CloudVision.

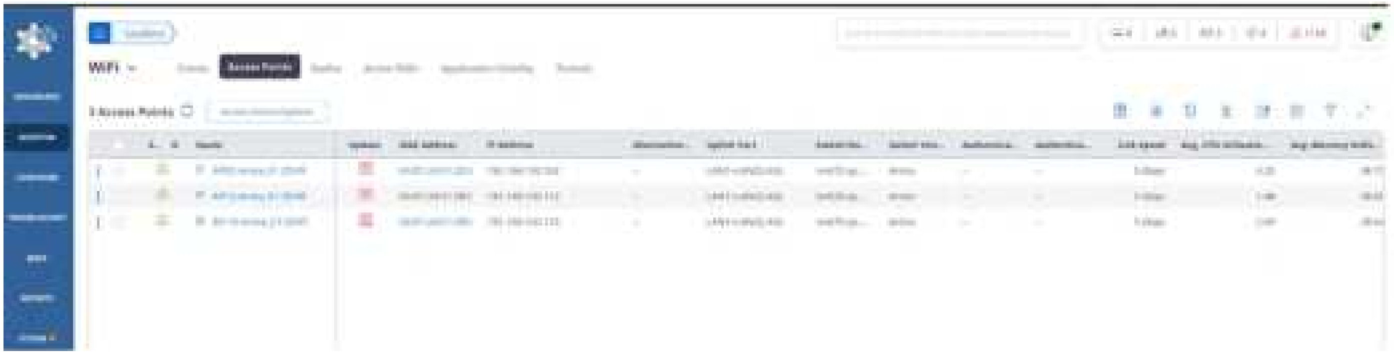



```
Leaf bri629
!
vrf instance ap-mgmt

!
interface Ethernet1
   description AP12
   switchport trunk native vlan 1000
   switchport mode trunk
   switchport source-interface tx
   channel-group 1 mode active
!
interface Port-Channel1
   description AP12
   switchport trunk native vlan 1000
   switchport mode trunk
   switchport source-interface tx
   port-channel lacp fallback individual
   port-channel lacp fallback timeout 5
   mlag 1
!
!
interface Vlan1000
   description l3-if-AP-NoDHCP_NoGateway
   vrf ap-mgmt
   ip dhcp relay all-subnets
   ip address virtual 192.168.100.1/24
!

!
interface Vxlan1
   vxlan vrf ap-mgmt vni 55555
!
ip routing vrf ap-mgmt
!
router bgp 64501

   !
   vlan 1000
```

```
      rd 1.135.1.1:64501
      route-target both 1000:1000
      redistribute learned
      redistribute dot1x

   !
   vrf ap-mgmt
      rd 55.135.1.1:64501
      route-target import evpn 55:55
      route-target export evpn 55:55
      redistribute connected
!
```

Successful onboarding requires connectivity between the AP management IP and CloudVision, which is cloud-based in our deployment example. This can be achieved by defining a NAT router in the network to convert the AP's internal management IP to a routable IP.

Once these steps are completed, we should see that the APs are successfully up and running in CloudVison



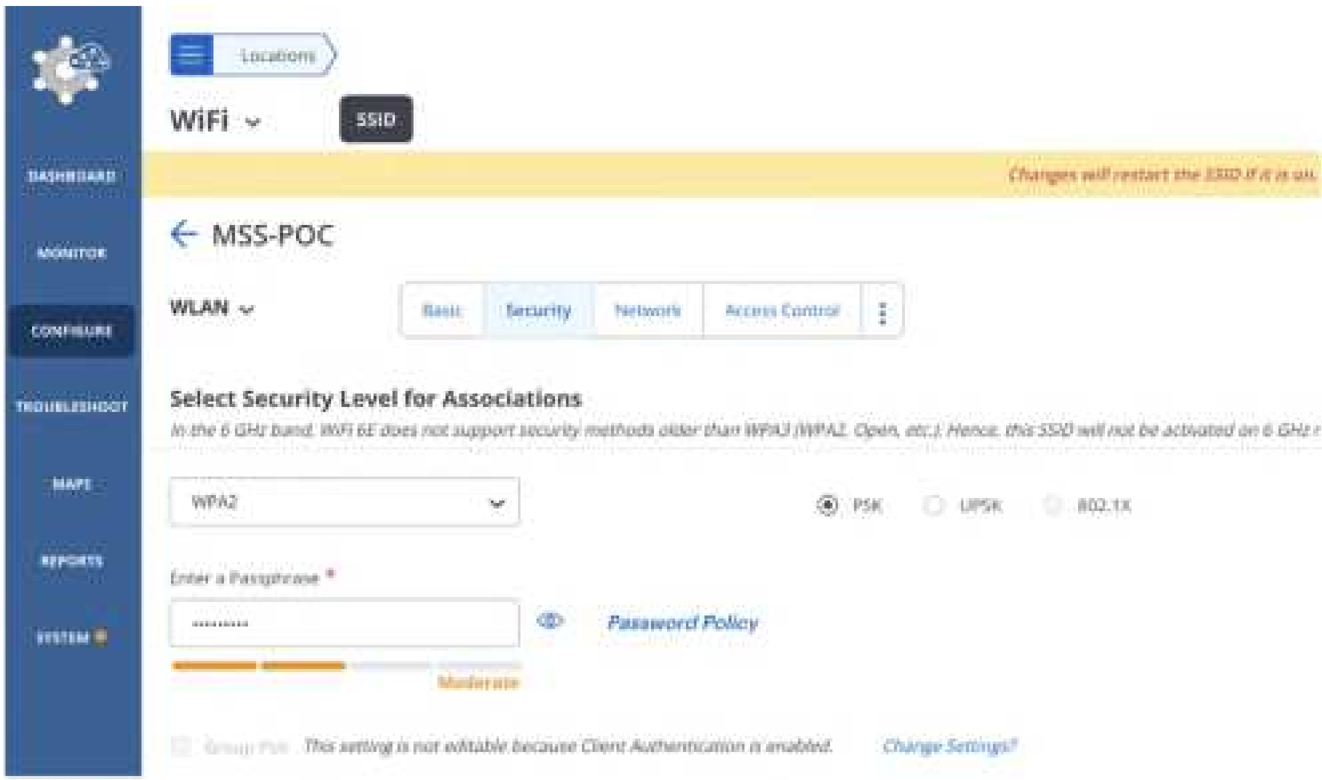
Device configuration

For uplink redundancy we have configured AP interfaces connecting to a pair of leaf switches as a LAG

SSID configuration

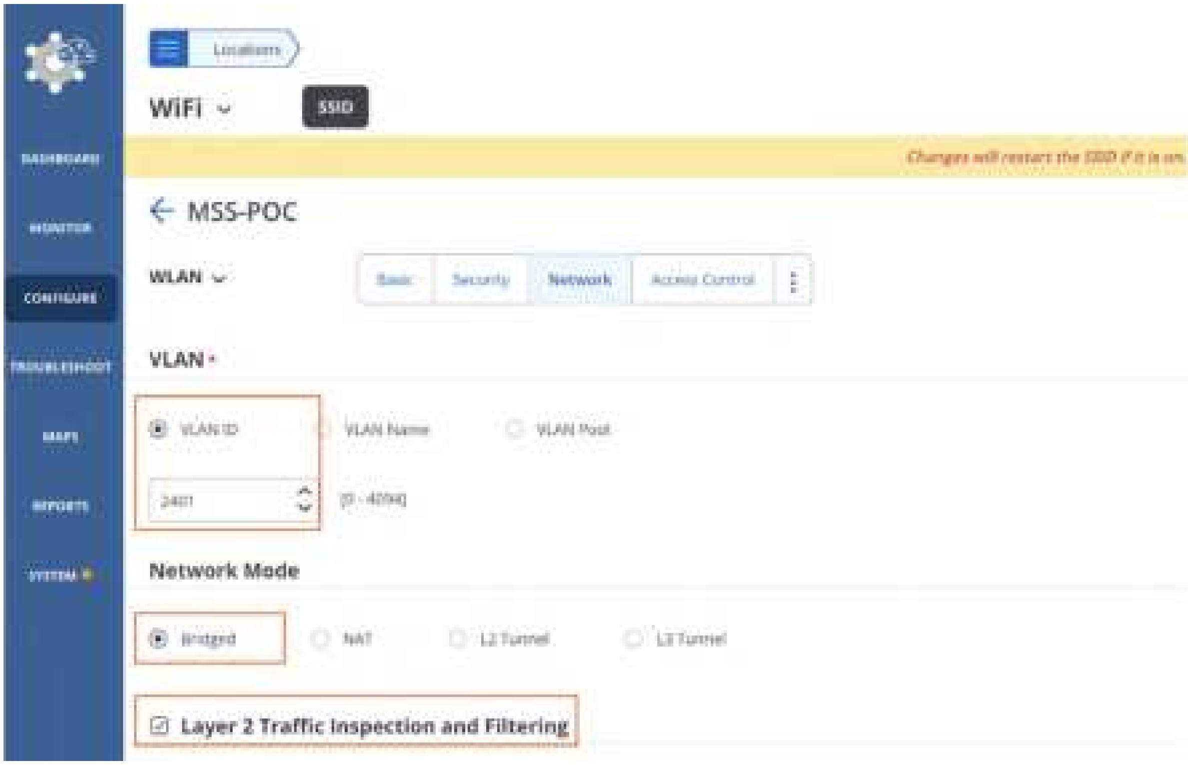
Once we establish the connectivity for the AP itself, next step is to define our SSID. This is where we specify the bridged/tunnel mode as well as configuration for access control

Here we have defined a couple of SSIDs. Let's take a closer look at the configuration



Under the **Security** tab, you can define the access details for the SSID.

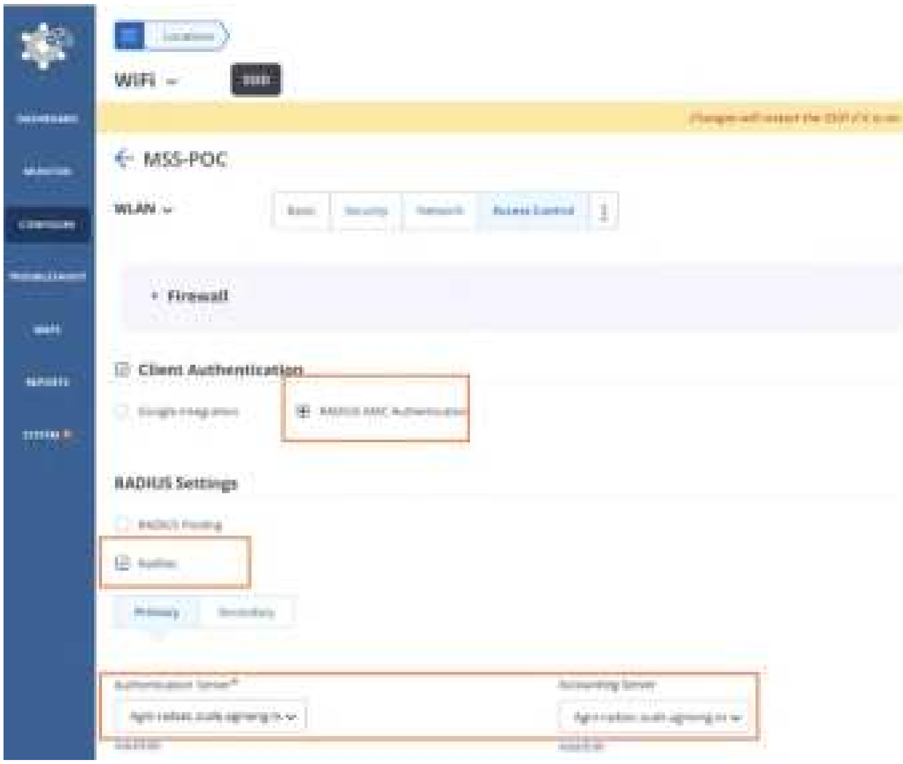
Under the **Network** tab, define the following parameters



As mentioned earlier, we have specified bridged mode. Here's where you can specify the VLAN ID for the SSID. In our case, VLAN 2401 is used (VLAN 2402 for MSS-POC2). Additionally, Layer 2 Traffic Inspection and Filtering has been enabled to prevent local switching on the AP. This ensures all traffic reaches the leaf switches, allowing for segmentation enforcement. These VLANs have already been added on the switch. Next, we will add the SVI configuration.

```
Leaf

!
interface Vlan2401
   description RaspberryPi-1-4
   mtu 9214
   vrf Campus
   ip address virtual 10.241.0.1/19
!
interface Vlan2402
   description RaspberryPi-5-8
   mtu 9214
   vrf Campus
   ip address virtual 10.242.0.1/19
!
```

Under the **Access Control** tab, we need to configure the following parameters



TheAPs will be responsible for authenticating wireless endpoints using the 802.1x protocol. In this setup, CloudVision AGNI will function as the RADIUS server, handling the Authentication, Authorization, and Accounting (AAA) for the network.

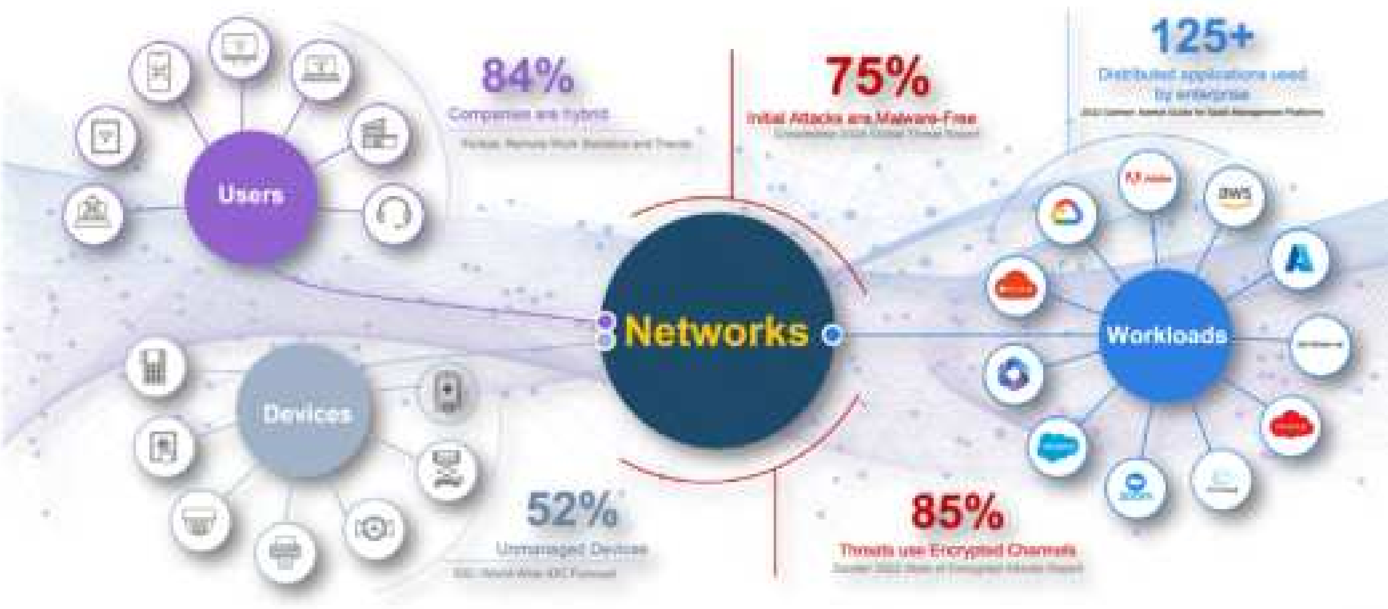
The communication between the APs and the CloudVision AGNI RADIUS server will be secured using Radius over TLS (RadSec), ensuring that sensitive authentication information is encrypted during transmission.

Security Services

The shift towards hybrid work models has significantly impacted the enterprise network security landscape. With employees accessing the network from diverse locations such as homes, cafes, and public transit, enterprise data is now exposed to a wider and constantly evolving threat landscape. This increased vulnerability is further compounded by the widespread adoption of BYOD (Bring Your Own Device) policies, IoT devices, and cloud applications, leading to a substantial rise in unmanaged assets and a reduced visibility into the actual attack surface.

This fundamental change in the way users interact with the network has rendered the traditional "castle and moat" security approach obsolete. This approach, which focused on fortifying the network perimeter and assuming everything within it was safe, is no longer effective in today's decentralized and distributed work environment. To adequately protect enterprise assets in this new paradigm, a more nuanced and dynamic approach is required.

The concept of micro-perimeters emerges as a solution to address these challenges. Instead of relying on a single, rigid network perimeter, micro-perimeters are created around individual assets or groups of assets based on their identity and context. This allows for granular and adaptive security policies that can be tailored to the specific needs of each asset, regardless of its location within the network. By shifting the focus from location-based security to identity-based security, enterprises can better protect their data and assets in the face of an increasingly complex and dynamic threat landscape.



