Purpose **4**

POC Setup **4**
Network Diagram 5
Traffic Flows 5

Deployment With Terraform: **6**
Verify the routing reachability 7
Activate fortiadc license with TFTP server 8

Deploy Manually **9**
Create VPC fortiADC VM 10
vswitches 10
Route table 10
Create FortiADC VM 11
Build fortiadc custom image 11
Create instance with custom image ID 13
Add Tag 13
VM in FortiADC VPC 14
Create HUBVPC 14
HUBVPC 15
vSwitches 15
internal vswitch 15
external vswitch 15
TR-landing vswitch 15
Routing Tables 16
TR-landing routing table 17
custom routing table 17
Create Fortigate instance 19
Create Fortigate with image from the marketplace. 19
ENI ports 19
Create Client-VM 20
Add Tag 20
VMs in HUBVPC 21
Create ACK1VPC 22
Create VPC for ACK1 22
CIDR: 10.0.0.0/8 22
Vswitch 22
ACK1 VPC route table 22

Component created by ACK 27
install kubectl on client VM 28
Deploy application on ACK1 29
Create ACK2 31
Create and Config CEN-TR 31
TR Connections 32
Route Table- defaul-north-south 32
route table association 32
Remove Route Propagation 33
route entry 33
Route Table east-west 34
route table association 34
Enable Route Propagation 34
The route Entry 35
Config Fortigate VIP for access FortiADC 36
GUI access VIP mapping 36
Firewall Policy 36
Create a policy to allow this traffic 36
SSH access VIP mapping 37
Firewall Policy 38
Config Fortigate VIP for access client-VM 39
Firewall policy 40
Config Fortigate Routing 40
Config static Route 40
Config Firewall Policy for East-west traffic and ACK egress traffic to internet 41
Verify the reachability 43
ACK1 to ACK2 43
ACK1 POD and Node IP 44
ACK2 POD and Node IP 44
Ping Result 44
FortiGate View 46
enable security feature for egress traffic. 46
Config FortiADC as Ingress controller **47**
Config FortiADC to connect ACK1 and ACK2 48
Create ServiceAccount on both ACK 48
config k8s sdn connector on fortiADC 49
Create ingress rule for ACK1 nodeport service 52
Create a Real Server Pool on FortiADC 52
Create Layer 7 Ingress Rule with RealServer Pool 54
Associate with WAF profile 56

Enable Monitoring 56
Verify the result 57
Access via fortiadc internal IP address (from client-vm) 57
access via fortiadc external IP 58
Access via Fortigate public IP if fortiadc has no public IP associated 58
Create a Virtual IP on fortigate 59
create virtual IP 0.0.0.0:8080 to map to 10.0.11.11:8080 59
Verify the result 60
scale out the node on ACK1 61
simulate the attack to nodeport service via ingress entry 61
Config FortiGate K8s Connector 62
Config K8S Connector 63
get the cluster IP and SA secret from ACK1 and ACK2 63
Config k8s connector on fortigate 65
Create Firewall Policy based on dynamic object 67
Define the Address object 67
create firewall policy 68

Purpose

Enterprises often want to have a central control to all traffic including both VM and Container traffic. and may also like to deploy or reuse existing fortiADC appliance/vm as ingress controller or load balancer for both ACK and VM workload.

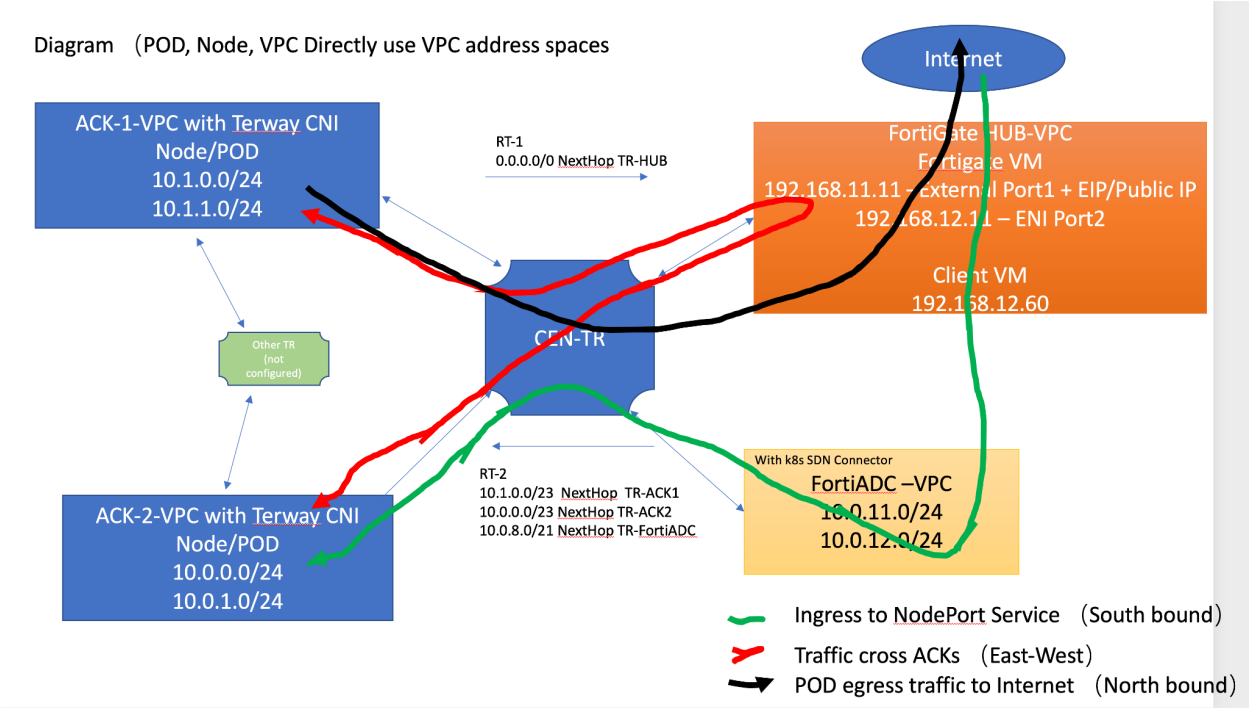
Below POC shows Fortinet leverages Next Generation Firewall Fortigate and Application Controller FortiADC to secure Enterprise ACK workload. In the Setup, FortiGate L3-L7 packet inspection for East-West and North-South traffic , Fortigate k8s SDN connector,FortiADC k8s SDN connector, FortiADC L7 LB as Ingress controller feature are detailed.

POC Setup

Here are the all components that are used in this POC Setup.

VPCs : 4
Public IP : 2 (one for fortigate, one for fortiadc)
ACK : 2
Alibaba SLB : 2 (created as part of each ACK, for ACK API access)
FortiADC VM : 1
Ubuntu Client VM :1
CEN-TR: 1 (attached to all 4 VPCs)

Network Diagram



Traffic Flows

CEN-TR0:

CEN-TR is attached to all 4 VPCS and as a central router to route the traffic between VPCs (East-West) and Internet bound traffic (North-South).

East-West Traffic :
Traffic between POD1 in ACK1 and POD2 in ACK2 is routed through FortiGate in HubVPC via CEN-TR0.

POD egress traffic:
All POD egress traffic is routed through Fortigate via CEN-TR0.

ACK1 ,ACK2 ingress Traffic:
Ingress to ACK1 and ACK2 service is via FortiADC Ingress controller for internal visit and external visit (when fortiadc associated a public IP) ,
FortiGate is used to take external traffic to ACK1, ACK2 , DNAT to FortiADC then to ACK Cluster when FortiADC does not have a public IP associated.

L7 Firewall Policy on Fortigate is used to inspect traffic between POD1 and POD2 (across VPCs). as well as POD egress traffic to the Internet. Fortigate is the central point for inspect traffic across ACKs and North-South traffices.

FortiADC is used as Ingress controller ,Layer 7 Ingress rule is configured on FortiADC to take ingress traffic to NodePort Type service on ACK1 and ACK2.

Traffic within ACK1 and ACK2 is subject to control by Alibaba Terway CNI with Network Policy.

Deploy With Terraform:

```
git clonegit@github.com:yagosys/fortinet_alibaba.git
cd securehub_ack
terraform apply
```

follow the instructions in the README file.

after deployment, you shall see following similar output at the console

```
Outputs:

FortigateAdminGUI_PORT = "8443"
PrimaryFortigateAvailability_zone = "cn-hongkong-b"
PrimaryFortigateID = "i-j6c8ukmv8inb7eul5ik5"
PrimaryFortigatePrivateIP = "192.168.11.11"
PrimaryFortigatePublicIP = "47.242.124.76"
PrimaryFortigateport2IP = "192.168.12.11"
ack1_worknode_ip = tolist([
  "10.1.0.164",
  "10.1.0.163",
])
ack2_worknode_ip = tolist([
  "10.0.0.250",
  "10.0.0.251",
])
client-vm = "192.168.12.60"
client-vm-password = "Welcome.123"
client-vm-ssh-port = "2022"
fortiadc_gui_https_port = 9443
```

```
fortiadc_instance_id = "i-j6c8ukmv8inb7eul5ik6"
fortiadc_private_ip = "10.0.11.11"
fortiadc_public_ip = "47.243.181.129"
fortiadc_ssh_port = 6022
```

Verify the routing reachability

client-vm shall be able to ping all endpoints in the topology. which include fortigate internal IP, fortiadc IP, ack1 and ack2 worker-node IP.

```
(base) i@ecs-148531:~/fortinet/cen-tr-ack$ ssh root@47.242.124.76 -p 2022
Warning: Permanently added '[47.242.124.76]:2022' (ECDSA) to the list of
known hosts.
root@47.242.124.76's password:
Welcome to Ubuntu 18.04.4 LTS (GNU/Linux 4.15.0-101-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

 * Super-optimized for small spaces - read how we shrank the memory
   footprint of MicroK8s to make it the smallest full K8s around.

   https://ubuntu.com/blog/microk8s-memory-optimisation
New release '20.04.3 LTS' available.
Run 'do-release-upgrade' to upgrade to it.


Welcome to Alibaba Cloud Elastic Compute Service !

Last login: Tue Feb 22 16:17:02 2022 from 192.168.12.11
root@iZj6cg0w474i2p8oykt7kqZ:~# ping 192.168.11.11 -c 1
PING 192.168.11.11 (192.168.11.11) 56(84) bytes of data.
64 bytes from 192.168.11.11: icmp_seq=1 ttl=255 time=0.358 ms

--- 192.168.11.11 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
```

```
rtt min/avg/max/mdev = 0.358/0.358/0.358/0.000 ms
root@iZj6cg0w474i2p8oykt7kqZ:~# ping 10.1.0.164 -c 1
PING 10.1.0.164 (10.1.0.164) 56(84) bytes of data.
64 bytes from 10.1.0.164: icmp_seq=1 ttl=63 time=1.59 ms

--- 10.1.0.164 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 1.593/1.593/1.593/0.000 ms
root@iZj6cg0w474i2p8oykt7kqZ:~# ping 10.1.0.163 -c 1
PING 10.1.0.163 (10.1.0.163) 56(84) bytes of data.
64 bytes from 10.1.0.163: icmp_seq=1 ttl=63 time=1.49 ms

--- 10.1.0.163 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 1.496/1.496/1.496/0.000 ms
root@iZj6cg0w474i2p8oykt7kqZ:~# ping 10.0.0.250 -c 1
PING 10.0.0.250 (10.0.0.250) 56(84) bytes of data.
64 bytes from 10.0.0.250: icmp_seq=1 ttl=63 time=1.65 ms

--- 10.0.0.250 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 1.651/1.651/1.651/0.000 ms
root@iZj6cg0w474i2p8oykt7kqZ:~# ping 10.0.0.251 -c 1
PING 10.0.0.251 (10.0.0.251) 56(84) bytes of data.
64 bytes from 10.0.0.251: icmp_seq=1 ttl=63 time=1.86 ms

--- 10.0.0.251 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 1.861/1.861/1.861/0.000 ms
root@iZj6cg0w474i2p8oykt7kqZ:~# ping 10.0.11.11 -c 1
PING 10.0.11.11 (10.0.11.11) 56(84) bytes of data.
64 bytes from 10.0.11.11: icmp_seq=1 ttl=63 time=1.39 ms

--- 10.0.11.11 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 1.393/1.393/1.393/0.000 ms
root@iZj6cg0w474i2p8oykt7kqZ:~#
```

Activate fortiadc license with TFTP server

The Fortigate has already applied with License and configuration file, however, the terraform script does not support fortiadc cloud init, therefore, we have to manually apply license to fortiadc. There are two ways to activate the fortiadc license, use command line or use GUI. If we use the command line, a TFTP server is required which our client-VM has already configured as a TFTP server. so we can use clientVM as a tftp server to configure the license.

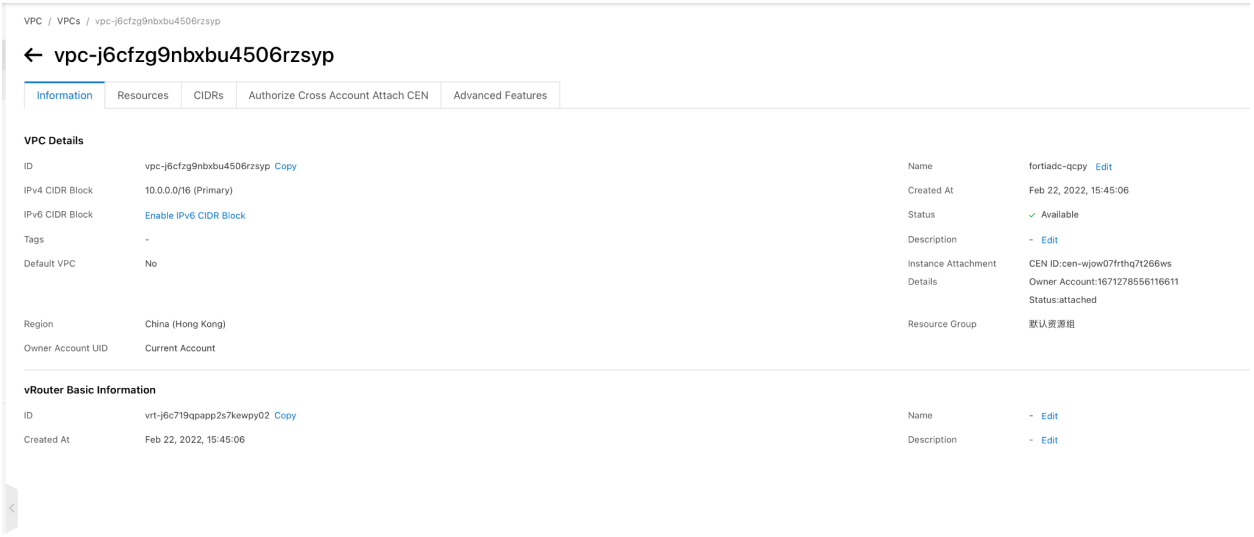
```
(base) i@ecs-148531:~/fortinet/cen-tr-ack$ ssh admin@47.242.124.76 -p 6022
Warning: Permanently added '[47.242.124.76]:6022' (ECDSA) to the list of
known hosts.
admin@47.242.124.76's password:
You are forced to change your password, please input a new password.
New Password: ***********
Confirm Password: ***********
FortiADC-ALI # execute vm license up
<Enter>

FortiADC-ALI # execute vm license tftp FADLICENSE.lic 192.168.12.60
This operation will replace the current vmware license and reload the
system!
Do you want to continue? (y/n)y

Connect to tftp server 192.168.12.60 ...
Please wait...

Get vmware license file from tftp server OK.
Connection to 47.242.124.76 closed by remote host.
```

Deploy Manually

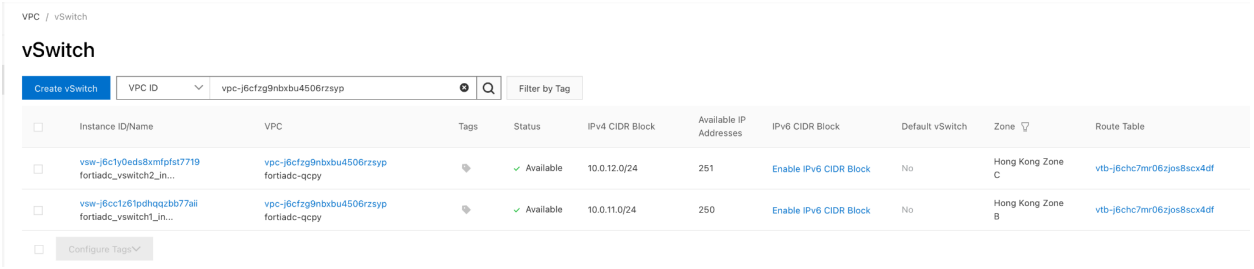
Below is for how to manually provision entire infrastructure without using terraform

Create VPC and fortiADC VM

FortiADC VPC has CIDR block 10.0.0.0/8 and two vswitches with subnet 10.0.12.0/24 and 10.0.11.0/24 , FortiADC VM is on vswitch 10.0.11.0/24. vswitch 10.0.12.0/24 is created for attach to CEN-TR purpose. as CEN-TR requires minimal 2 vswitch attachment.

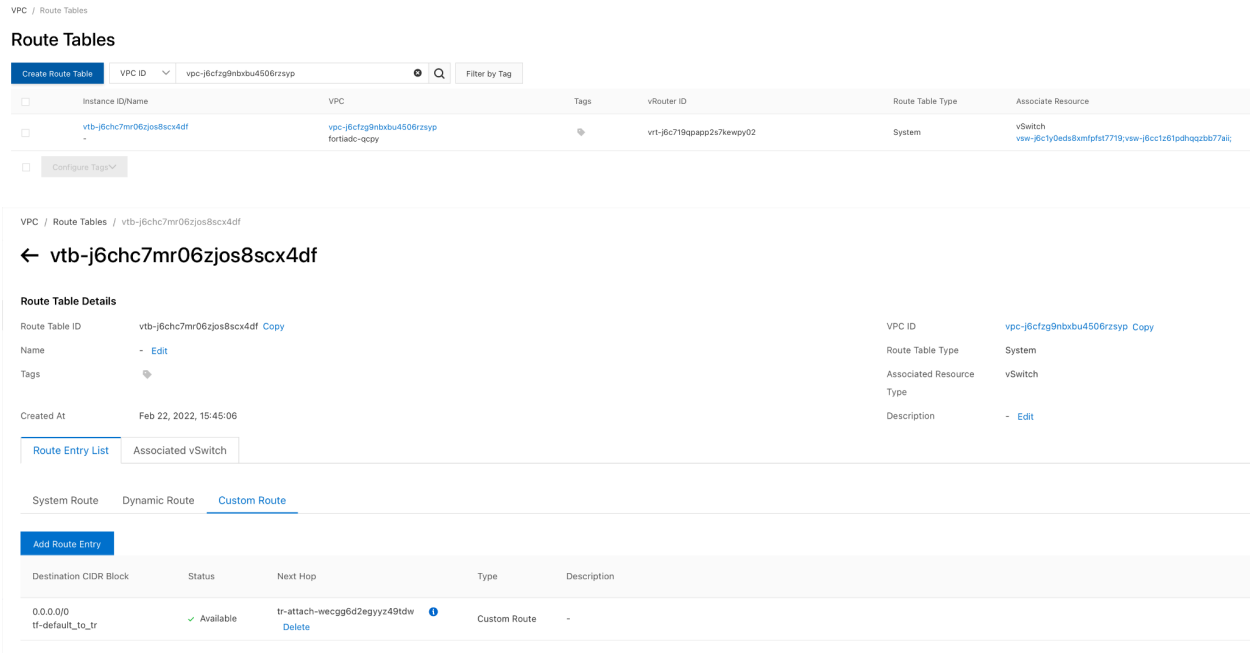


vswitches



Route table

FortiADC VPC has one system default routing table with default route point to CEN-TR.



Create FortiADC VM

FortiADC is not available on alibaba marketplace, therefore, build a custom image for fortiadc on alibaba cloud is required.

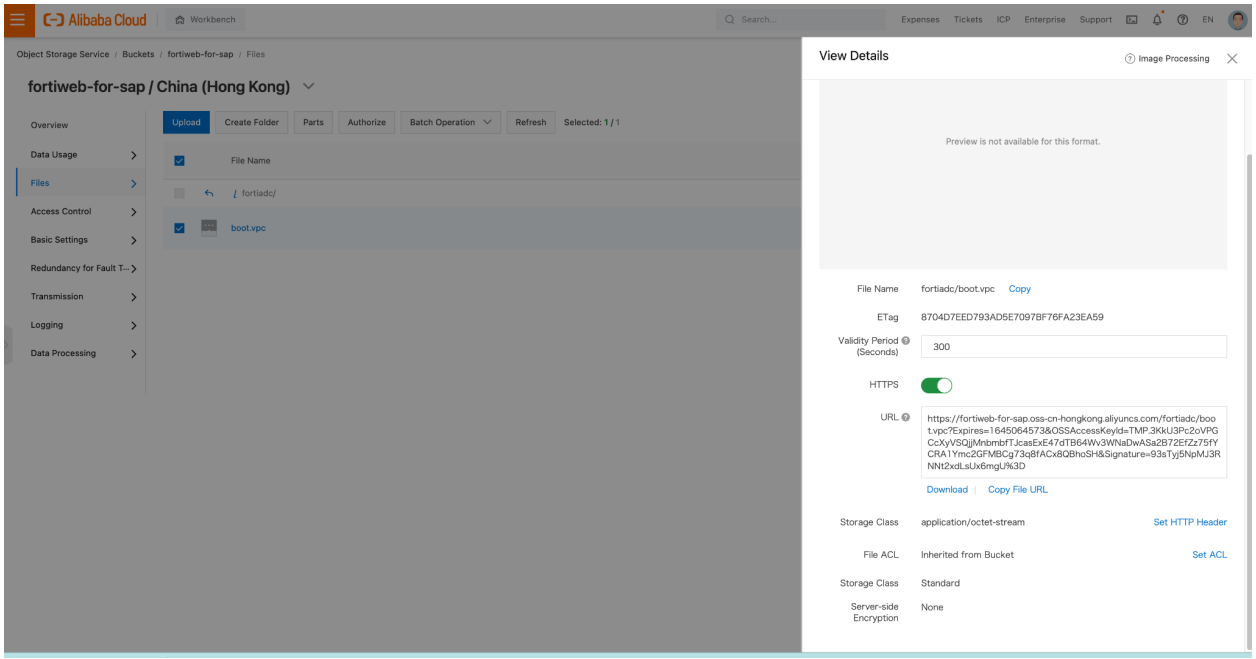
Build fortiadc custom image

official document
https://docs.fortinet.com/document/fortiadc-public-cloud/latest/alibaba-cloud-deployment-guide

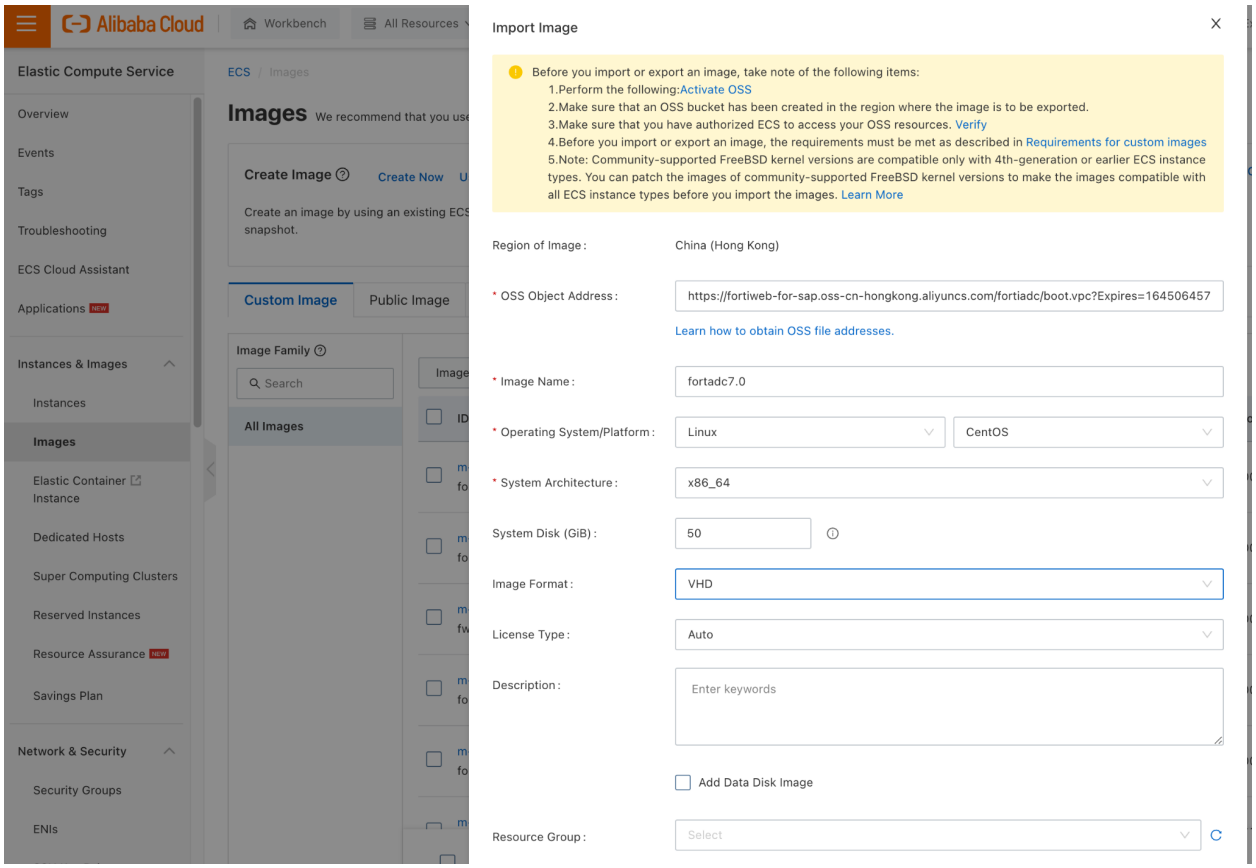
you must have an existing Alibaba Cloud Account to enable you to use Alibaba Cloud products. Get a released image from Fortinet https://support.fortinet.com/Download/FirmwareImages.aspx with file name something like “XXX.FORTINET.out.ali.zip”, then unzip this file to get boot.vpc

FAD_ALI-V700-build0014-FORTINET.out	117,854	2022-02-01 09:02:57	2022-02-01 09:02:10	HTTPS Checksum
FAD_ALI-V700-build0014-FORTINET.out.ali.zip	117,513	2022-02-01 09:02:21	2022-02-01 09:02:32	HTTPS Checksum

