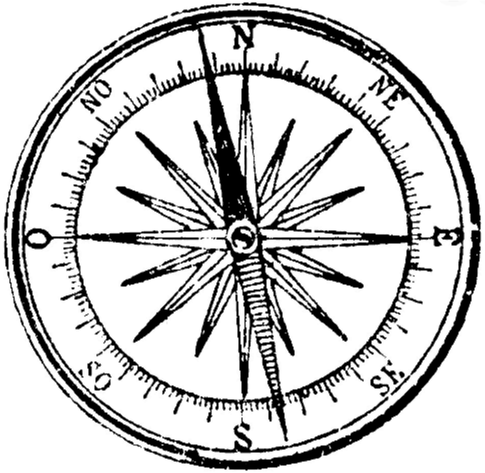
ARISTA

# HYBRID MULTICLOUD ORCHESTRATION

DIRECT

NATIVE



3RD
PARTY

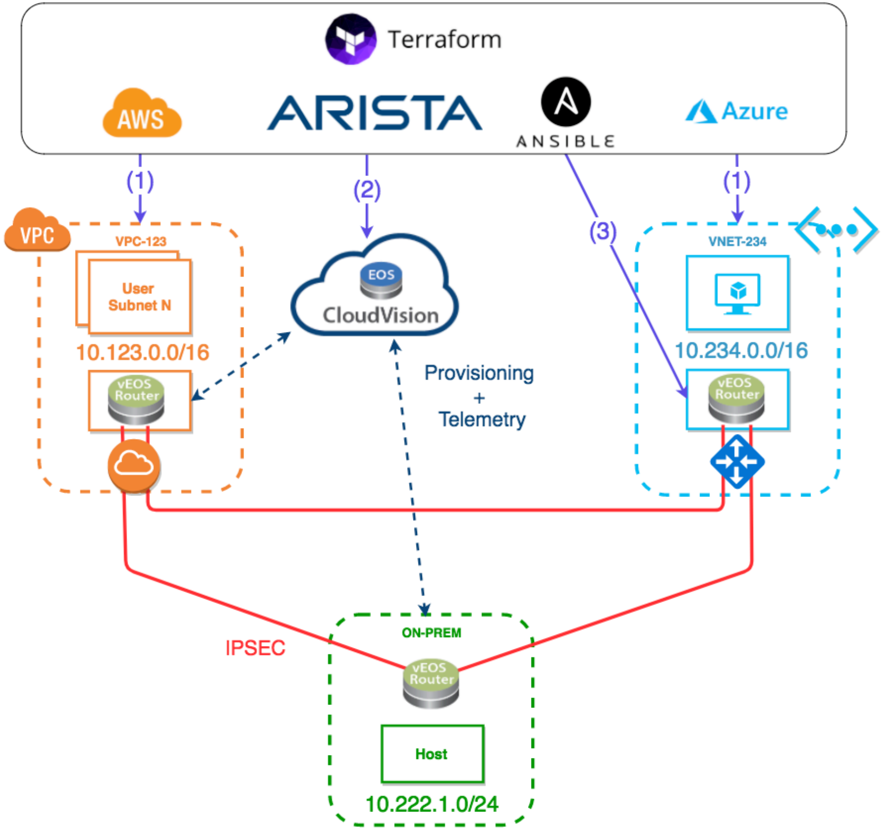
INTERNET
IPSEC

|  | Ansible | Terraform | CloudFormation |
|---|---|---|---|
| Syntax | YAML | HCL / JSON | JSON |
| State Management | Some | Yes | No |
| Manage already created resources | Yes | Hard | No |
| Providers support | +++ | ++ | AWS |

# Hybrid Multi-Cloud orchestration

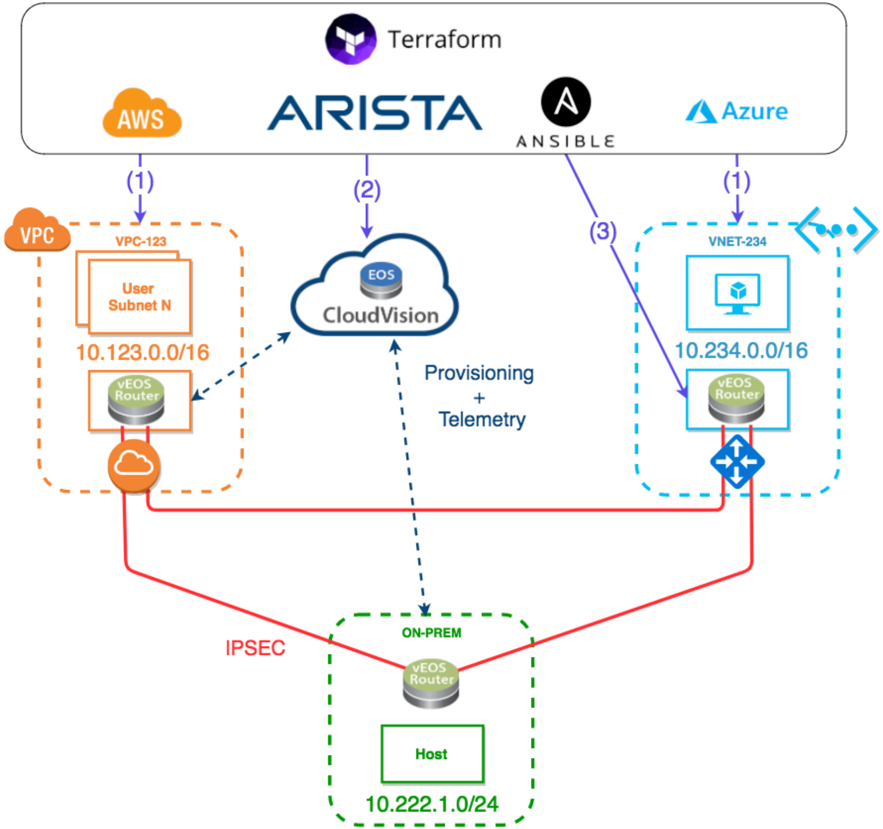
amazon
web services
Azure
vmware
EOS
CloudVision
vEOS
Router
KVM

# Hybrid Multi-Cloud orchestration



1. Use Terraform to orchestrate hybrid multi-Cloud deployments
2. Use CVP API to configure vEOS Routers
3. Use Ansible to configure vEOS Router (or any third party), integrated into Terraform.

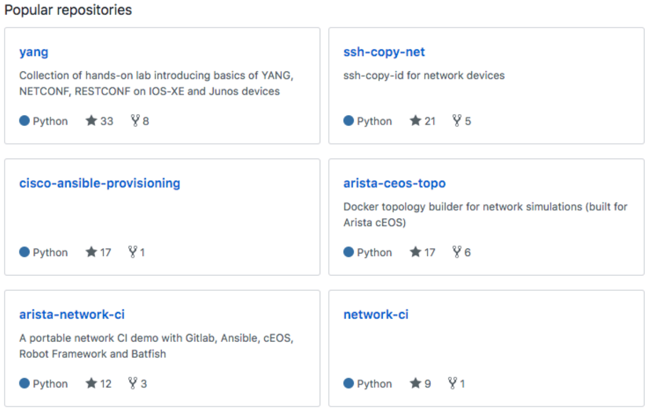
# Hybrid Multi-Cloud orchestration demo



https://github.com/networkop/tf-mcloud-demo

A lot more of really good stuff here:

https://github.com/networkop/



Popular repositories

**yang**
Collection of hands-on lab introducing basics of YANG, NETCONF, RESTCONF on IOS-XE and Junos devices
● Python ★ 33 ⑂ 8

**ssh-copy-net**
ssh-copy-id for network devices
● Python ★ 21 ⑂ 5

**cisco-ansible-provisioning**
● Python ★ 17 ⑂ 1

**arista-ceos-topo**
Docker topology builder for network simulations (built for Arista cEOS)
● Python ★ 17 ⑂ 6

**arista-network-ci**
A portable network CI demo with Gitlab, Ansible, cEOS, Robot Framework and Batfish
● Python ★ 12 ⑂ 3