Zero Trust Principles

The Cybersecurity and Infrastructure Security Agency (CISA) has developed a Zero Trust Maturity Model that offers a structured framework and actionable guidance to both government agencies and private sector organizations. This model is designed to assist these entities in progressively adopting and implementing Zero Trust security principles, ultimately achieving a robust Zero Trust security posture. The Zero Trust model is predicated on the concept of "never trust, always verify," which necessitates continuous validation of trust for every user, device, and transaction within a network, regardless of its origin or perceived trustworthiness. By implementing the Zero Trust Maturity Model, organizations can significantly enhance their overall security posture, mitigate the risk of cyberattacks, and safeguard their critical assets and data.

It identifies 3 main principles to attain Zero Trust

- Minimize Lateral Movement
- Never Trust
- Always Verify

Arista Zero Trust Networking Solution

The three key principles of Zero Trust are implemented throughout Arista's ZTN solution:

1. Arista Guardian for Network Identity or AGNI is responsible for authorizing (i.e. never trust) Campus endpoint access to the network and continuously verifying their posture. As part of the access control capabilities AGNI also manages the identity-based microperimeters necessary for Arista Multi-domain Segmentation Services (MSS).
2. Arista Multi-domain Segmentation Services or MSS is responsible for defining the microperimeters segmentation policies to minimize lateral movement. Furthermore MSS provides granular and stateful conversation visibility between microperimeters to recommend segmentation rules to explicitly permit only trusted traffic and to verify the impact of segmentation rule with session level visibility of traffic matching specific policies (MSS can effectively provide Firewall-like visibility for all the lateral (intra-zone) traffic that is not inspected by a Firewall).
3. Arista Network Detection and Response or NDR is responsible for continuously monitoring the traffic permitted by MSS both north-south and east-west and use AI to correlate the traffic against an ever evolving database of adversarial models based on the Mitre Attack Framework (https://attack.mitre.org/). Arista NDR is capable of detecting a large number of complex attacks, developed over a long period of time, like exfiltration, phishing

The picture below shows the ZTN services along with the many external integrations to manage identity and posture.

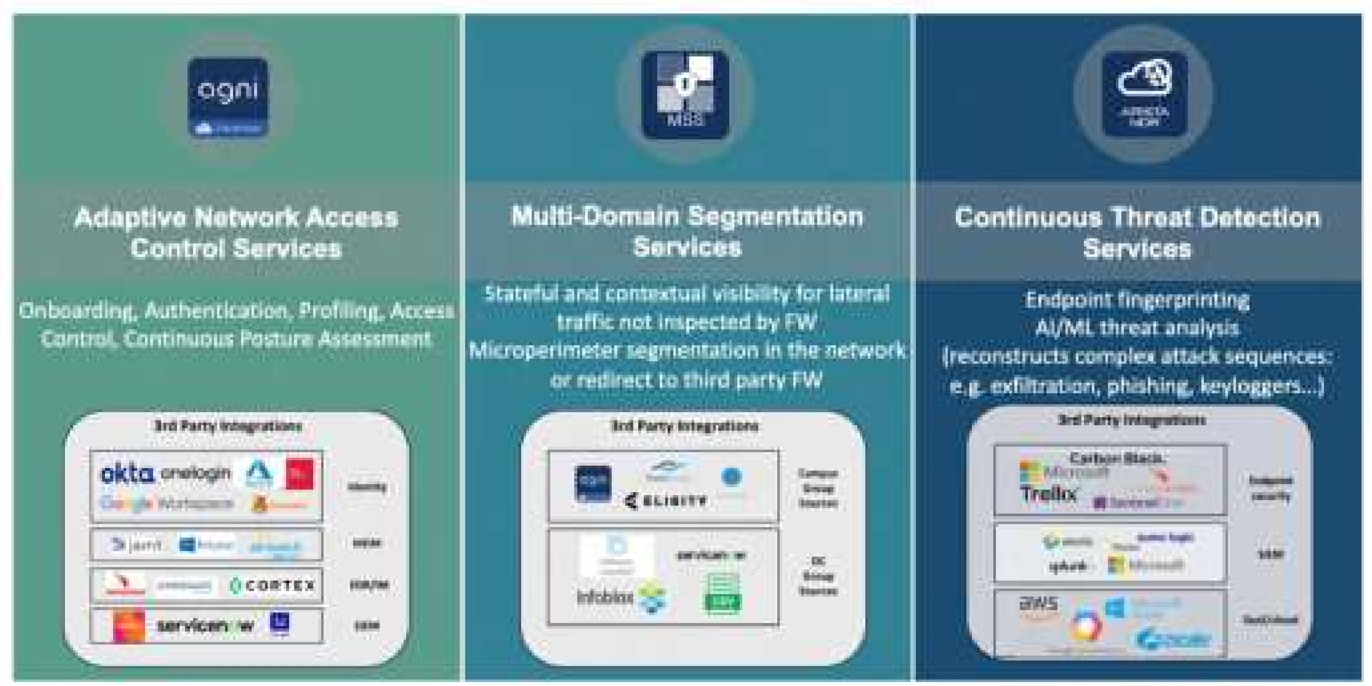
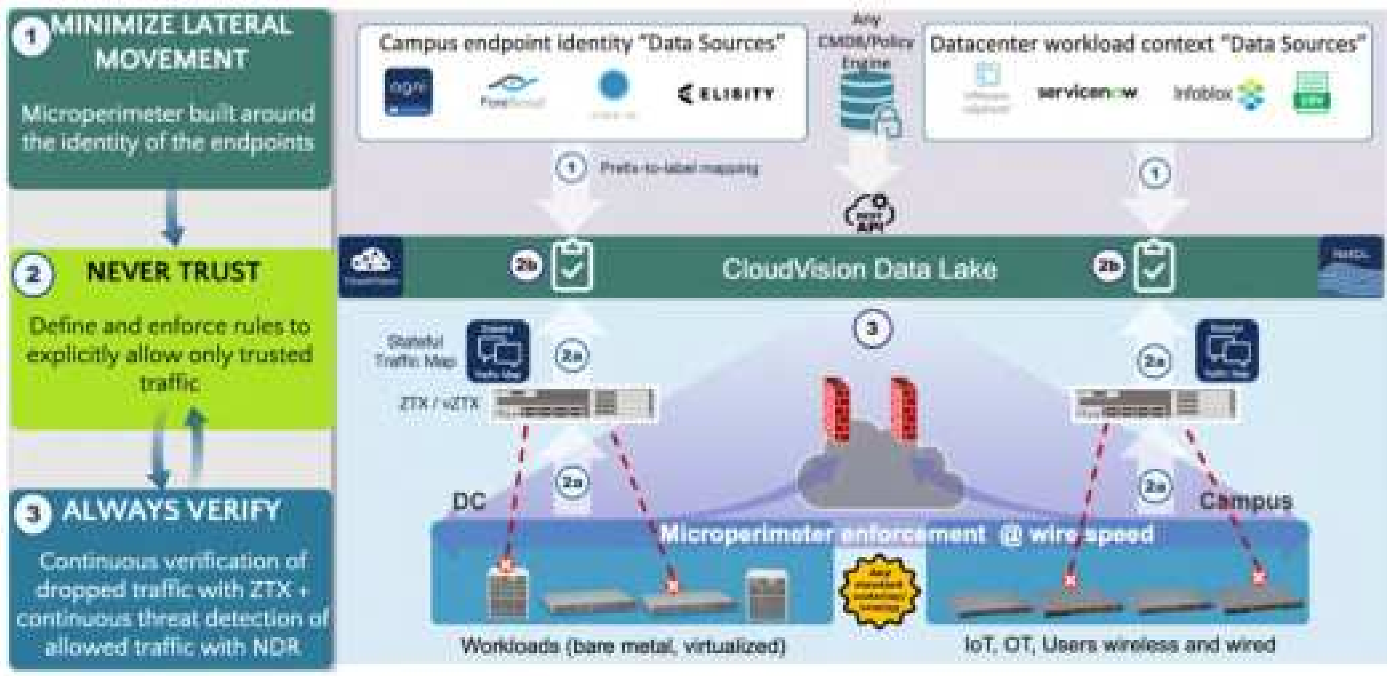


Figure below summarizes how the three key different ZTN services come together to deliver the ZTN principles

Zero Trust Principles of Arista Multi-Domain Segmentation

Arista MSS implements a set of services to iteratively achieve Zero Trust Segmentation Policies and continuously adapt these policies to changes in the population of endpoints and microperimeters and to adapt to changes in application traffic.

This section will discuss how Arista MSS minimizes lateral movement with dynamic (or locally defined) microperimeters, defines segmentation rules to explicitly allow only trusted traffic (never trust) and continuously verifies changes to the traffic to adapt the zero trust policies (always verify).



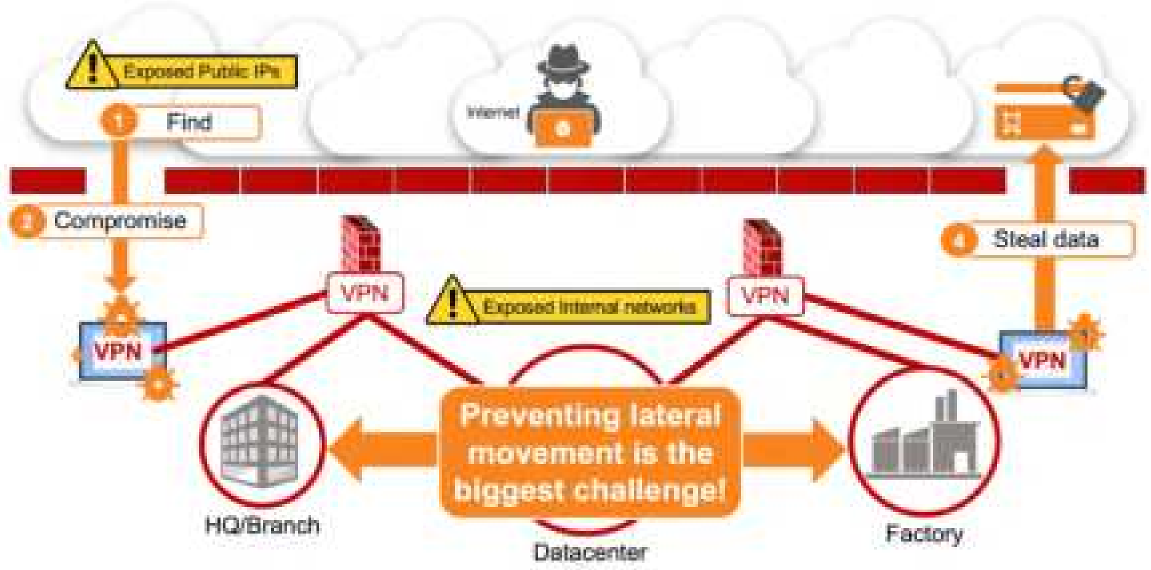
Minimize Lateral Movement

Firewalls and Endpoint Detection and Response (EDR) platforms are essential components of network security, designed to detect and block cyberattacks. However, the sophistication and persistence of modern cyber threats mean that breaches are increasingly inevitable. Attackers continually evolve their tactics, rendering traditional defenses less effective.

Once a network is breached, the attacker's objective often shifts from mere access to lateral movement within the network. This involves compromising additional systems and accounts to expand their control and reach high-value assets such as sensitive data, intellectual property, and critical infrastructure. Lateral movement allows attackers to evade detection, maintain persistence, and inflict significant damage.

To counter these advanced threats, organizations must adopt a multi-layered defense strategy that goes beyond perimeter protection. This includes implementing zero trust principles, network segmentation, continuous monitoring, and advanced threat detection and response capabilities. By assuming that breaches are inevitable and focusing on limiting the impact of attacks, organizations can enhance their resilience and protect their critical assets.

Let's consider the example below.



Attackers exploit exposed public IPs to infiltrate and compromise trusted end-user devices. Given the continuous development and increasing sophistication of cyberattacks, the probability of successful breaches is significant. Upon gaining access to a trusted device within the enterprise, threat actors can then execute lateral movement to access sensitive data, potentially leading to ransom demands.

A critical aspect of an effective cybersecurity strategy involves containing these breaches by restricting lateral movement, thereby mitigating potential crises. The establishment of an identity-based microperimeter is paramount in limiting such lateral movement, as will be detailed in this document. While firewalls and Endpoint Detection and Response (EDR) platforms serve as essential network security components to prevent cyberattacks, the growing sophistication of threats renders breaches unavoidable. Attackers typically seek to achieve lateral movement within an organization to target high-value assets and sensitive information, rather than merely gaining initial network access.

A microperimeter is a security concept that focuses on creating granular access controls around specific resources or assets. It is identity-based, meaning that access is granted or denied based on the user's identity and their associated privileges. By implementing microperimeters, organizations can limit lateral movement by restricting access to sensitive areas of the network based on user roles and responsibilities.

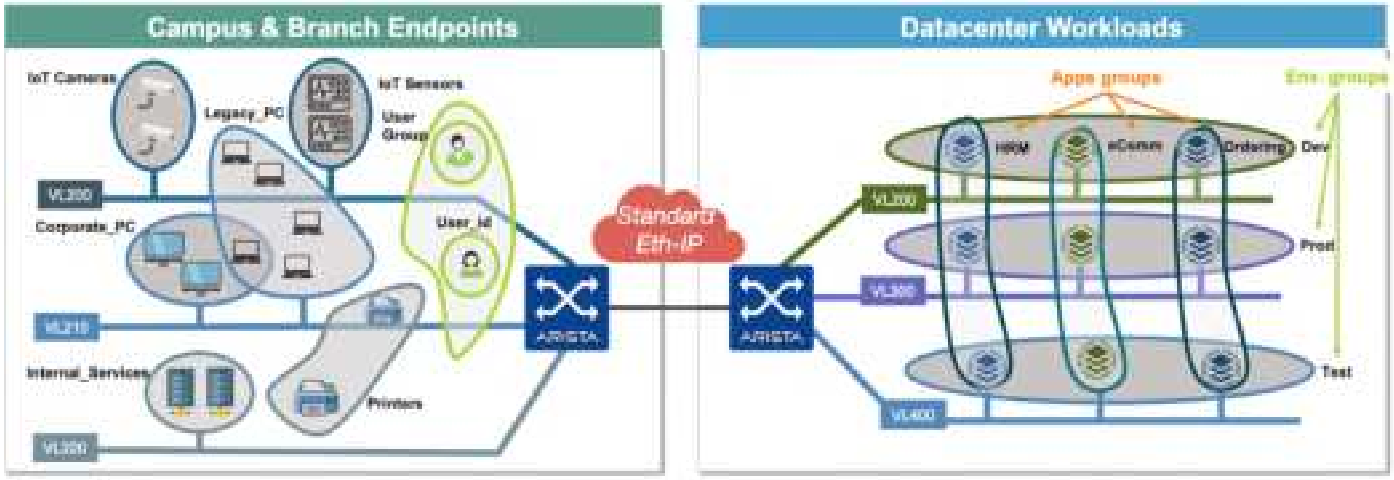
By implementing a multi-layered defense strategy that includes microperimeters, organizations can significantly improve their ability to contain breaches, limit lateral movement, and protect their valuable assets from cyberattacks.

EOS CLI/CloudVision can be used to locally define microperimeters, or they can be dynamically defined from various supported data sources including Arista AGNI maintaining endpoint-to-microperimter mapping consistency throughout the network preventing any blindspots.

(see picture below).

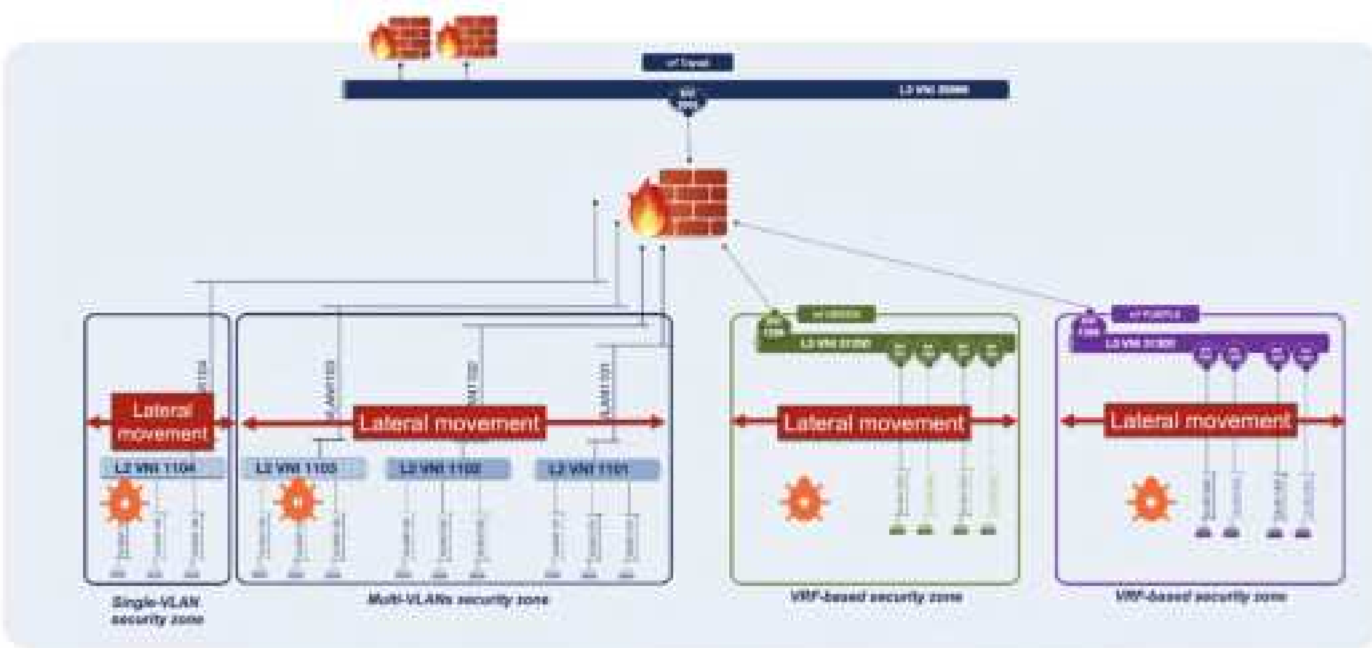
Here are some of the key properties of MSS identity based microperimeters

- Endpoints belonging to the same microperimeter can be part of **same or different VLANs/Subnets**
- Endpoints can belong to **multiple microperimeters**
- Mapping of endpoints to micro-perimeters is resolved locally within the EOS switches and **does not need any tags** to be transported in the dataplane with VXLAN (or other) encapsulations.