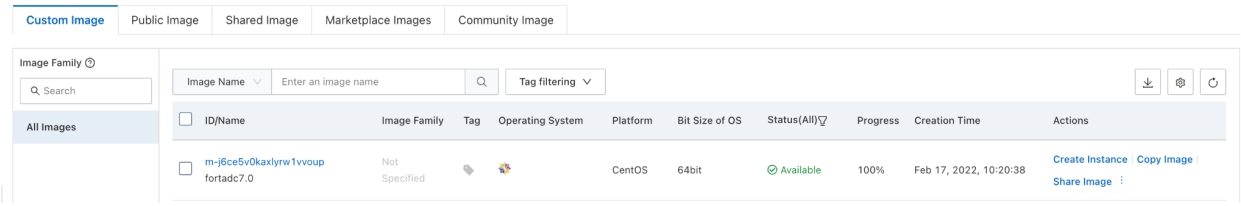
Upload boot.vpc file to OSS and obtain the URL for download



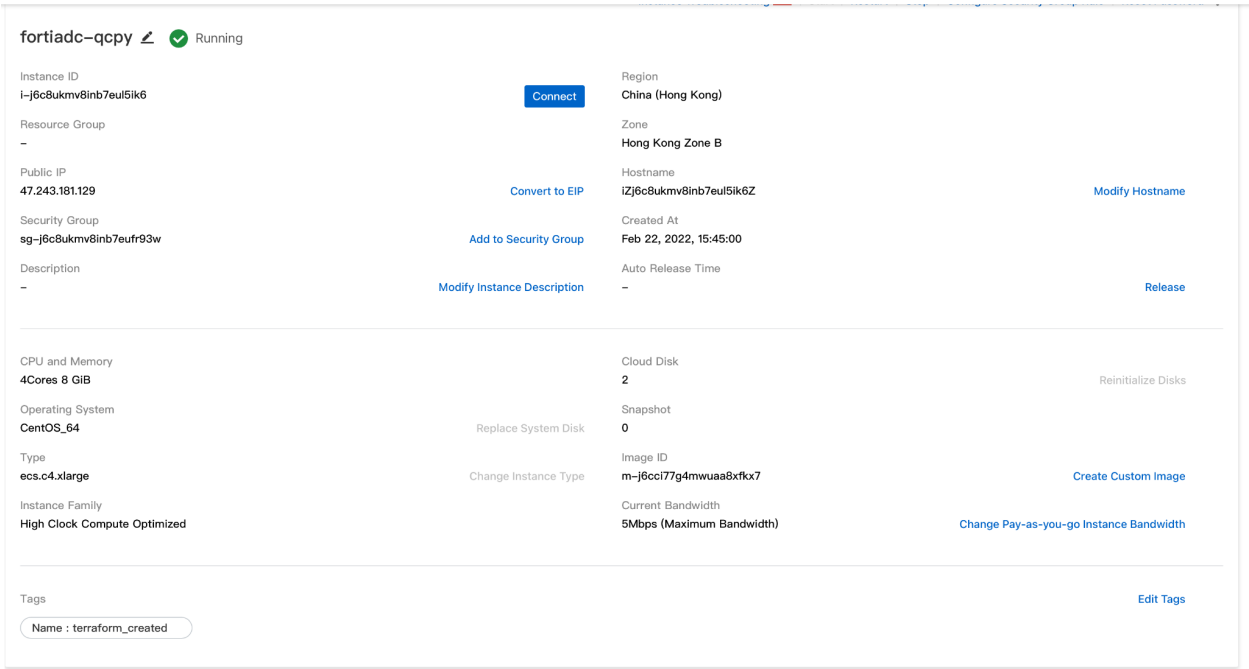
Import Image from URL

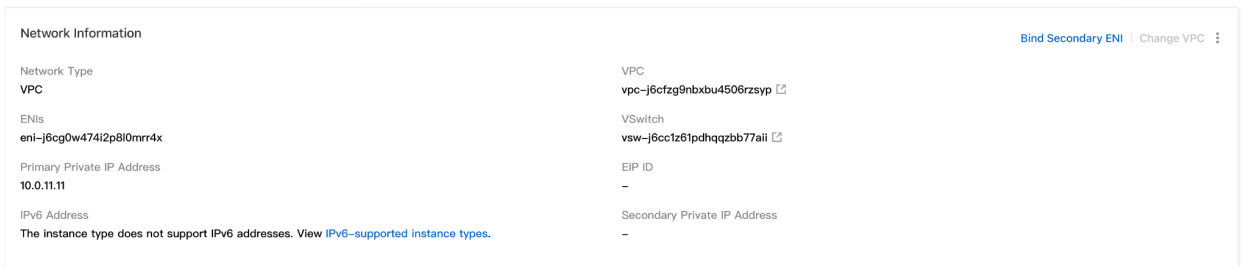


you will got the image after a few minutes



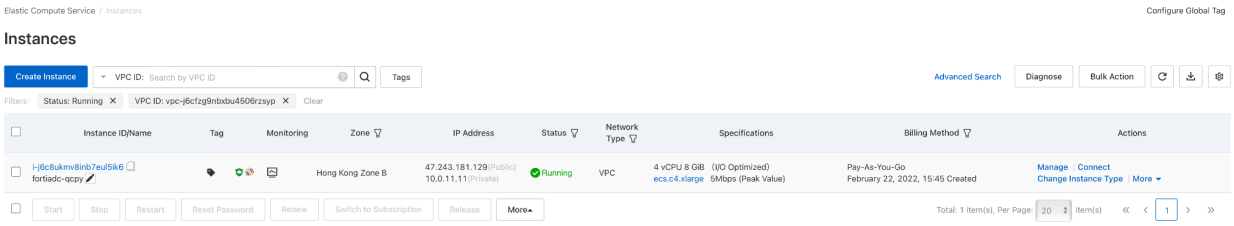
Create instance with custom image ID





For this POC, only one ENI is required.

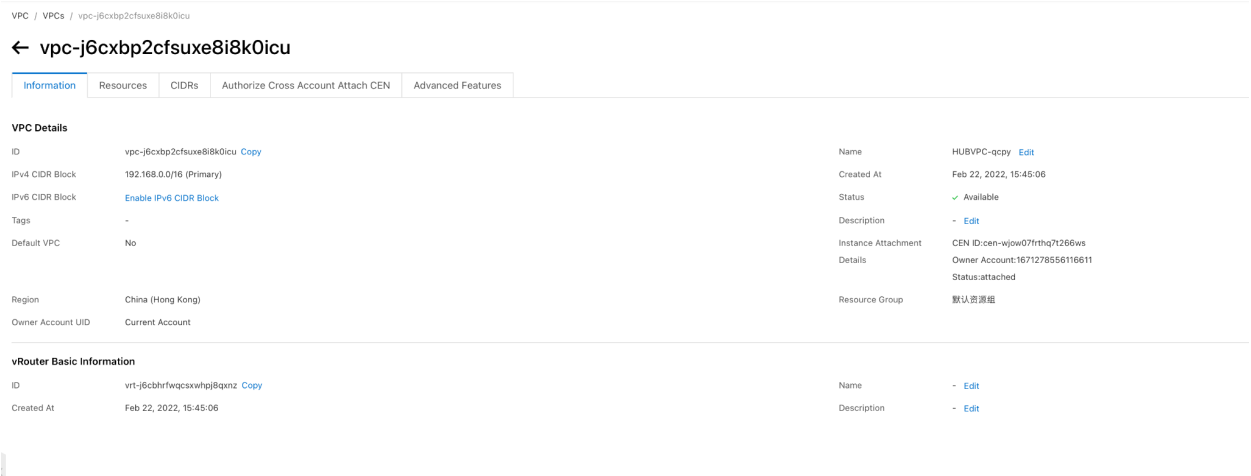
VM in FortiADC VPC



Create HUBVPC

HUBVPC uses CIDR 192.168.0.0/16. 3 Paires of Vswitches is required, one Vswitch is for external Internet access, Fortigate Port1 is on this vswitch, one Vswitch is for internal traffic, an ENI is created on this vswitch and attached to the fortigate as Port2. The cross VPCs traffic will be routed to this vSwitch so traffic can go into Fortigate VM. one pair of Vswitch for CEN-TR landing is required for landing TR traffic into this vswitch.

HUBVPC



vSwitches

internal vswitch

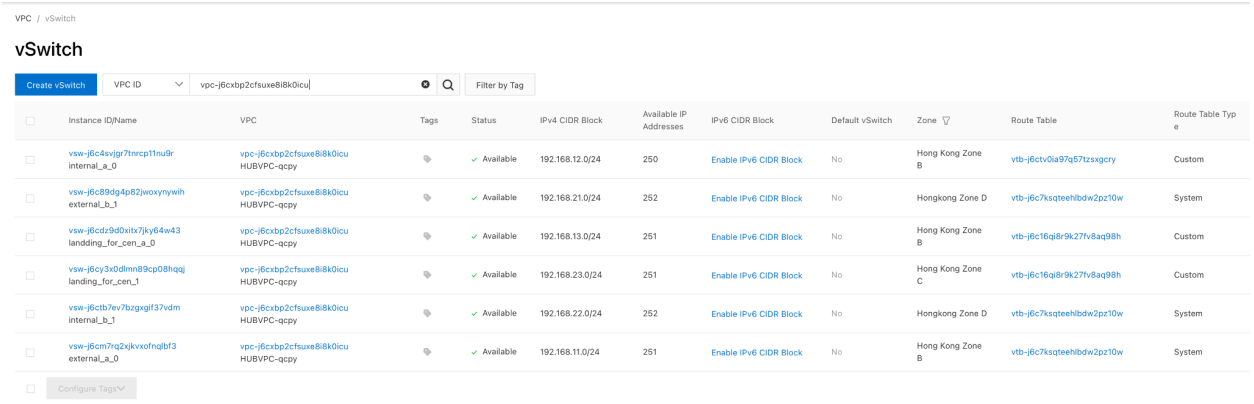
VSwitch for internal traffic, fortigate Port 2 (corresponding to ECS ENI Port2) is on this vswitch.

external vswitch

VSwitch for External traffic, fortigate Port 1 is on this vswitch to accept traffic from external. Port1 will be associated with a public IP or EIP

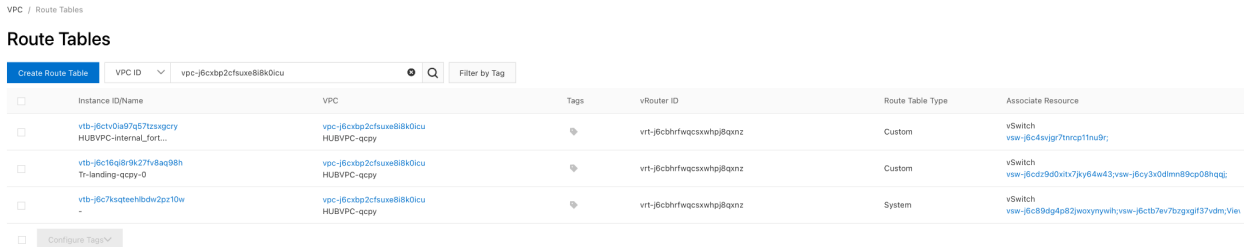
TR-landing vswitch

a Pair of VSwitch for TR-Landing, when CEN-TR connects to this HUBVPC, the TR-Landing vswitch will take the traffic. since CEN-TR requires minimal two VSwitch to attach for redundancy purposes. therefore. two VSwitch is required. each one is a separated zone.

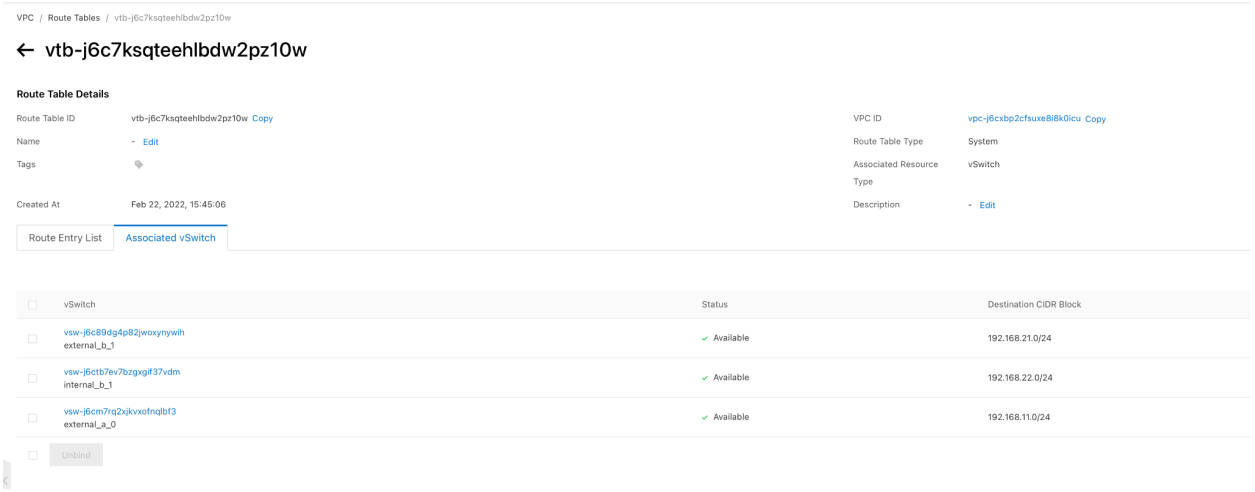


Routing Tables

In this VPC, 3 Routing Tables are required.

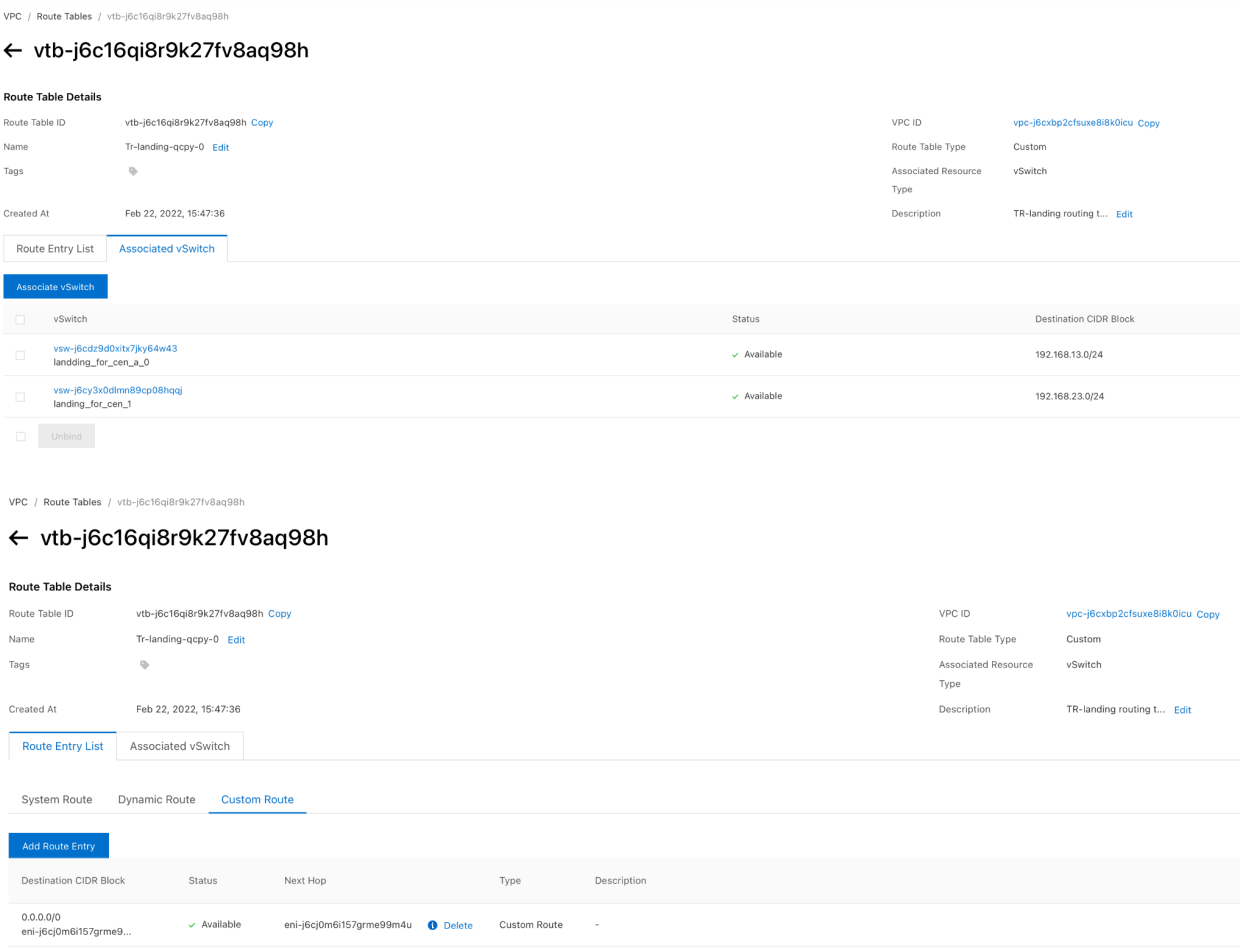


The default route is point to default IGW. and associated with external vswitch, IGW can be replaced with NAT GW as well.



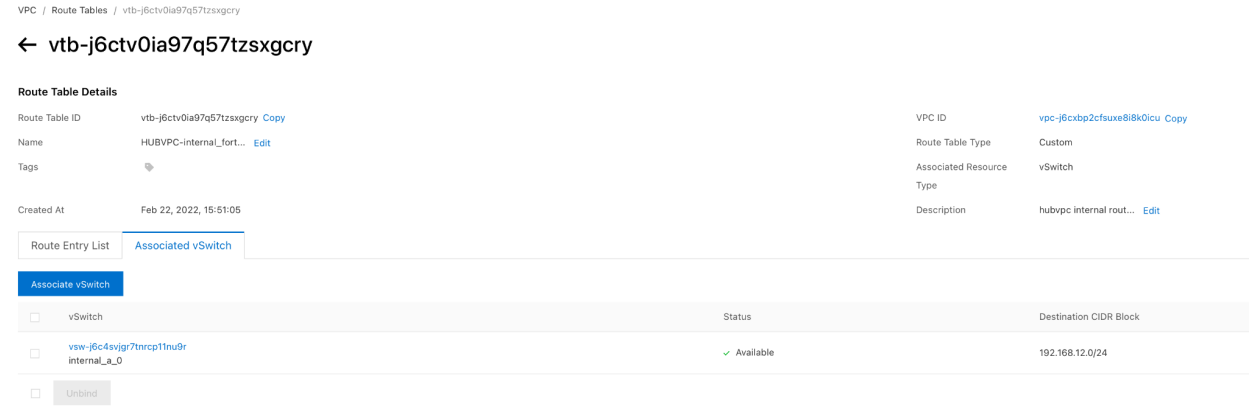
TR-landing routing table

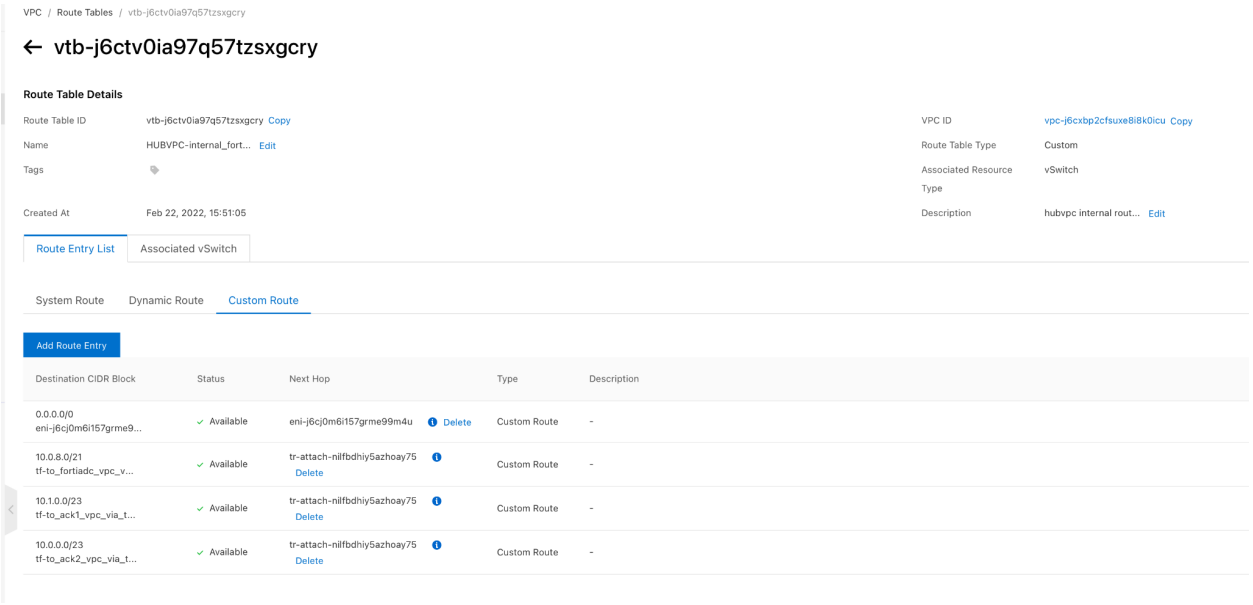
TR-landing is for traffic from TR Router and associated with TR-landing vswitch, where the nexthop will be sent to ECS VM via ENI Port 2. A default route with next hop to ENI port 2 is configured in this routing table. all traffices that from TR arriving to this vswitch will be routed to Fortigate VM via ENI Port 2.



custom routing table

to_VPC_ACK_1_2_ADC custom routing table is attached with internal vswitch, so when traffic exit from ENI Port 2 (as it is on internal vswitch), this routing table will be looked up , it will use nexthop TR router to reach VPC ACK-1 and VPC ACK-2 and FortiADC VPC.

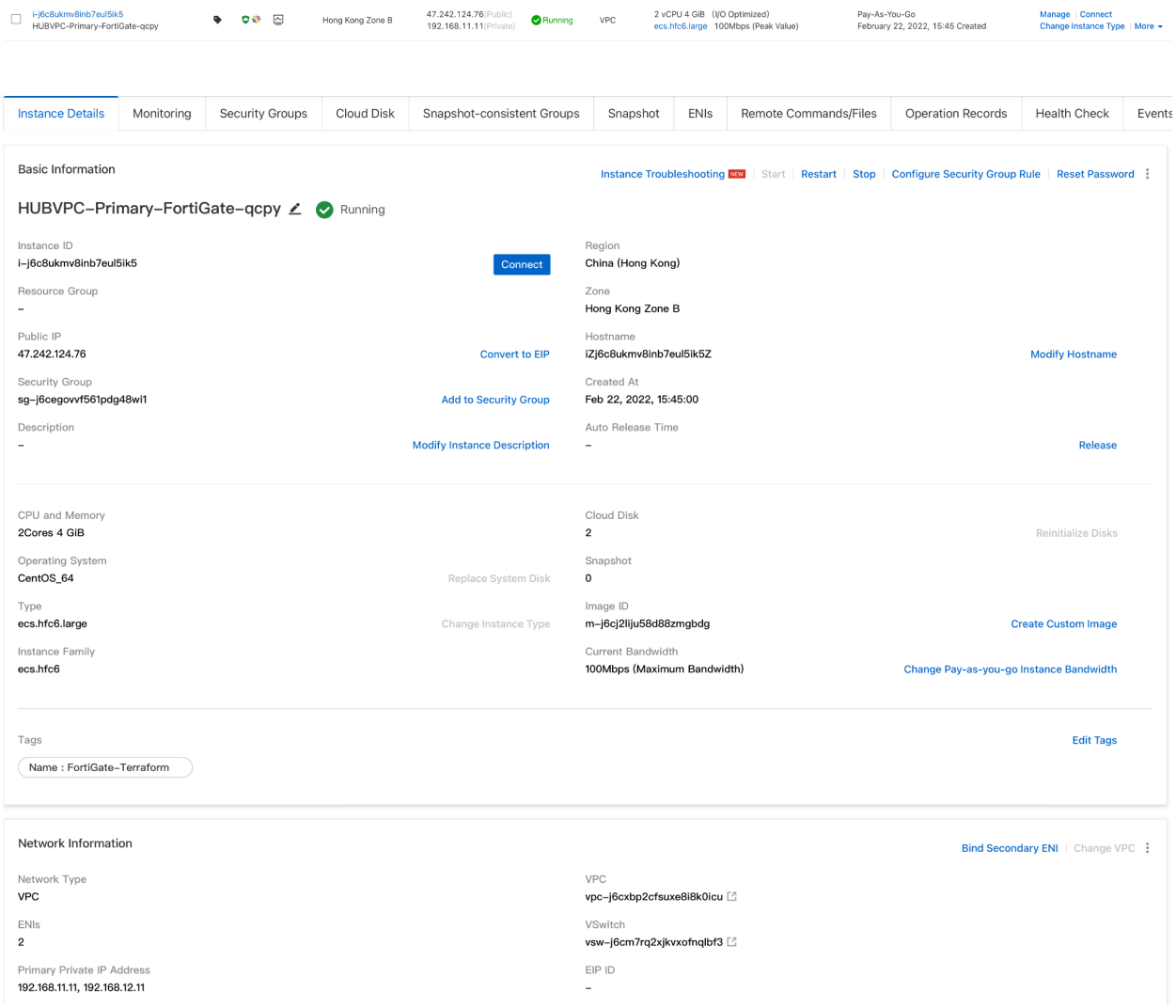




FortiGate is in HUBVPC where all North-South traffic and Cross ACK traffic will be routed into this VPC via CEN-TR. Also Fortigate is located in this VPC. so this VPC will require internet access. FortiGate in this VPC will be associated with a public IP or EIP.

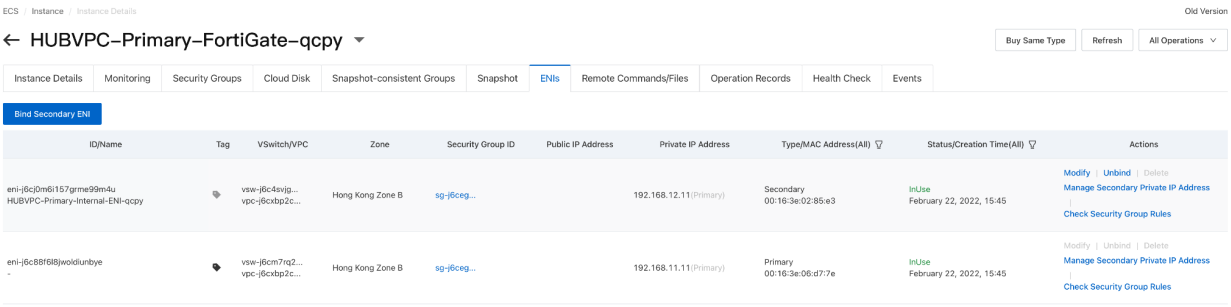
Create Fortigate instance

Create Fortigate with image from the marketplace.