**network-ci**
● Python ★ 9 ⑂ 1

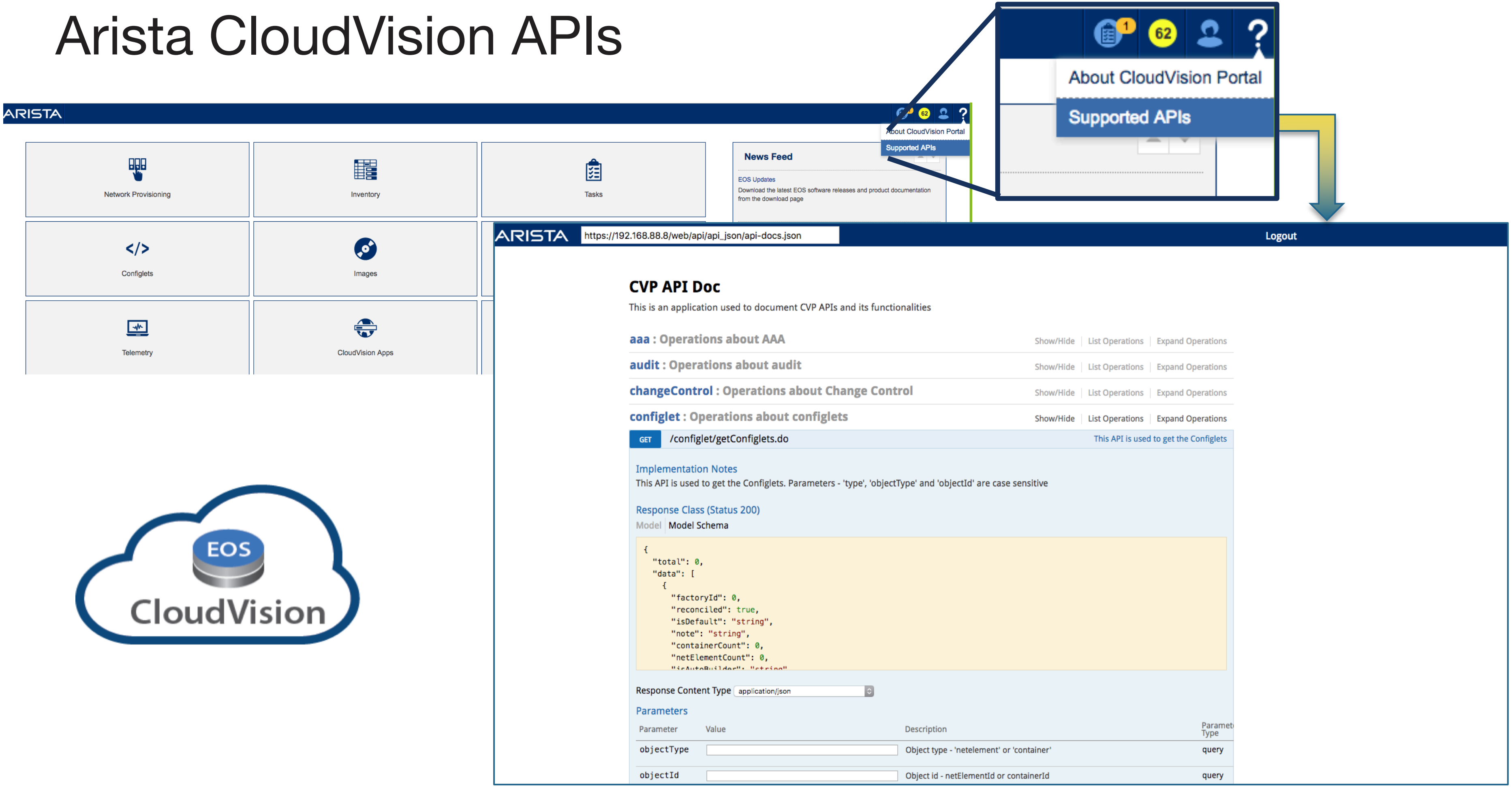
# Arista CloudVision APIs

- CVP API can be used directly using http calls with POST method sending JSON data
- RESTful HTTP methods and needed JSON data structures can be found at http://<insert_your_CVP_IP_HERE>/web/api
- Can be cumbersome to use the CVP API direct so CVP RAC wrapper is available to simplify
- CVP RAC is created and maintained by Arista
- CVP RAC exists for several languages like Python, Ruby and GO
- Everything that can be done in CVP GUI can be done via the API

https://github.com/aristanetworks/cvprac

https://github.com/networkop/terraform-cvp

# Arista CloudVision APIs



## CVP API Doc

This is an application used to document CVP APIs and its functionalities

**aaa** : Operations about AAA — Show/Hide | List Operations | Expand Operations

**audit** : Operations about audit — Show/Hide | List Operations | Expand Operations

**changeControl** : Operations about Change Control — Show/Hide | List Operations | Expand Operations

**configlet** : Operations about configlets — Show/Hide | List Operations | Expand Operations

GET /configlet/getConfiglets.do — This API is used to get the Configlets

Implementation Notes

This API is used to get the Configlets. Parameters - 'type', 'objectType' and 'objectId' are case sensitive

Response Class (Status 200)

Model | Model Schema

```
{
  "total": 0,
  "data": [
    {
      "factoryId": 0,
      "reconciled": true,
      "isDefault": "string",
      "note": "string",
      "containerCount": 0,
      "netElementCount": 0,
      "isAutoBuilder": "string"
```

Response Content Type: application/json

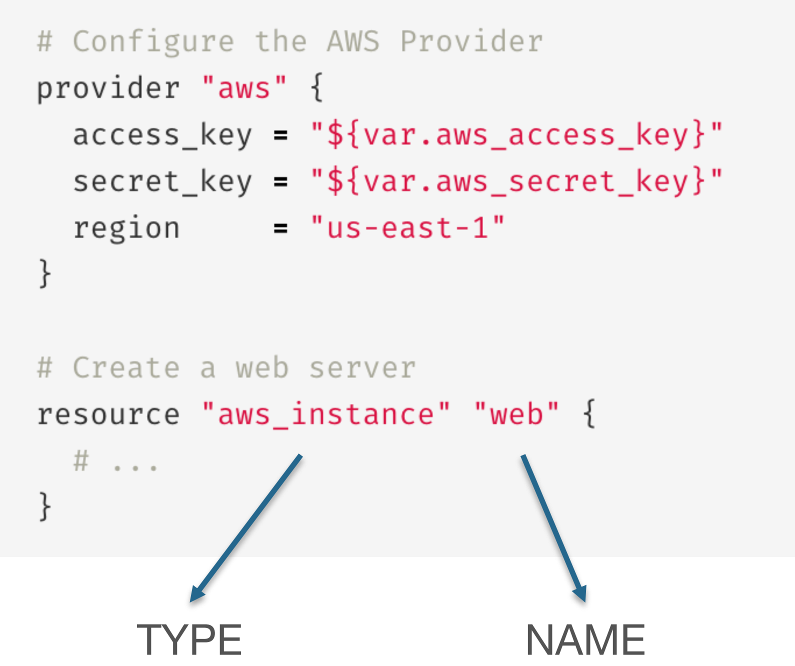
Parameters

| Parameter | Value | Description | Parameter Type |
|---|---|---|---|
| objectType | | Object type - 'netelement' or 'container' | query |
| objectId | | Object id - netElementId or containerId | query |

# What is Terraform?

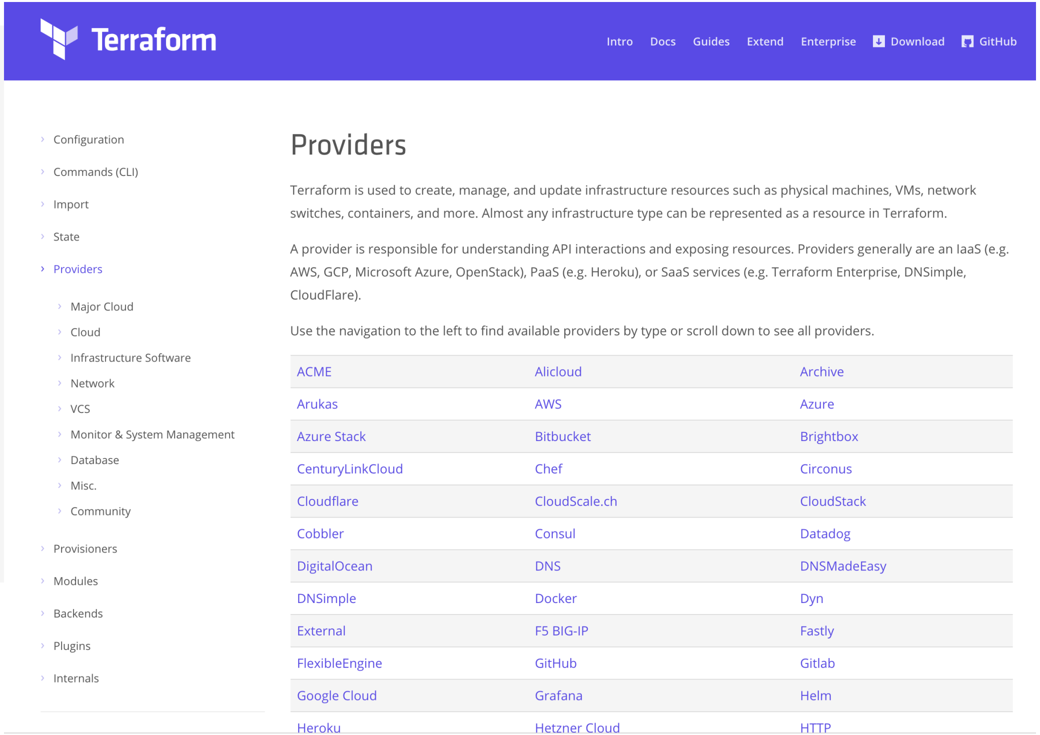
- Infrastructure as code ← This is exactly what this demo is about
- A tool to manage virtual server life cycles (AWS, Azure, VMWare, etc.)
- A tool to manage supporting services (DNS, Email)
- A tool to manage system servies (MySQL, PostgreSQL)
- Configuration files can be JSON or HCL (**HashiCorp** configuration language)
- Created by Hashicorp (Vagrant, Vault, et al.)
- Written in Go

# Terraform – Working with resources and providers



```
# Configure the AWS Provider
provider "aws" {
  access_key = "${var.aws_access_key}"
  secret_key = "${var.aws_secret_key}"
  region     = "us-east-1"
}

# Create a web server
resource "aws_instance" "web" {
  # ...
}
```

https://www.terraform.io/docs/providers/



## Providers

Terraform is used to create, manage, and update infrastructure resources such as physical machines, VMs, network switches, containers, and more. Almost any infrastructure type can be represented as a resource in Terraform.

A provider is responsible for understanding API interactions and exposing resources. Providers generally are an IaaS (e.g. AWS, GCP, Microsoft Azure, OpenStack), PaaS (e.g. Heroku), or SaaS services (e.g. Terraform Enterprise, DNSimple, CloudFlare).