Never Trust

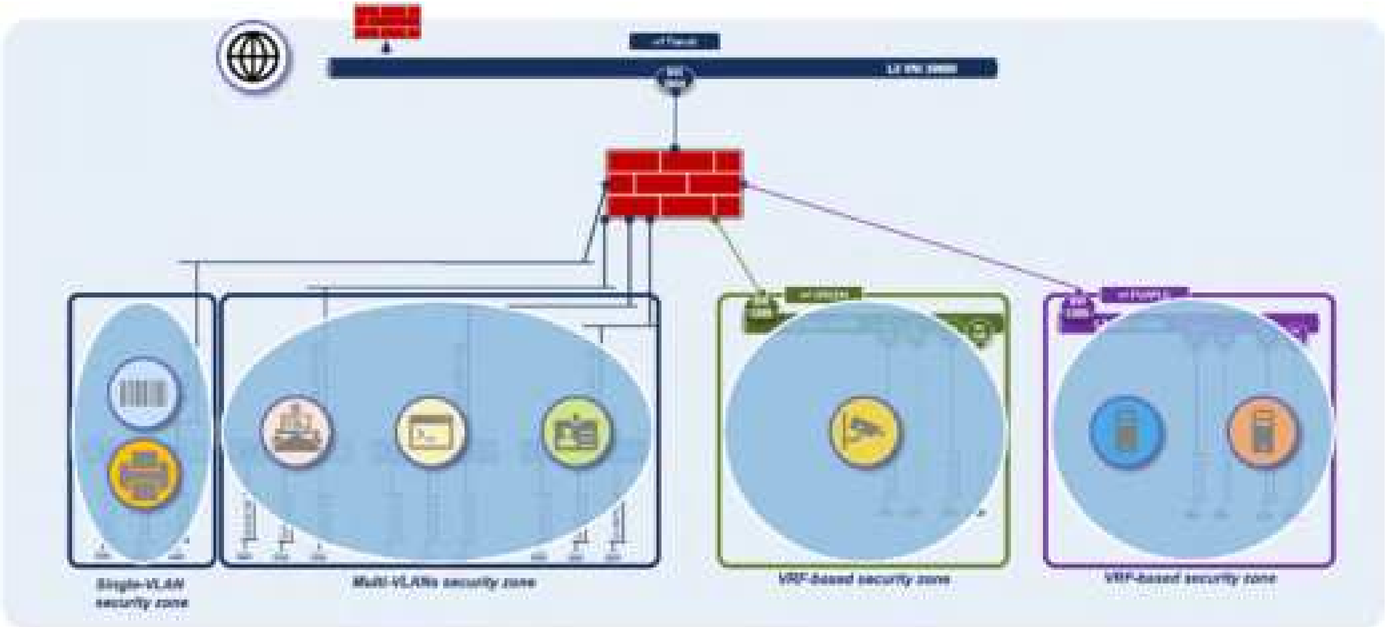
Zero Trust security policies establish a default deny posture, requiring explicit authorization for communication between endpoints with a security rule. This contrasts with traditional network segmentation, which uses VLANs and VRFs to create location-based macro-perimeters. These perimeters align with security zones, and traffic between zones typically passes through a firewall for monitoring and inspection.



However, traffic within a zone is implicitly trusted, as it's not cost-effective to inspect all traffic using expensive firewall bandwidth. This implicit trust creates a larger attack surface for lateral movement within the zone. Traditional location-based macroperimeter approaches can lead to a communication matrix with significant areas of uninspected traffic.

Source	Dest	Scanner	Printer	Room	Terminal	Camera	Reader	Service-1	Internet	Service-2
	Scanner	✓	✓	✓	✓	✓	✓	✓	✓	✓
	Printer	✓	✓	✓	✓	✓	✓	✓	✓	✓
	Room	✓	✓	✓	✓	✓	✓	✓	✓	✓
	Terminal	✓	✓	✓	✓	✓	✓	✓	✓	✓
	Camera	✓	✓	✓	✓	✓	✓	✓	✓	✓
	Reader	✓	✓	✓	✓	✓	✓	✓	✓	✓
	Service-1	✓	✓	✓	✓	✓	✓	✓	✓	✓
	Internet								✓	
	Service-2	✓	✓	✓	✓	✓	✓	✓	✓	✓

Zero trust principles recommend establishing identity-based micro-perimeters and defining explicit policies that enable communication between them, as opposed to constructing network-based perimeters.

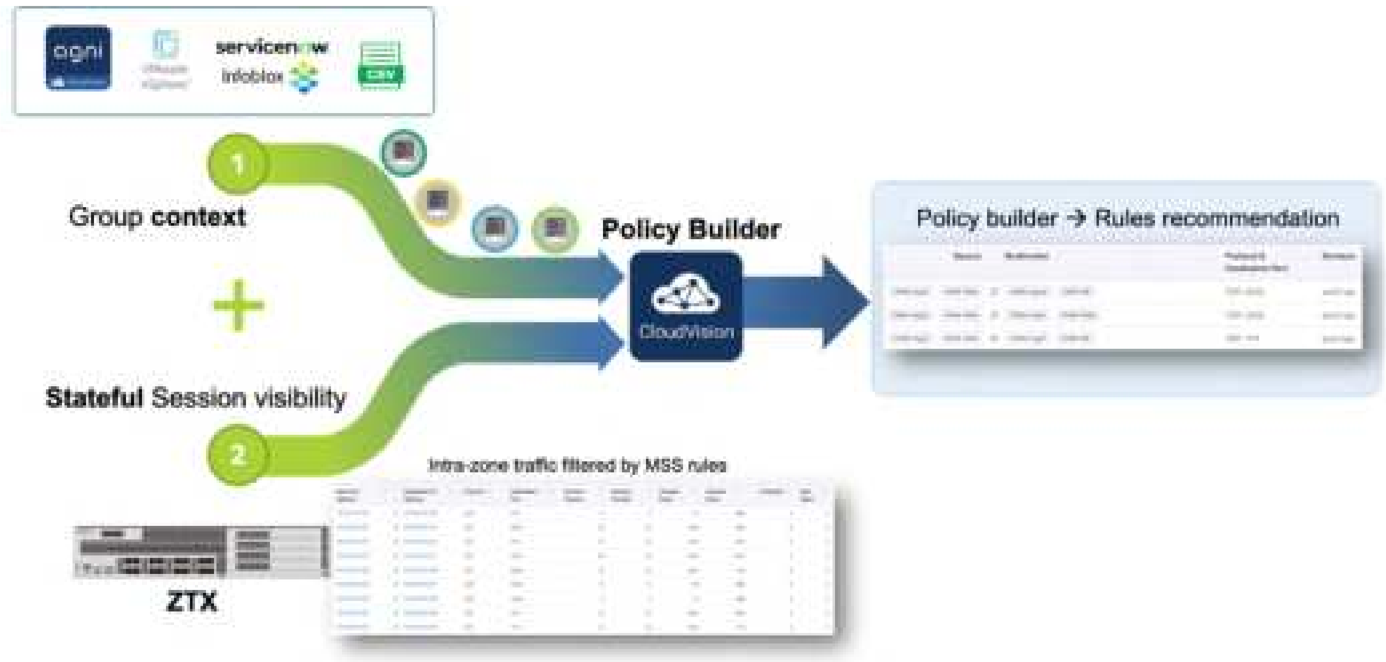


By default, all communication is blocked and only communication allowed via explicit policies is permitted.

Source \ Dest	Scanner	Printer	Room	Terminal	Camera	Reader	Service-1	Internet	Service-2
Scanner		✓							✓
Printer									✓
Room			✓					✓	✓
Terminal							✓		
Camera							✓		
Reader							✓		
Service-1				✓	✓	✓	✓	✓	
Internet								✓	
Service-2	✓	✓	✓					✓	✓

In order to implement policies that allow or deny communication between or within microperimeters, it is crucial to have granular visibility into all existing network conversations and determine which should be permitted or prohibited. MSS conversation (or session) visibility is provided by the ZTX appliance

Arista ZTX appliance, operating out of band, receives truncated mirror east-west traffic appropriately filtered with an MSS rule from production switches and provides the stateful flow level visibility required to discover these network conversations. CloudVision Policy Builder then leverages the micro-perimeter data and the flow data from ZTX to recommend micro-perimeter-based policies, which the security admin can review and implement on the EOS switches with a single click.



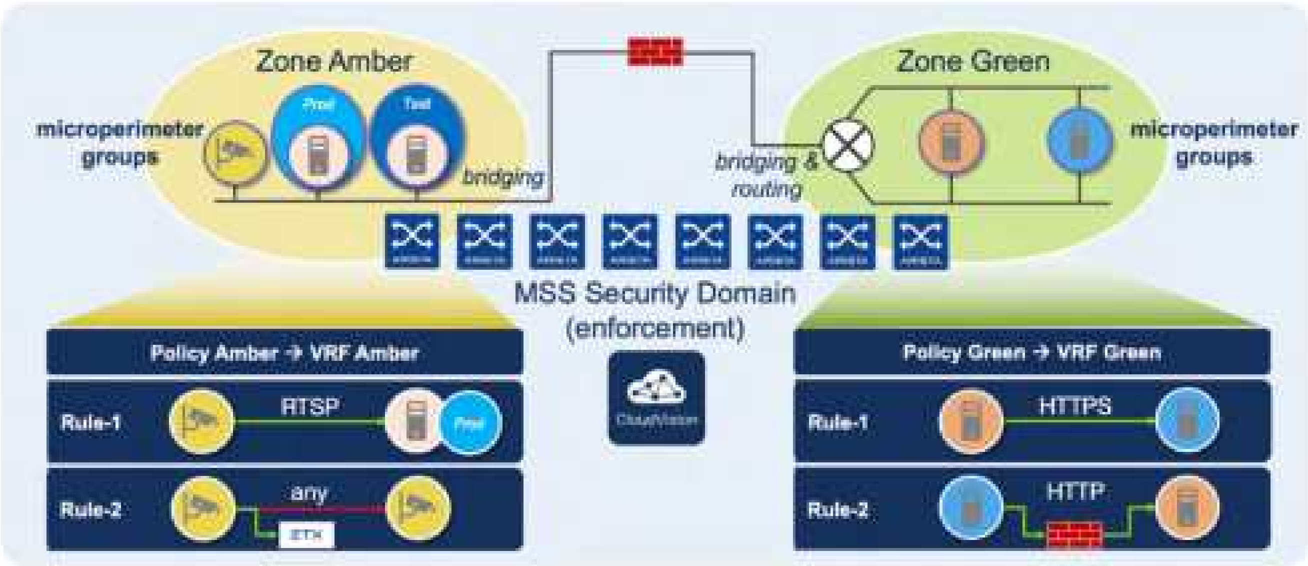
Understanding MSS Security Domains, Policies, Rules, Policy Objects

The picture below summarizes the MSS Policy model and its various components:

1. A Security Domain is a constrict of CloudVision to identify a collection of switches which shares the same policies and microperimeter objects. The switches may have multiple VRFs configured or being just simple L2 switches.
2. An MSS policy is an EOS construct that represents a collection of rules. A policy to be effective, must be attached to a security domain and within a domain to a specific VRF. When no VRFs are configured the Policy attaches to the default VRF (this is also the case for L2 switches).
3. MSS rules are structured to filter traffic at wirespeed on a combination of:
    a. source group object(s): if more than one object is defined as source or destination (e.g. [IoT-Cameras, Building-1] an AND operation is implied across the prefixes composing the groups.
    b. destination group object(s)
    c. service objects: TCP/UDP+L4 port or ICMP (types/codes)

When the traffic matches the policy objects, the rule can define multiple enforcement actions:

1. forward: to explicitly allow the traffic to be routed or bridged to destination
2. deny: to block the traffic
3. forward+monitor: to simultaneously allow the traffic and create a copy of the traffic to be GRE encapsulated and mirrored to a ZTX appliance. The mirrored traffic can also be configured to be truncated and rate limited out of every switch.
4. Within a policy it is possible to apply a monitor action to each rule of the policy.
5. deny+monitor: to simultaneously allow the traffic and create a copy of the traffic to be GRE encapsulated and mirrored to a ZTX appliance.
6. Redirect: to steer the traffic to a third party Firewall Gateway. The network design needs to be properly implemented to allow the redirect action to work.



Always Verify with Arista MSS

The traffic mapping/discovery phase's default rule is "forward+monitor," which sends truncated mirrored traffic to the ZTX node while forwarding traffic uninterrupted. After defining explicit policies, the default rule becomes "drop+monitor," achieving true zero-trust. This drops any traffic not explicitly allowed and sends a truncated mirrored copy to the ZTX node. Security admins can then verify policy violations and troubleshoot valid communication dropped by the default rule, converting them to explicit permit rules as needed.

Monitoring permitted traffic for malicious behavior is also crucial. Arista NDR, with advanced machine learning and AI, constantly monitors for threat signatures and initiates quarantine upon detection.

Configuration Deployment

Let's integrate the concepts we've explored with a practical deployment example.

As mentioned previously in the wireless service section, our primary focus will be on the Leaf/Access enforcement deployment model. This approach aims to establish microperimeter-based security and enforcement through CloudVision, thereby achieving a Zero Trust architecture. We will delve into the Spine/Aggregation enforcement option in a subsequent version of this document.

We'll begin by examining AGNI (Arista Guardian for Network Identity), which is Arista's Network Access Control (NAC) solution. AGNI plays a crucial role in our Zero Trust architecture by controlling and managing network access based on device and user identity. It helps ensure that only authorized devices and users can connect to the network, and that they are granted access only to the resources they are entitled to.

By integrating AGNI with CloudVision and our Leaf/Access enforcement model, we can create a comprehensive and dynamic security framework that adapts to the evolving threat landscape. This framework allows us to enforce granular access policies, segment network traffic, and isolate potential threats, all while maintaining high network performance and availability.