ENI ports

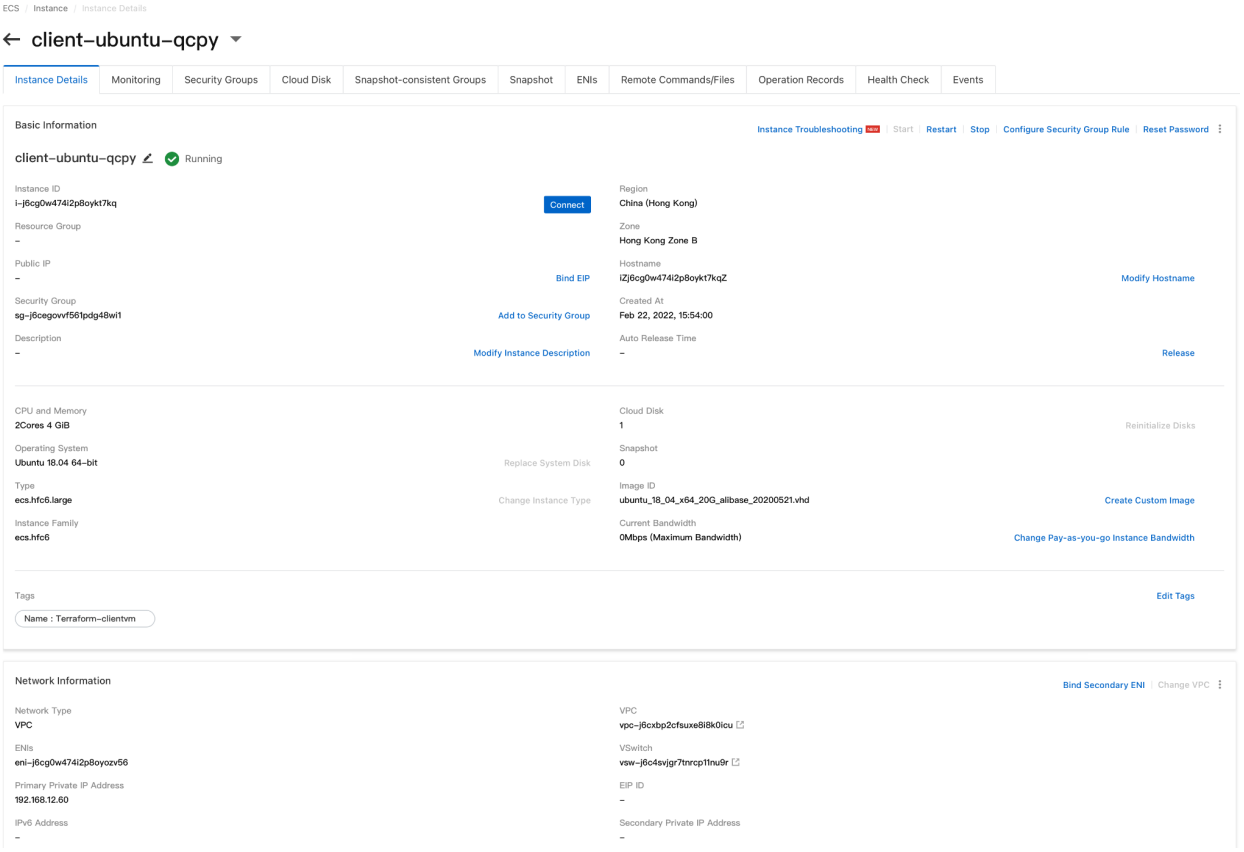
ENI Port with IP 192.168.12.11 is map to fortigate Port2.
Primary Port with IP 192.168.11.11 is map to fortigate Port1.



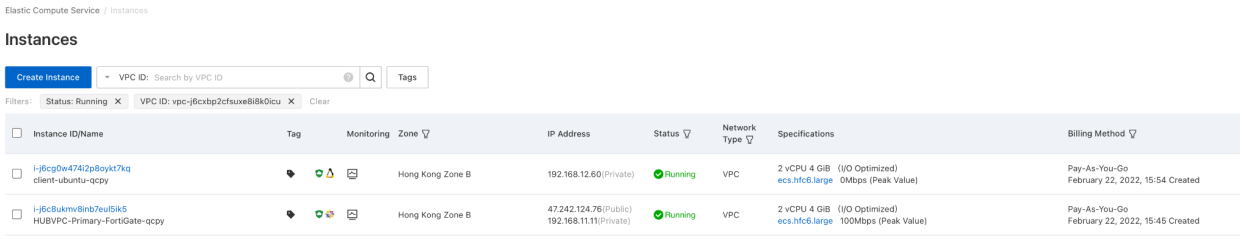
Create Client-VM

Create a Ubuntu VM in HUBVPC as a kubectl client. You can also create this VM in a dedicated VPC or any other VPCs as long as it has a route to reach all other components you want to manage. in this POC setup. We just use HUB VPC.
This Ubuntu VM does not need a public IP address associated , We will use Fortigate do a DNAT to access this VM, To to add, We need to add an Tag for this VM , so fortigate can automatically locate this VM via Tag when Fortigate configured with SDN connector to alibaba Cloud.
You can also attach a Public IP for this client VM temporarily for external SSH access if Fortigate has not ready yet. after you config a VIP/DNAT on fortigate for this client VM, you can detach its public IP.

You will need to install kubectl on this client VM to manage ACK1 and ACK2.



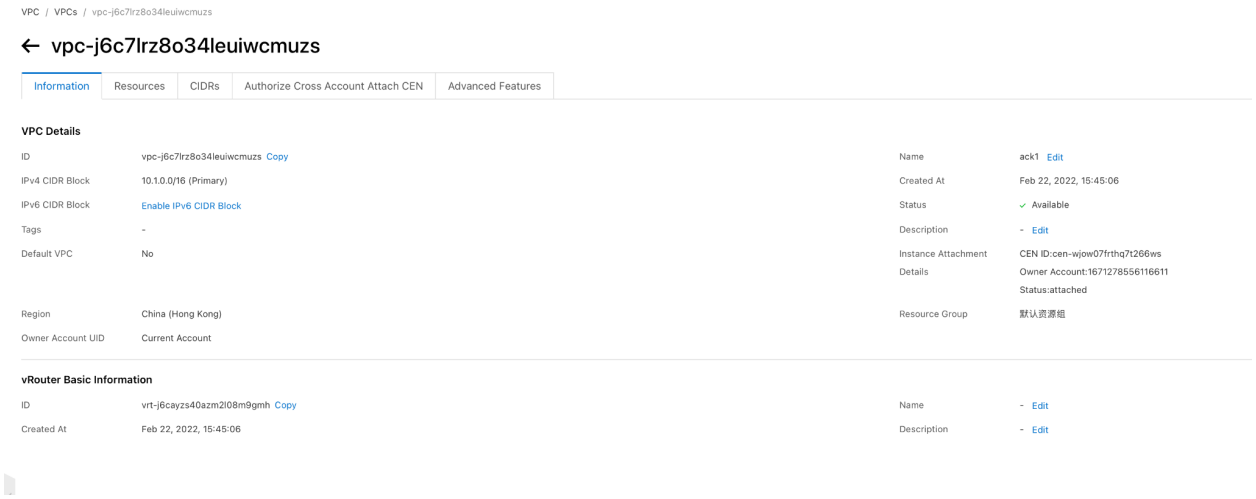
VMs in HUBVPC



Create ACK1VPC

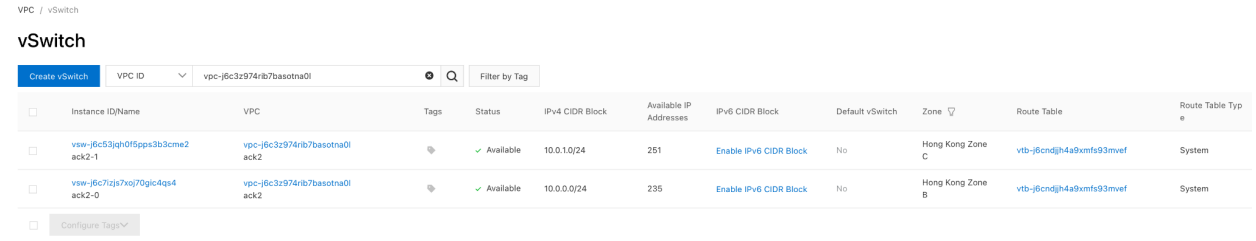
Create VPC for ACK1

CIDR: 10.0.0.0/8



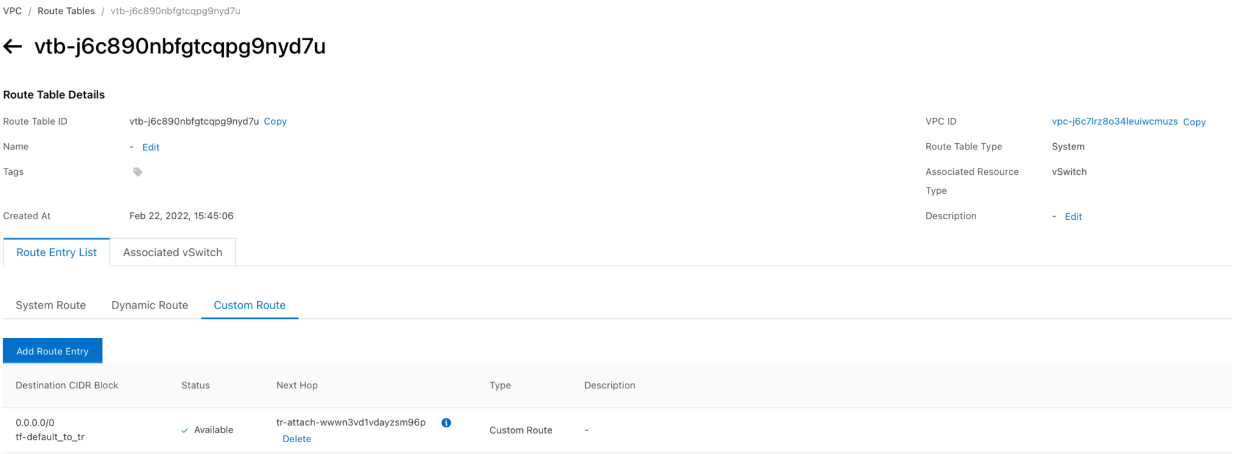
Vswitch

10.1.1.0/24 and 10.1.0.0/24 subnet for both POD and ACK worknode.



ACK1 VPC route table

In ACK1 default route table, you may configure route all traffic to CEN-TR , where CEN-TR will send all traffic to HUB VPC fortigate for inspection and internet access.



If you want POD egress to the internet use NAT Gateway, config 0.0.0.0/0 to NAT gateway, then configure specific route to CEN-TR.
Please be aware before you set up a fortigate for egress traffic to the internet, your ACK will not be able to reach the internet if you route all traffic to TR. you may configure 0.0.0.0/0 to NAT Gateway temporarily for internet access before Fortigate and CEN-TR are ready.

Create an ACK Cluster , In this POC, we choose a standard cluster. You can choose another model if you like.

Choose Terway as CNI as it will use VPC CIDR subnet for POD. also toggle Network-Policy option for network policy inside cluster. With Terway CNI, the POD will directly use the VPC CIDR address. no SNAT for POD is needed. POD , ACK nodes are directly using VPC CIDR.

Public access to ACK API is optional , as we will use client VM to access ACK with internal IP addresses. but you may toggle “Expose API with EIP” if you want.

You can choose not to install Ingress controller as we will use fortiadc as ingress controller. but you can install an ingress controller if you want to use both built-in ingress controller and FortiADC.

Cluster Name

ack-2022-|

The name must be 1 to 63 characters in length and can contain letters, Chinese characters, digits, and hyphens (-).

Cluster Specification

Professional | Standard edition | Contrast

Professional Kubernetes clusters are developed based on managed Kubernetes clusters. This type of cluster provides higher reliability and security for workloads in large-scale production environments, and is covered by the service-level agreement (SLA) that includes compensation clauses. Learn More

Region

How to select a region

China (Beijing)	China (Zhangjiakou)	China (Hohhot)	China (Ulanqab)	China (Hangzhou)	China (Shanghai)	China (Shenzhen)	China (Heyuan)
China (Guangzhou)	China (Chengdu)	China (Hong Kong)	Japan (Tokyo)	Singapore	Australia (Sydney)	Malaysia (Kuala Lumpur)	Philippines (Manila)
Indonesia (Jakarta)	India (Mumbai)	US (Virginia)	US (Silicon Valley)	UK (London)	Germany (Frankfurt)		

Billing Method

Pay-As-You-Go | Subscription | Contrast

Kubernetes Version

1.22.3-aliyun.1 | 1.20.11-aliyun.1 | Release Notes

Container Runtime

Containerd 1.4.8 | Docker 19.03.15 | Sandboxed-Container 2.2.0 | How to choose the container runtime?

VPC

VPC-ACK-1 (vpc-j6crkomejzxh81uyt4ia4, 10.0.0....

Create VPC | Plan Kubernetes CIDR blocks in VPC networks

Network Plug-in

Flannel | Terway | How to select a network plug-in for a Kubernetes cluster

☐ IPVLAN This feature combines IPVLAN and eBPF to enable NIC virtualization and sharing. Only Alibaba Cloud Linux 2 is supported.

☑ Support for NetworkPolicy Policy-based network traffic control is provided.

vSwitch

Select 1~5 vSwitches. We recommend that you select vSwitches in different zones to ensure high availability for the cluster.

	Name	ID	Zone	CIDR	Available IP Addresses
☑	ACK-1-10-1-1	vsw-j6co50qducn3jawn7nkvk	China (Hong Kong) ZoneC	10.1.1.0/24	248
☑	ACK-1-10-1-0-0	vsw-j6cns9z7kgjp71mppcpor	China (Hong Kong) ZoneB	10.1.0.0/24	236

Create vSwitch

Pod vSwitch

All | ZoneC (1 / 1) | ZoneB (1 / 1)

	Name	ID	Zone	CIDR	Supported Pods
☑	ACK-1-10-1-1	vsw-j6co50qducn3jawn7nkvk	China (Hong Kong) ZoneC	10.1.1.0/24	252
☑	ACK-1-10-1-0-0	vsw-j6cns9z7kgjp71mppcpor	China (Hong Kong) ZoneB	10.1.0.0/24	252

Create vSwitch

The prefix length of the VSwitch address is recommended to be no greater than 19 bits.

Service CIDR

192.168.0.0/16 Recommended Value:192.168.0.0/16

Valid values: 10.0.0.0/16-24, 172.16-31.0.0/16-24, and 192.168.0.0/16-24.

The specified CIDR block cannot overlap with that of the VPC 10.0.0.0/8 or those of the ACK clusters that are deployed in the VPC. The CIDR block cannot be modified after the cluster is created.

Configure SNAT

☐ Configure SNAT for VPC Recommended

Nodes and applications in the cluster do not have Internet access. To enable Internet access for nodes and application after the cluster is created, you must manually create and configure a NAT gateway for the VPC. For more information, see Manually enable SNAT.

Access to API Server

slb.s2.small | SLB Instance Specifications

By default, an internal-facing SLB instance is created for the API server. You can modify the specification of the SLB instance. If you delete the SLB instance, you cannot access the API server.

☐ Expose API Server with EIP Note

If this option is disabled, you cannot access the cluster API server through the Internet.

RDS Whitelist

Select RDS Instance

We recommend that you go to the RDS console to add the CIDR blocks of the specified nodes and specified pods to a whitelist of the RDS instance. Otherwise, if the RDS instance is not in the running state, the node pool cannot be scaled out.

Security Group

Create Basic Security Group | Create Advanced Security Group

By default, advanced security groups that are automatically created allow the communication between IP addresses within the VPC. You can ALSO manually modify security group rules based on your requirements. Security group overview

Deletion Protection

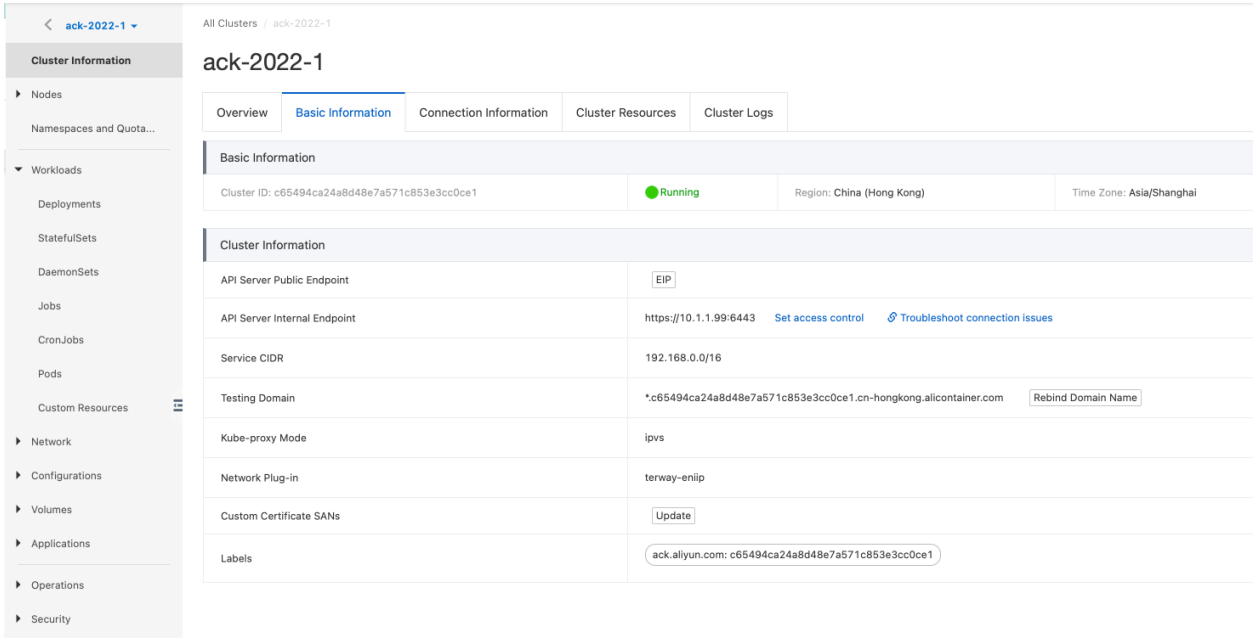
☐ Enable

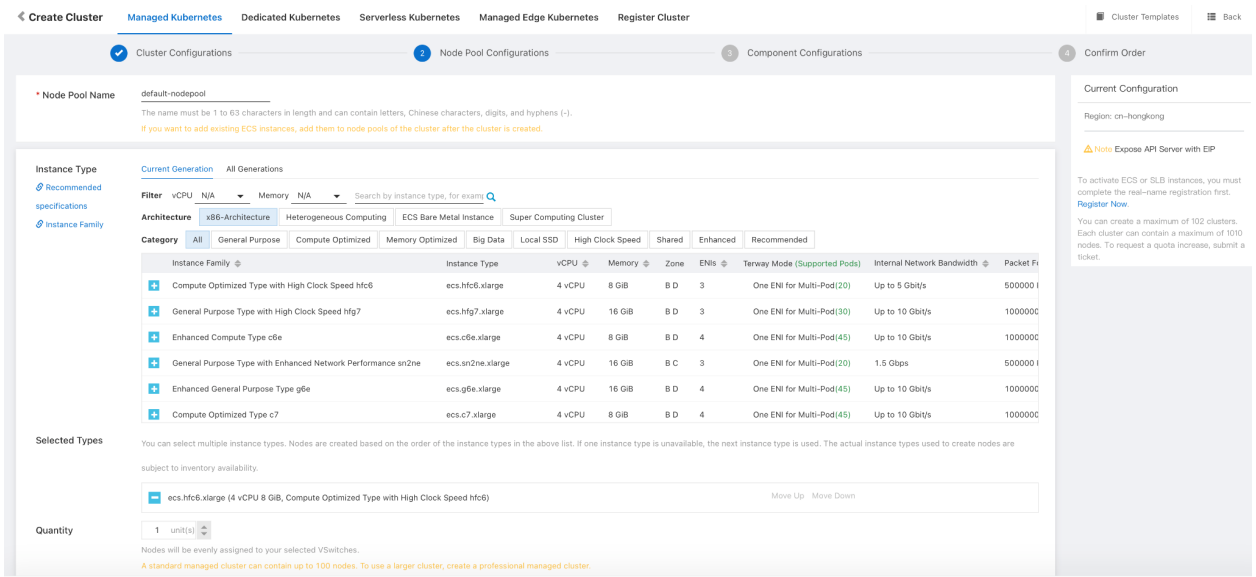
Cluster Cannot Be Deleted in Console or by Calling API

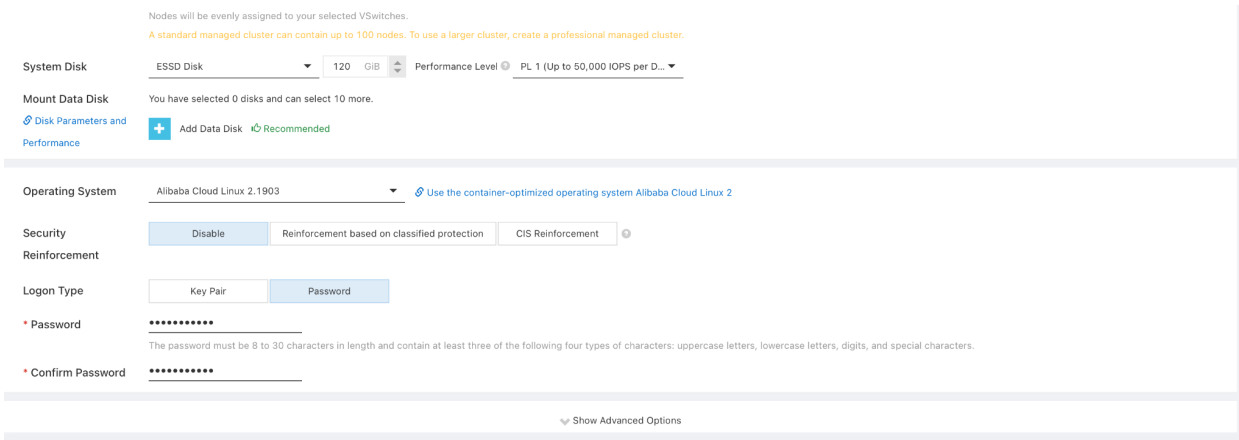
Resource Group

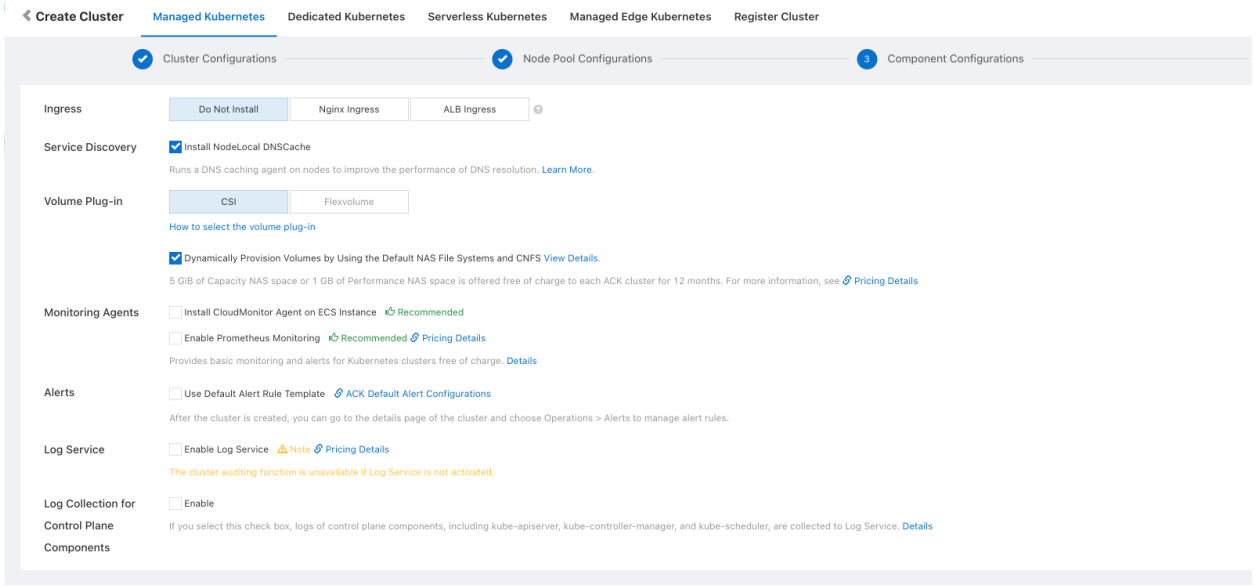
Not Selected

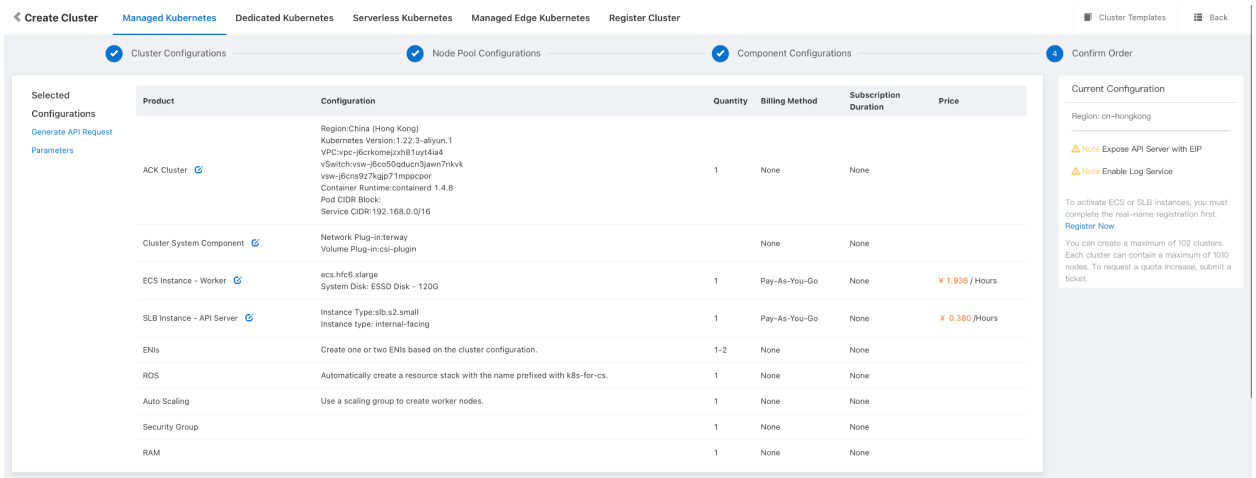
To create a resource group, click here.