Use the navigation to the left to find available providers by type or scroll down to see all providers.

| | | |
|---|---|---|
| ACME | Alicloud | Archive |
| Arukas | AWS | Azure |
| Azure Stack | Bitbucket | Brightbox |
| CenturyLinkCloud | Chef | Circonus |
| Cloudflare | CloudScale.ch | CloudStack |
| Cobbler | Consul | Datadog |
| DigitalOcean | DNS | DNSMadeEasy |
| DNSimple | Docker | Dyn |
| External | F5 BIG-IP | Fastly |
| FlexibleEngine | GitHub | Gitlab |
| Google Cloud | Grafana | Helm |
| Heroku | Hetzner Cloud | HTTP |

# Terraform – Working with variables

- Terraform loads all files ending in .tf in a directory
- If a default value is set, the variable is optional. Otherwise, the variable is required, so Terraform will prompt you for the values for unset string variables during run time.
- Terraform will also read environment variables in the form of TF_VAR_name

**variables.tf**

```
variable "access_key" {}
variable "secret_key" {}
variable "region" {
  default = "us-east-1"
}
```

### Assigning variables from a file:

**terraform.tfvars**

```
access_key = "foo"
secret_key = "bar"
```

export TF_VAR_name="baz"

### Using variables in configuration:

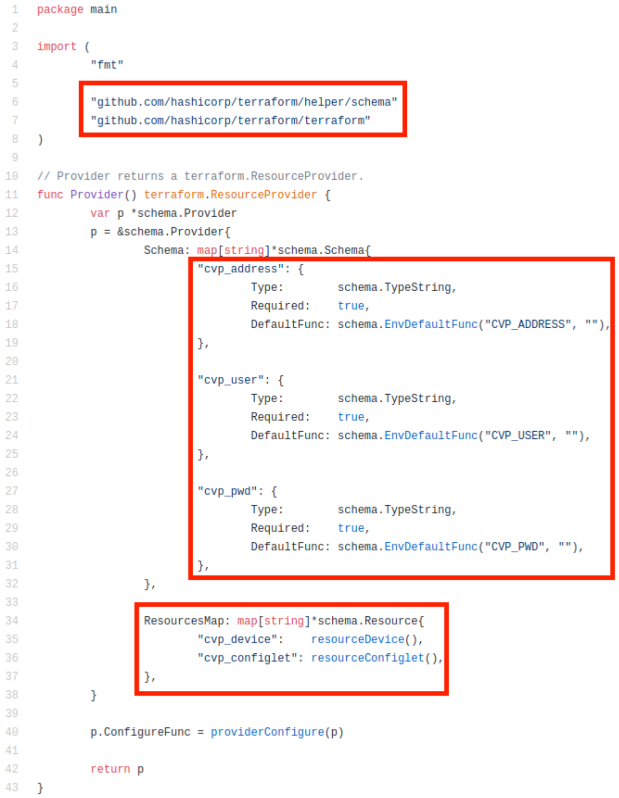
**main.tf**

```
provider "aws" {
  access_key = "${var.access_key}"
  secret_key = "${var.secret_key}"
  region     = "${var.region}"
}
```

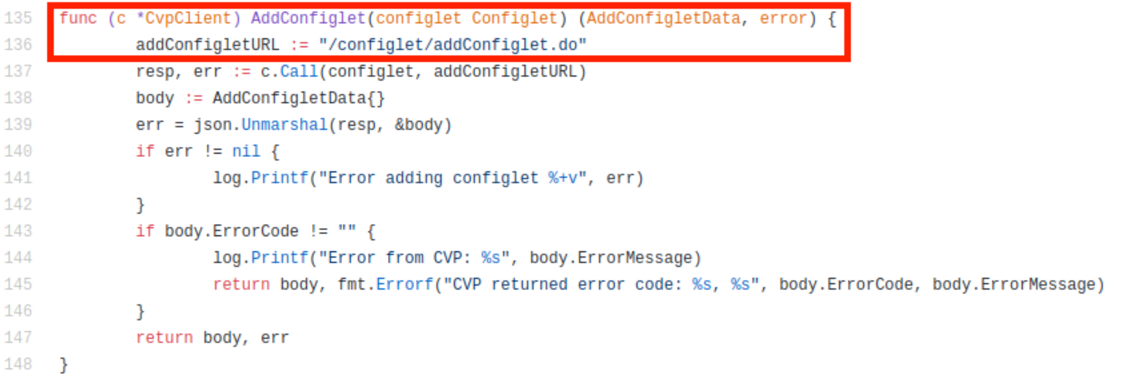
# Terraform – CVP

https://github.com/networkop/terraform-cvp

https://github.com/networkop/cvpgo

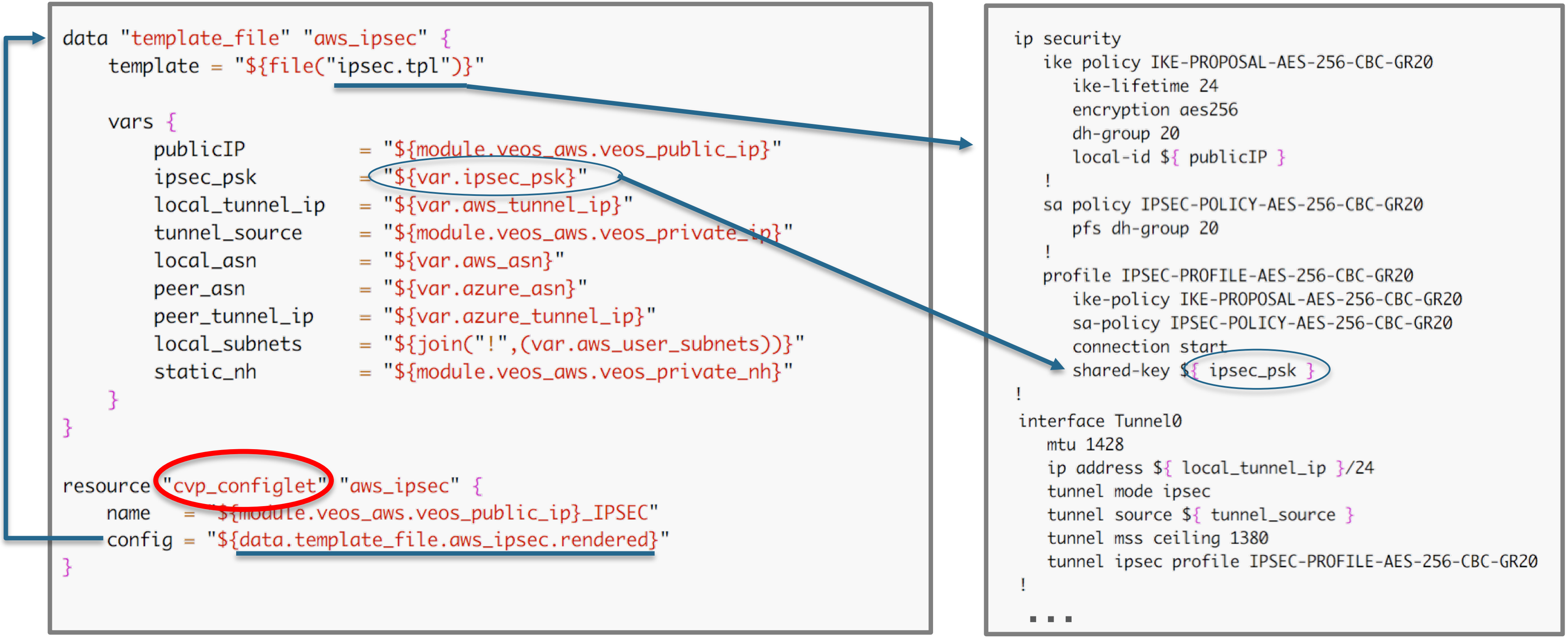


```go
package main

import (
        "fmt"

        "github.com/hashicorp/terraform/helper/schema"
        "github.com/hashicorp/terraform/terraform"
)

// Provider returns a terraform.ResourceProvider.
func Provider() terraform.ResourceProvider {
        var p *schema.Provider
        p = &schema.Provider{
                Schema: map[string]*schema.Schema{
                        "cvp_address": {
                                Type:        schema.TypeString,
                                Required:    true,
                                DefaultFunc: schema.EnvDefaultFunc("CVP_ADDRESS", ""),
                        },

                        "cvp_user": {
                                Type:        schema.TypeString,
                                Required:    true,
                                DefaultFunc: schema.EnvDefaultFunc("CVP_USER", ""),
                        },

                        "cvp_pwd": {
                                Type:        schema.TypeString,
                                Required:    true,
                                DefaultFunc: schema.EnvDefaultFunc("CVP_PWD", ""),
                        },
                },

                ResourcesMap: map[string]*schema.Resource{
                        "cvp_device":    resourceDevice(),
                        "cvp_configlet": resourceConfiglet(),
                },
        }

        p.ConfigureFunc = providerConfigure(p)

        return p
}
```



```go
func (c *CvpClient) AddConfiglet(configlet Configlet) (AddConfigletData, error) {
        addConfigletURL := "/configlet/addConfiglet.do"
        resp, err := c.Call(configlet, addConfigletURL)
        body := AddConfigletData{}
        err = json.Unmarshal(resp, &body)
        if err != nil {
                log.Printf("Error adding configlet %+v", err)
        }
        if body.ErrorCode != "" {
                log.Printf("Error from CVP: %s", body.ErrorMessage)
                return body, fmt.Errorf("CVP returned error code: %s, %s", body.ErrorCode, body.ErrorMessage)
        }
        return body, err
}
```