AGNI Configuration

For MSS, AGNI performs two critical functions - Admission control and Profiling

For admission control, we need the access points and the switches to act as the authenticators and form RadSec Tunnel with AGNI. Below are the steps to follow

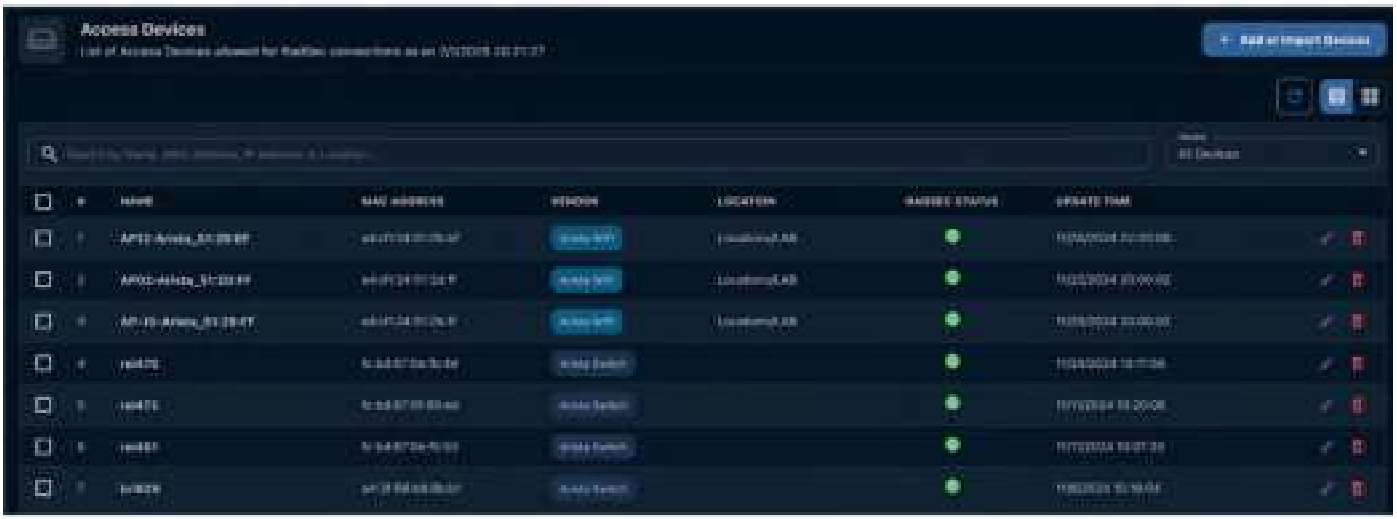
[Steps for adding APs as authenticators for AGNI]

[Steps for adding Switches as authenticators for AGNI]

Once you complete the steps to add the switch as authenticator, you should see configuration similar to the one shown below:

```
Leaf
!
management security
   ssl profile agni-server
      certificate poc_bri629.pem key rit311.key
      trust certificate poc_radsec.pem
!
radius-server host radsec.scale.agnieng.net tls ssl-profile agni-server
!
aaa group server radius agni-server-group
   server radsec.scale.agnieng.net tls
!
aaa authentication dot1x default group radius
aaa accounting dot1x default start-stop group radius
!
```

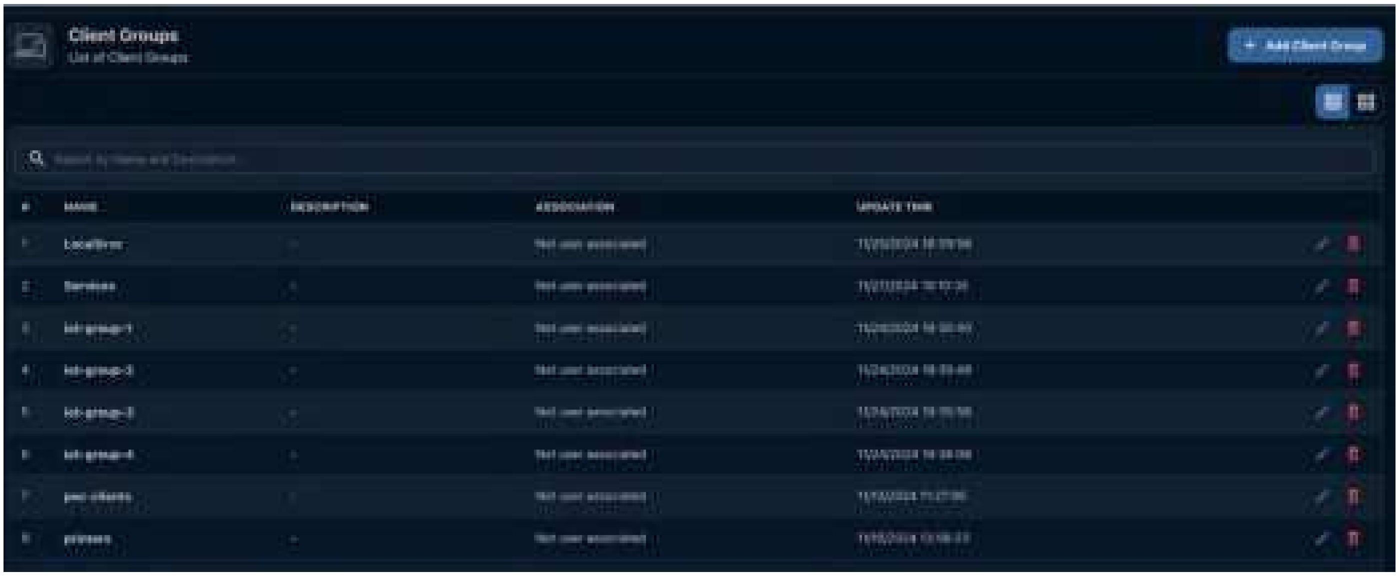
Once the steps mentioned above are completed, we can see that the APs and the leaf switches are now added as authenticators in AGNI and the RadSec tunnel is up.

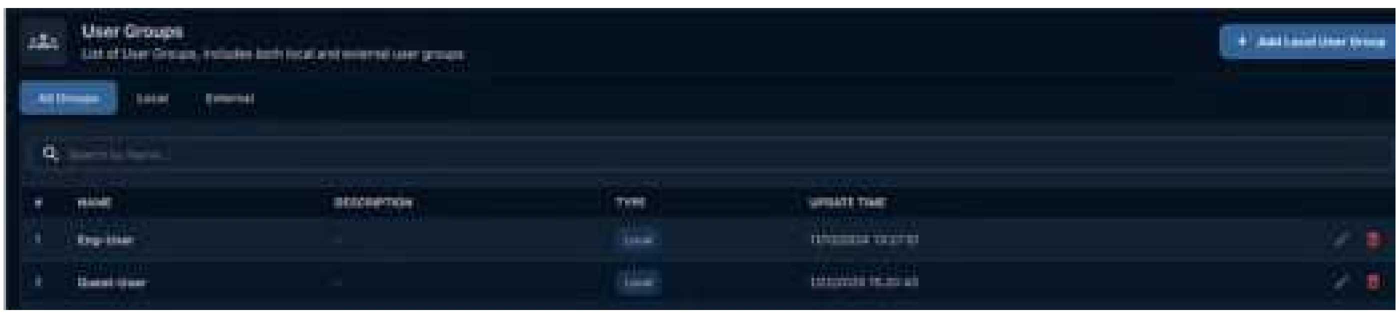


We need to add the below configuration to the interfaces connecting to the endpoints for dot1x authentication

```
Leaf

bri629
!
interface Ethernet49
   description ixs342-2
   switchport mode trunk
   switchport source-interface tx
   dot1x pae authenticator
   dot1x reauthentication
   dot1x port-control auto
   dot1x host-mode multi-host authenticated
   dot1x mac based authentication host-mode common
!

roi470

!
interface Ethernet22
   description poc-srv-51.vmnic1
   switchport mode trunk
   dot1x pae authenticator
   dot1x reauthentication
   dot1x port-control auto
   dot1x host-mode multi-host authenticated
   dot1x mac based authentication host-mode common
!
```

In AGNI, the IOT endpoints are classified into Client groups and the users are classified into user groups. For the purpose of this demo we have created a few Client groups and couple of user groups as shown





Once the endpoints successfully authenticate, the mapping of the endpoint IPs to these groups will be sent to CloudVision via event notification, after we add AGNI as our datasource.

Support for AGNI segments will be added soon.

CloudVision Configuration

Enable MSS

MSS is not enabled by default in CloudVision. Follow the steps below to enable it.

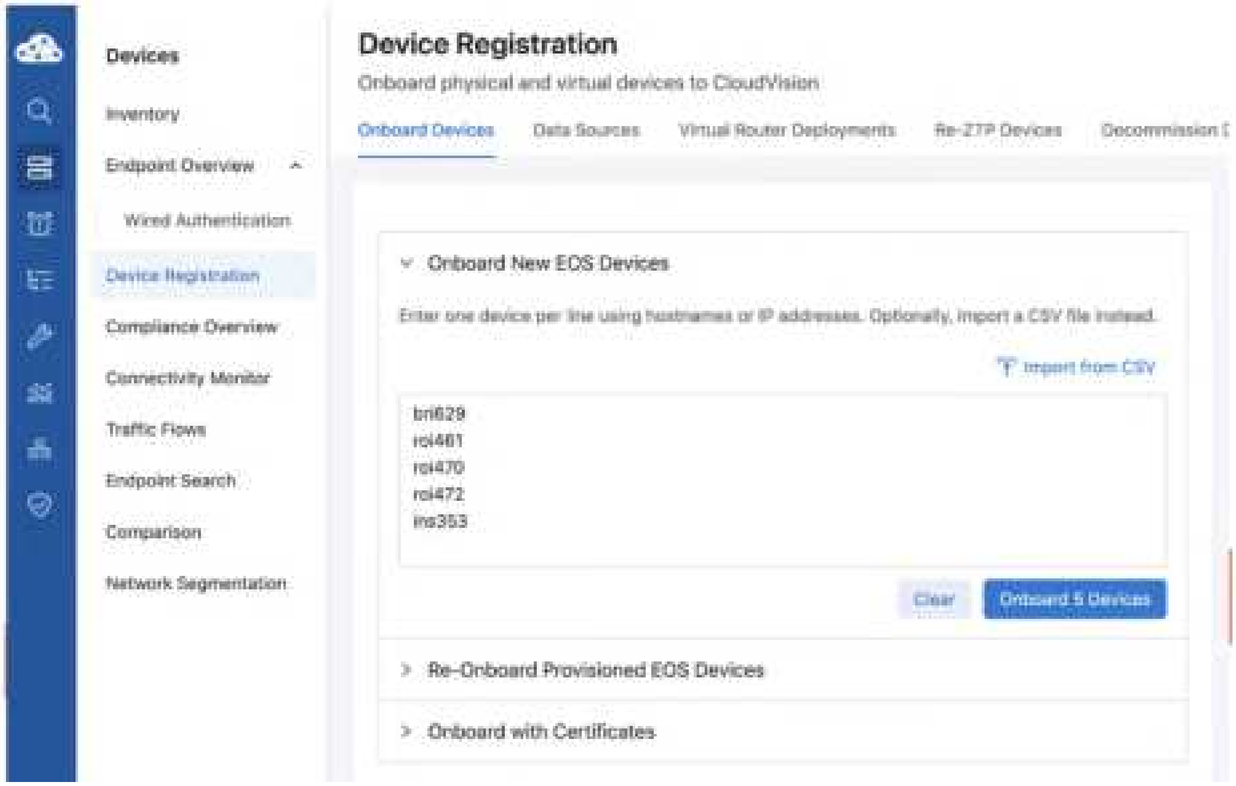
1. Go to Setting
2. Enable Network Security - MSS and Studios-MSS Studio knobs

```
daemon TerminAttr
   exec /usr/bin/TerminAttr -smashexcludes=ale,flexCounter,hardware,kni,pulse,strata
-cvaddr=172.28.137.75:9910,172.28.130.47:9910,172.28.133.90:9910 -cvauth=token,/tmp/token -cvvrf=default
-taillogs -cvtargetconfigs mss
```

Make sure the following CLI is enabled in the Traffic Policy configuration on the Trident based enforcement switches

```
traffic-policies
   vrf Campus
   transforms interface prefix common source-destination
```

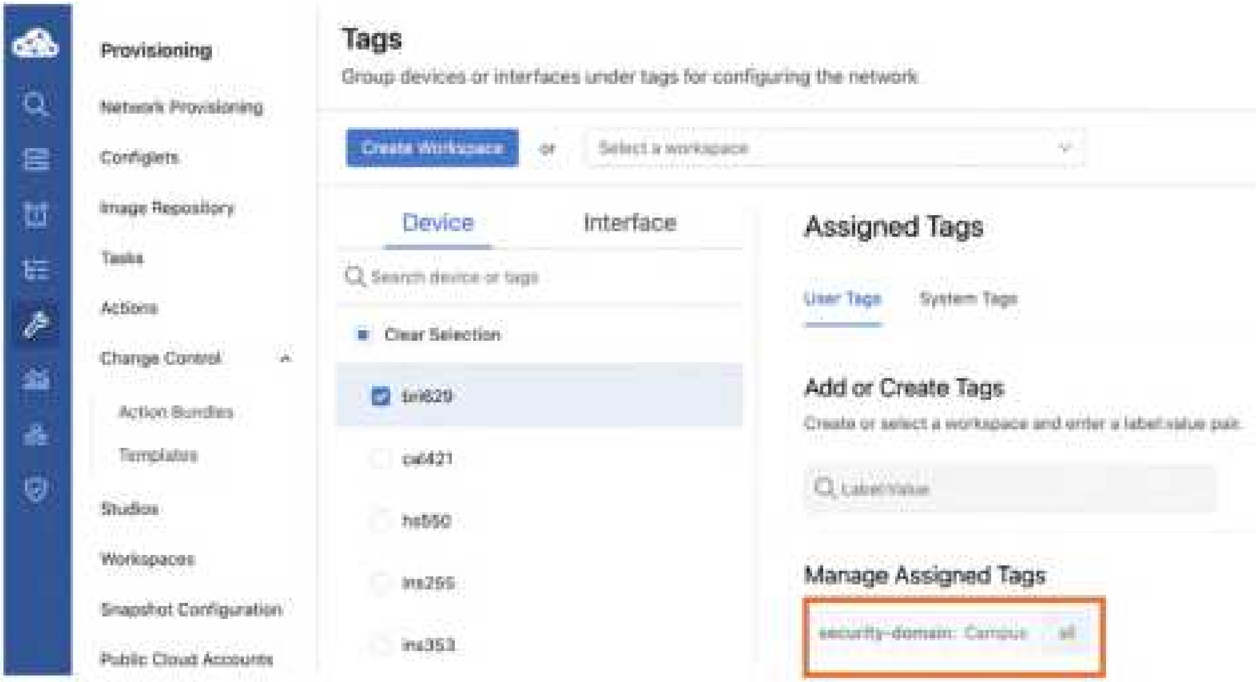
Onboard the Enforcement switches and ZTX appliance



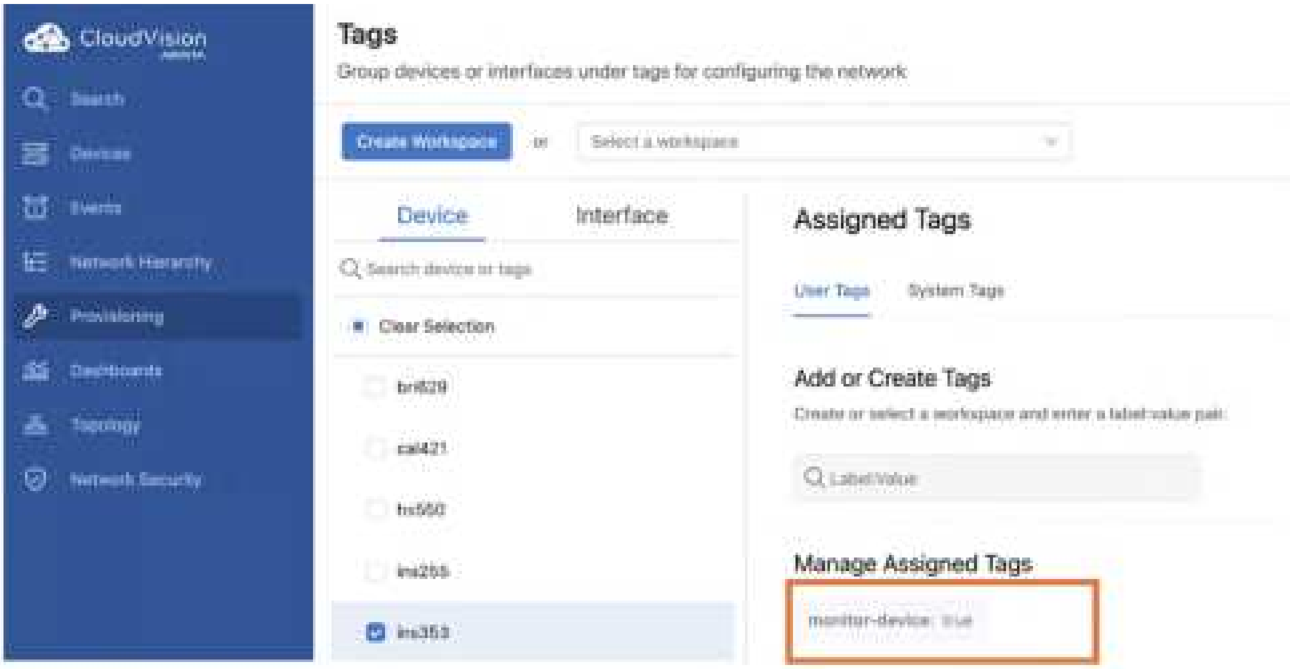
Accept the changes in the Inventory and Topology Studio

Define the Tags

Add the security-domain tag to all the enforcement switches (all the leaf switches in our example)



Add the monitor-device tag for the ZTX node

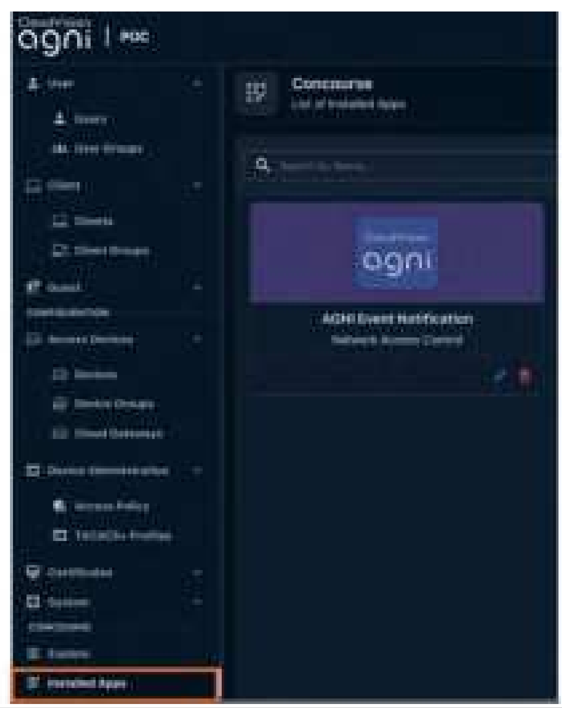


Add Datasource

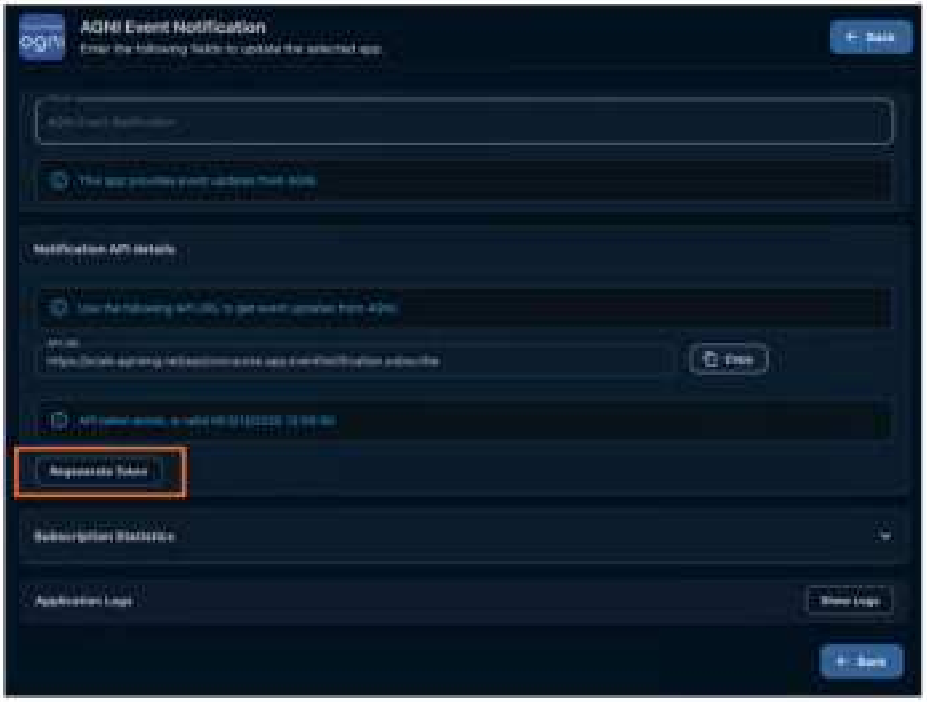
Before we add AGNI as a data source in CloudVision, we need to gather the Organization ID and Event notification API token

To obtain the Organization ID, click on the user name Icon on the far right and copy the organization ID.