ack-2022-1
Cluster Information
Nodes
Namespaces and Quota...
Workloads
Deployments
StatefulSets
DaemonSets
Jobs
CronJobs
Pods
Custom Resources
Network
Configurations
Volumes
Applications
Operations
Security
All Clusters / ack-2022-1
ack-2022-1
Overview
Basic Information
Connection Information
Cluster Resources
Cluster Logs
Basic Information
Cluster ID: c65494ca24a8d48e7a571c853e3cc0ce1
Running
Region: China (Hong Kong)
Time Zone: Asia/Shanghai
Cluster Information
API Server Public Endpoint
EIP
API Server Internal Endpoint
https://10.1.1.99:6443
Set access control
Troubleshoot connection issues
Service CIDR
192.168.0.0/16
Testing Domain
*.c65494ca24a8d48e7a571c853e3cc0ce1.cn-hongkong.alicontainer.com
Rebind Domain Name
Kube-proxy Mode
ipvs
Network Plug-in
terway-eniip
Custom Certificate SANs
Update
Labels
ack.aliyun.com: c65494ca24a8d48e7a571c853e3cc0ce1


Create Cluster
Managed Kubernetes
Dedicated Kubernetes
Serverless Kubernetes
Managed Edge Kubernetes
Register Cluster
Cluster Templates
Back
Cluster Configurations
Node Pool Configurations
Component Configurations
Confirm Order
* Node Pool Name
default-nodepool
The name must be 1 to 63 characters in length and can contain letters, Chinese characters, digits, and hyphens (-).
If you want to add existing ECS instances, add them to node pools of the cluster after the cluster is created.
Current Configuration
Region: cn-hongkong
Note Expose API Server with EIP
To activate ECS or SLB instances, you must complete the real-name registration first. Register Now.
You can create a maximum of 102 clusters. Each cluster can contain a maximum of 1010 nodes. To request a quota increase, submit a ticket.
Instance Type
Recommended specifications
Instance Family
Current Generation
All Generations
Filter
vCPU
Memory
Search by instance type, for exam
Architecture
x86-Architecture
Heterogeneous Computing
ECS Bare Metal Instance
Super Computing Cluster
Category
All
General Purpose
Compute Optimized
Memory Optimized
Big Data
Local SSD
High Clock Speed
Shared
Enhanced
Recommended
Instance Family
Instance Type
vCPU
Memory
Zone
ENIs
Terway Mode (Supported Pods)
Internal Network Bandwidth
Compute Optimized Type with High Clock Speed hfc6
ecs.hfc6.xlarge
4 vCPU
8 GiB
B D
3
One ENI for Multi-Pod(20)
Up to 5 Gbit/s
General Purpose Type with High Clock Speed hfg7
ecs.hfg7.xlarge
4 vCPU
16 GiB
B D
3
One ENI for Multi-Pod(30)
Up to 10 Gbit/s
Enhanced Compute Type c6e
ecs.c6e.xlarge
4 vCPU
8 GiB
B D
4
One ENI for Multi-Pod(45)
Up to 10 Gbit/s
General Purpose Type with Enhanced Network Performance sn2ne
ecs.sn2ne.xlarge
4 vCPU
16 GiB
B C
3
One ENI for Multi-Pod(20)
1.5 Gbps
Enhanced General Purpose Type g6e
ecs.g6e.xlarge
4 vCPU
16 GiB
B D
4
One ENI for Multi-Pod(45)
Up to 10 Gbit/s
Compute Optimized Type c7
ecs.c7.xlarge
4 vCPU
8 GiB
B D
4
One ENI for Multi-Pod(45)
Up to 10 Gbit/s
Selected Types
You can select multiple instance types. Nodes are created based on the order of the instance types in the above list. If one instance type is unavailable, the next instance type is used. The actual instance types used to create nodes are subject to inventory availability.
ecs.hfc6.xlarge (4 vCPU 8 GiB, Compute Optimized Type with High Clock Speed hfc6)
Move Up
Move Down
Quantity
1
unit(s)
Nodes will be evenly assigned to your selected VSwitches.
A standard managed cluster can contain up to 100 nodes. To use a larger cluster, create a professional managed cluster.


Nodes will be evenly assigned to your selected VSwitches.
A standard managed cluster can contain up to 100 nodes. To use a larger cluster, create a professional managed cluster.
System Disk
ESSD Disk
120
GiB
Performance Level
PL 1 (Up to 50,000 IOPS per D...
Mount Data Disk
You have selected 0 disks and can select 10 more.
Disk Parameters and Performance
Add Data Disk
Recommended
Operating System
Alibaba Cloud Linux 2.1903
Use the container-optimized operating system Alibaba Cloud Linux 2
Security Reinforcement
Disable
Reinforcement based on classified protection
CIS Reinforcement
Logon Type
Key Pair
Password
* Password
The password must be 8 to 30 characters in length and contain at least three of the following four types of characters: uppercase letters, lowercase letters, digits, and special characters.
* Confirm Password
Show Advanced Options





Dependency Check

Item	Status	Details
ACK activation check	Passed	
Account Status Check	Passed	
RAM Role Authorization Check	Passed	
Dependent Service Activation Status	Passed	
Auto Scaling Status Check	Passed	
Service Quota Check	Passed	
System Disk Size Check	Passed	
Data Disk Size Check	Passed	
Account Balance Check	Passed	

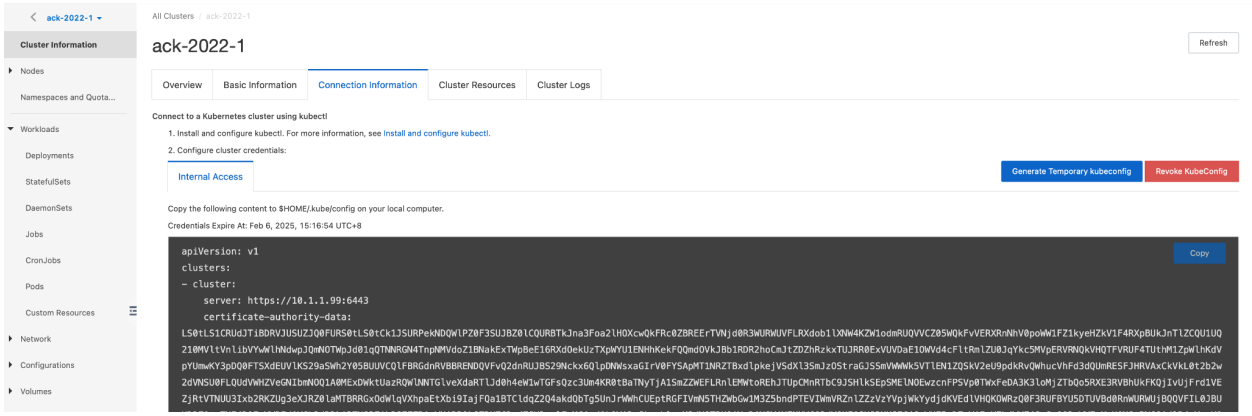
* Terms of Service

During the cluster creation process, the following operations may be performed depending on cluster configurations:

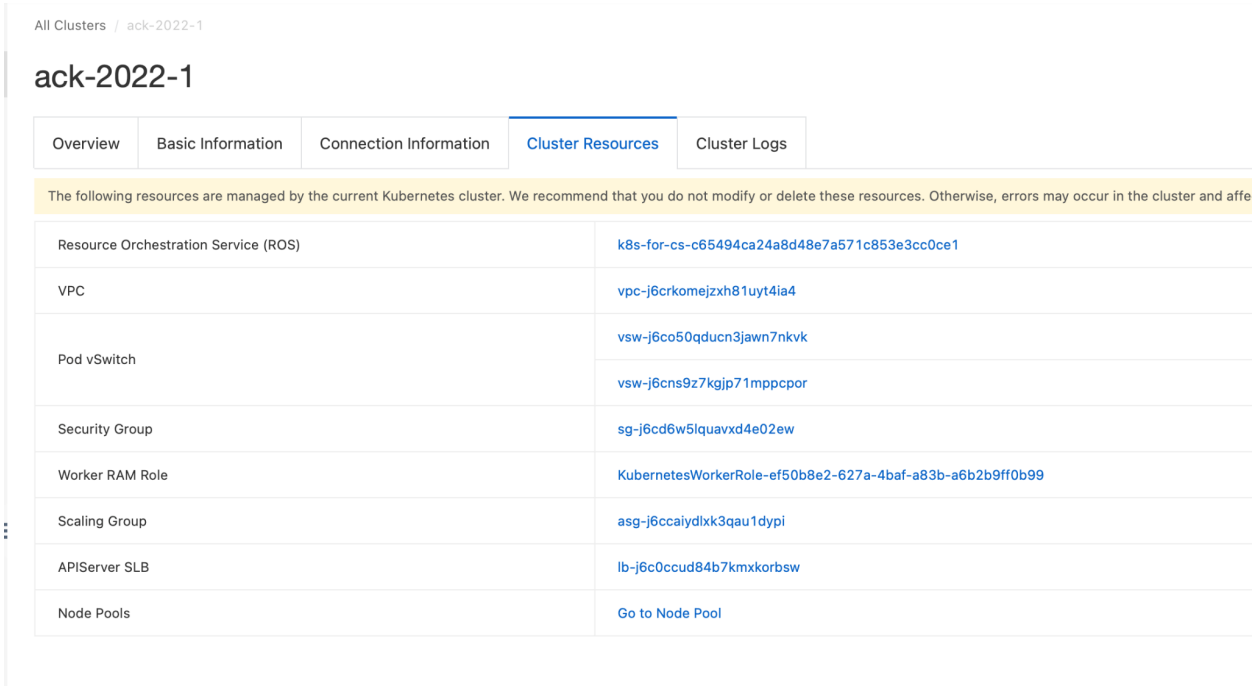
- Create ECS instances, configure a public key to enable SSH logon from master nodes to other nodes, and configure the Kubernetes cluster through CloudInit.
- Create a security group that allows access to the VPC network over ICMP.
- Create VPC routing rules.
- Create a NAT gateway and Elastic IP addresses.
- Create a RAM role and grant it the following permissions: query, create, and delete ECS instances, create and delete cloud disks, and all permissions on SLB instances, CloudMonitor, VPC, Log Service, and NAS. The Kubernetes cluster dynamically creates SLB instances, cloud disks, and VPC routing rules based on your settings.
- Create an internal SLB instance and open port 6443.
- When you use a dedicated or managed Kubernetes cluster, the system collects log and monitoring information about control components on master nodes to help ensure cluster stability.
- If the cluster creation fails, the created resources will be charged. We recommend that you delete unused resources at the earliest opportunity.

I have read and understand the preceding statement. I also have read and accept the Terms of Service and Disclaimer.

An internal SLB is created for access to k8s API server , the SLB has IP:PORT pair 10.1.1.99:6443 in this POC. We can directly copy the kubeconfig context into client-VM kubeconfig file.



Component created by ACK



1 VPC, two Vswitch (the POD and Node can be in both of these vswitch subnet, so when you create route entry in CEN-TR and Fortigate , do not forget add both), a APIServer SLB for k8s API server, a Node Pools

install kubectl on client VM

ssh into client VM by using fortigate public IP (after you configure VIP for client-VM on fortigate).
login into client-VM to install kubectl
https://kubernetes.io/docs/tasks/tools/?spm=5176.2020520152.0.0.1c6e61b1ojKsuW

```
(base) i@ecs-148531:~/fortinet/cen-tr-ack$ ssh root@47.242.124.76 -p 2022
root@iZj6cg0w474i2p8oykt7kqZ:~# head /root/.kube/config_ack1

apiVersion: v1
clusters:
- cluster:
    server: https://10.1.0.160:6443
    certificate-authority-data:
LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0tCk1JSURPekNDQWlPZ0F3SUJBZ0lCQURBTkJna3F
oa2lHOXcwQkFRc0ZBREErTVNjd0R3WURWUVFLRXdob1lXNW4KZW1odmRUQVVCZ05WQkFvVERXRn
NhV0poWW1FZ1kyeHZkV1F4RXpBUkJnTlZCQU1UQ210MVltVnlibVYwWlhNdwpJQmNOTWpJd01qS
XlNRGMwTlRVM1doZ1BNakExTWpBeU1UVXdOelExTlRkYU1ENHhKekFQQmdOVkJBb1RDR2hoCmJt
ZDZhRzkxTUJRR0ExVUVDaE1OWVd4cFltRmlZU0JqYkc5MVpERVRNQkVHQTFVRUF4TUthM1ZpWlh
KdVpYUmwKY3pDQ0FTSXdEUVlKS29aSWh2Y05BUUVCQlFBRGdnRVBBRENDQVFvQ2dnRUJBTGx6T1
JncG11K1lNd2FVZ0l0dApLcjY5c0UxRWhWMUdETVdiUEEzUFNJaDYvV29UNTJlV3JLTXpTQThHW
UVyU1JjQmJwTXoxUjNwVEZ0eC9oYVU2CjNIbmNLdHdQeC9KOXVmVmF1S1hadk1HcEhMWFFjWVdP
ZXplbG9vdUp2dTFhbGdiR2pCZjdoU1FhRjVjNHFreGwKTTNwekN4RTloZ1hYVllHMlhRRkM3SEJ
3VTFsbm9IWUlmRHoxdXlEOThsYUhOVW1WMWcrNkovYk1SZzZrNlRQVApibUdmWENoaFRqeHdxQ2
xUUmxWc0tLbW9lRXNETWRndXExc21ZbmRSaEx2Y0F5Mm4wbXJUNThDMnozbXRiRGNZCkNrWXY5S
nRvM3JGbHJWMmk1dEs4NlRjL2NIS0pWZ1pxWDdVQlU3VmRsanJSU2pvUUNXOGd0L2FuTXhuQ3RT
VkMKUUQwQ0F3RUFBYU5DTUVBd0RnWURWUjBQQVFIL0JBUURBZ0trTUE4R0ExVWRFd0VCL3dRRk1
BTUJBZjh3SFFZRApWUjBPQkJZRUZKWTZ4a0xEdWlMdWN2SjJGcWhSK3dzV1JaTXZNQTBHQ1NxR1
NJYjNEUUVCQ3dVQUE0SUJBUUNmCndRa3Q2cnJSTE9LTE1sU01uODJMWThlRW9NVHRBL1ZmRWpsN
XdYaFJ4K0kxWGFGRE9iRnUrY3V3SWN6OVBOYUMKemFrSWpZdWN5NWdlNmZnQWRFVzNOL0RDR1dP
WEJFNTh6N3R2a0V0U2N5bGV5cEdjV0U3ZlZKeWtjVU1NcThyaQo1TlI0WXBoTjRJNlMvbGtHeTE
5c0J6NzI3STFqSGwyR3lSQVlXZlhkQnZUeUV4WFo5MUlpZmNVWUVBMUJBQkJ3Cld2WGZaK3Mxb3
g2dTNFaUFOZGVlUkNXeS9yWis1NTg5TFRzVUtsQ3BMTTFhTlhHRjhiQjZFMW1kdERqNGVxMmsKY
ndnbVFEWTAyWlJScVJQVHBWR1JlTXdpZytGSmRPWU41Q1FnQzc0T1pUTnVJQ0F2aTZDRUcvNUI4
NktpOWJyWQp3L2ovOVFaNW9pa1lvVmVyY3pyTgotLS0tLUVORCBDRVJUSUZJQ0FURS0tLS0tCg==
=
  name: kubernetes
contexts:
- context:
    cluster: kubernetes
```

you shall be able to use kubectl get the node information.

```
root@iZj6cg0w474i2p8oykt7kqZ:~# kubectl get node
--kubeconfig=/root/.kube/config_ack1 -o wide
NAME                     STATUS   ROLES    AGE    VERSION
INTERNAL-IP   EXTERNAL-IP   OS-IMAGE                KERNEL-VERSION
CONTAINER-RUNTIME
cn-hongkong.10.1.0.163   Ready    <none>   147m   v1.22.3-aliyun.1
10.1.0.163    <none>        CentOS Linux 7 (Core)
3.10.0-1160.15.2.el7.x86_64   containerd://1.4.8
cn-hongkong.10.1.0.164   Ready    <none>   147m   v1.22.3-aliyun.1
10.1.0.164    <none>        CentOS Linux 7 (Core)
3.10.0-1160.15.2.el7.x86_64   containerd://1.4.8
```

Deploy application on ACK1

```
root@iZj6cg0w474i2p8oykt7kqZ:~# cat simple-deployment.yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx-deployment
  labels:
    app: nginx
spec:
  replicas: 2
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
      - name: nginx
        image: nginx:1.14.2
        ports:
```

```
        - containerPort: 80

root@iZj6cd1vfw01yb6cq9jsqyZ:~# kubectl apply -f simple-deployment.yaml
root@iZj6cg0w474i2p8oykt7kqZ:~# cat k8s_svc_nginx.yaml
kind: Service
apiVersion: v1
metadata:
  name: nginx-30163
spec:
  # Expose the service on a static port on each node
  # so that we can access the service from outside the cluster
  type: NodePort

  # When the node receives a request on the static port (30163)
  # "select pods with the label 'app' set to 'echo-hostname'"
  # and forward the request to one of them
  selector:
    app: nginx

  ports:
    # Three types of ports for a service
    # nodePort - a static port assigned on each the node
    # port - port exposed internally in the cluster
    # targetPort - the container port to send requests to
    - nodePort: 30163
      port: 80
      targetPort: 80

root@iZj6cd1vfw01yb6cq9jsqyZ:~# kubectl apply -f k8s_svc_nginx.yaml

root@iZj6cg0w474i2p8oykt7kqZ:~# kubectl get svc
--kubeconfig=/root/.kube/config_ack1
NAME          TYPE        CLUSTER-IP     EXTERNAL-IP   PORT(S)        AGE
kubernetes    ClusterIP   172.21.0.1     <none>        443/TCP        153m
nginx-30163   NodePort    172.21.8.187   <none>        80:30163/TCP   135m
root@iZj6cg0w474i2p8oykt7kqZ:~#
```

Create ACK2

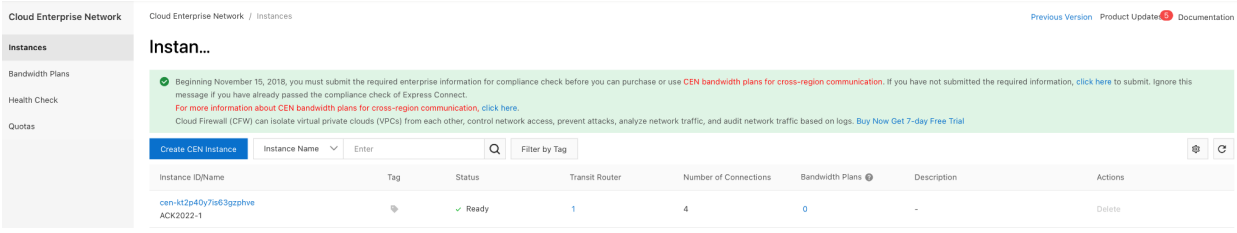
Do the same as ACK1 . except it has a different vswitch IP address, so POD and NODE have different IP address subnets.

```
root@iZj6cg0w474i2p8oykt7kqZ:~# kubectl get deployment
--kubeconfig=/root/.kube/config_ack2
NAME               READY   UP-TO-DATE   AVAILABLE   AGE
nginx-deployment   2/2     2            2           136m
root@iZj6cg0w474i2p8oykt7kqZ:~# kubectl get svc
--kubeconfig=/root/.kube/config_ack2
NAME          TYPE        CLUSTER-IP      EXTERNAL-IP   PORT(S)        AGE
kubernetes    ClusterIP   172.21.0.1      <none>        443/TCP        154m
nginx-30163   NodePort    172.21.14.153   <none>        80:30163/TCP   136m
```

Create and Config CEN-TR

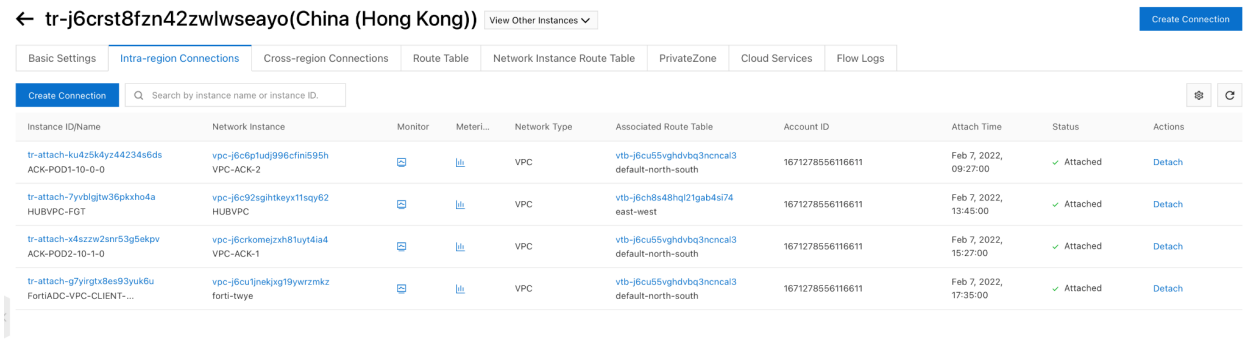
A CEN-TR is required to attach ACK1 ,ACK2 , HUBVPC VPC as well as fortiADC VPC.

Two Routing Tables in CEN-TR are required, one is the default Routing Table, other one is custom Routing table for HUVPC route traffic to ACK1 , ACK2, and fortiADC VPC.
default Routing table is for ACK1,ACK2, fortiADC VPC to reach HUBVPC where it has only one 0.0.0.0/0 route with nexthop set to TR router.



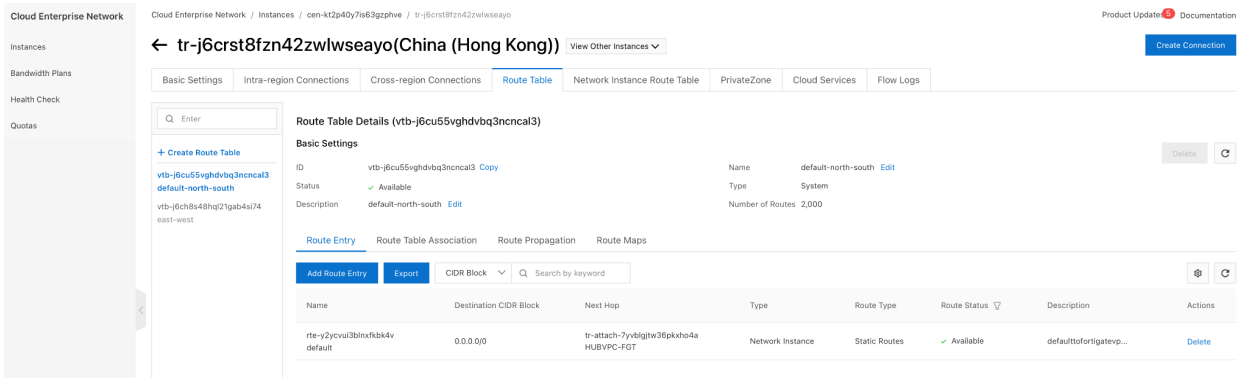
so, this CEN TR router will have 4 Connections, and 2 Route Tables.

TR Connections



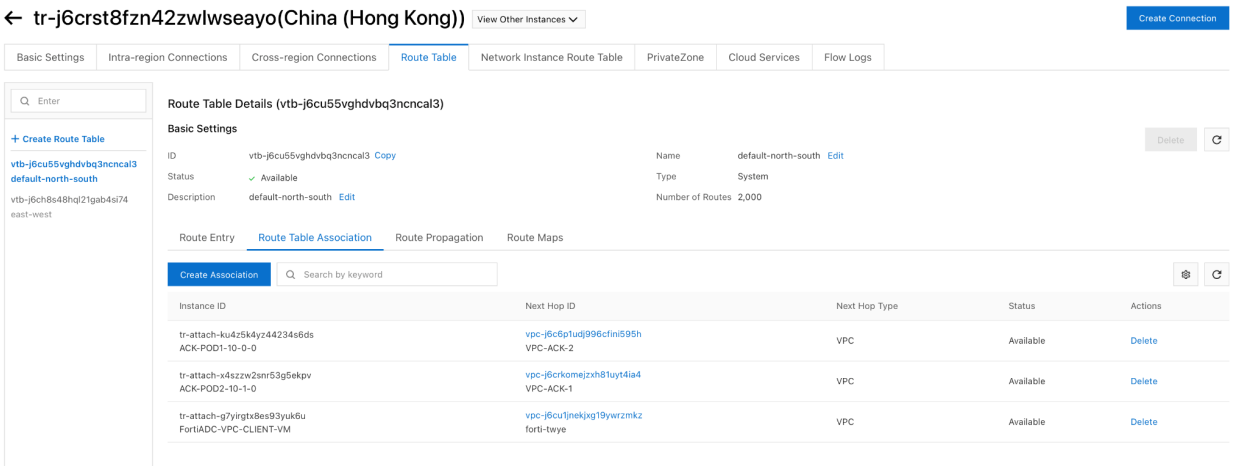
Route Table- defaul-north-south

default north-south route table is for north-south traffic (ACK to internet and ACK to FortiADC, fortiADC to ACK etc)



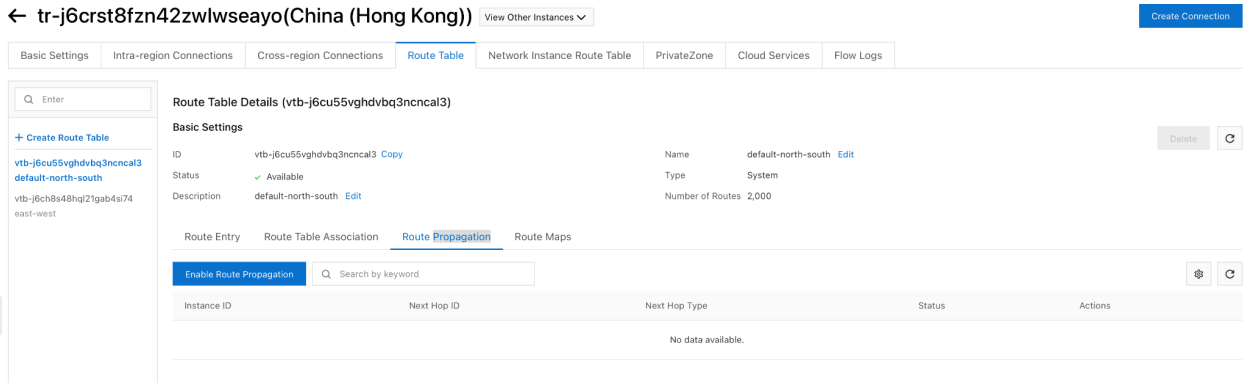
route table association

default-north-south route table is associated with ACK1, ACK2 and FortiADC-Client VPC.