# Terraform – CVP resource: Configlet



**main.tf**

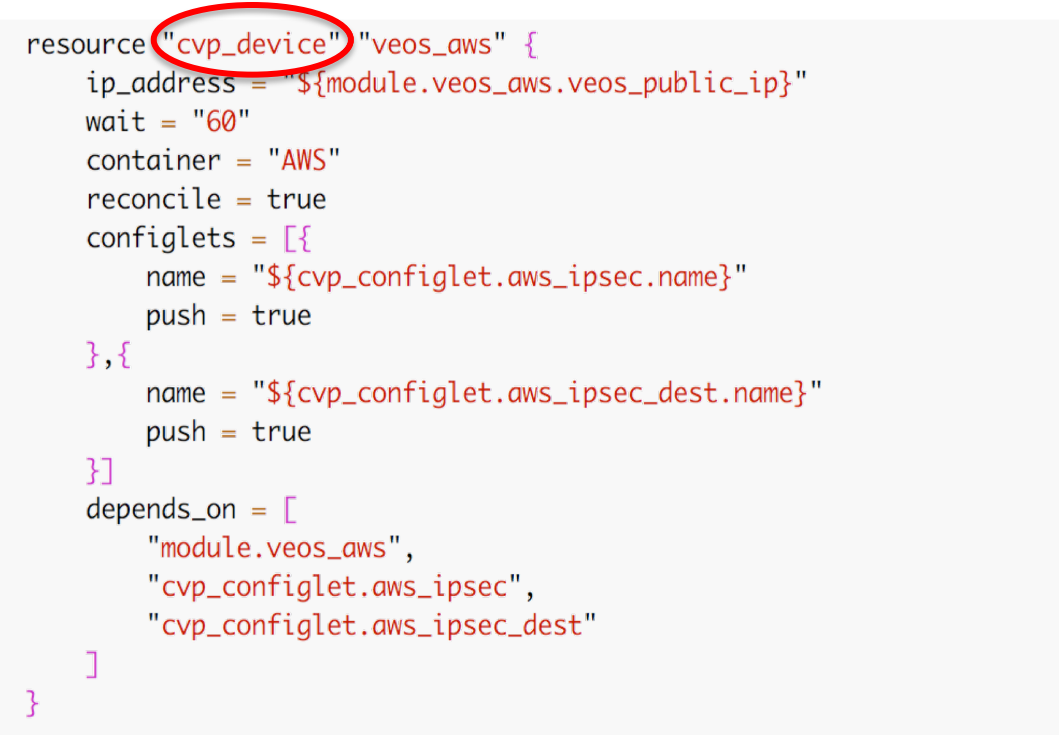
```
data "template_file" "aws_ipsec" {
    template = "${file("ipsec.tpl")}"

    vars {
        publicIP            = "${module.veos_aws.veos_public_ip}"
        ipsec_psk           = "${var.ipsec_psk}"
        local_tunnel_ip     = "${var.aws_tunnel_ip}"
        tunnel_source       = "${module.veos_aws.veos_private_ip}"
        local_asn           = "${var.aws_asn}"
        peer_asn            = "${var.azure_asn}"
        peer_tunnel_ip      = "${var.azure_tunnel_ip}"
        local_subnets       = "${join("!",(var.aws_user_subnets))}"
        static_nh           = "${module.veos_aws.veos_private_nh}"
    }
}

resource "cvp_configlet" "aws_ipsec" {
    name   = "${module.veos_aws.veos_public_ip}_IPSEC"
    config = "${data.template_file.aws_ipsec.rendered}"
}
```

**ipsec.tpl**

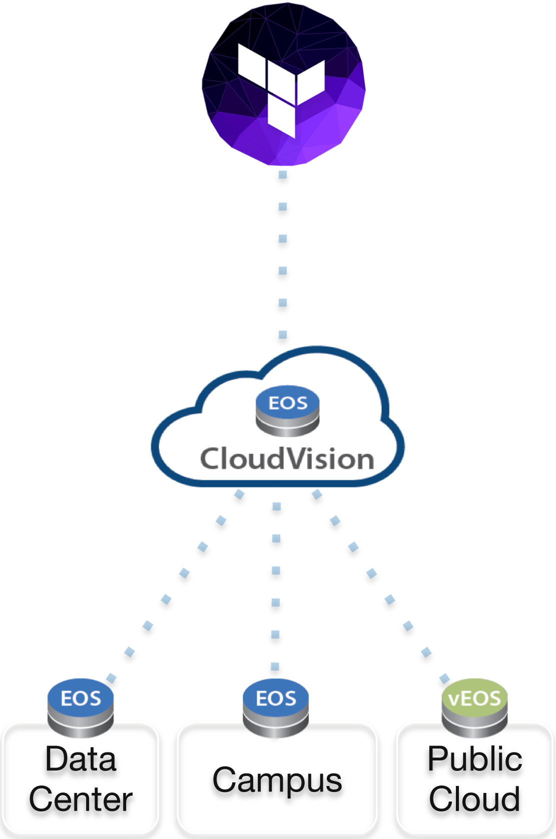
```
ip security
   ike policy IKE-PROPOSAL-AES-256-CBC-GR20
      ike-lifetime 24
      encryption aes256
      dh-group 20
      local-id ${ publicIP }
   !
   sa policy IPSEC-POLICY-AES-256-CBC-GR20
      pfs dh-group 20
   !
   profile IPSEC-PROFILE-AES-256-CBC-GR20
      ike-policy IKE-PROPOSAL-AES-256-CBC-GR20
      sa-policy IPSEC-POLICY-AES-256-CBC-GR20
      connection start
      shared-key ${ ipsec_psk }
!
interface Tunnel0
   mtu 1428
   ip address ${ local_tunnel_ip }/24
   tunnel mode ipsec
   tunnel source ${ tunnel_source }
   tunnel mss ceiling 1380
   tunnel ipsec profile IPSEC-PROFILE-AES-256-CBC-GR20
!
 ...
```

**Cloud Builders**

# Terraform – CVP resource: Device



```
resource "cvp_device" "veos_aws" {
    ip_address = "${module.veos_aws.veos_public_ip}"
    wait = "60"
    container = "AWS"
    reconcile = true
    configlets = [{
        name = "${cvp_configlet.aws_ipsec.name}"
        push = true
    },{
        name = "${cvp_configlet.aws_ipsec_dest.name}"
        push = true
    }]
    depends_on = [
        "module.veos_aws",
        "cvp_configlet.aws_ipsec",
        "cvp_configlet.aws_ipsec_dest"
    ]
}
```



https://github.com/networkop/terraform-cvp

# What is Ansible?

- Super simple, yet extremely powerful tool to automate software provisioning, configuration management, and application deployment.
- Very low barrier for entry, no coding skills needed
- Uses SSH or API as transport
- Not just for network devices – servers, cloud providers, VMware, whatever
- Python based, so easily extended
- YAML driven, making it extremely easy to use and is human readable

# Ansible Modules

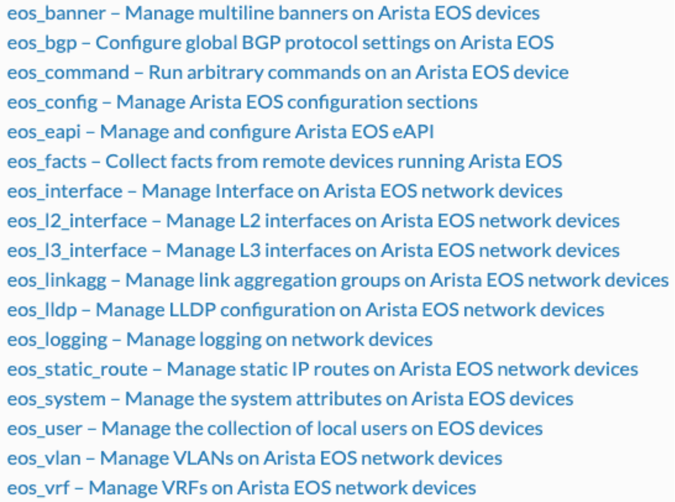
1200+ built-in modules including:

apt, yum, copy, command, cron, dns, docker, easy_install, ec2 (amazon modules), file, filesystem, find, git, known_hosts, mysql, mongodb, nagios, npm, openstack, rax (rackspace), pip, shell, snmp_facts, **eos_*, cv_*…**

## eos_* Core Modules

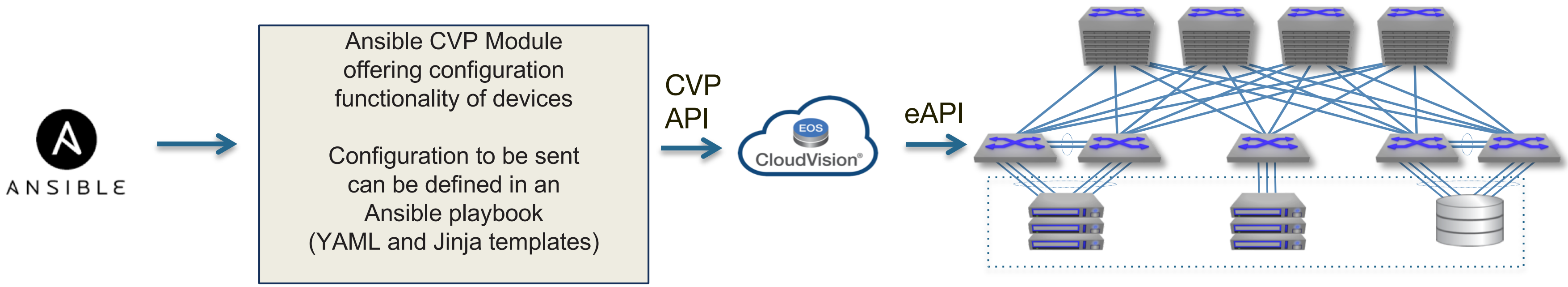
**Advantages**

- No third-party libraries needed
- No additional configuraton or client running on the switch
- Leverages eAPI/CLI(SSH) connection
- Work directly with running-configuration
- Easy to use and understand
- Offline-mode (generate configuration lines)



eos_banner – Manage multiline banners on Arista EOS devices
eos_bgp – Configure global BGP protocol settings on Arista EOS
eos_command – Run arbitrary commands on an Arista EOS device
eos_config – Manage Arista EOS configuration sections
eos_eapi – Manage and configure Arista EOS eAPI
eos_facts – Collect facts from remote devices running Arista EOS
eos_interface – Manage Interface on Arista EOS network devices
eos_l2_interface – Manage L2 interfaces on Arista EOS network devices
eos_l3_interface – Manage L3 interfaces on Arista EOS network devices
eos_linkagg – Manage link aggregation groups on Arista EOS network devices
eos_lldp – Manage LLDP configuration on Arista EOS network devices
eos_logging – Manage logging on network devices
eos_static_route – Manage static IP routes on Arista EOS network devices
eos_system – Manage the system attributes on Arista EOS devices
eos_user – Manage the collection of local users on EOS devices
eos_vlan – Manage VLANs on Arista EOS network devices
eos_vrf – Manage VRFs on Arista EOS network devices