To obtain the Event Notification token, go to the AGNI Event Notification Application in the Concourse section.



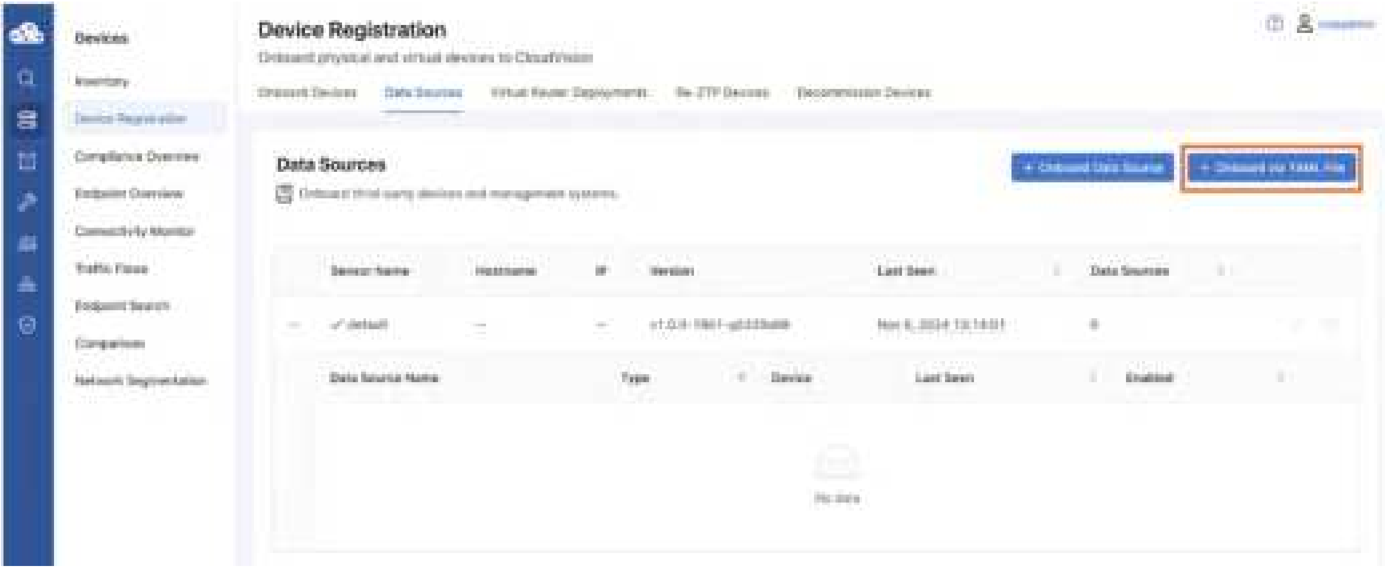
You can generate/regenerate the token from here. Save the token when it's generated as you will not be able to view it afterward.



CloudVision supports multiple data sources that can be added as identity/context source for the campus or datacenter endpoints and tags. Each of the supported data sources have a specific yaml configuration template that can be used to onboard them.

In our example, we will be using AGNI as the data source. Follow the steps below on CloudVision to onboard the data source.

Go to **Devices** > **Device Registration** > **Data Sources.**



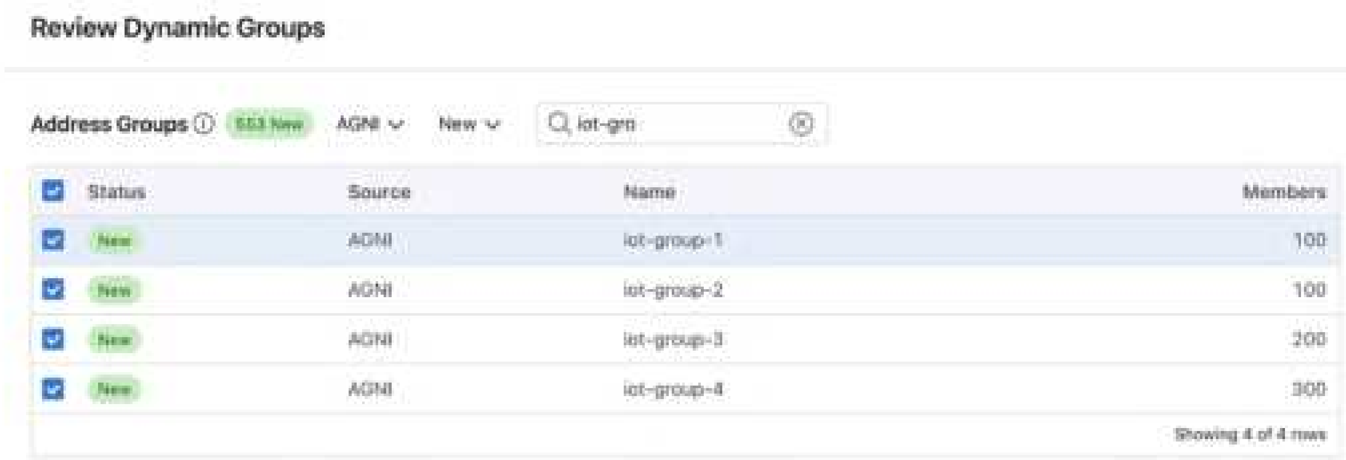
Click on **Onboard via YAML File** and paste the AGNI YAML file template. Modify the parameter as per your topology and then press **Onboard.**

```
- Type: AGNI
  DeviceID: <Add device ID. This will be used as a prefix for the groups learned via this datasource>
  Sensor: default
  LogLevel: LOG_LEVEL_INFO
  Enabled: true
  Options:
    base_url: https://<provide AGNI URL>/api/
    device_id: <Add device ID. This will be used as a prefix for the groups learned via this datasource>
    org_id: <add organization id here>
    poll_interval: 30s
    trafficPolicy: true
  Credentials:
    auth_token: <add event authentication token here>
```

Once the data source is added, CloudVision will receive the client-groups and user-groups from AGNI as well as the endpoint IPs associated with these groups. These can be viewed from the Network security tab in CloudVision

Go to **Network Security** > **Policy Manager** > **Groups**

Here you can click on the **Review Groups** to see dynamically learned groups from AGNI. Select all the groups that you want to import to CloudVision for the policy creation workflow



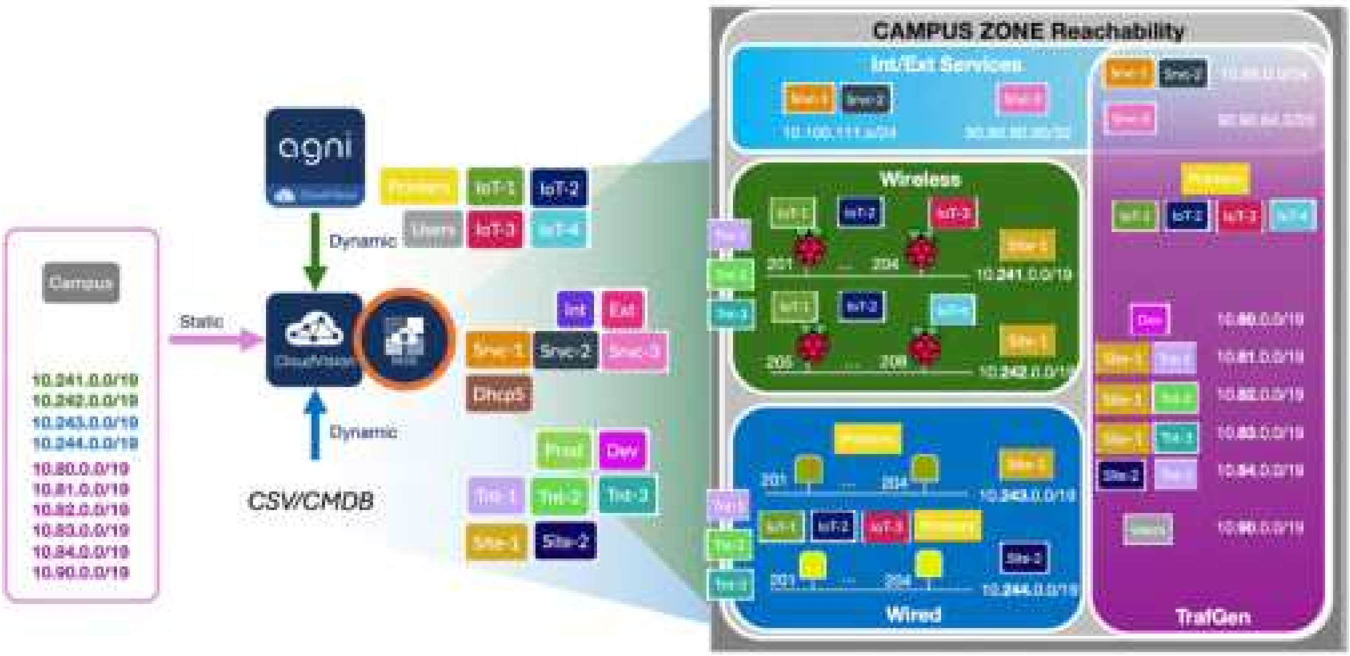
If you are maintaining groups in another datasource, we can add then using the datasource specific template. We are adding some groups (like Env, Site, Tenant, SrvcType, SrvcCode) that are part of a CSV file.

```
- Type: mss_csv
  DeviceID: CDB
  Sensor: default
  LogLevel: LOG_LEVEL_INFO
  Enabled: true
  Options:
    cat_1: Env
    cat_2: Site
    cat_3: Tenant
    cat_4: SrvcType
    cat_5: SrvcCode
    device_id: CDB
    endpoint: Hostname
    ip_prefix: Ipaddr
```

```
    poll_interval: 20s
    sftp_filename: <CSV file location on the host e.g /root/poc-campus.csv>
    sftp_host: <host IP/DNS name>
    sftp_username: root
  Credentials:
    sftp_password: <password>
```

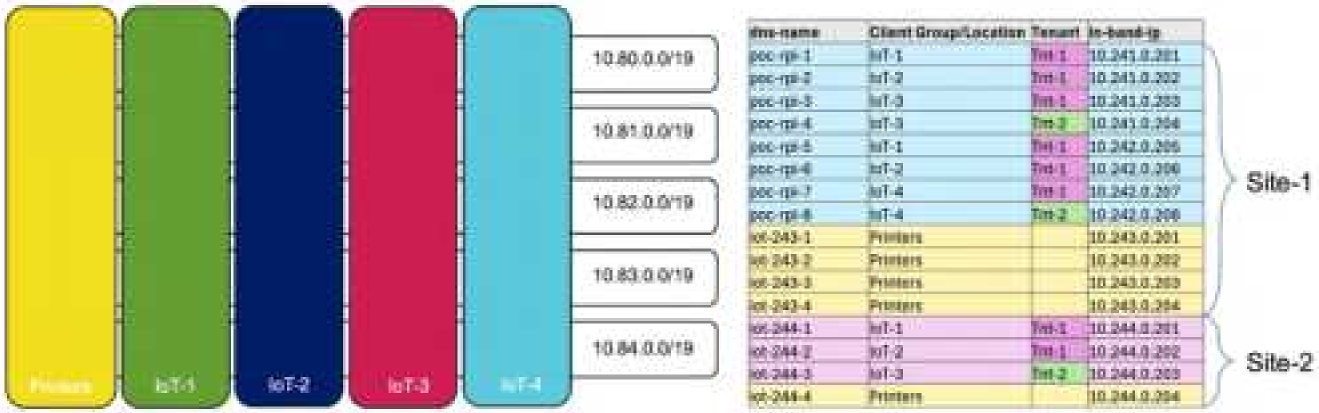
Once the groups from the data source are added, they will be available to use in the policy creation workflow

Below are the groups that are defined for this deployment example



We have defined groups for IOT devices (IOT-1, IOT-2, IOT-3, IOT-4 and Printers), Users (Eng-Users), Campus Site (Site-1 and Site-2), Tenants (Tnt-1, Tnt-2, Tnt-3) and Services (Srvc-1, Srvc-2 and Srvc-3)

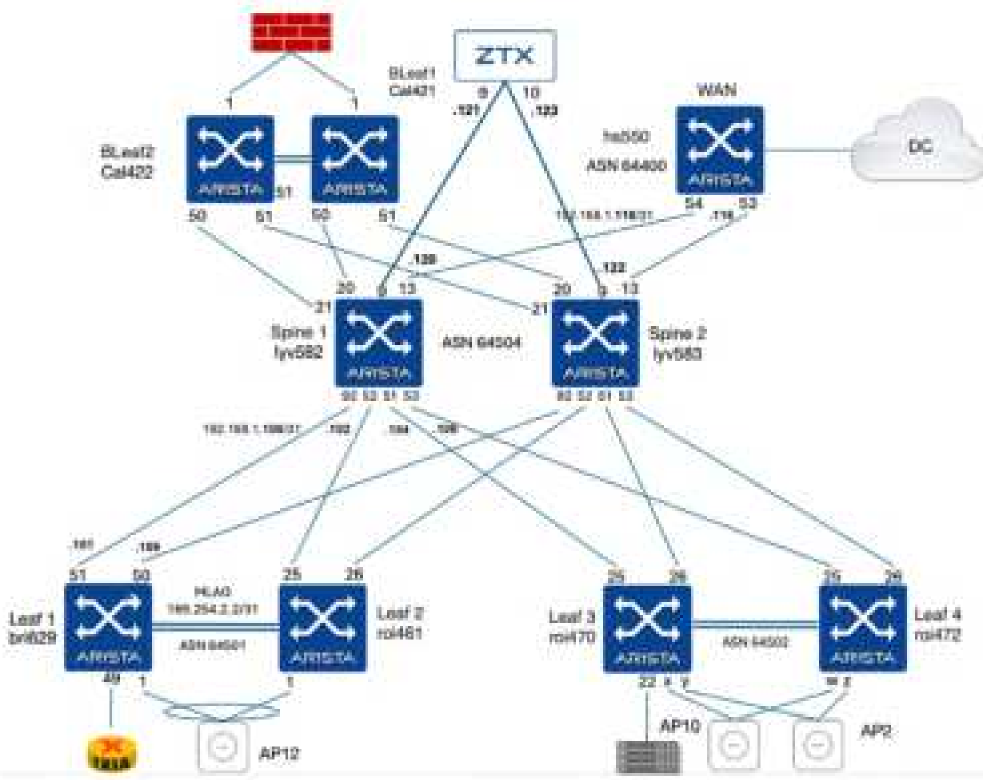
If we look closely at how the endpoints are spread across different groups, we can see that there is no dependency between the IP location/VLAN/Subnet to the group mapping. Also we see that the endpoints belong to multiple groups. As an example, poc-rpi-1 (10.241.0.201) belongs to Tnt-1, Site-1 and IOT-1



dns-name	Client Group/Location	Tenant	In-band-ip
poc-rpi-1	IoT-1	Tnt-1	10.241.0.201
poc-rpi-2	IoT-2	Tnt-1	10.241.0.202
poc-rpi-3	IoT-3	Tnt-1	10.241.0.203
poc-rpi-4	IoT-3	Tnt-2	10.241.0.204
poc-rpi-5	IoT-1	Tnt-1	10.242.0.205
poc-rpi-6	IoT-2	Tnt-1	10.242.0.206
poc-rpi-7	IoT-4	Tnt-1	10.242.0.207
poc-rpi-8	IoT-4	Tnt-2	10.242.0.208
iot-243-1	Printers		10.243.0.201
iot-243-2	Printers		10.243.0.202
iot-243-3	Printers		10.243.0.203
iot-243-4	Printers		10.243.0.204
iot-244-1	IoT-1	Tnt-1	10.244.0.201
iot-244-2	IoT-2	Tnt-1	10.244.0.202
iot-244-3	IoT-3	Tnt-2	10.244.0.203
iot-244-4	Printers		10.244.0.204

Policy Creation workflow

In this section, we will demonstrate how we can use ZTX to create explicit policies. The ZTX node has been connected to the spine switches in our example.