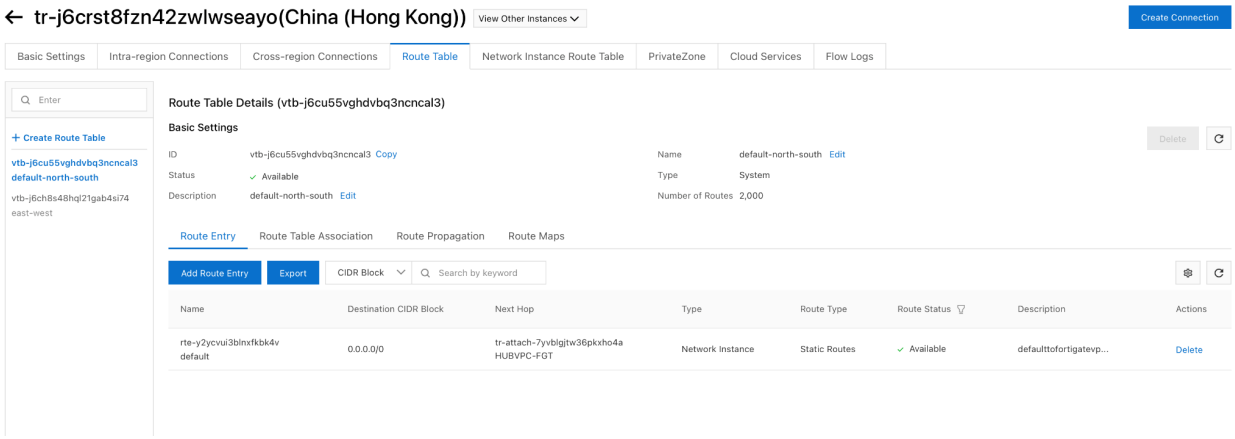
Remove Route Propagation

but it will not enable Route propagation from those VPCs, it only need default route to reach HUBVPC. so route propagation from ACK1,ACK2,FortiADC-Client VPC must be removed.



route entry

a default route 0.0.0.0/0 with nexthop to TR-attach to HUBVPC must be configured. so all traffic from ACK1,ACK2,FortADC-Client VPC will be sent to TR to reach HUBVPC.

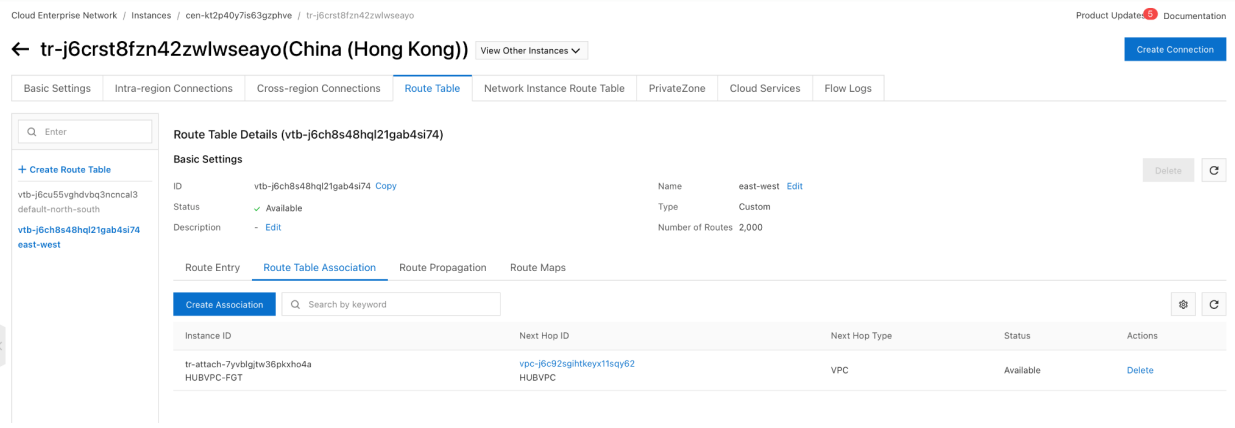


Route Table east-west

Route table east-west is for ACK cross VPC traffic, the traffic between ACKs VPC. those traffic subject be inspected by HUBVPC fortigate , Fortigate must learn ACK1 ,ACK2, FortiADC-clientVM subnet , so HUBVPC know where to send packet back to ACK1, ACK2, FortiADC after it done the inspection.

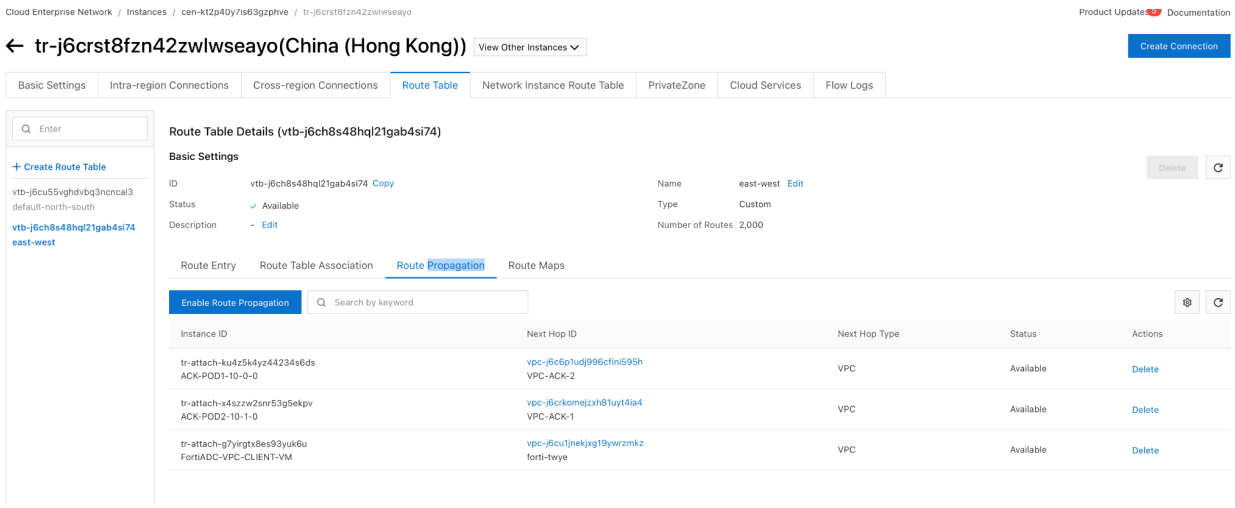
route table association

this route table is only associated with HUBVPC.

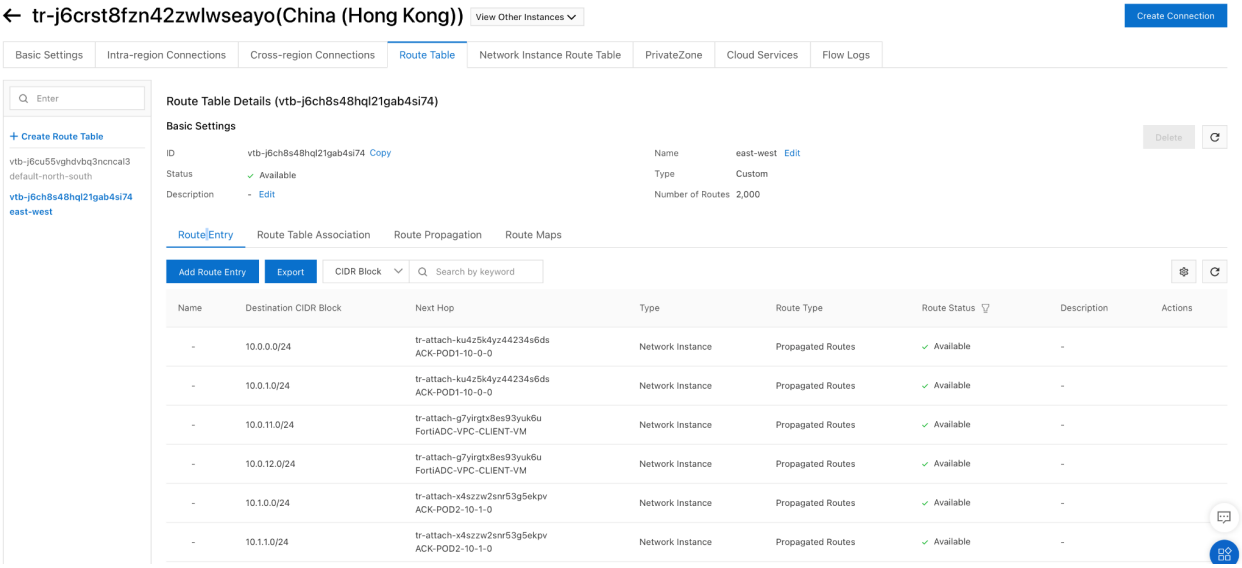


Enable Route Propagation

Route Propagation from ACK1,ACK2, fortiADC-client must be enabled. for HUBVPC know how to route traffic.



The route Entry



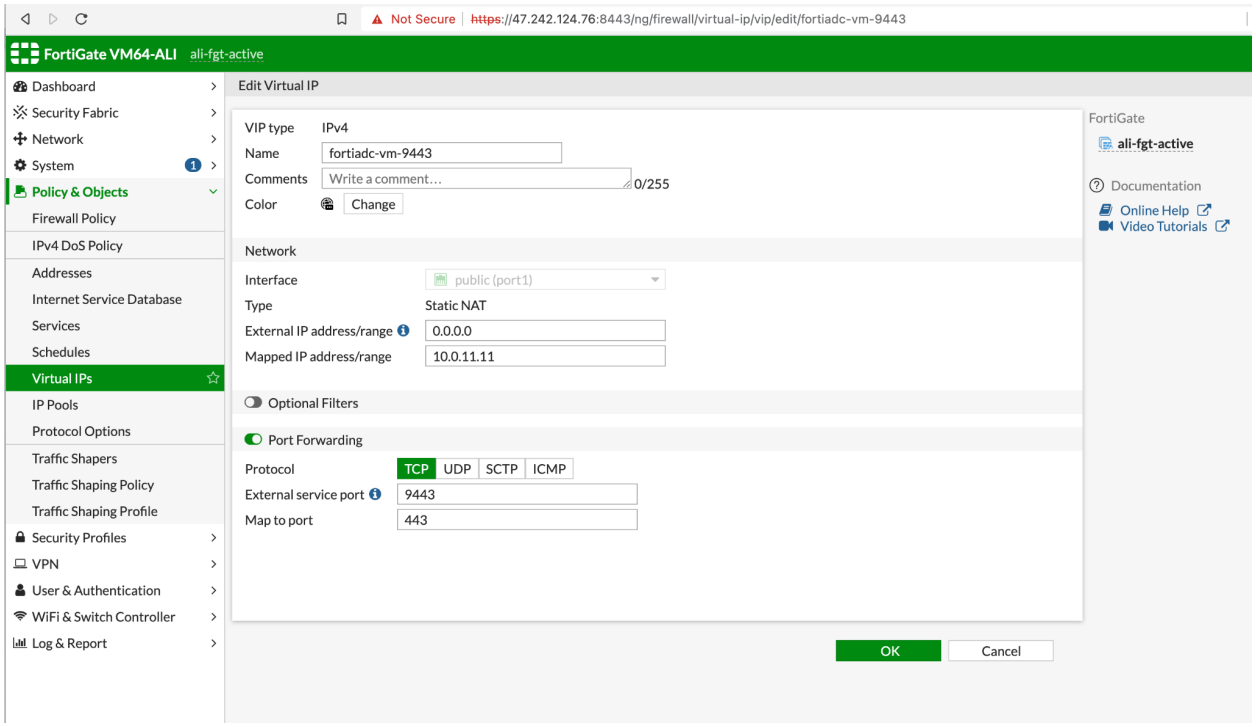
a total of six routes from ACK1, ACK2, fortiADC VPC will be populated in this routing table.

if the Access key/secret from Alicloud is not available for the fortigate to use. Therefore configuring the SDN connector on fortigate to retrieve client-VM and fortiADC VM is not possible, you will need to get the IP address of client VM and fortiADC VM, then create static entry on fortigate with Virtual IP to access client VM and fortiadc instead use SDN connector.

Config Fortigate VIP for access FortiADC

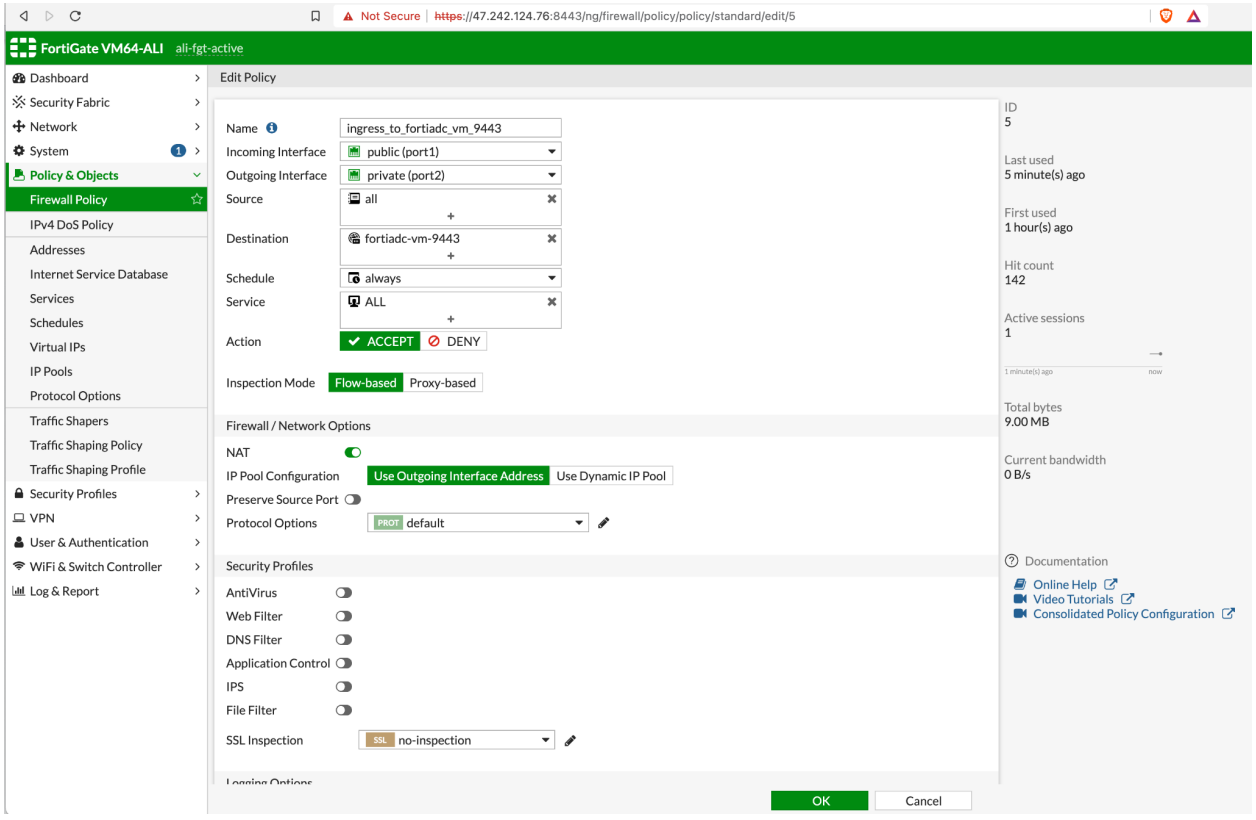
If FortiADC sits behind fortigate and does not a public IP, we can config a VIP on Fortigate to access fortiadc. below we config a GUI access to fortiadc via VIP:Port on FortiGate and a SSH access to fortiadc.