# Ansible CVP Integration*



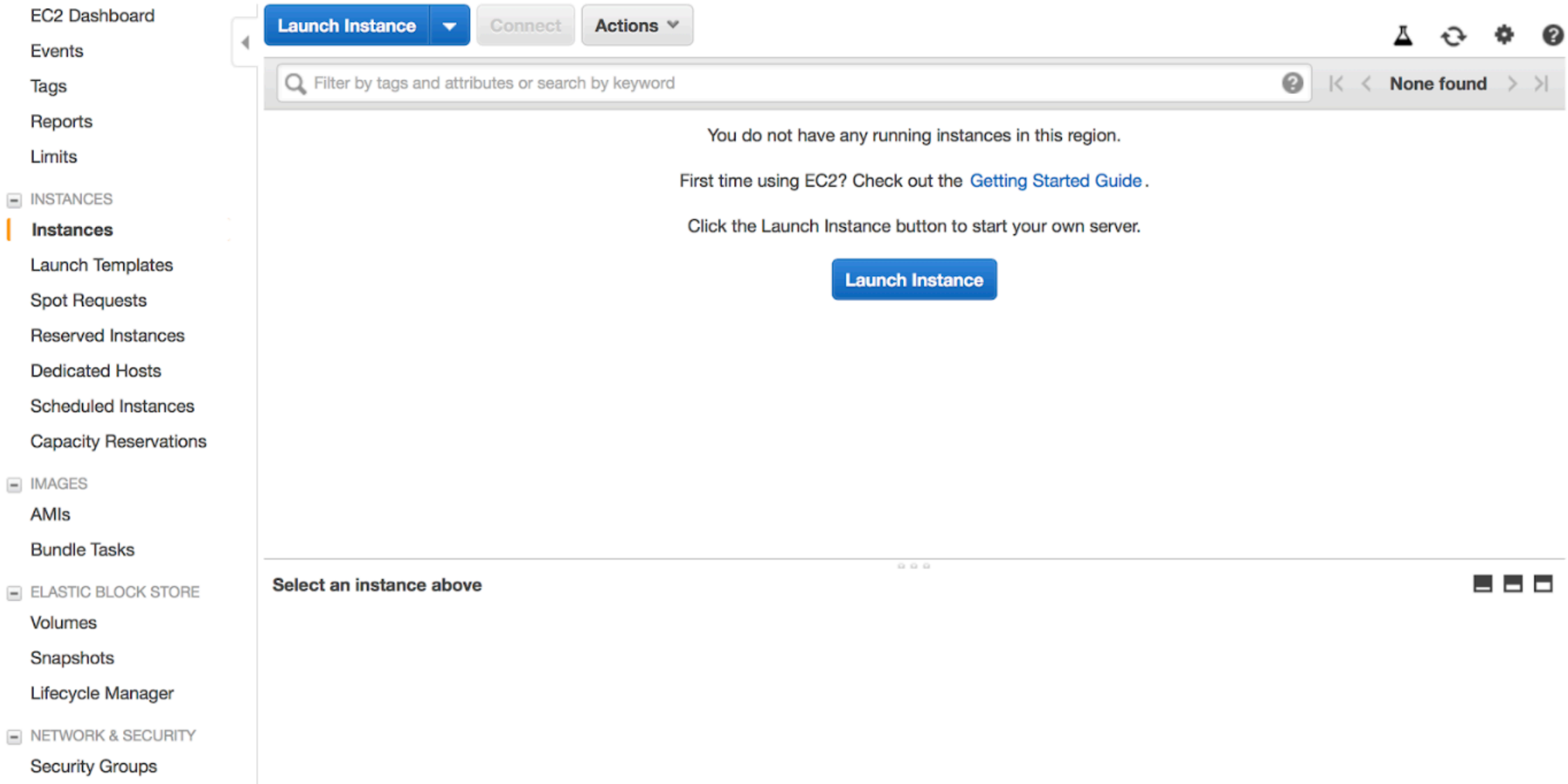
Ansible CVP Module offering configuration functionality of devices

Configuration to be sent can be defined in an Ansible playbook (YAML and Jinja templates)

- Makes delegation of discrete parts of the configuration possible
- Change control and rollback achieved when config is sent through CVP
- Firm procedure and review possibilities if integration with ServiceNow is used
- Gives traceability and audibility through logs in CVP for performed tasks

*NOT USED IN THIS DEMO

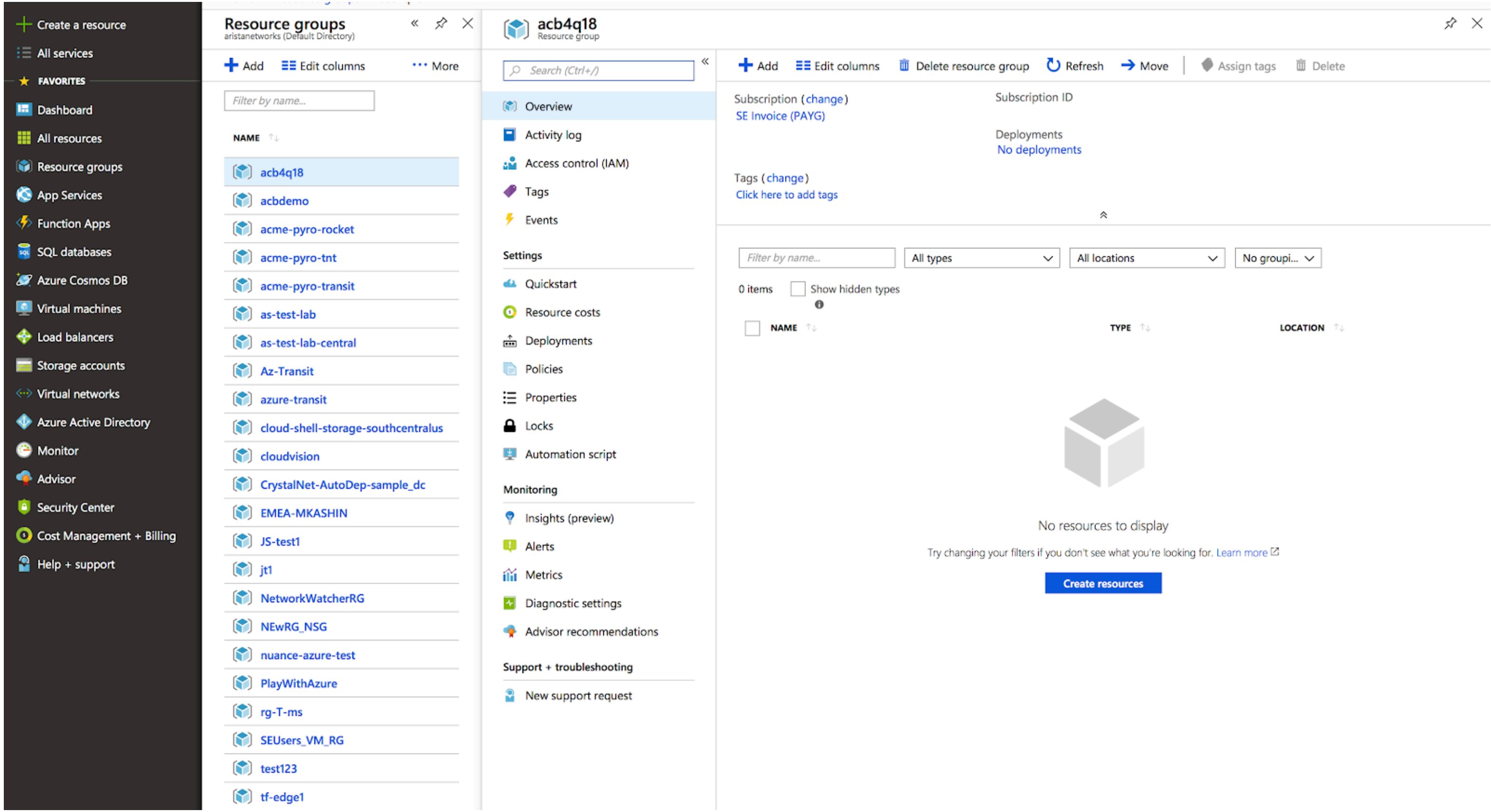
# "Talk is cheap. Show me the code."*



You do not have any running instances in this region.

First time using EC2? Check out the Getting Started Guide.

Click the Launch Instance button to start your own server.



- Create a resource
- All services
- FAVORITES
- Dashboard
- All resources
- Resource groups
- App Services
- Function Apps
- SQL databases
- Azure Cosmos DB
- Virtual machines
- Load balancers
- Storage accounts
- Virtual networks
- Azure Active Directory
- Monitor
- Advisor
- Security Center
- Cost Management + Billing
- Help + support

**Resource groups**
aristanetworks (Default Directory)

Add | Edit columns | More

Filter by name...

NAME

- acb4q18
- acbdemo
- acme-pyro-rocket
- acme-pyro-tnt
- acme-pyro-transit
- as-test-lab
- as-test-lab-central
- Az-Transit
- azure-transit
- cloud-shell-storage-southcentralus
- cloudvision
- CrystalNet-AutoDep-sample_dc
- EMEA-MKASHIN
- JS-test1
- jt1
- NetworkWatcherRG
- NEwRG_NSG
- nuance-azure-test
- PlayWithAzure
- rg-T-ms
- SEUsers_VM_RG
- test123
- tf-edge1

**acb4q18**
Resource group

Search (Ctrl+/)

- Overview
- Activity log
- Access control (IAM)
- Tags
- Events

Settings

- Quickstart
- Resource costs
- Deployments
- Policies
- Properties
- Locks
- Automation script

Monitoring

- Insights (preview)
- Alerts
- Metrics
- Diagnostic settings
- Advisor recommendations

Support + troubleshooting

- New support request

Add | Edit columns | Delete resource group | Refresh | Move | Assign tags | Delete

Subscription (change)
SE Invoice (PAYG)

Subscription ID

Deployments
No deployments

Tags (change)
Click here to add tags

Filter by name... | All types | All locations | No groupi...