As discussed before, ZTX will build a stateful traffic map that will be sent to CloudVision to assist in the policy creation workflow. Before we do that we need to take care of a few pre-requesites.

Create a loopback interface on ZTX

```
interface Loopback0
   description router-id
   ip address 10.135.2.13/32
```

This will serve as the source of the L2GRE tunnel that will be automatically created to each of the leaf switches in the security domain

Define the loopback interface on the enforcement switches

This loopback interface will serve as the tunnel destination for the L2GRE tunnel that will be created by the ZTX

```
bri629.12:31:48#show running-config section loopback0
interface Loopback0
   description router-id
   ip address 10.135.1.1/32

roi461.12:32:49#show running-config section loopback0
interface Loopback0
   description router-id
   ip address 10.135.1.2/32

roi470.12:33:27#show running-config section loopback0
interface Loopback0
   description router-id
   ip address 10.135.1.3/32

roi472.12:33:55#show running-config section loopback0
interface Loopback0
   description router-id
   ip address 10.135.1.4/32
```

Routing for loopback connectivity

In our example we are using BGP for loopback connectivity between the ZTX and enforcement switches. You can also create simple static routes if needed.

```
ZTX

!
interface Ethernet1/9
   description Spine1-E9
   no switchport
   ip address 192.168.1.121/31
!
interface Ethernet1/10
   description Spine2-E9
   no switchport
   ip address 192.168.1.123/31
!

!
router bgp 64453
   router-id 10.135.2.13
   distance bgp 20 200 200
   maximum-paths 2
   neighbor UNDERLAY peer group
   neighbor UNDERLAY maximum-routes 1200
   neighbor 192.168.1.120 peer group UNDERLAY
   neighbor 192.168.1.120 remote-as 64504
   neighbor 192.168.1.120 description Spine1
   neighbor 192.168.1.122 peer group UNDERLAY
   neighbor 192.168.1.122 remote-as 64504
   neighbor 192.168.1.122 description Spine2
   redistribute connected
   !
   address-family ipv4
      neighbor UNDERLAY activate
!


roi472.12:34:01#show ip route 10.135.2.13

VRF: default
Source Codes:
       C - connected, S - static, K - kernel,
       O - OSPF, IA - OSPF inter area, E1 - OSPF external type 1,
       E2 - OSPF external type 2, N1 - OSPF NSSA external type 1,
       N2 - OSPF NSSA external type2, B - Other BGP Routes,
```

```
       B I - iBGP, B E - eBGP, R - RIP, I L1 - IS-IS level 1,
       I L2 - IS-IS level 2, O3 - OSPFv3, A B - BGP Aggregate,
       A O - OSPF Summary, NG - Nexthop Group Static Route,
       V - VXLAN Control Service, M - Martian,
       DH - DHCP client installed default route,
       DP - Dynamic Policy Route, L - VRF Leaked,
       G  - gRIBI, RC - Route Cache Route,
       CL - CBF Leaked Route

 B E      10.135.2.13/32 [20/0]
           via 192.168.1.106, Ethernet25
           via 192.168.1.114, Ethernet26

ins353.12:38:53#show ip route 10.135.1.4

VRF: default
Source Codes:
       C - connected, S - static, K - kernel,
       O - OSPF, IA - OSPF inter area, E1 - OSPF external type 1,
       E2 - OSPF external type 2, N1 - OSPF NSSA external type 1,
       N2 - OSPF NSSA external type2, B - Other BGP Routes,
       B I - iBGP, B E - eBGP, R - RIP, I L1 - IS-IS level 1,
       I L2 - IS-IS level 2, O3 - OSPFv3, A B - BGP Aggregate,
       A O - OSPF Summary, NG - Nexthop Group Static Route,
       V - VXLAN Control Service, M - Martian,
       DH - DHCP client installed default route,
       DP - Dynamic Policy Route, L - VRF Leaked,
       G  - gRIBI, RC - Route Cache Route,
       CL - CBF Leaked Route

Gateway of last resort:
 B E      0.0.0.0/0 [20/0]
           via 192.168.1.120, Ethernet1/9
           via 192.168.1.122, Ethernet1/10
```

Policy Manager Configuration

Add Security Domain

Here we will add the Campus Security Domain which is the tag we added to the enforcement switches in the previous step

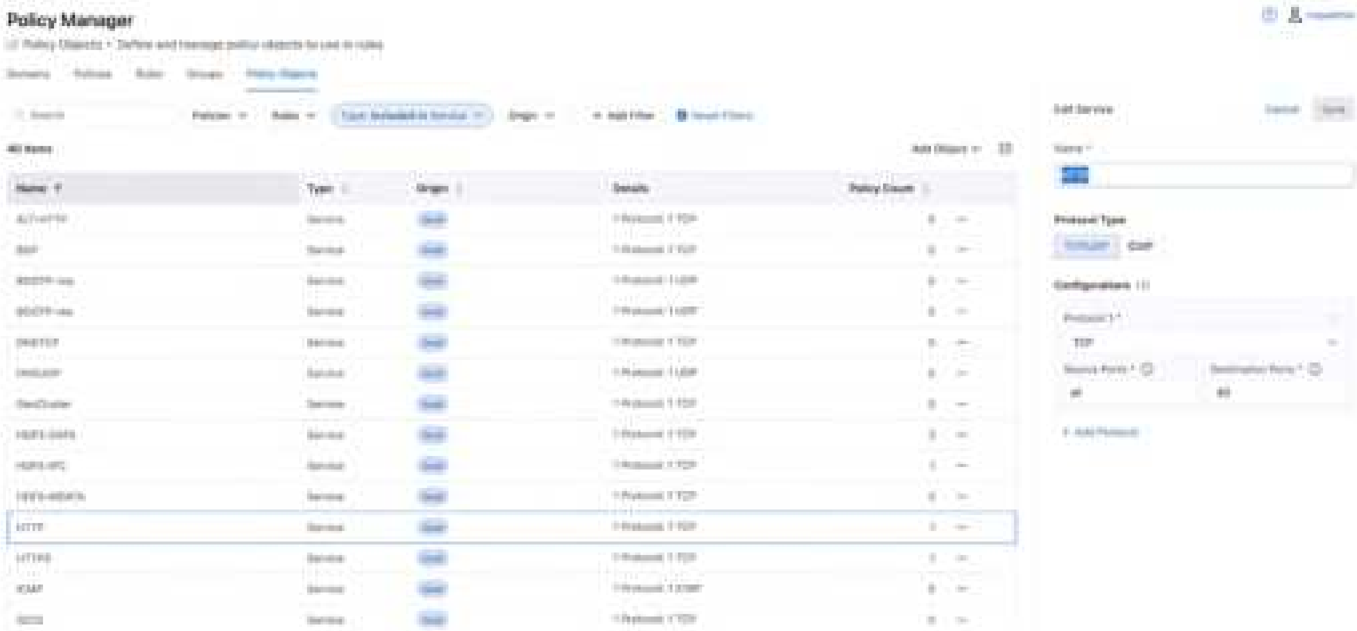
Add Policy Object

Next we create the policy object that will hold our monitoring policy as well as the policy recommendations that we receive via ZTX. We also tie the VRF and the domain under policy configuration

Next we will create a static group in the MSS Service studio that matches all the internal IPs. The goal is to use this in a monitoring rule so that we can monitor all the traffic within the campus and send a truncated mirrored copy to ZTX node to build the traffic map

Add Services

Next, we can predefine a set of L4 services. As we can see we can attach multiple L4 ports to the same service. This will help us identify the traffic better when audit the rules generated by ZTX



Add Monitor Object

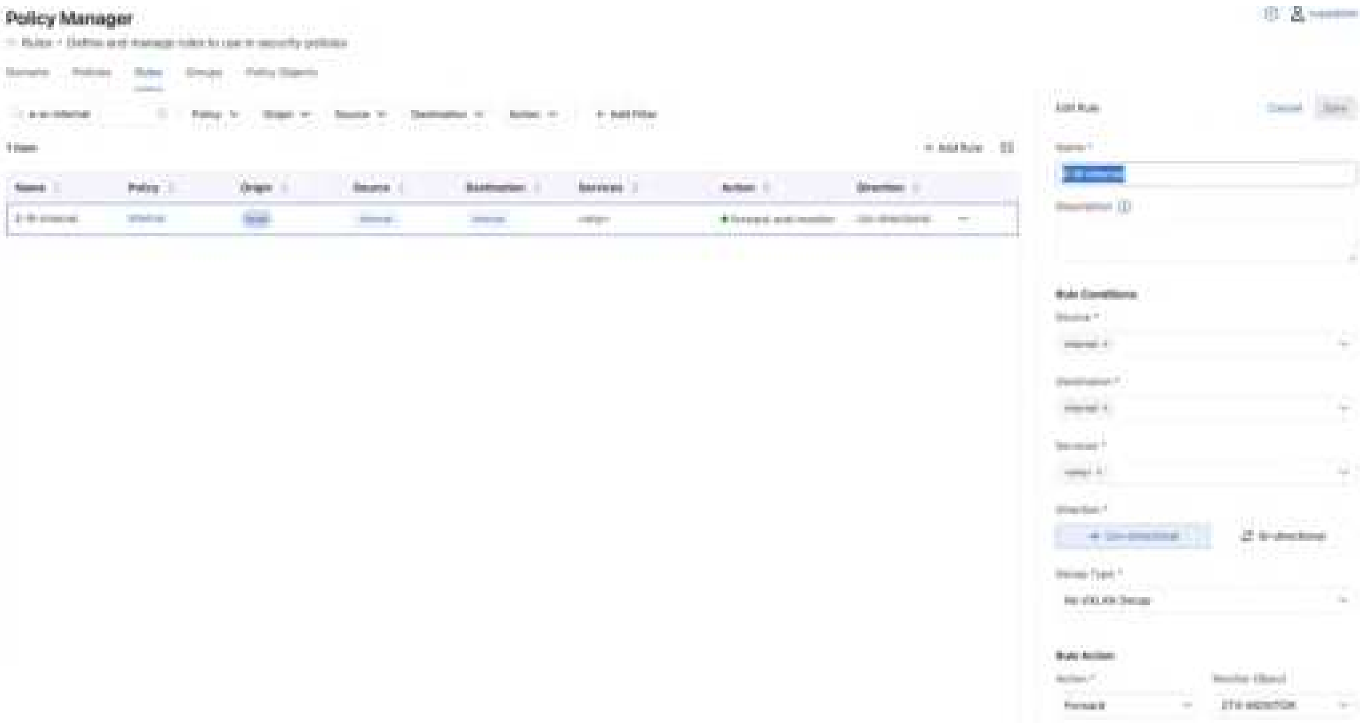
One last step before we move on to defining the policy is to create a Monitor object that will reference the ZTX node.

Monitor Node: Provide the ZTX node name
Exporter Interface: ZTX Loopback interface
Active Timeout: Frequency at which the ZTX node will export session data to CloudVision
Tunnel Destination IP: Loopback IP of ZTX
Tunnel Source Interface: Loopback interface name of the TOR switches
Truncation: Mirror truncated copy or the full copy of the packet

Traffic discovery and Policy building

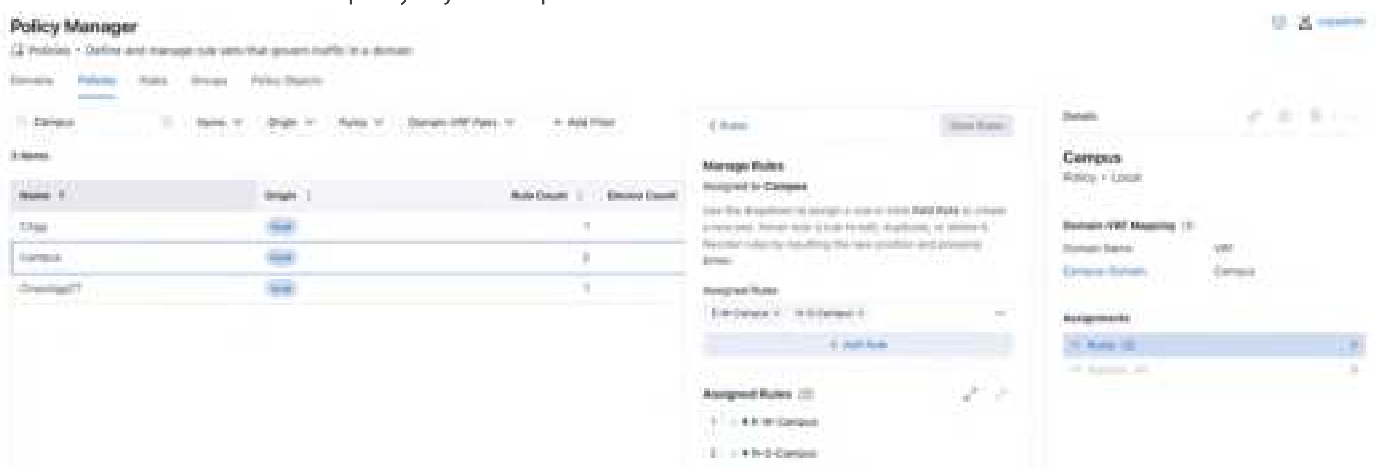
The objective of Zero Trust is to establish a default 'drop and monitor' policy that only permits traffic explicitly allowed by the rules. The 'monitor' action allows for observation and validation of any policy violations from the dropped traffic. However, to construct these explicit rules, we begin with a default 'forward and monitor' rule. This enables ZTX to observe all traffic, build a traffic map, and provide rule recommendations. Once accepted, these recommendations are converted into explicit policy rules and placed above the default rule. Consequently, the default rule will only forward and monitor traffic that doesn't match the explicit rules. This process allows for the discovery and conversion of new traffic into explicit rules. Once satisfied with the established rules, the default policy can be changed to 'drop and monitor.'

Next, we define our E-W-Campus monitoring rule to capture all traffic within the Campus group.



Finally, we define the North-South rule as a forward rule. This means that any traffic that is not affected by the preceding rules will be forwarded as usual. Most likely, this will be traffic going out of the campus pod, which would then be inspected by the N-S firewall.