GUI access VIP mapping

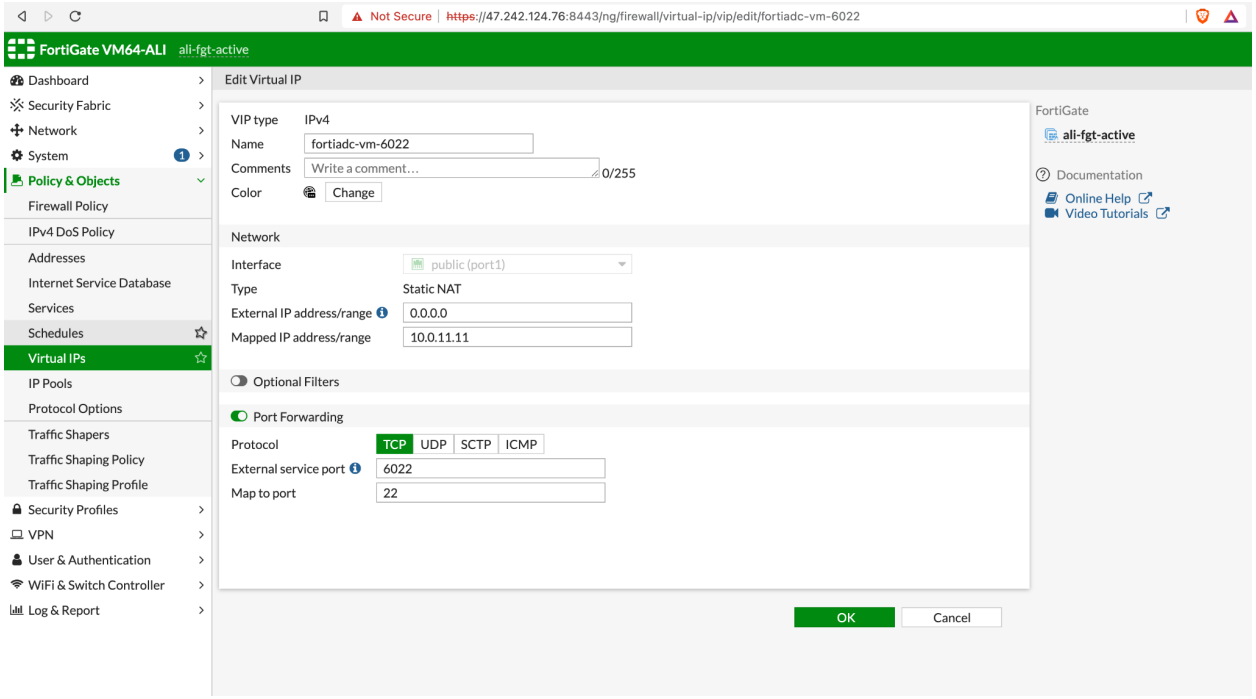


Firewall Policy

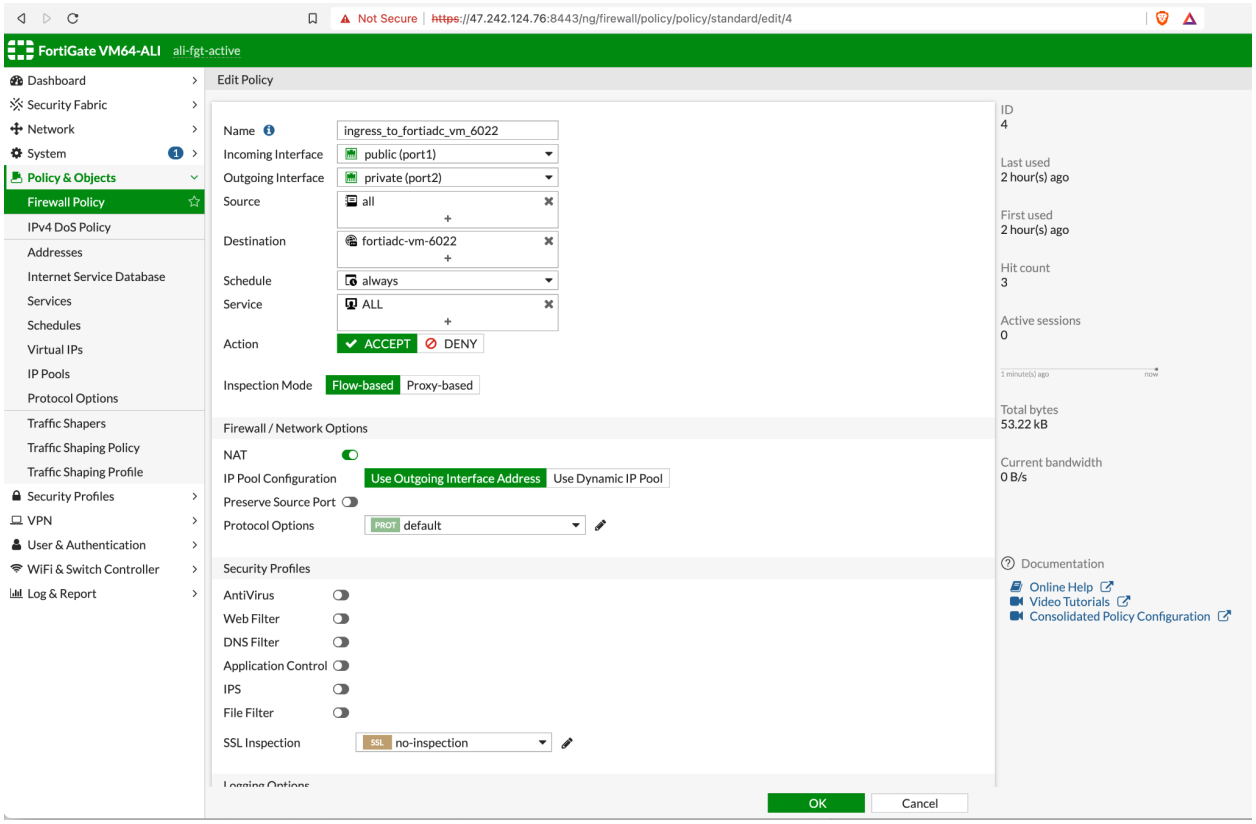
Create a policy to allow this traffic



SSH access VIP mapping

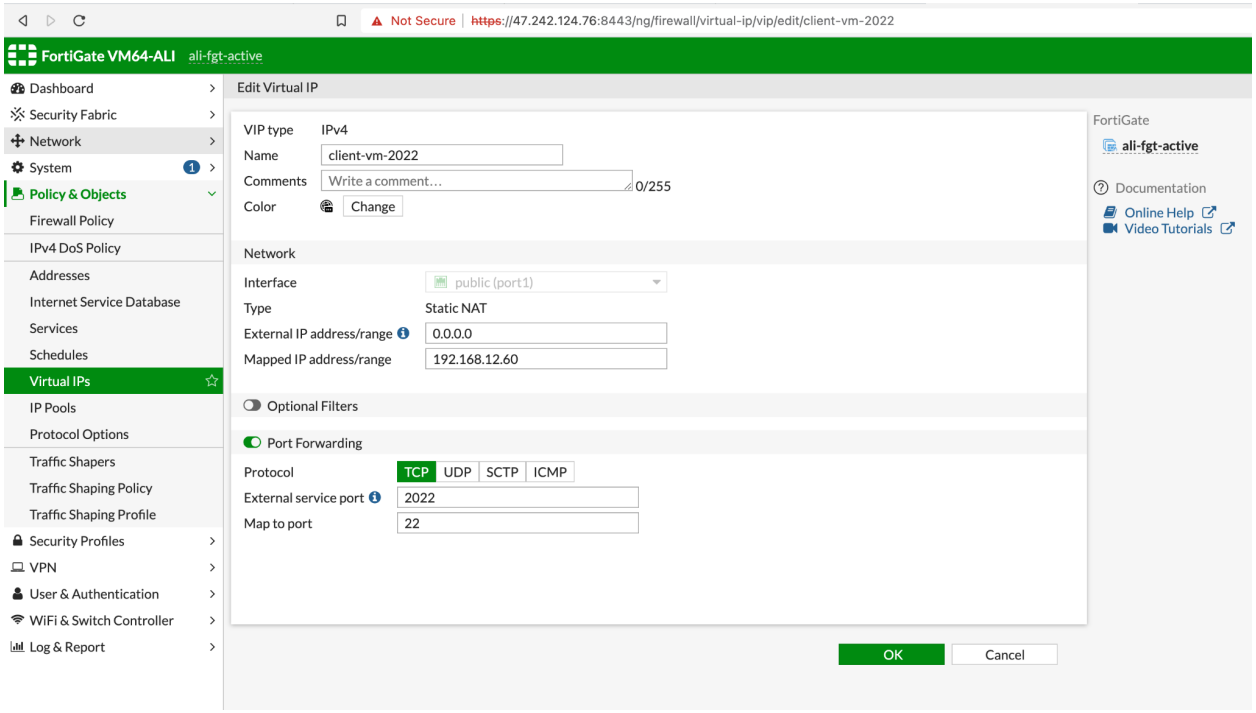


Firewall Policy

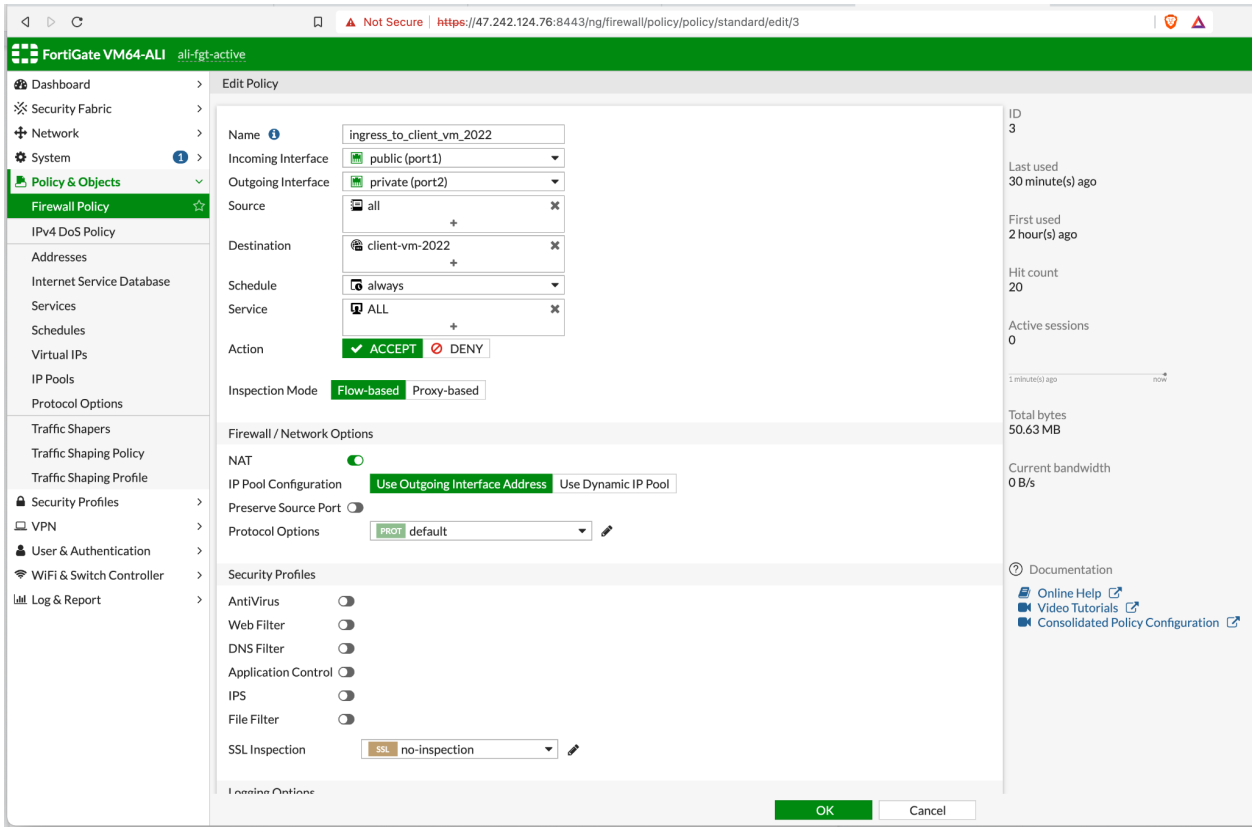


Config Fortigate VIP for access client-VM

SSH access VIP mapping



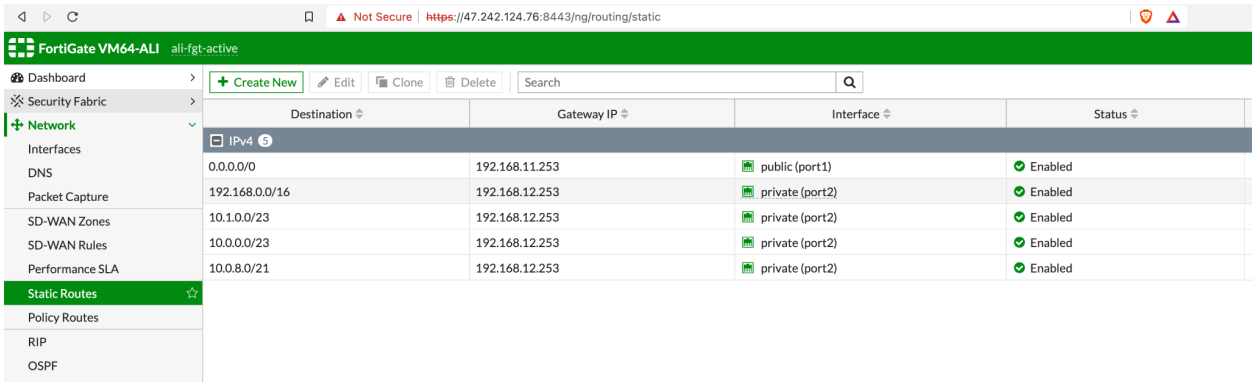
Firewall policy



Config Fortigate Routing

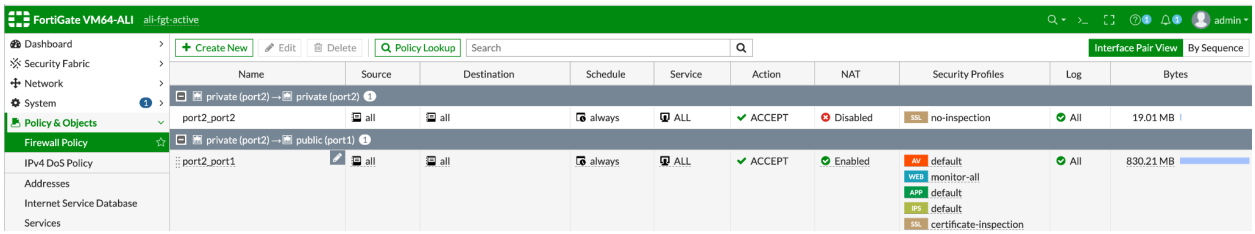
Config static Route

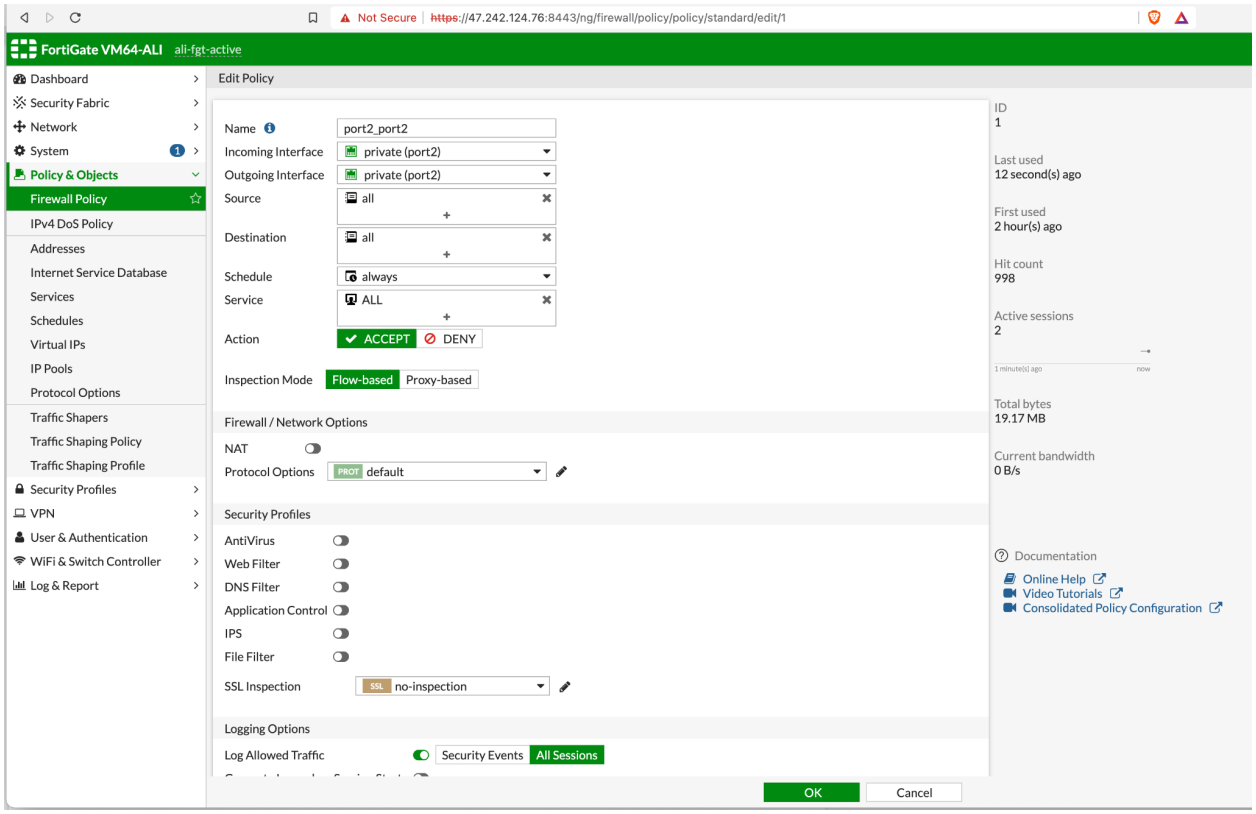
On the fortigate, route entry to ACK-1 , ACK-2, fortiADC subnet is needed. When traffic from TR reaches the fortigate (via TR-landing vswitch), the fortigate needs to lookup the VM inside the routing table to make a decision where to route the traffic. in this POC, the traffic will be sent back to port2.

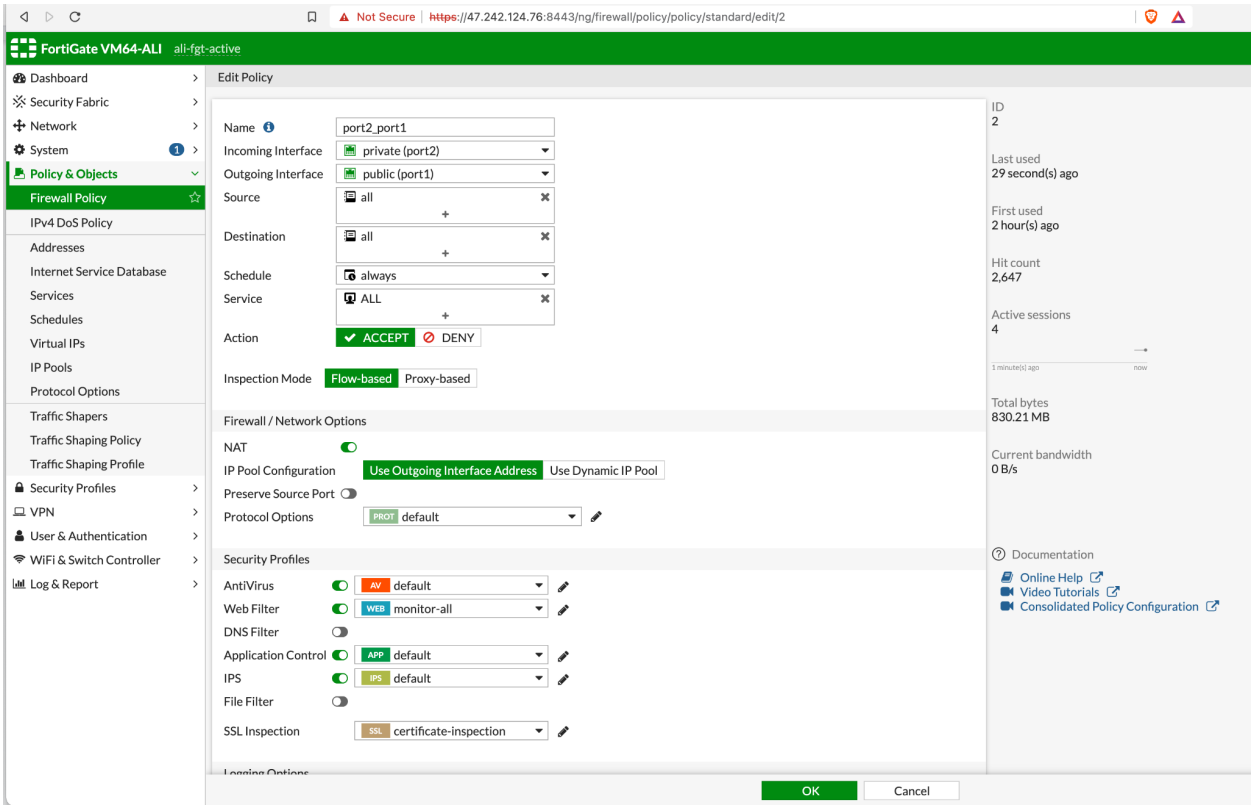


Config Firewall Policy for East-west traffic and ACK egress traffic to internet

Two Firewall policies are required. One is for traffic from ACKs to the internet. Another one is for traffic from ACK1 to ACK2 as well as ACK1,ACK2 to fortiADC VM.




Not Secure
https://47.242.124.76:8443/ng/firewall/policy/policy/standard/edit/1
FortiGate VM64-ALI
ali-fgt-active
Dashboard
Security Fabric
Network
System
Policy & Objects
Firewall Policy
IPv4 DoS Policy
Addresses
Internet Service Database
Services
Schedules
Virtual IPs
IP Pools
Protocol Options
Traffic Shapers
Traffic Shaping Policy
Traffic Shaping Profile
Security Profiles
VPN
User & Authentication
WiFi & Switch Controller
Log & Report
Edit Policy
Name
port2_port2
Incoming Interface
private (port2)
Outgoing Interface
private (port2)
Source
all
Destination
all
Schedule
always
Service
ALL
Action
ACCEPT
DENY
Inspection Mode
Flow-based
Proxy-based
Firewall / Network Options
NAT
Protocol Options
PROT
default
Security Profiles
AntiVirus
Web Filter
DNS Filter
Application Control
IPS
File Filter
SSL Inspection
SSL
no-inspection
Logging Options
Log Allowed Traffic
Security Events
All Sessions
OK
Cancel
ID
1
Last used
12 second(s) ago
First used
2 hour(s) ago
Hit count
998
Active sessions
2
1 minute(s) ago
now
Total bytes
19.17 MB
Current bandwidth
0 B/s
Documentation
Online Help
Video Tutorials
Consolidated Policy Configuration

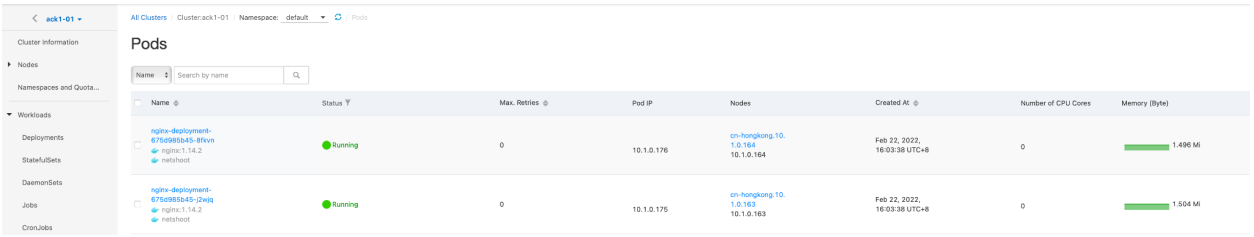


Verify the reachability

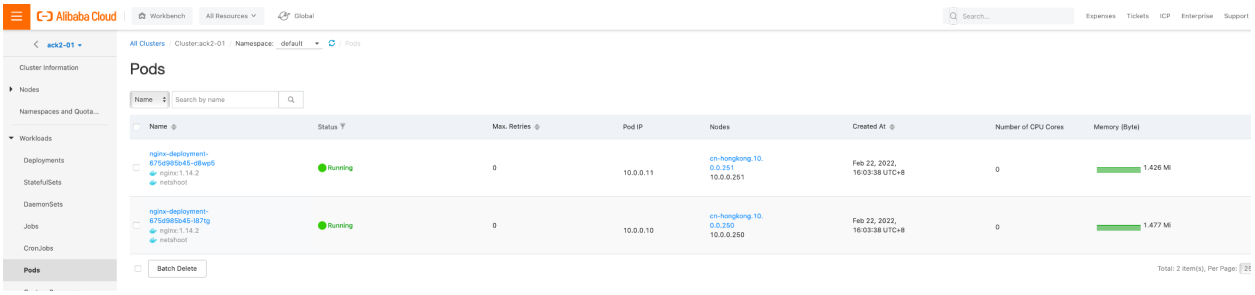
ACK1 to ACK2

POD1 to POD2 and PODs to Node IP

ACK1 POD and Node IP



ACK2 POD and Node IP



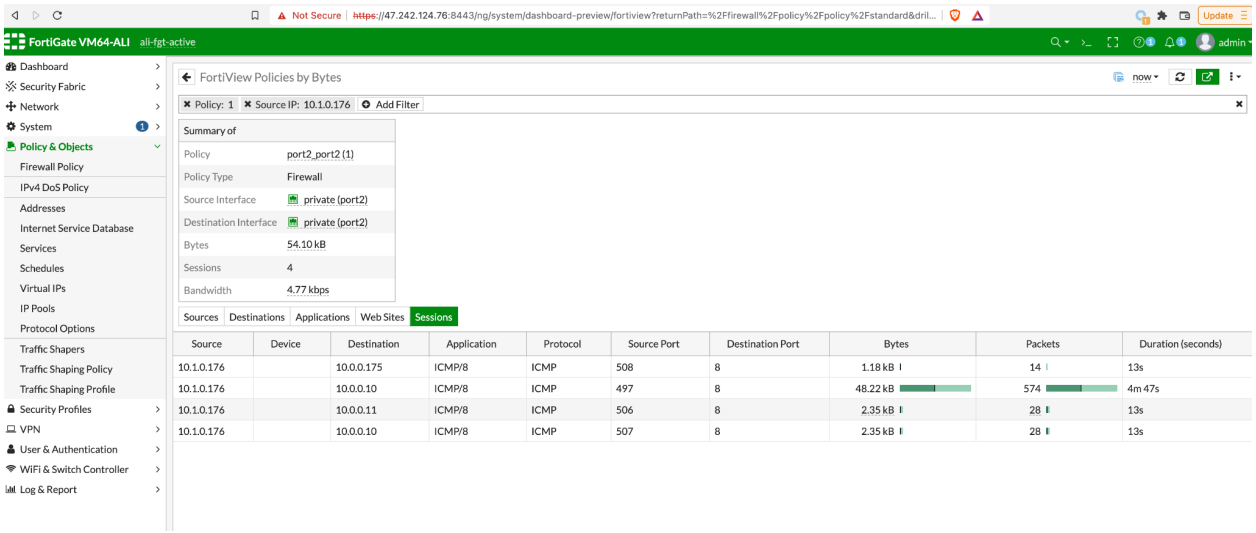
Ping Result

ACK1 POD1 can successfully ping ACK2 POD2 and is also able to access the internet . and traffic can also be inspected by fortigate.

```
root@iZj6cg0w474i2p8oykt7kqZ:~# kubectl1 exec -it
po/nginx-deployment-675d985b45-8fkvn -- sh
Defaulted container "nginx" out of: nginx, netshoot
# apt update
Hit:1 http://security.debian.org/debian-security stretch/updates InRelease
Ign:2 http://deb.debian.org/debian stretch InRelease
Hit:3 http://deb.debian.org/debian stretch-updates InRelease
Hit:4 http://deb.debian.org/debian stretch Release
Reading package lists... Done
Building dependency tree
Reading state information... Done
32 packages can be upgraded. Run 'apt list --upgradable' to see them.
```

```
# apt install iputils-ping
Reading package lists... Done
Building dependency tree
Reading state information... Done
iputils-ping is already the newest version (3:20161105-1).
0 upgraded, 0 newly installed, 0 to remove and 32 not upgraded.
# ping 10.0.0.10 -c 1
PING 10.0.0.10 (10.0.0.10) 56(84) bytes of data.
64 bytes from 10.0.0.10: icmp_seq=1 ttl=59 time=3.25 ms

--- 10.0.0.10 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 3.256/3.256/3.256/0.000 ms
# ping 10.0.0.11 -c 1
PING 10.0.0.11 (10.0.0.11) 56(84) bytes of data.
64 bytes from 10.0.0.11: icmp_seq=1 ttl=59 time=3.53 ms

--- 10.0.0.11 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 3.530/3.530/3.530/0.000 ms
# ping 10.1.0.176 -c 1
PING 10.1.0.176 (10.1.0.176) 56(84) bytes of data.
64 bytes from 10.1.0.176: icmp_seq=1 ttl=64 time=0.029 ms

--- 10.1.0.176 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 0.029/0.029/0.029/0.000 ms
# ping 10.1.0.175 -c 1
PING 10.1.0.175 (10.1.0.175) 56(84) bytes of data.
64 bytes from 10.1.0.175: icmp_seq=1 ttl=62 time=0.304 ms

--- 10.1.0.175 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 0.304/0.304/0.304/0.000 ms
#
```

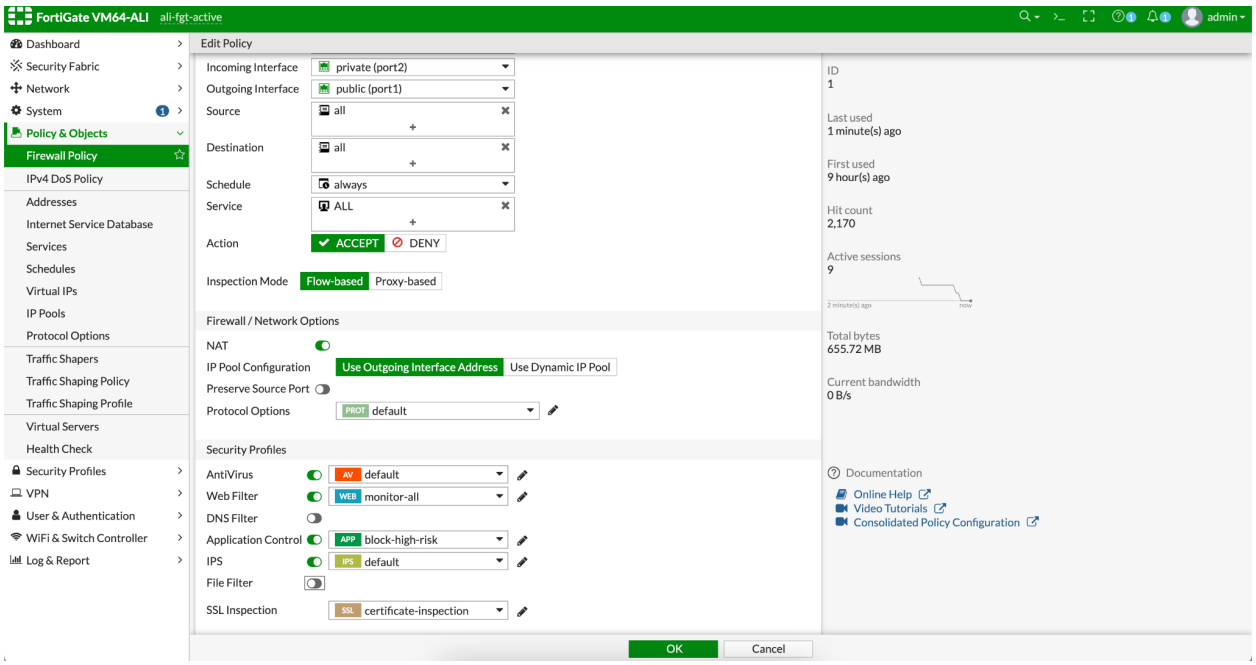
FortiGate View



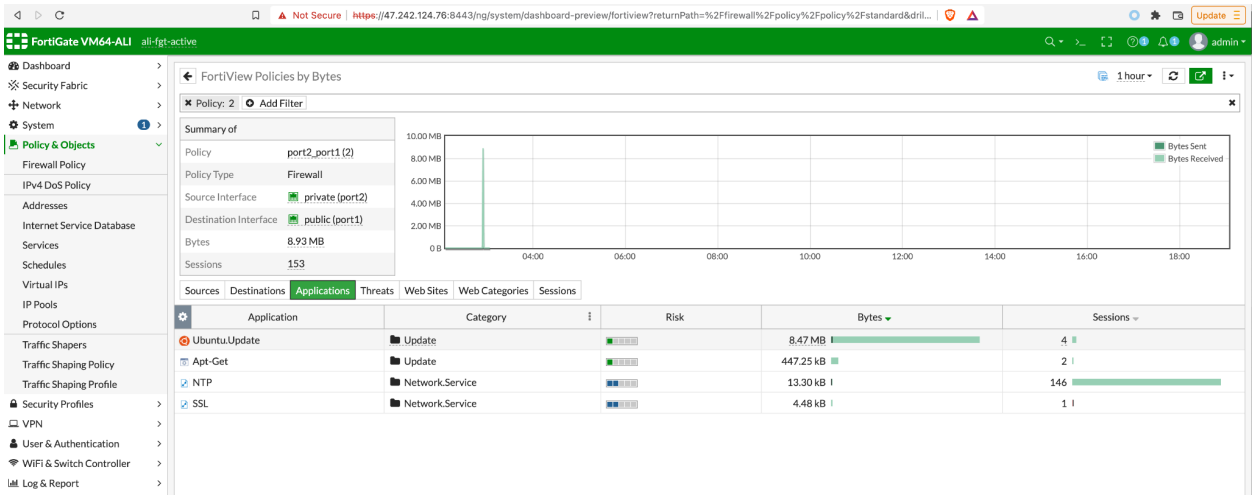
you can see the traffic between POD (10.1.0.176) and other POD 10.0.0.175,10.0.0.10,10.0.0.11,10.0.0.10 etc

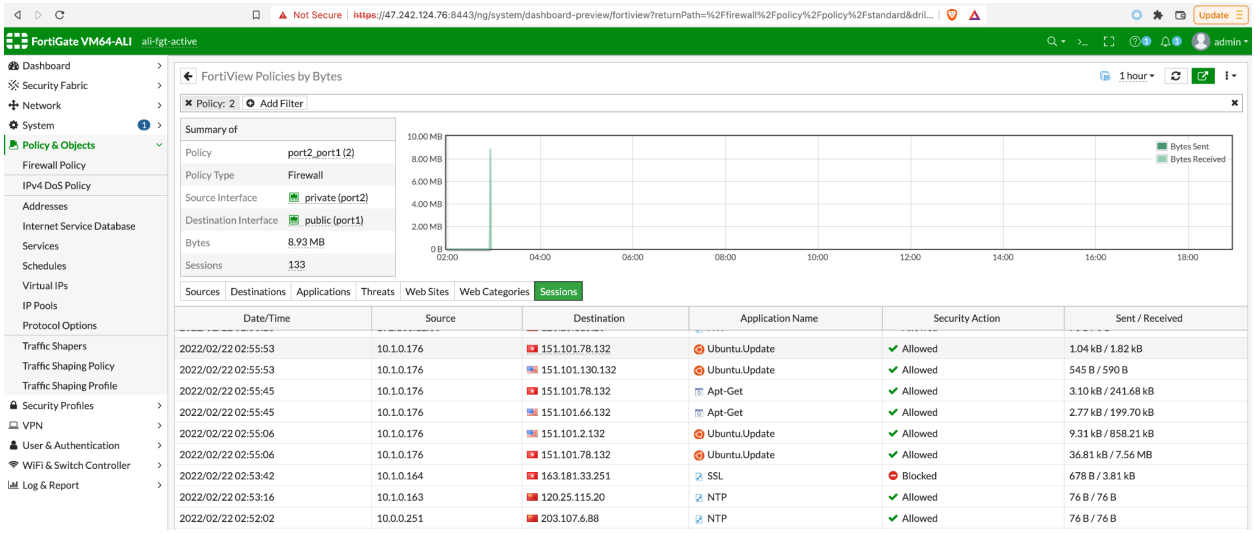
enable security feature for egress traffic.

to fine control what traffic allows to egress to the internet, you can enable fortigate builtin Layer 7 security feature.