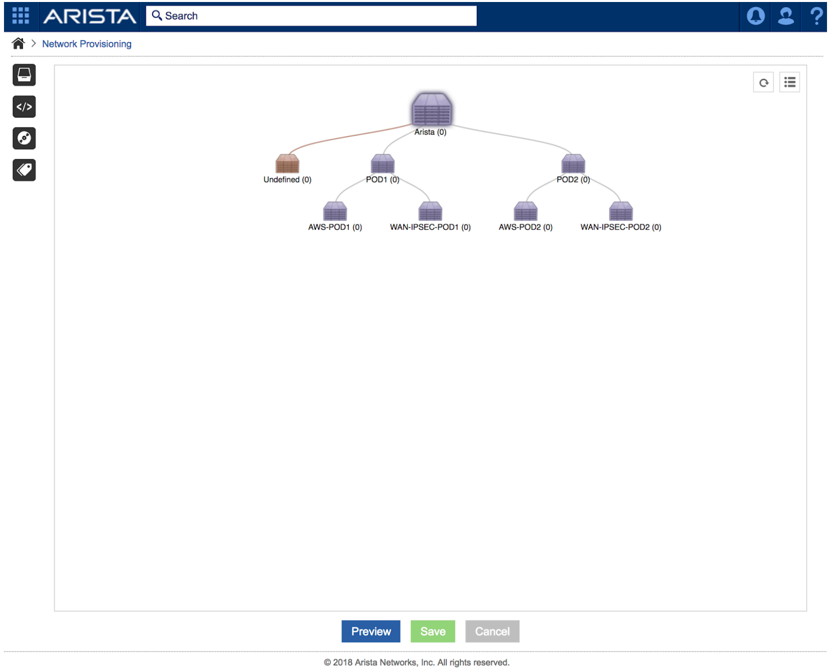
0 items □ Show hidden types

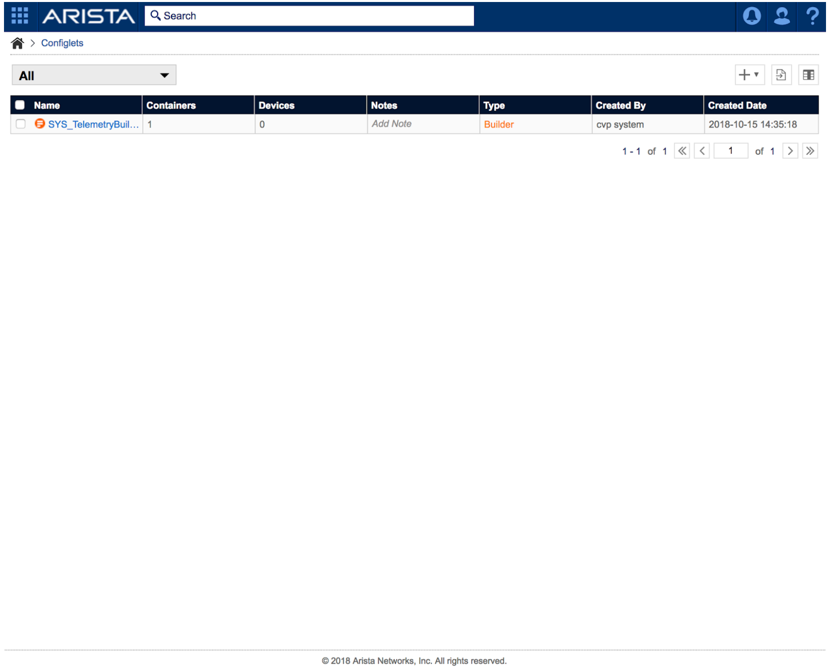
□ NAME | TYPE | LOCATION

No resources to display

Try changing your filters if you don't see what you're looking for. Learn more

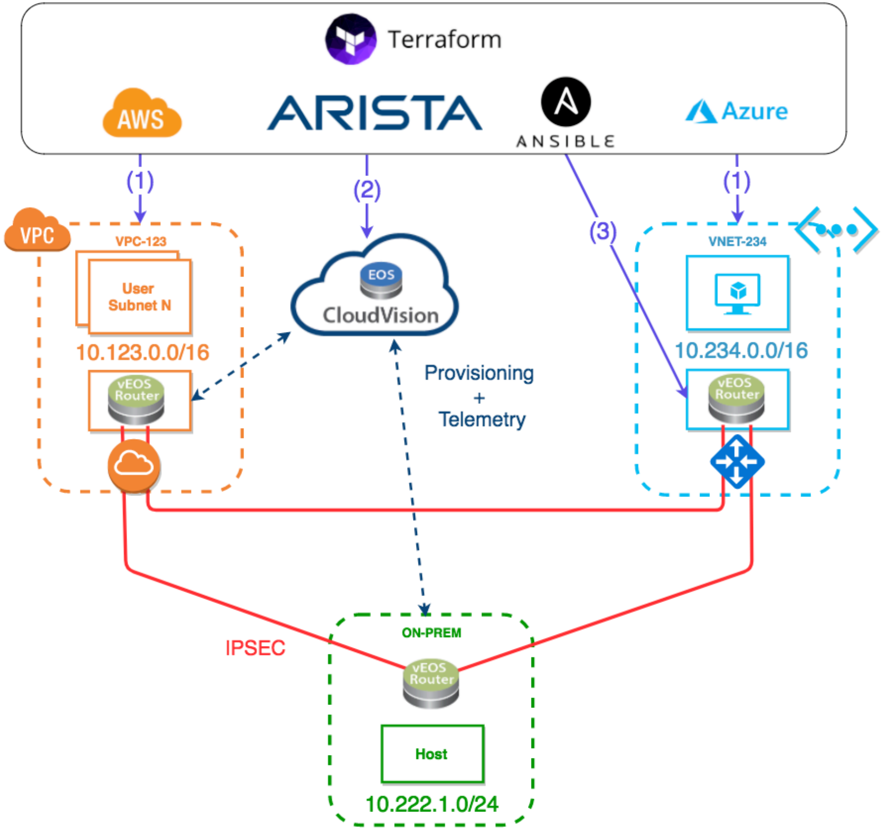
Create resources





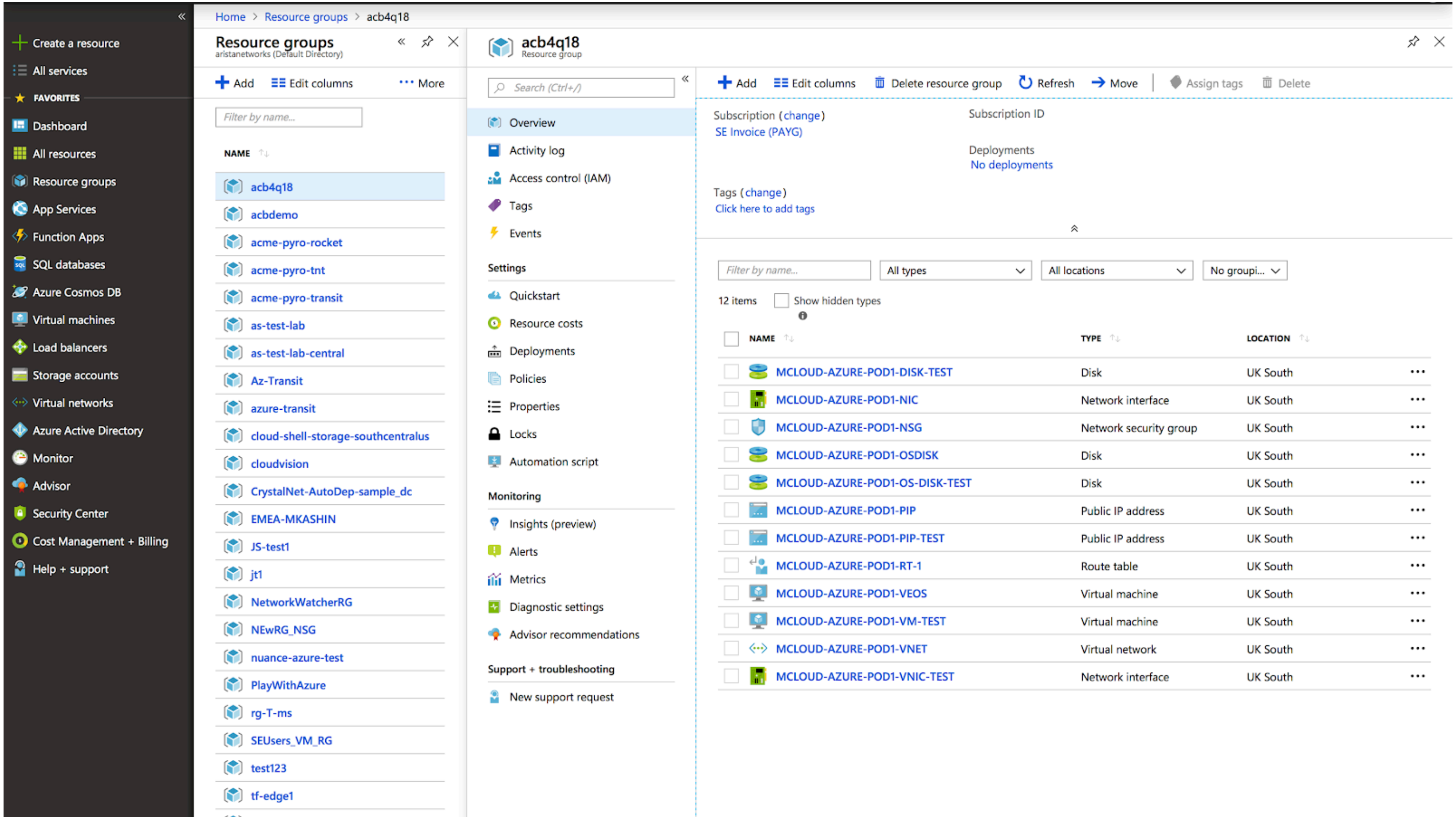
| □ Name | Containers | Devices | Notes | Type | Created By | Created Date |
|---|---|---|---|---|---|---|
| □ SYS_TelemetryBuil… | 1 | 0 | Add Note | Builder | cvp system | 2018-10-15 14:35:18 |

# Ok, I’m ready to see some action now…



# RUN!

- `terraform init`
- `terraform apply`
- `terraform destroy`



Home > Resource groups > acb4q18

**Resource groups**
aristanetworks (Default Directory)

Filter by name...