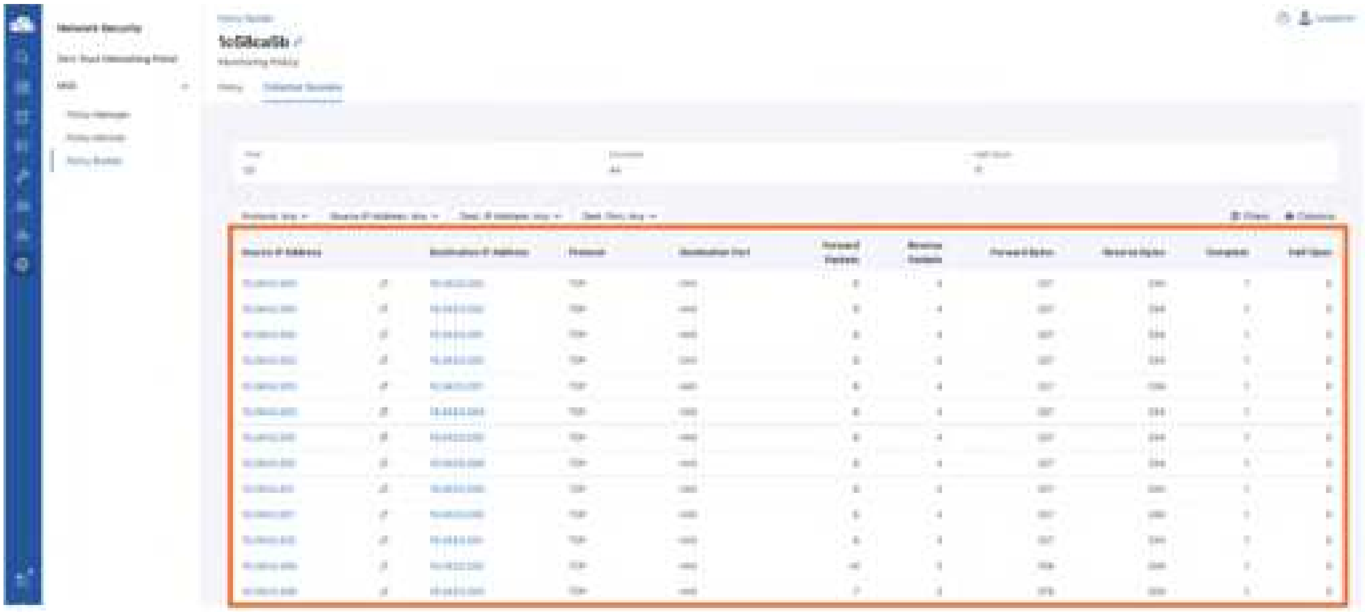
Both these rules are added to our policy object 'Campus'



We have started some traffic that goes from endpoints in Site 1 to Printers in Site 1

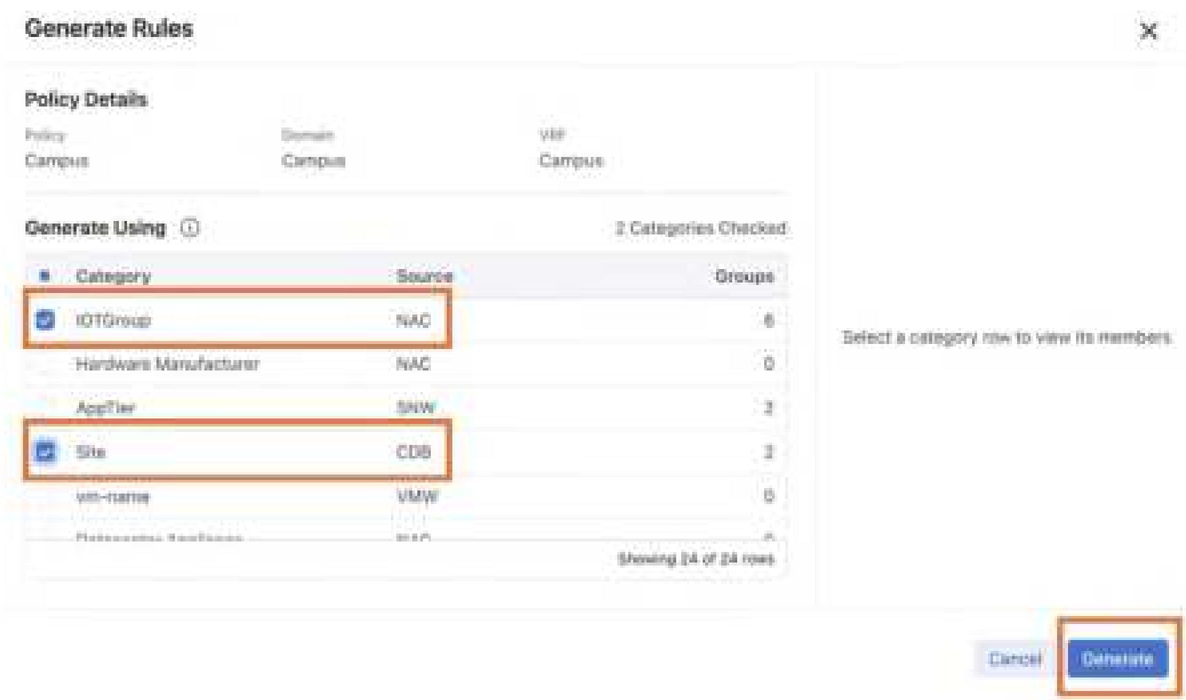
Go to the **Network Security** > **Policy Builder**

Click on **Collected Sessions** to see the stateful traffic flows

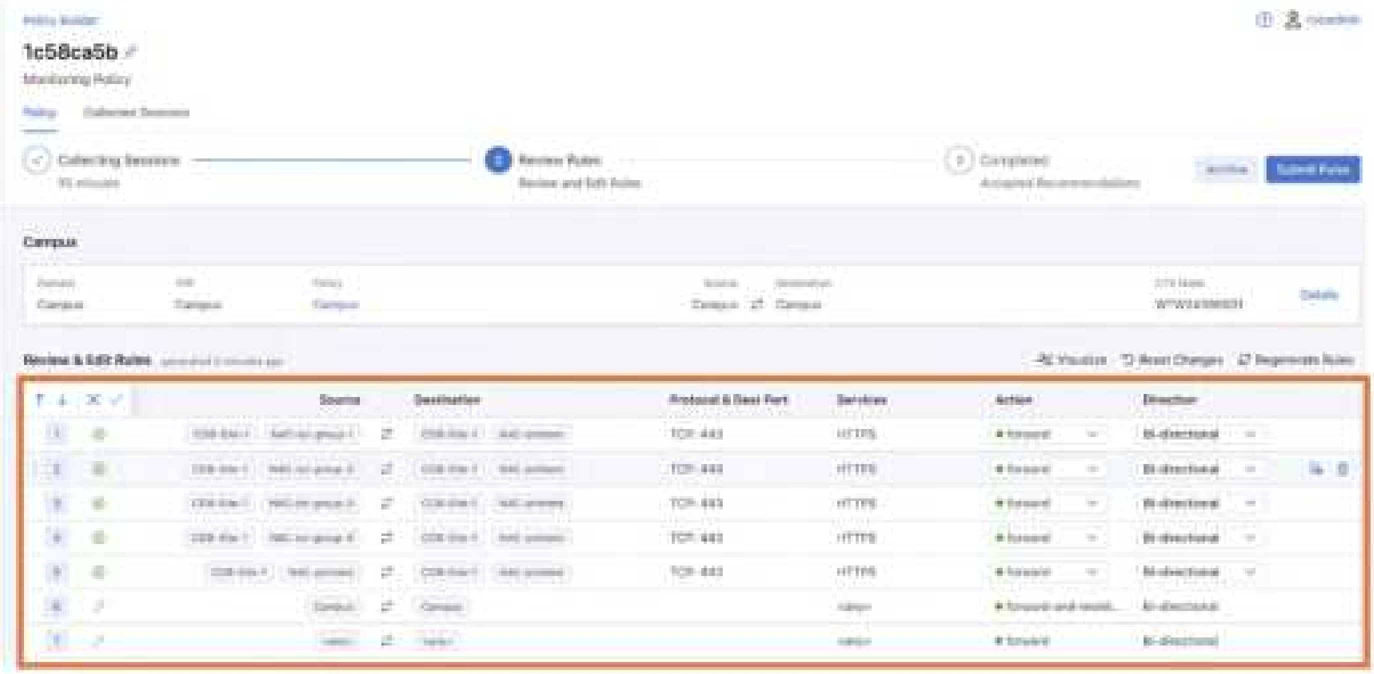


Go to **Policy** tab and then click **Generate Rules** to color these flows with the group tags that we leaned previously

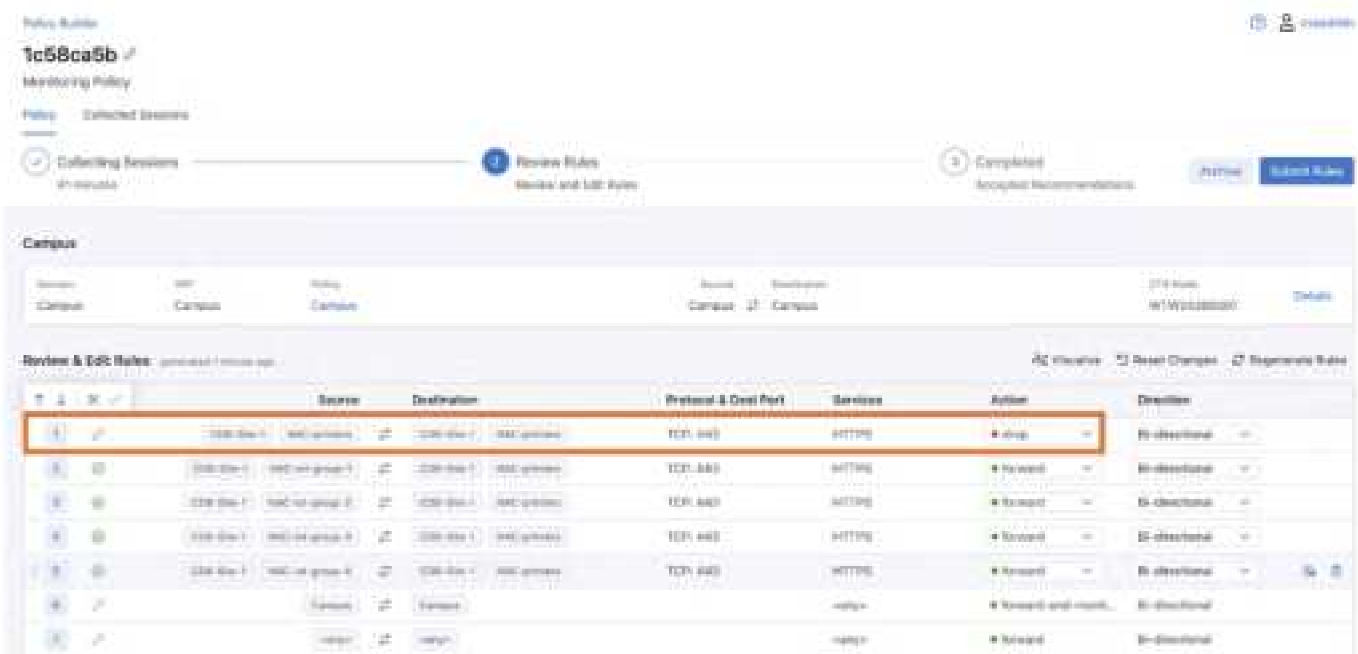
Select the relevant tags and click **Generate**. Here you can select broader tags or more specific tags to build either a broader policy rule or a very specific policy rule



Now we get presented with the policy rules in terms of the tags we selected. As an Admin, you can now audit these rules, change the order of the rules, change the action or delete the recommendations if you choose so.



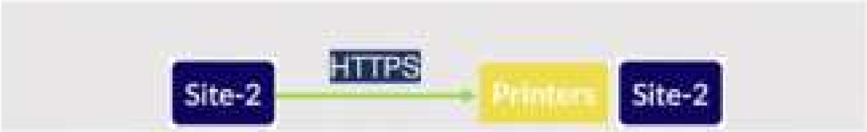
We see that members of IOT-group-1, IOT-group-2, IOT-group-3, IOT-group-4 as well as Printers within Site-1 are sending traffic to the Printers in Site-1. Everything looks good, except that we do not want the printers to talk to each other. At this point we can move the Printers-Printers rule at the top and change the action to drop.

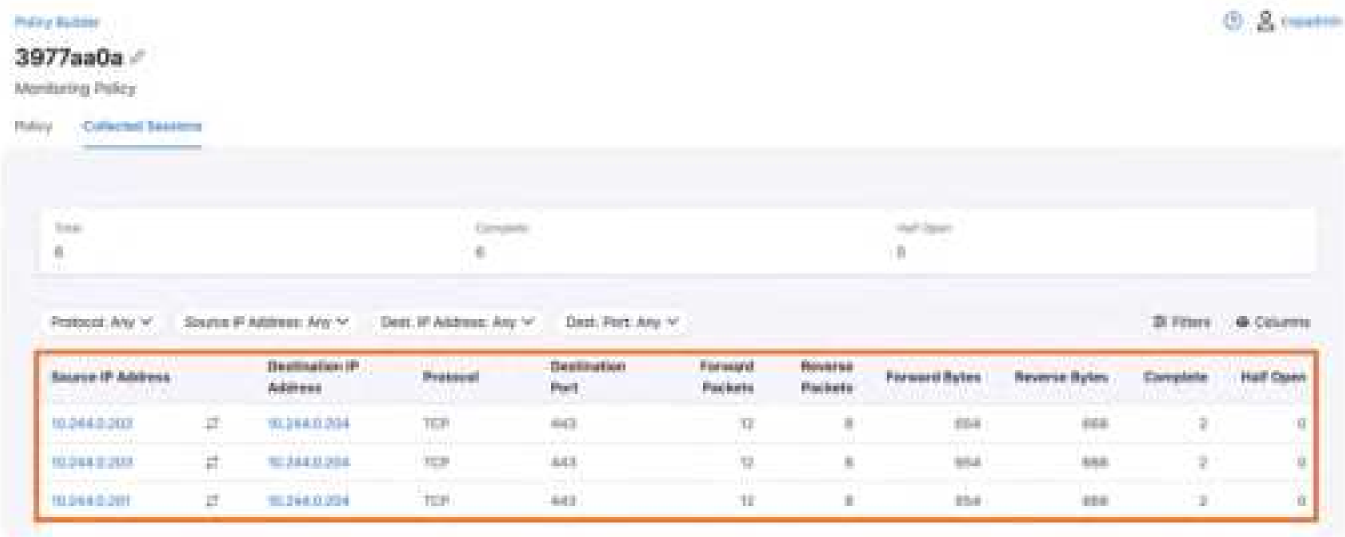


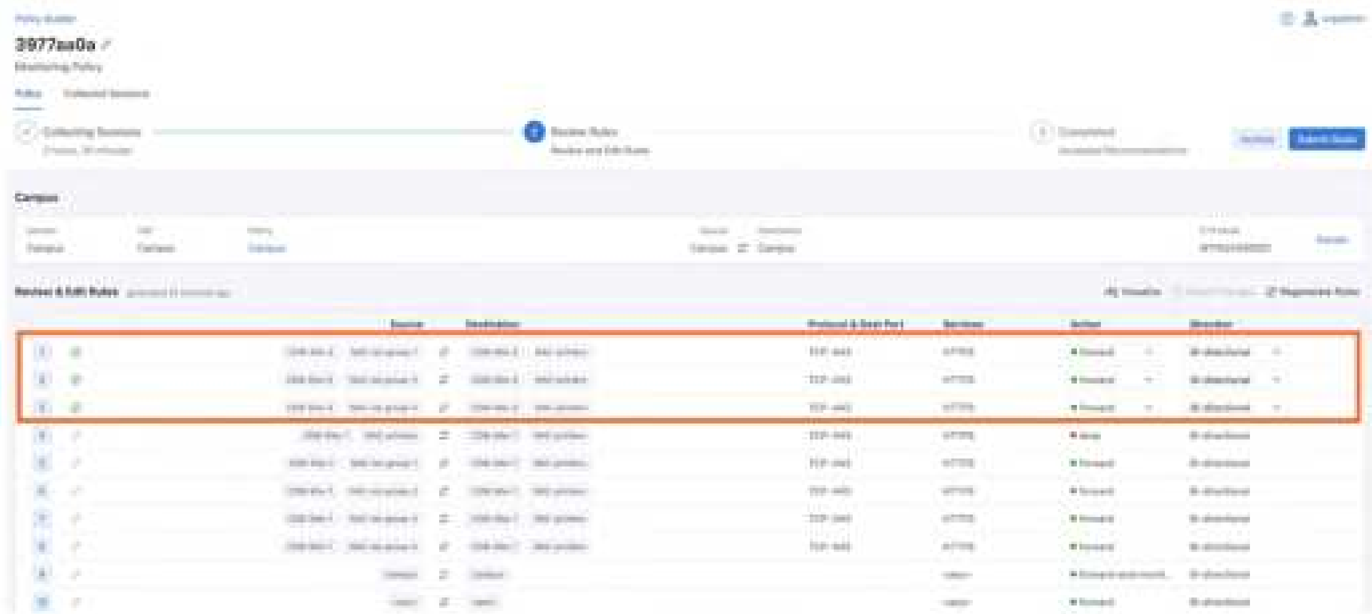
Now we are ready to push the rules to the switches by clicking **Submit Rule**. This will trigger the change control workflow

Now we can see the explicit rule above the default forward and monitor policy. Now any new traffic that shows up will first go through the explicitly defined rules, and if there is no match, will get processed by the default rule and show up in the collected session.