After that, by default, you will see more detailed traffic information as well as you can see some suspicious traffic is dropped by default policy.





Config FortiADC as Ingress controller

FortiADC has a built- in k8s connector which can dynamically pull ACK SVC information and use that as a real server as target to create Layer 4 Load Balancer or Layer 7 Ingress Rule.

Config FortiADC to connect ACK1 and ACK2

Create ServiceAccount on both ACK

in the terraform script, create a Service Account already included. this step only required if you choose manually to deploy infrastructure .

```
root@iZj6cd1vfw01yb6cq9jsqyZ:~# kubectl cluster-info
Kubernetes control plane is running at https://10.0.0.221:6443
metrics-server is running at
https://10.0.0.221:6443/api/v1/namespaces/kube-system/services/heapster/pro
xy
KubeDNS is running at
https://10.0.0.221:6443/api/v1/namespaces/kube-system/services/kube-dns:dns
/proxy

To further debug and diagnose cluster problems, use 'kubectl cluster-info
dump'.
root@iZj6cd1vfw01yb6cq9jsqyZ:~#


root@iZj6cd1vfw01yb6cq9jsqyZ:~# kubectl -n kube-system create
serviceaccount ftntconnector
serviceaccount/ftntconnector created



root@iZj6cd1vfw01yb6cq9jsqyZ:~# kubectl create clusterrolebinding
service-admin --clusterrole=cluster-admin
--serviceaccount=kube-system:ftntconnector
clusterrolebinding.rbac.authorization.k8s.io/service-admin created
```

```
root@iZj6cd1vfw01yb6cq9jsqyZ:~#  kubectl describe secrets ftntconnector -n
kube-system
Name:         ftntconnector-token-r2zrt
Namespace:    kube-system
Labels:       <none>
Annotations:  kubernetes.io/service-account.name: ftntconnector
              kubernetes.io/service-account.uid:
b22929da-808c-462b-97aa-441054631fa4

Type:  kubernetes.io/service-account-token

Data
====
namespace:  11 bytes
token:
eyJhbGciOiJSUzI1NiIsImtpZCI6Ill6RTk0STNuME9feHBNNkhhTlpoMDRhbjI1ZzBkN29HeEN
KbHprVWM1dE0ifQ.eyJpc3MiOiJrdWJlcm5ldGVzL3NlcnZpY2VhY2NvdW50Iiwia3ViZXJuZXR
lcy5pby9zZXJ2aWNlYWNjb3VudC9uYW1lc3BhY2UiOiJrdWJlLXN5c3RlbSIsImt1YmVybmV0ZX
MuaW8vc2VydmljZWFjY291bnQvc2VjcmV0Lm5hbWUiOiJmdG50Y29ubmVjdG9yLXRva2VuLXIye
nJ0Iiwia3ViZXJuZXRlcy5pby9zZXJ2aWNlYWNjb3VudC9zZXJ2aWNlLWFjY291bnQubmFtZSI6
ImZ0bnRjb25uZWN0b3IiLCJrdWJlcm5ldGVzLmlvL3NlcnZpY2VhY2NvdW50L3NlcnZpY2UtYWN
jb3VudC51aWQiOiJiMjI5MjlkYS04MDhjLTQ2MmItOTdhYS00NDEwNTQ2MzFmYTQiLCJzdWIiOi
JzeXN0ZW06c2VydmljZWFjY291bnQ6a3ViZS1zeXN0ZW06ZnRudGNvbm5lY3RvciJ9.N_77kMHu
8SEjAtgeDtzISGlaaFUY51Ekd0Y1jrESlWpt8sjxhhipYGrdyqFPfql2UyAlqDeI3W9Lnuf8_uW
FJywhoF2o8v209bTZ_SJ5F4_mnlFjx8JhcNXRyCdy5aIuzpIKyi4UBJ6MoyMgx1fBJJ5UzpVhsr
fhCC87fOCL5f8vtmcz5NDbSzmPeacwbcbN1UDhXeKWamn1Yl5UgkBY8WGqZesBmbtxnPPR5eeq4
s-PLwyRrtDNA9cakxKveLHQCPZCQKmRxEs_BZBrwloJnDLPC6-ZBSCKl2neo6jBAKDzUdBVSGHx
wWTxyb-HHItshfBFwsghblrTGdI-IyNbyQ
ca.crt:     1180 bytes

root@iZj6cd1vfw01yb6cq9jsqyZ:~#
```

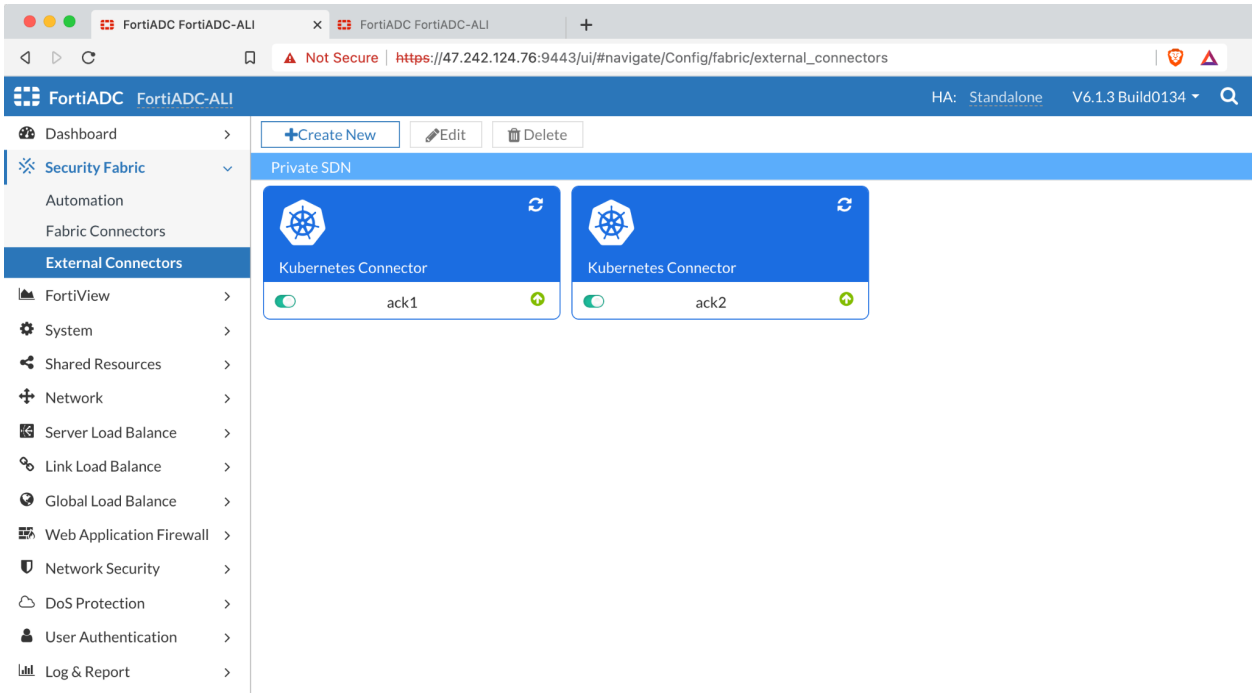
config k8s sdn connector on fortiADC

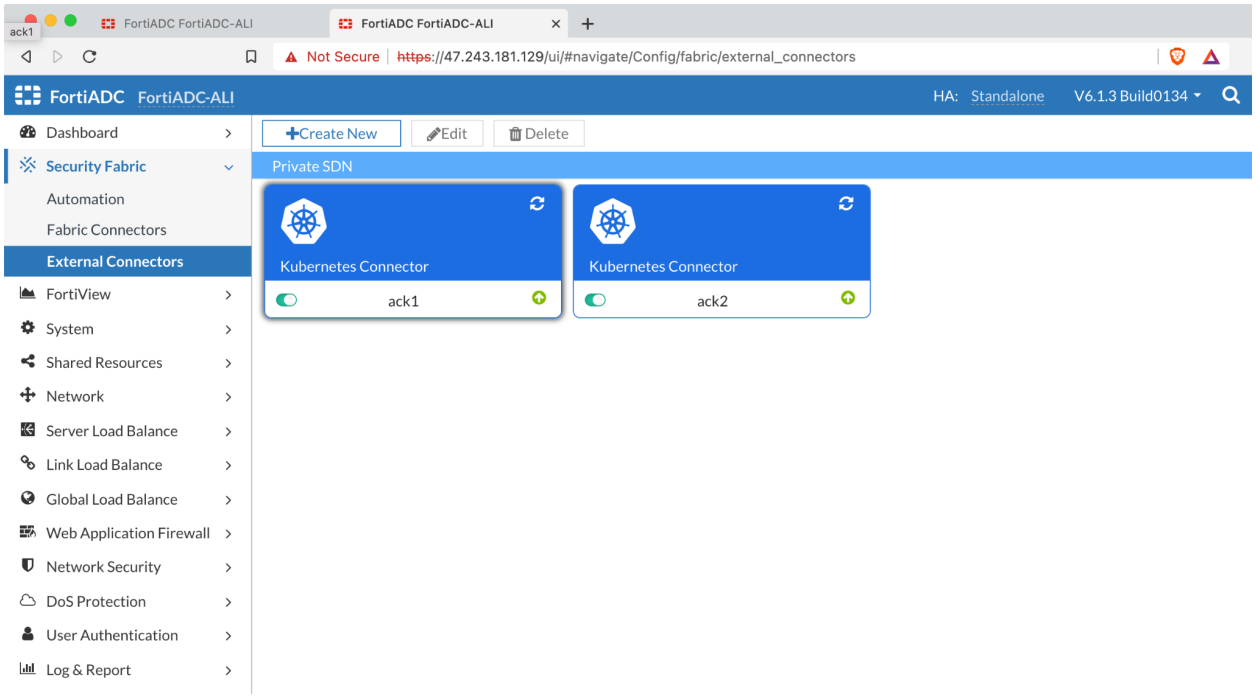
```
root@iZj6cd1vfw01yb6cq9jsqyZ:~# ssh admin@10.0.11.11
admin@10.0.11.11's password:
FortiADC-ALI # config system sdn-connector
```

```
FortiADC-ALI (sdn-connector) # edit ack1

FortiADC-ALI (ack1) # set server 10.1.0.160

FortiADC-ALI (ack1) # set server-port 6443

FortiADC-ALI (ack1) # set secret-token
eyJhbGciOiJSUzI1NiIsImtpZCI6IldTLUVKbkNoakNhLUQ3dExSY2tKeWlLR09VWVJZNlVYd1B
HU0N3elFJc28ifQ.eyJpc3MiOiJrdWJlcm5ldGVzL3NlcnZpY2VhY2NvdW50Iiwia3ViZXJuZXR
lcy5pby9zZXJ2aWNlYWNjb3VudC9uYW1lc3BhY2UiOiJrdWJlLXN5c3RlbSIsImt1YmVybmV0ZX
MuaW8vc2VydmljZWFjY291bnQvc2VjcmV0Lm5hbWUiOiJmdG50Y29ubmVjdG9yLXRva2VuLXc3Z
mJyIiwia3ViZXJuZXRlcy5pby9zZXJ2aWNlYWNjb3VudC9zZXJ2aWNlLWFjY291bnQubmFtZSI6
ImZ0bnRjb25uZWN0b3IiLCJrdWJlcm5ldGVzLmlvL3NlcnZpY2VhY2NvdW50L3NlcnZpY2UtYWN
jb3VudC51aWQiOiJjYTQzYzNjZi0zNzg5LTQzYjQtODRiYy1iMjM4MzBhMDNiNTciLCJzdWIiOi
JzeXN0ZW06c2VydmljZWFjY291bnQ6a3ViZS1zeXN0ZW06ZnRudGNvbm5lY3RvciJ9.lLQhahSo
OMTkEqQFX77bep7yxV3hoxEdt6R8Rj3dhdJLSAO5k7OWh9s6GsJ6KU6j7OowYdC2XVJUbJi8v2p
dV8Ok5rS2LHlk4iSNLDVHjQhNdnz-us0G3JAlO9zAseqyG_XBYfhNtebK5LFJOK1zrwL7r5SaIJ
3fRsElgXYFqZCdio3f-6_-QirL_MwQ5Uv6XSj8joc803TGp472deixwzwnMWMHpTa3U1IJtNUqD
Y0xnFYqE56YkQMM7sqaUR85SbHSgFf8kYgU21iFQQltpsMlBLow7IvViZ_xEf1IvSGm8uFrRe0l
43OmJ4vT1VvGVvCXsFAb-o5ew5CIOCtufA

FortiADC-ALI (ack1) # next

FortiADC-ALI (sdn-connector) # end

FortiADC-ALI # config system sdn-connector

FortiADC-ALI (sdn-connector) # edit ack2

FortiADC-ALI (ack2) # set server 10.0.0.247

FortiADC-ALI (ack2) # set server-port 6443

FortiADC-ALI (ack2) # set secret-token
eyJhbGciOiJSUzI1NiIsImtpZCI6IlBOQ3JuWlhvRU9NVXloVVpzQm15OVEyZjZJSDFwTk1YZXN
ueHJZS2ZRSlEifQ.eyJpc3MiOiJrdWJlcm5ldGVzL3NlcnZpY2VhY2NvdW50Iiwia3ViZXJuZXR
lcy5pby9zZXJ2aWNlYWNjb3VudC9uYW1lc3BhY2UiOiJrdWJlLXN5c3RlbSIsImt1YmVybmV0ZX
MuaW8vc2VydmljZWFjY291bnQvc2VjcmV0Lm5hbWUiOiJmdG50Y29ubmVjdG9yLXRva2VuLWd6d
3o0Iiwia3ViZXJuZXRlcy5pby9zZXJ2aWNlYWNjb3VudC9zZXJ2aWNlLWFjY291bnQubmFtZSI6
ImZ0bnRjb25uZWN0b3IiLCJrdWJlcm5ldGVzLmlvL3NlcnZpY2VhY2NvdW50L3NlcnZpY2UtYWN
jb3VudC51aWQiOiIyMDQ1MTMwMy02YmYxLTQ1MGQtYTVkNi0wMGE4NDU1ZTAzZjYiLCJzdWIiOi
JzeXN0ZW06c2VydmljZWFjY291bnQ6a3ViZS1zeXN0ZW06ZnRudGNvbm5lY3RvciJ9.YE48oJo-
cvIFzeUj7kSjzGJprY7aw3o6RLv9npizXGel7SfVb_nJaqc3DBVCge4CdMOfAhd0FA81RkiQi2W
```

```
LLFk6ya4ClsjoWYP6wviOpxAWBDMK7qyfrKOJJj9ipyjtRIMWDz4b8MiYuzJxsNyRfXglU1pNDf
LzSrJPKbeSzAN8pUci3KGgiyCZM8G06EEMFOPWyKSUTk0HQDLibMrFi60PJj1D9LW9-CQWfh-FC
yp9zu2husdHBl7itHxhsI17KwVlu_7fu5zkBOn3mnw0XUzT8EY8h2pnlXzsan6sCahIcNkyNCSk
P3Ny7fkXI9MCTsac7M_Y9JxBoSn2m_sW-A

FortiADC-ALI (ack2) # end

FortiADC-ALI #
```

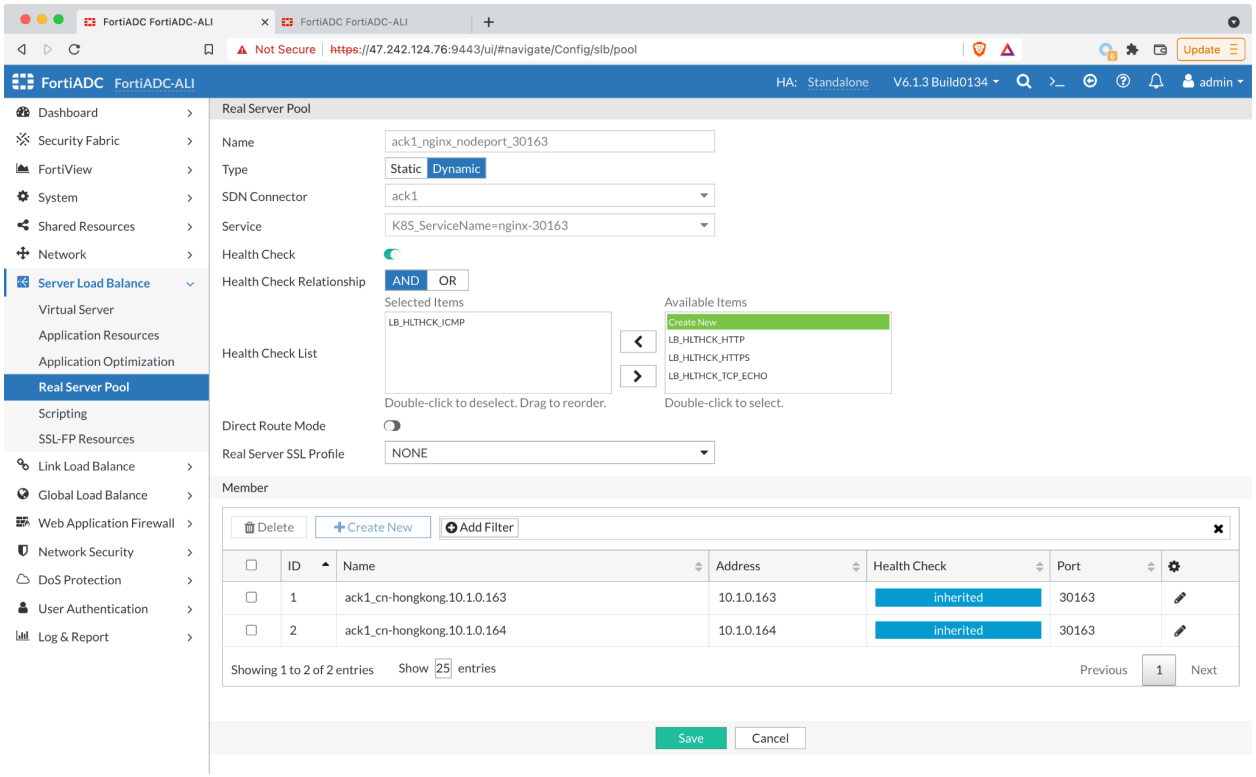
once done. on the fortiADC console. You will see below, an green icon indicating the success connection between FortiADC and both ACKs.
you can use the browser to logon fortiadc GUI to take a look.
The fortiadc GUI can be accessed via fortigate VIP:9443 or fortiadc public IP:443 (if configured).





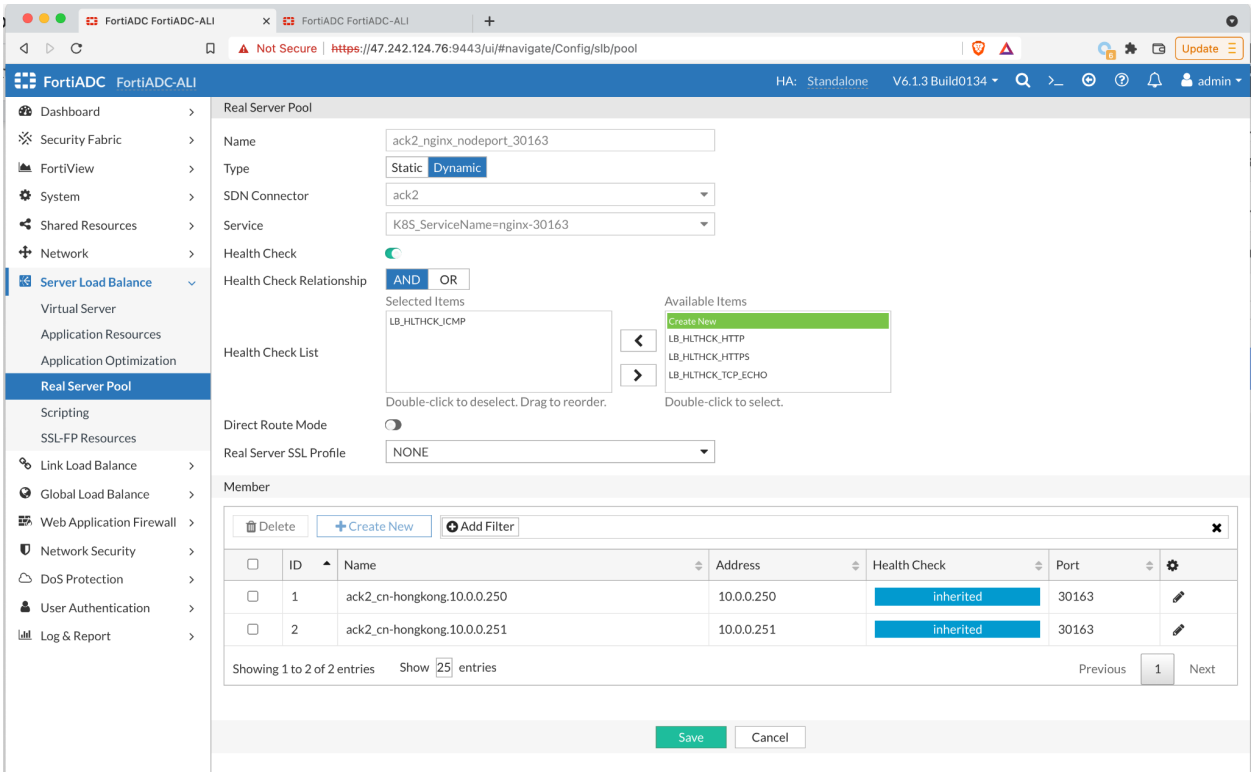
Create ingress rule for ACK1 nodeport service

Create a Real Server Pool on FortiADC

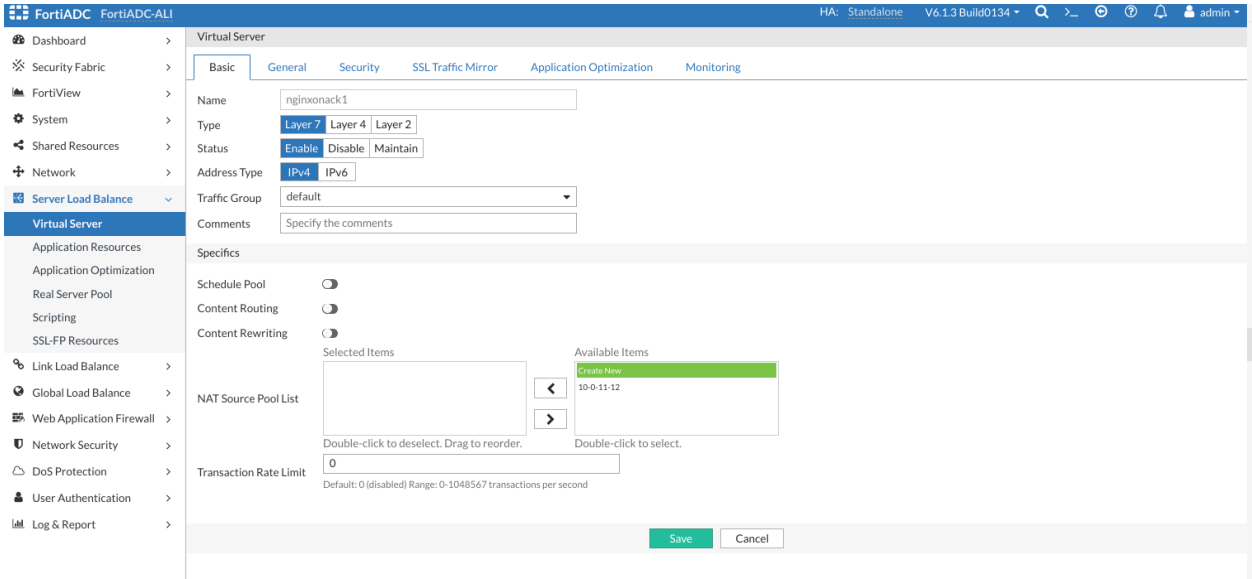


After clicking Save, the Member will be automatically populated.