NAME

- acb4q18
- acbdemo
- acme-pyro-rocket
- acme-pyro-tnt
- acme-pyro-transit
- as-test-lab
- as-test-lab-central
- Az-Transit
- azure-transit
- cloud-shell-storage-southcentralus
- cloudvision
- CrystalNet-AutoDep-sample_dc
- EMEA-MKASHIN
- JS-test1
- jt1
- NetworkWatcherRG
- NEwRG_NSG
- nuance-azure-test
- PlayWithAzure
- rg-T-ms
- SEUsers_VM_RG
- test123
- tf-edge1

**acb4q18**
Resource group

Subscription (change)
SE Invoice (PAYG)

Subscription ID

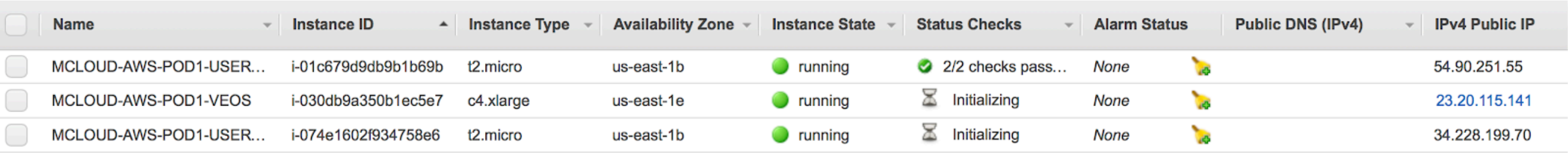
Deployments
No deployments

Tags (change)
Click here to add tags

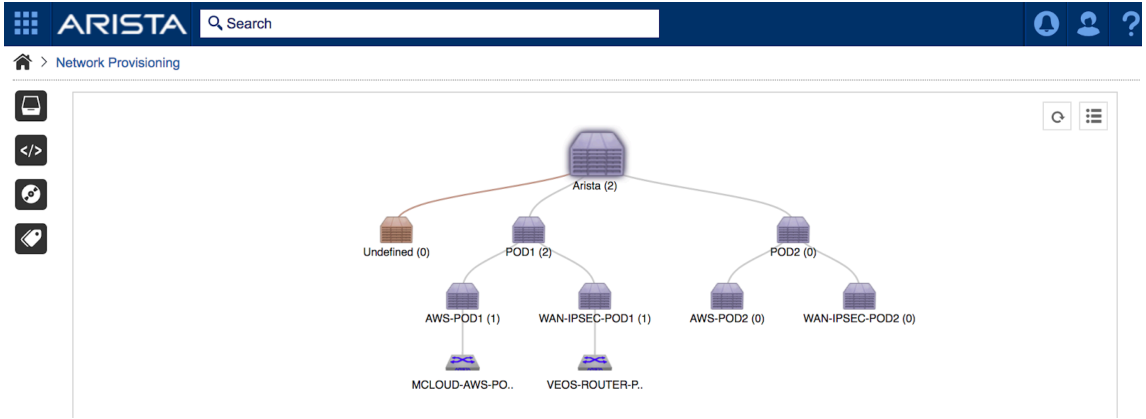
Filter by name... | All types | All locations | No groupi...

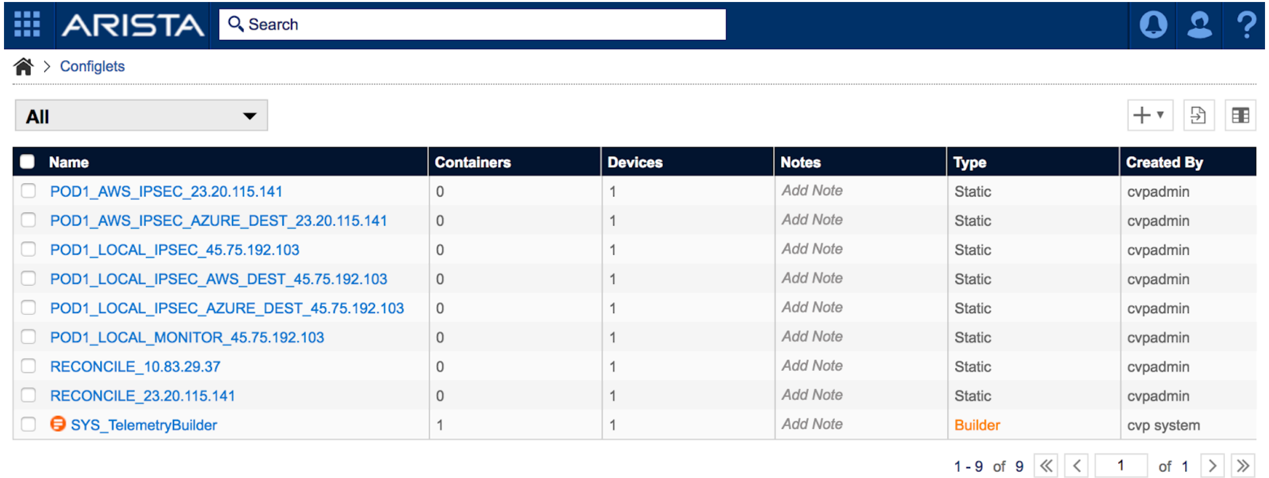
12 items □ Show hidden types

| □ | NAME | TYPE | LOCATION | |
|---|---|---|---|---|
| □ | MCLOUD-AZURE-POD1-DISK-TEST | Disk | UK South | ··· |
| □ | MCLOUD-AZURE-POD1-NIC | Network interface | UK South | ··· |
| □ | MCLOUD-AZURE-POD1-NSG | Network security group | UK South | ··· |
| □ | MCLOUD-AZURE-POD1-OSDISK | Disk | UK South | ··· |
| □ | MCLOUD-AZURE-POD1-OS-DISK-TEST | Disk | UK South | ··· |
| □ | MCLOUD-AZURE-POD1-PIP | Public IP address | UK South | ··· |
| □ | MCLOUD-AZURE-POD1-PIP-TEST | Public IP address | UK South | ··· |
| □ | MCLOUD-AZURE-POD1-RT-1 | Route table | UK South | ··· |
| □ | MCLOUD-AZURE-POD1-VEOS | Virtual machine | UK South | ··· |
| □ | MCLOUD-AZURE-POD1-VM-TEST | Virtual machine | UK South | ··· |
| □ | MCLOUD-AZURE-POD1-VNET | Virtual network | UK South | ··· |
| □ | MCLOUD-AZURE-POD1-VNIC-TEST | Network interface | UK South | ··· |