Now lets start some more traffic as shown below. We follow the same process and generate more rules using the same tags as before

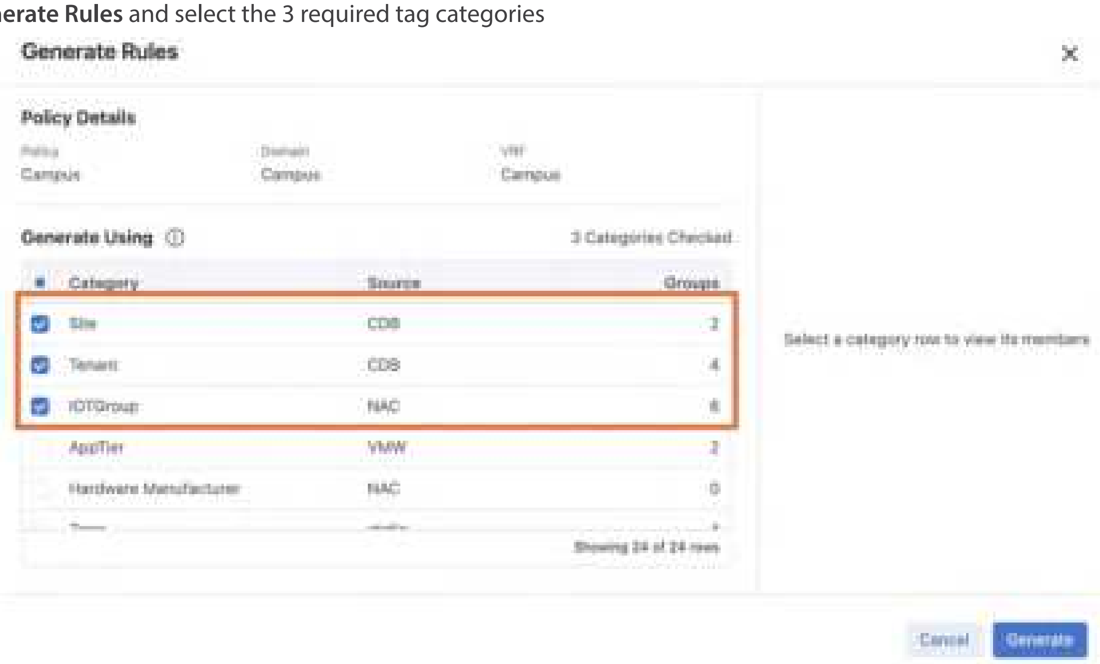




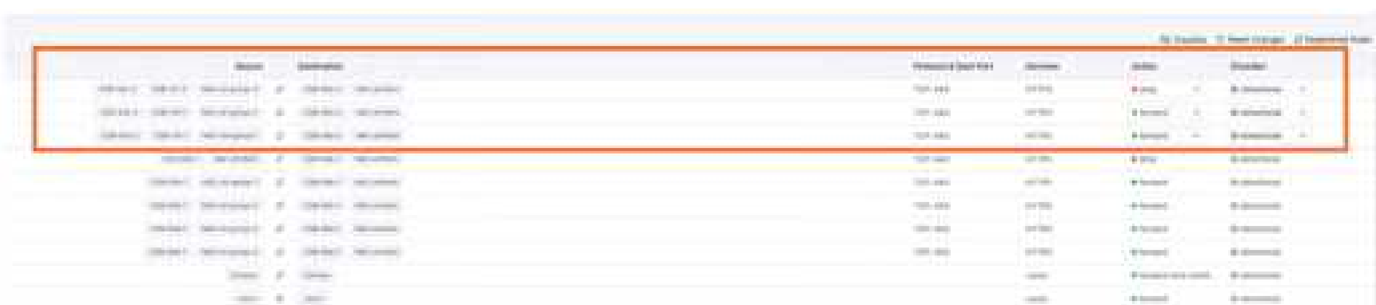


Now Site-2 has both Tnt-1 and Tnt-2. We only want Tnt-1 to be able to access the printers and not Tnt-2. We cannot do that with the set of policies that are present at the moment. So we can regenerate the policies and also include the Tenant tag to get a 3 tag policy recommendation that also includes the Tenant tags

Click on **Regenerate Rules** and select the 3 required tag categories



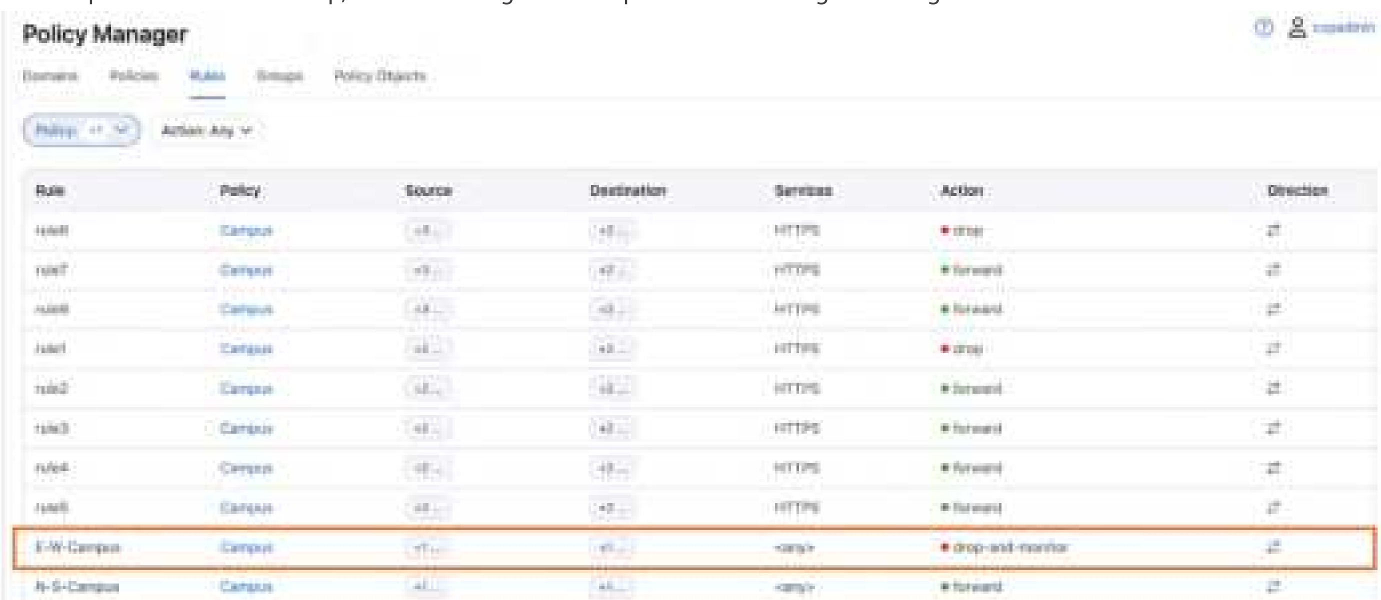
We can mark the action '**drop**' for the Tnt-2 rule specifically now



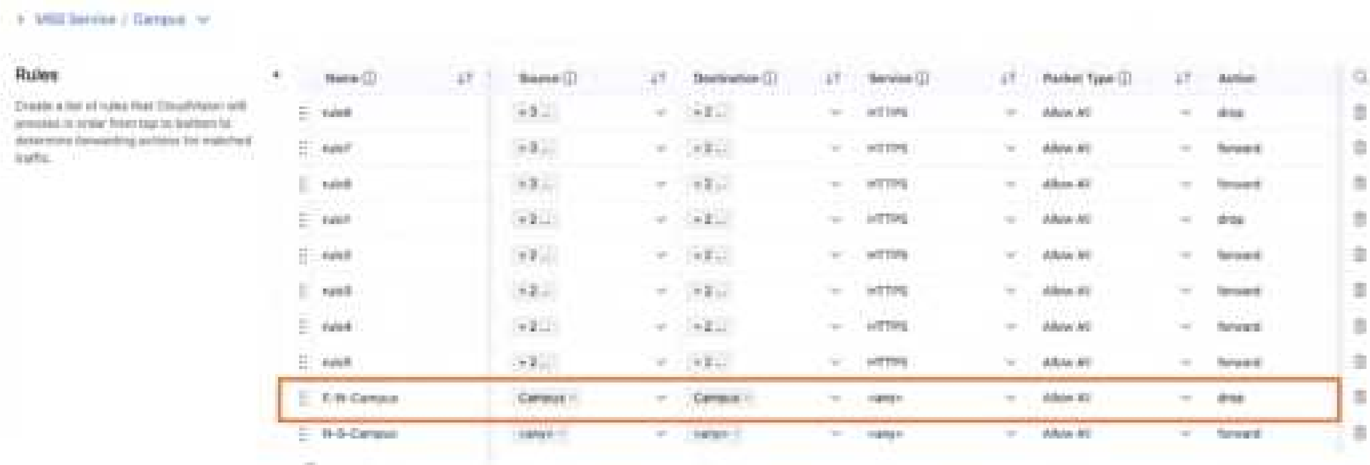
This way by combining multiple tags, we can get a more granular control of our policies

Once we push the rules, we will have more explicitly defined rules added to our existing policy

Continue monitoring and adding specific rules until you have accounted for all typical network traffic. This point is reached when no new flows appear in the collected sessions, as all traffic matches the explicitly defined rules. We can then move to the zero trust posture by converting the default E-W rule to a drop and monitor. Do this by navigating to the MSS Service studio, changing the E-W-Campus rule's action to 'drop,' and submitting the workspace and executing the change control.



Any new traffic that doesn't match the explicitly defined rules will be dropped. The corresponding flow will appear in the collected session due to the 'drop and monitor' action. This allows us to convert the dropped traffic into an explicit policy, using the same procedure outlined above.



Traffic intended to travel North-South will not match any explicitly defined rules or the default East-West-Campus rule. Instead, this traffic will hit the North-South-Campus rule and be processed according to the defined action. In this instance, it is marked as forward. As a result, it will adhere to standard routing/forwarding and reach its destination. A monitor action can be added to this rule (similar to the East-West-Campus rule), and explicit policies can be created for North-South traffic, mirroring the approach taken for East-West traffic.

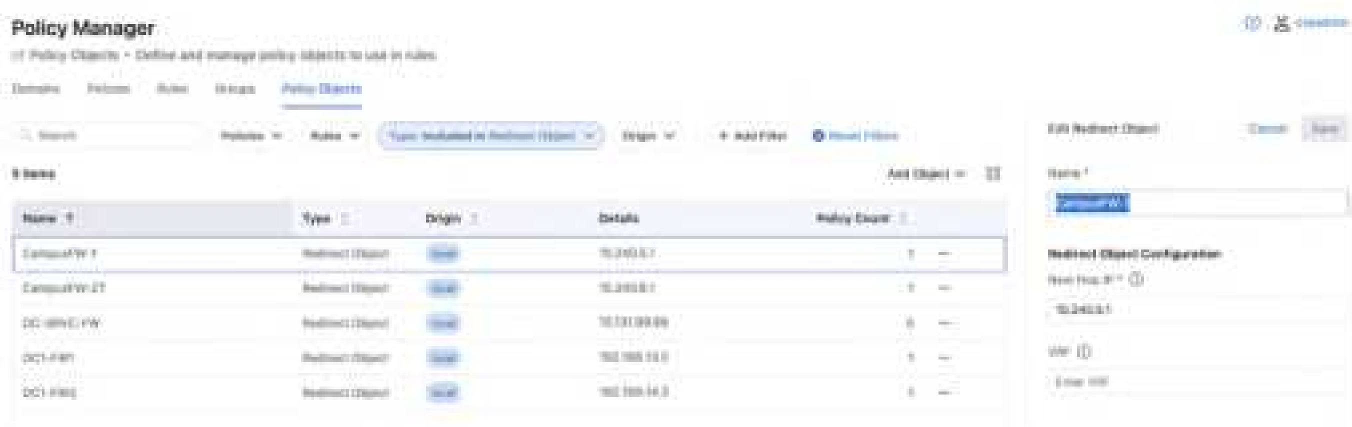
Traffic Redirection

MSS can create zero trust policies to enforce traffic within a zone/VRF that would typically go uninspected by the firewall. Traffic for another zone/VRF is routed to the firewall for inspection via the default N-S-Campus rule.

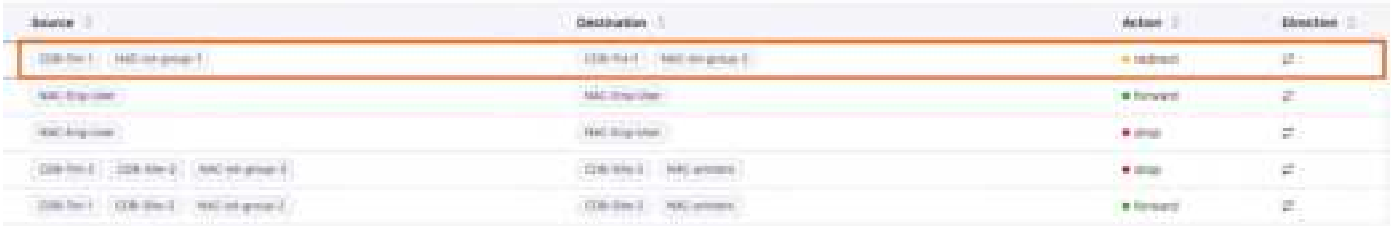
However, it is possible to redirect specific intrazone traffic to the firewall for deep packet inspection as well. This is achieved by defining a redirect object which contains the firewall IP address.

Traffic Redirection Configuration

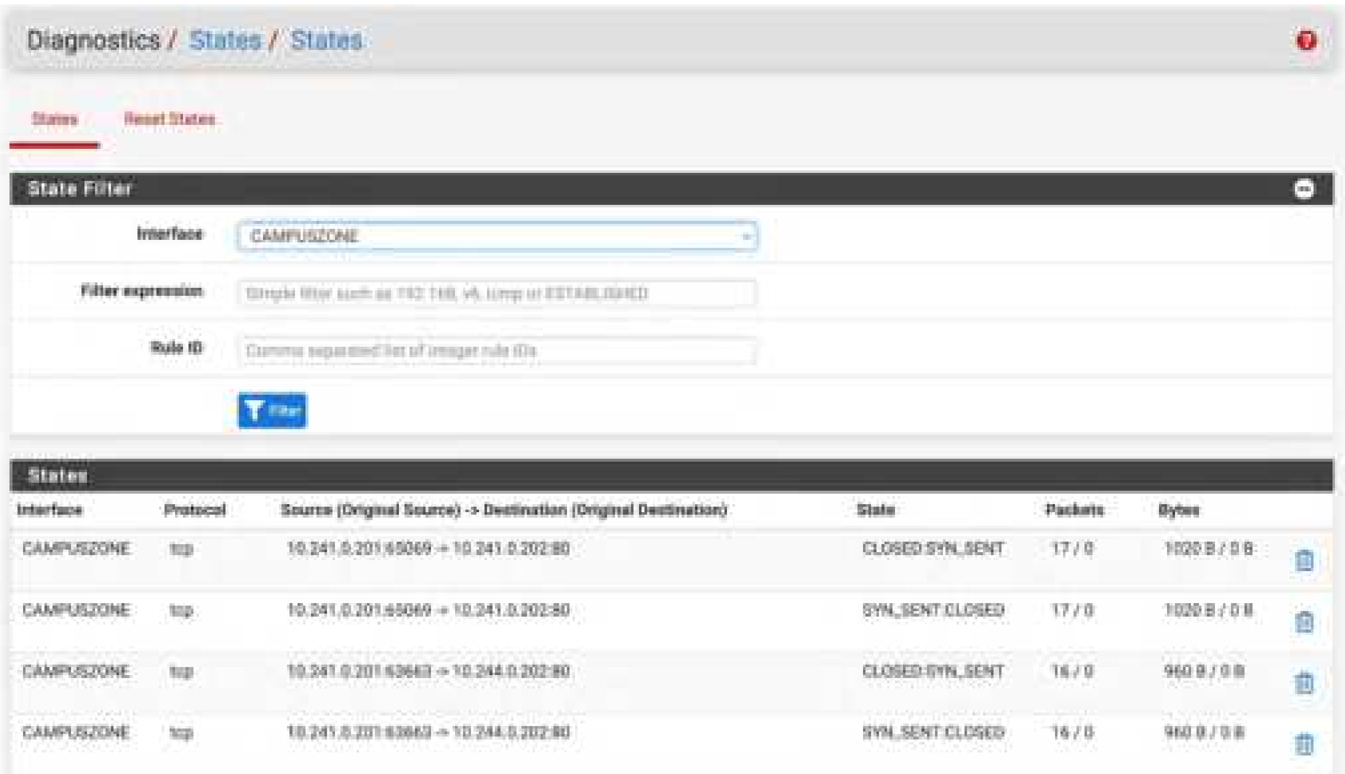
A redirect object will be created for traffic redirection—this is similar to creating a monitor object for monitoring. The next hop IP is the IP of the firewall interface when you want to redirect traffic



Traffic can be redirected to the firewall interface matching the redirect object IP, once the redirect object has been defined in the policy.



The firewall interface should display the traffic as long as the IP address can be reached.



Hardware and Scale

For hardware support, scale and licensing requirements please refer to the ZTN datasheet

Additional Resources

MSS Datasheet

MSS Deployment Guide

ZTX Datasheet

ZTX Deployment Guide

Santa Clara—Corporate Headquarters
5453 Great America Parkway,
Santa Clara, CA 95054

Phone: +1-408-547-5500
Fax: +1-408-538-8920
Email: info@arista.com

Ireland—International Headquarters
3130 Atlantic Avenue
Westpark Business Campus
Shannon, Co. Clare
Ireland

Vancouver—R&D Office
9200 Glenlyon Pkwy, Unit 300
Burnaby, British Columbia
Canada V5J 5J8

San Francisco—R&D and Sales Office 1390
Market Street, Suite 800
San Francisco, CA 94102

India—R&D Office
Global Tech Park, Tower A, 11th Floor
Marathahalli Outer Ring Road
Devarabeesanahalli Village, Varthur Hobli
Bangalore, India 560103

Singapore—APAC Administrative Office
9 Temasek Boulevard
#29-01, Suntec Tower Two
Singapore 038989

Nashua—R&D Office
10 Tara Boulevard
Nashua, NH 03062

Copyright © 2025 Arista Networks, Inc. All rights reserved. CloudVision, and EOS are registered trademarks and Arista Networks is a trademark of Arista Networks, Inc. All other company names are trademarks of their respective holders. Information in this document is subject to change without notice. Certain features may not yet be available. Arista Networks, Inc. assumes no responsibility for any errors that may appear in this document. August 13, 2025 07-0019-02