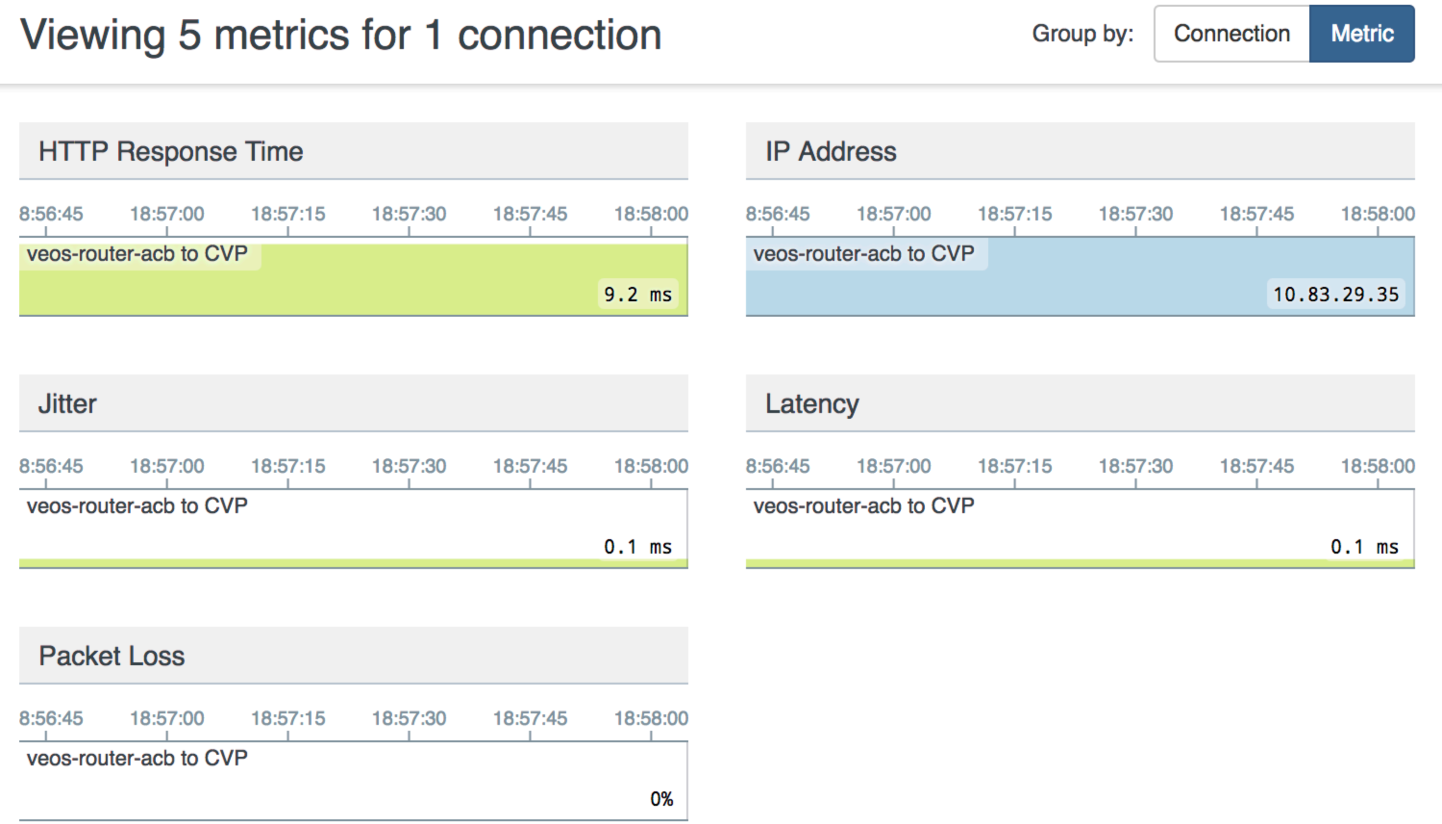


| □ | Name | Instance ID | Instance Type | Availability Zone | Instance State | Status Checks | Alarm Status | Public DNS (IPv4) | IPv4 Public IP |
|---|---|---|---|---|---|---|---|---|---|
| □ | MCLOUD-AWS-POD1-USER... | i-01c679d9db9b1b69b | t2.micro | us-east-1b | ● running | ✓ 2/2 checks pass... | *None* |  | 54.90.251.55 |
| □ | MCLOUD-AWS-POD1-VEOS | i-030db9a350b1ec5e7 | c4.xlarge | us-east-1e | ● running | ⌛ Initializing | *None* |  | 23.20.115.141 |
| □ | MCLOUD-AWS-POD1-USER... | i-074e1602f934758e6 | t2.micro | us-east-1b | ● running | ⌛ Initializing | *None* |  | 34.228.199.70 |





| □ Name | Containers | Devices | Notes | Type | Created By |
|---|---|---|---|---|---|
| □ POD1_AWS_IPSEC_23.20.115.141 | 0 | 1 | Add Note | Static | cvpadmin |
| □ POD1_AWS_IPSEC_AZURE_DEST_23.20.115.141 | 0 | 1 | Add Note | Static | cvpadmin |
| □ POD1_LOCAL_IPSEC_45.75.192.103 | 0 | 1 | Add Note | Static | cvpadmin |
| □ POD1_LOCAL_IPSEC_AWS_DEST_45.75.192.103 | 0 | 1 | Add Note | Static | cvpadmin |
| □ POD1_LOCAL_IPSEC_AZURE_DEST_45.75.192.103 | 0 | 1 | Add Note | Static | cvpadmin |
| □ POD1_LOCAL_MONITOR_45.75.192.103 | 0 | 1 | Add Note | Static | cvpadmin |
| □ RECONCILE_10.83.29.37 | 0 | 1 | Add Note | Static | cvpadmin |
| □ RECONCILE_23.20.115.141 | 0 | 1 | Add Note | Static | cvpadmin |
| □ SYS_TelemetryBuilder | 1 | 1 | Add Note | Builder | cvp system |



Viewing 5 metrics for 1 connection

Group by: Connection **Metric**

HTTP Response Time

veos-router-acb to CVP 9.2 ms

IP Address

veos-router-acb to CVP 10.83.29.35

Jitter

veos-router-acb to CVP 0.1 ms

Latency

veos-router-acb to CVP 0.1 ms

Packet Loss

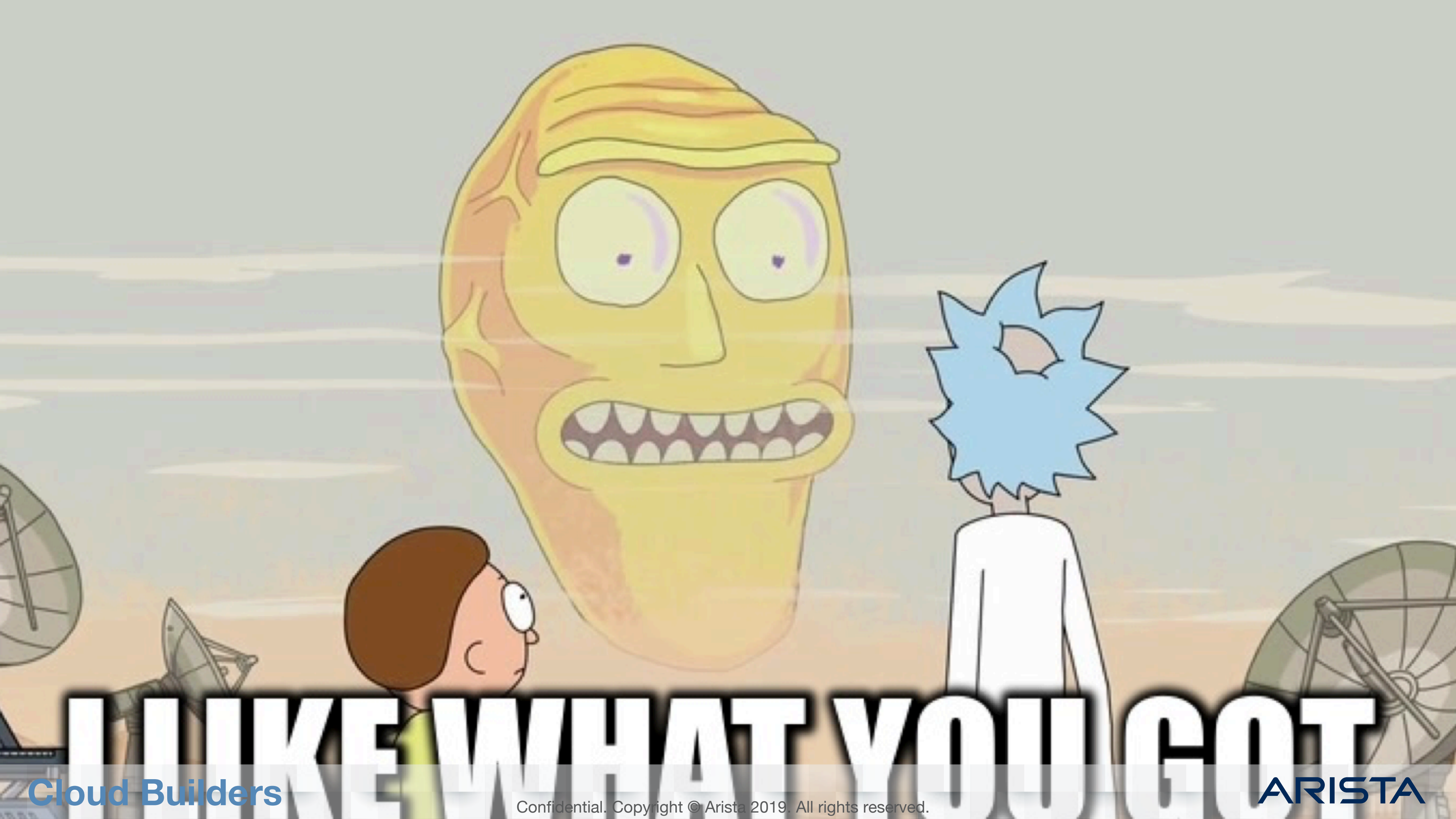
veos-router-acb to CVP 0%

```
VEOS-ROUTER-POD1#show ip route bgp

VRF: default
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 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,
       DH - DHCP client installed default route, M - Martian

 B E    10.123.1.0/24 [200/0] via 169.254.1.10, Tunnel1
 B E    10.123.2.0/24 [200/0] via 169.254.1.10, Tunnel1
 B E    10.234.1.0/24 [200/0] via 169.254.2.20, Tunnel2
```

**AWS** prefixes

**AZURE** prefixes

```
acb@acb4q2018:~/mcloud-pod1$ ping 10.123.1.137
PING 10.123.1.137 (10.123.1.137) 56(84) bytes of data.
64 bytes from 10.123.1.137: icmp_seq=1 ttl=62 time=92.4 ms
64 bytes from 10.123.1.137: icmp_seq=2 ttl=62 time=92.6 ms
64 bytes from 10.123.1.137: icmp_seq=3 ttl=62 time=92.4 ms
64 bytes from 10.123.1.137: icmp_seq=4 ttl=62 time=92.3 ms
64 bytes from 10.123.1.137: icmp_seq=5 ttl=62 time=92.3 ms
^C
--- 10.123.1.137 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4001ms
rtt min/avg/max/mdev = 92.347/92.468/92.681/0.352 ms

acb@acb4q2018:~/mcloud-pod1$ ping 10.234.1.4
PING 10.234.1.4 (10.234.1.4) 56(84) bytes of data.
64 bytes from 10.234.1.4: icmp_seq=1 ttl=62 time=3.57 ms
64 bytes from 10.234.1.4: icmp_seq=2 ttl=62 time=2.97 ms
64 bytes from 10.234.1.4: icmp_seq=3 ttl=62 time=3.75 ms
64 bytes from 10.234.1.4: icmp_seq=4 ttl=62 time=3.30 ms
^C
--- 10.234.1.4 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3004ms
rtt min/avg/max/mdev = 2.978/3.403/3.752/0.300 ms
```


I LIKE WHAT YOU GOT

# Thank You