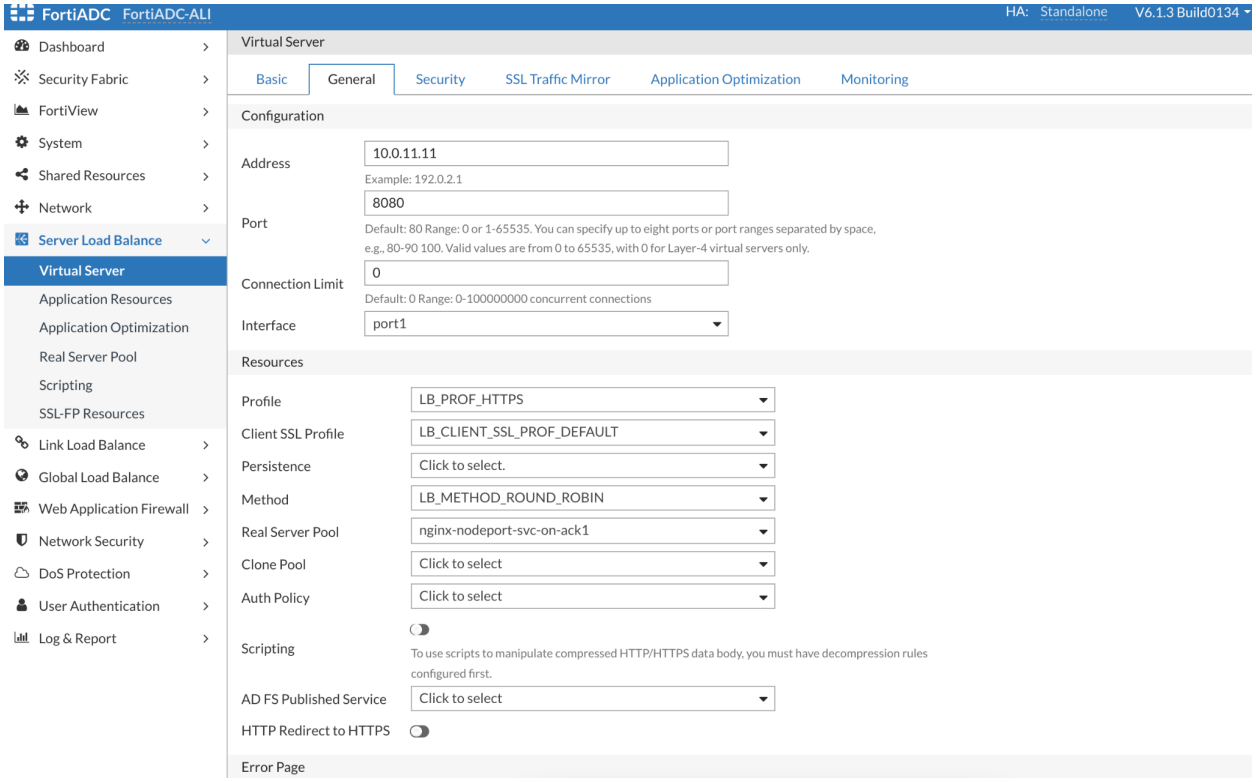
Do same for nginx on ACK2



Create Layer 7 Ingress Rule with RealServer Pool

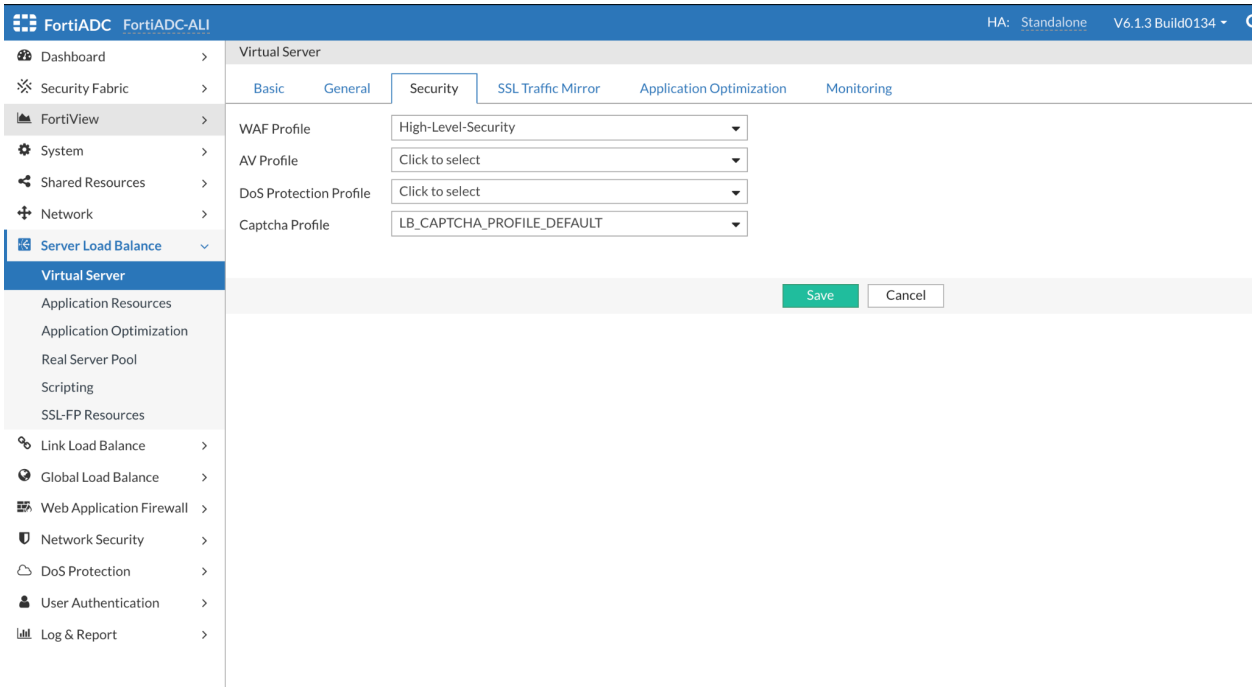


Choose Layer 7 ingress.

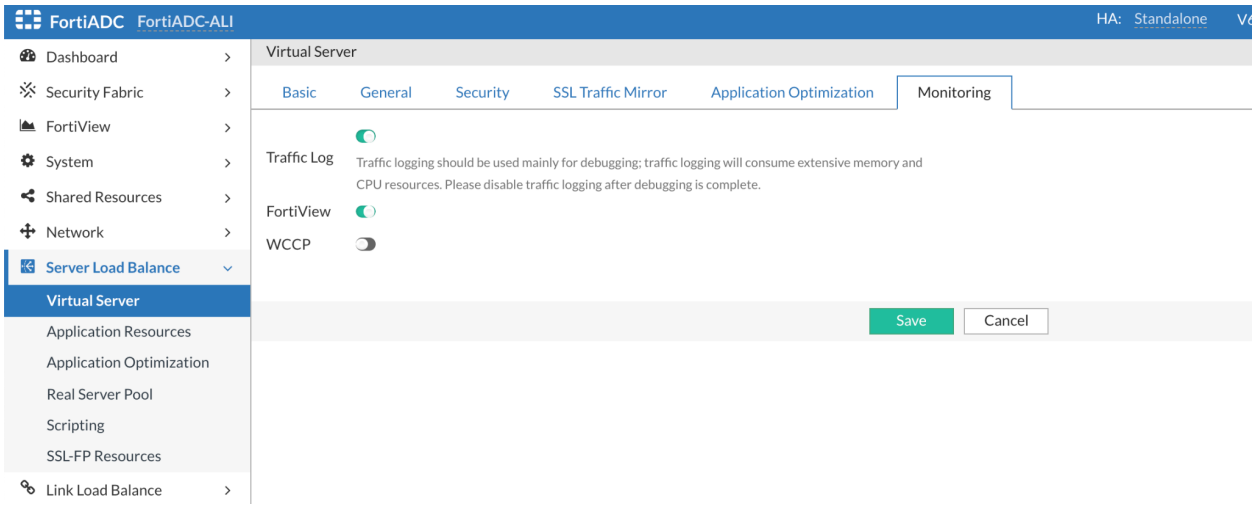


The Server IP is fortiADC interface IP address; you can also choose another secondary IP once that IP has been added to fortiADC ECS. The port is set to 8080. The Profile uses LB_PROF_HTTPS which is default for HTTPS and also chooses the default Client SSL Profile and in the RealServer Pool , selects the correct real server pool.

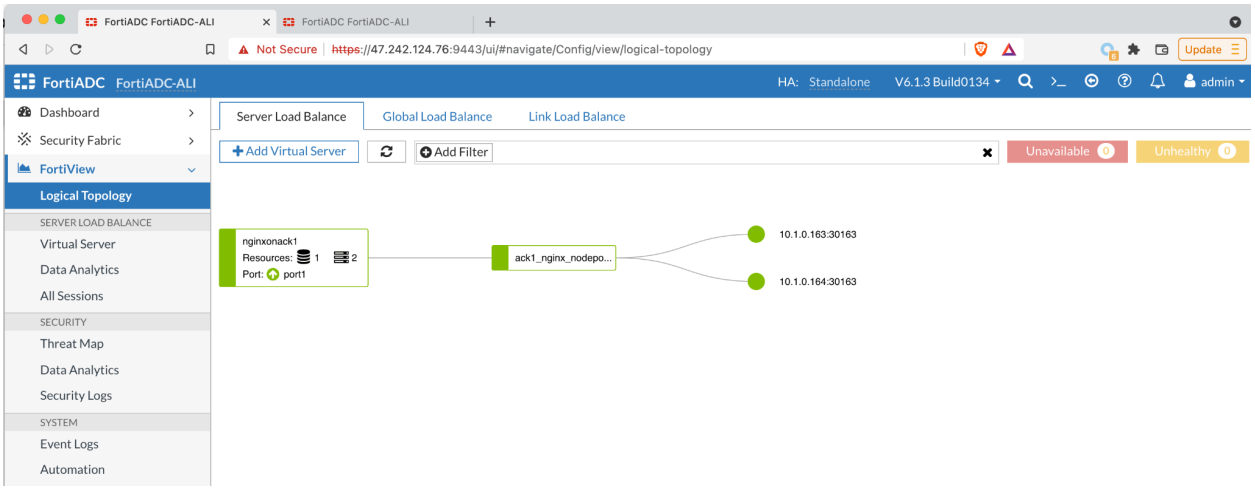
Associate with WAF profile



Enable Monitoring



after that, you will see a ingress rule has been created for nodeport service on ACK1



Verify the result

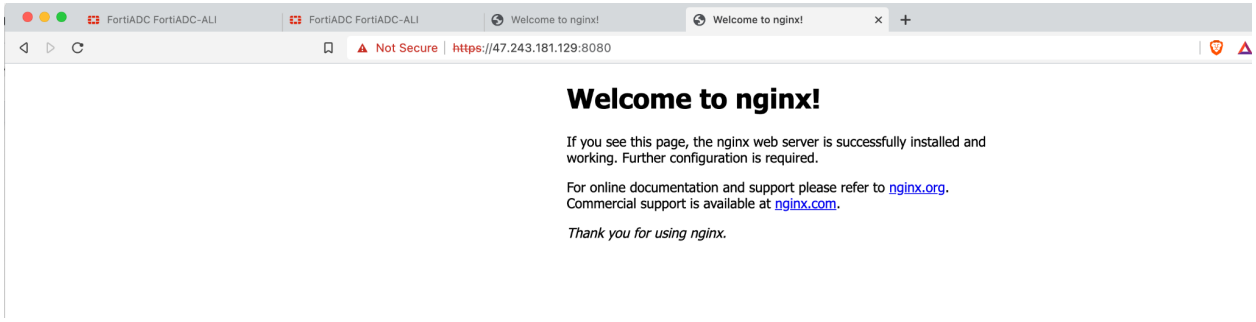
Access via fortiadc internal IP address (from client-vm)

```
root@iZj6cd1vfw01yb6cq9jsqyZ:~# curl -k https://10.0.11.11:8080
<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
<style>
    body {
        width: 35em;
        margin: 0 auto;
        font-family: Tahoma, Verdana, Arial, sans-serif;
    }
</style>
</head>
<body>
<h1>Welcome to nginx!</h1>
<p>If you see this page, the nginx web server is successfully installed and
working. Further configuration is required.</p>

<p>For online documentation and support please refer to
<a href="http://nginx.org/">nginx.org</a>.<br/>
```

```
Commercial support is available at
<a href="http://nginx.com/">nginx.com</a>.</p>

<p><em>Thank you for using nginx.</em></p>
</body>
</html>
root@iZj6cd1vfw01yb6cq9jsqyZ:~#
```

access via fortiadc external IP

If we have a public IP or EIP assigned to fortiadc , we can also use public IP to access nginx instead of using internal IP.

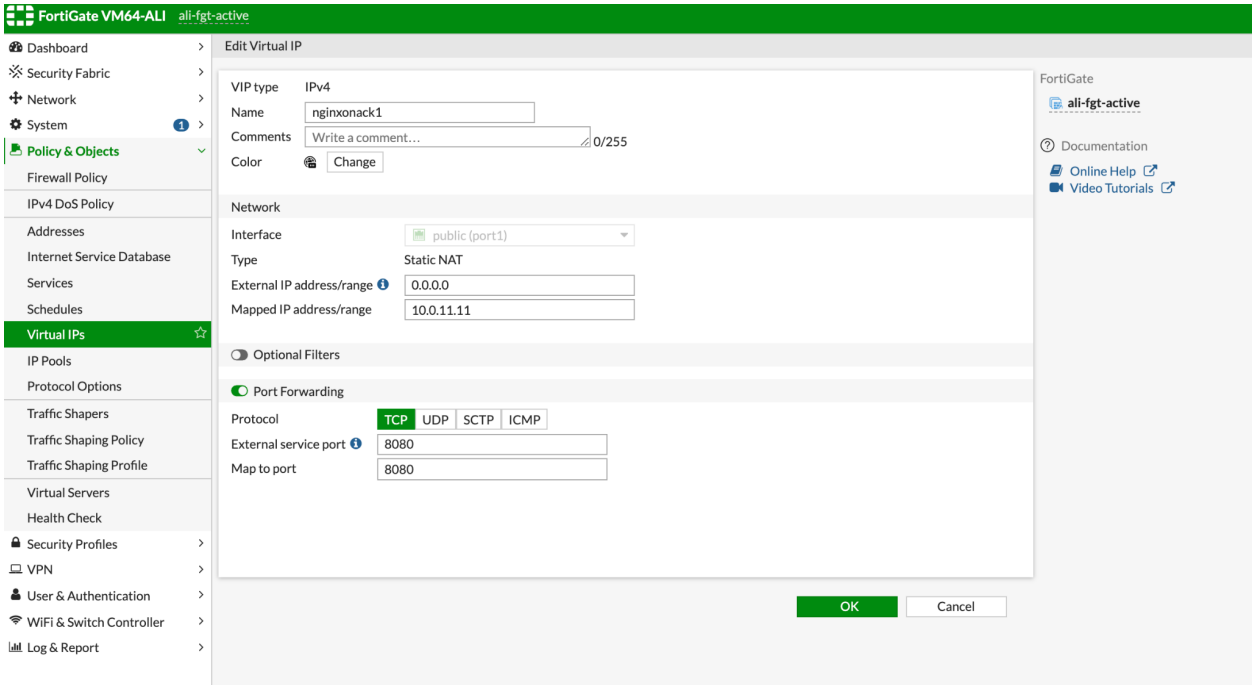


Access via Fortigate public IP if fortiadc has no public IP associated

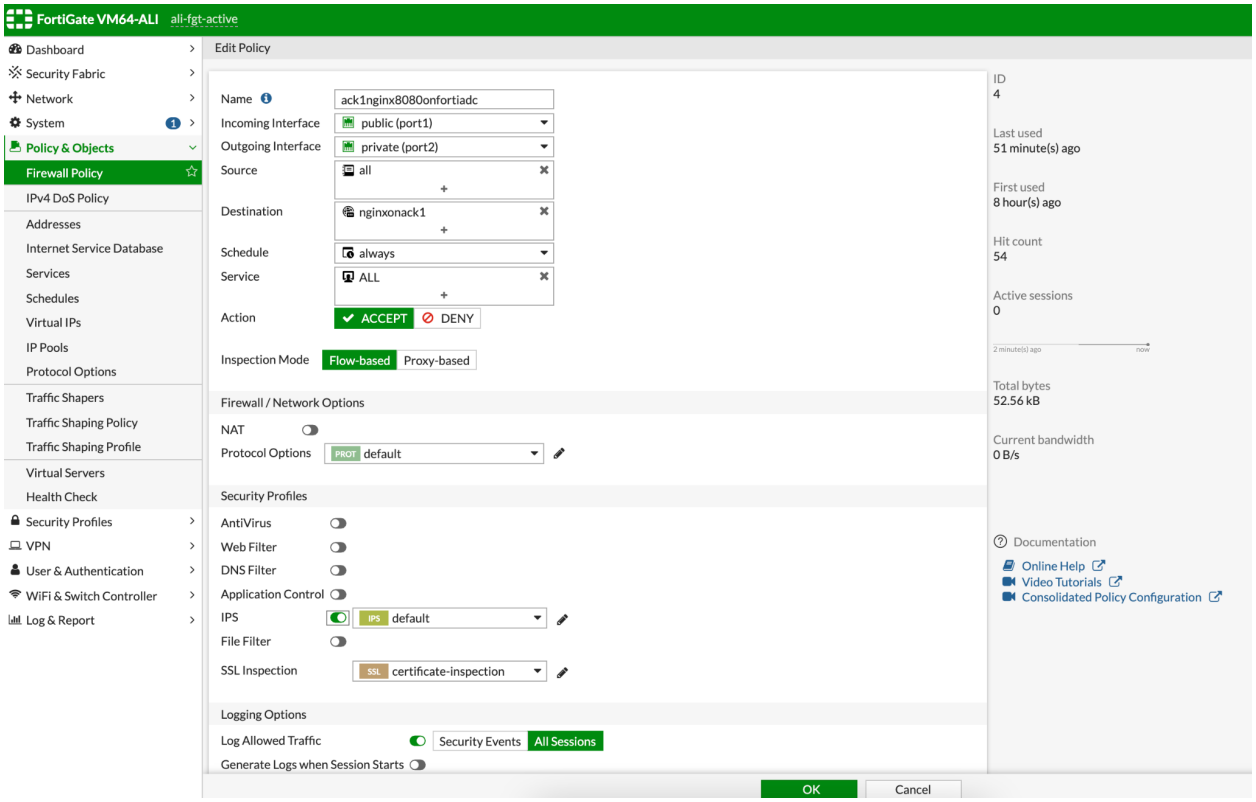
The ingress for nginx nodeport service is configured with an internal IP address on fortiadc, what if we want to give external access to this nodeport service ? We can create a Layer 4 Entry on Fortigate for that with a VIP/DNAT entry. This portion of config has already been included in the terraform script. Below is only required if created manually.

Create a Virtual IP on fortigate

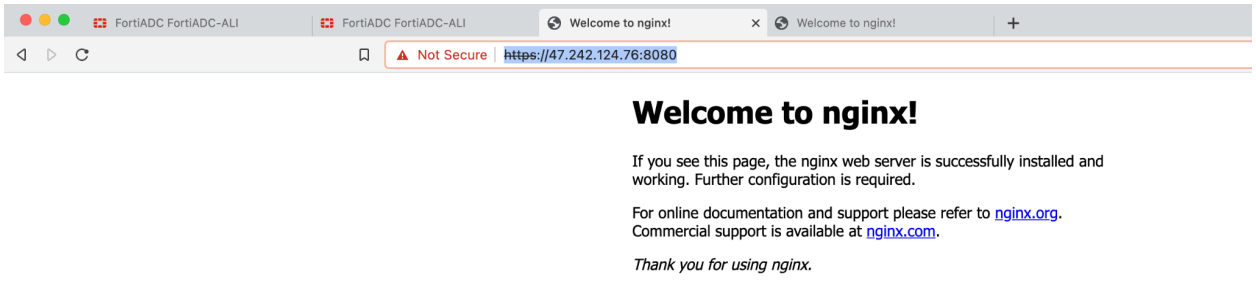
create virtual IP 0.0.0.0:8080 to map to 10.0.11.11:8080



Create a Firewall Policy to enable this traffic and apply security policy such as IPS policy etc.,

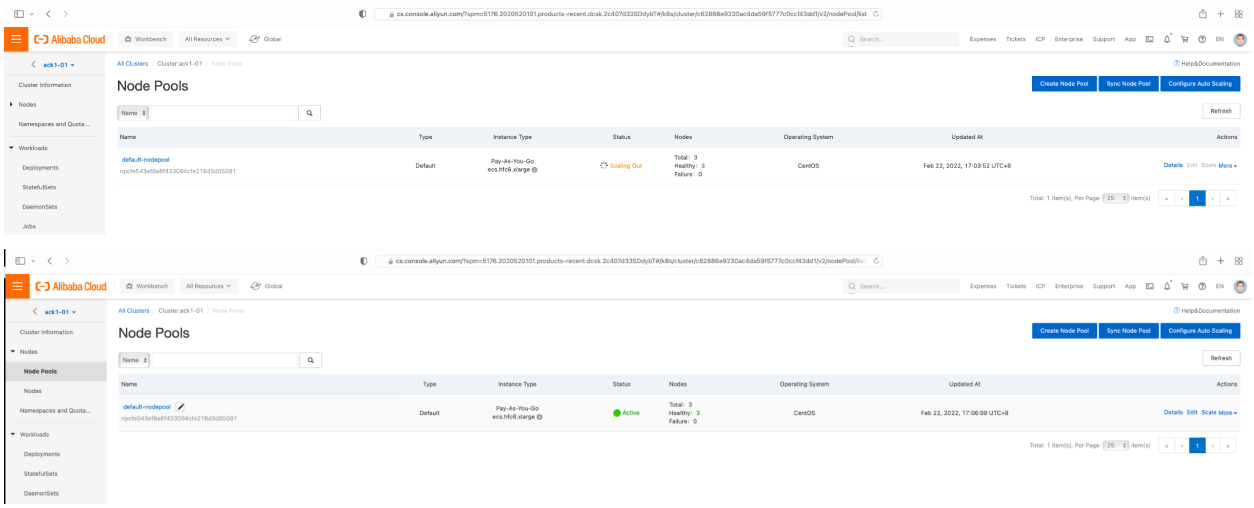


Verify the result

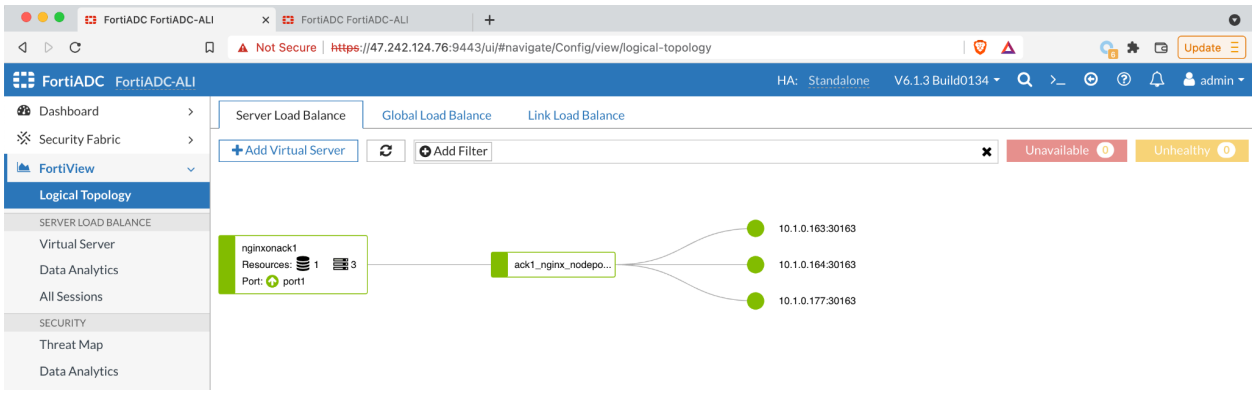


scale out the node on ACK1

let’s scale out the ACK worknode from 2 to 3.



FortiADC will automatically update the real server

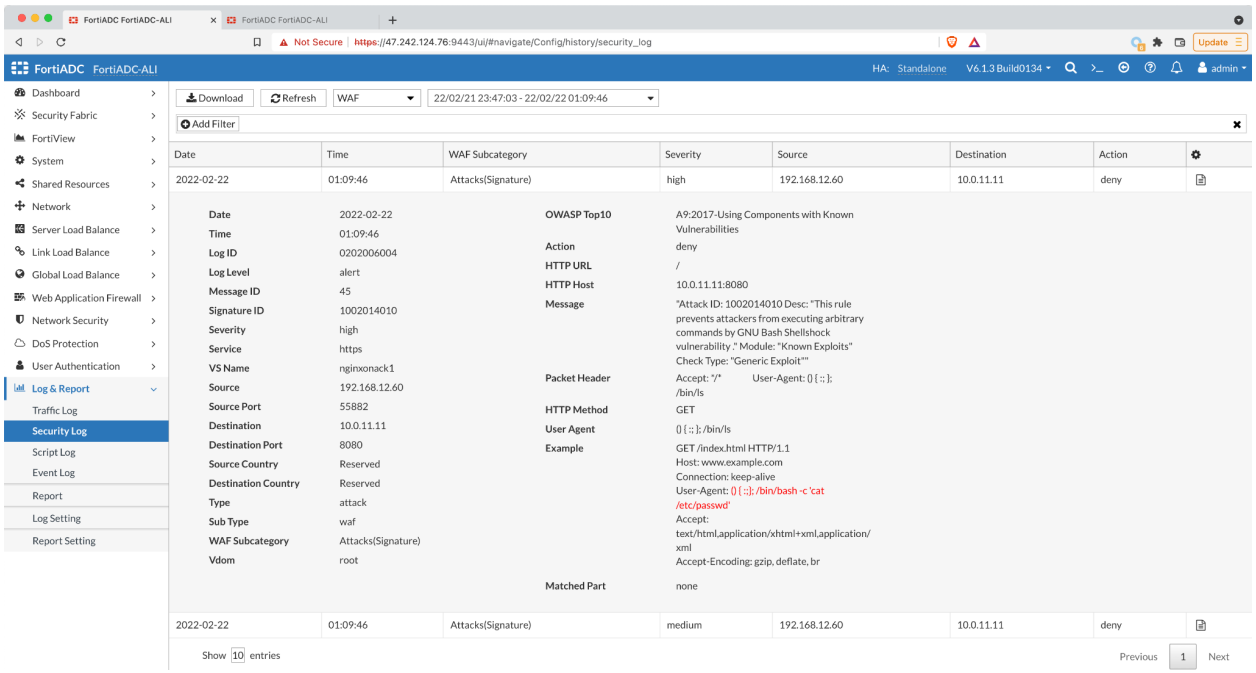


simulate the attack to nodeport service via ingress entry

```
root@iZj6cd1vfw01yb6cq9jsqyZ:~# curl -k -H "User-Agent: () { :; }; /bin/ls"
https://10.0.11.11:8080
<html><body><h1>
403 Forbidden</h1>
Request forbidden by administrative rules.
</body></html>
```

```
root@iZj6cd1vfw01yb6cq9jsqyZ:~#
```

you can see the attack to the ingress has been administratively forbidden . You can also check the fortiADC security log for that detail (after enabling security log in log setting). FortiADC has captured it’s ShellShock attack and action accordingly .



Config FortiGate K8s Connector for refined control

Config K8S Connector

To refine the control of K8S objects, for example, down to individual pods based on Label or based on service name. We can configure the K8S Connector on Fortigate to communicate with ACK to retrieve k8s objects dynamically.

get the cluster IP and SA secret from ACK1 and ACK2

```
root@iZj6cg0w474i2p8oykt7kqZ:~# kubectl cluster-info
--kubeconfig=/root/.kube/config_ack1

Kubernetes control plane is running at https://10.1.0.160:6443
metrics-server is running at
https://10.1.0.160:6443/api/v1/namespaces/kube-system/services/heapster/pro
xy
KubeDNS is running at
https://10.1.0.160:6443/api/v1/namespaces/kube-system/services/kube-dns:dns
/proxy

To further debug and diagnose cluster problems, use 'kubectl cluster-info
dump'.
root@iZj6cg0w474i2p8oykt7kqZ:~# kubectl1 describe secret ftntconnector -n
kube-system --kubeconfig=/root/.kube/config_ack1
Name:         ftntconnector-token-w7fbr
Namespace:    kube-system
Labels:       <none>
Annotations:  kubernetes.io/service-account.name: ftntconnector
              kubernetes.io/service-account.uid:
ca43c3cf-3789-43b4-84bc-b23830a03b57

Type:  kubernetes.io/service-account-token

Data
====
ca.crt:     1180 bytes
namespace:  11 bytes
token:
eyJhbGciOiJSUzI1NiIsImtpZCI6IldTLUVKbkNoakNhLUQ3dExSY2tKeWlLR09VWVJZNlVYd1B
```

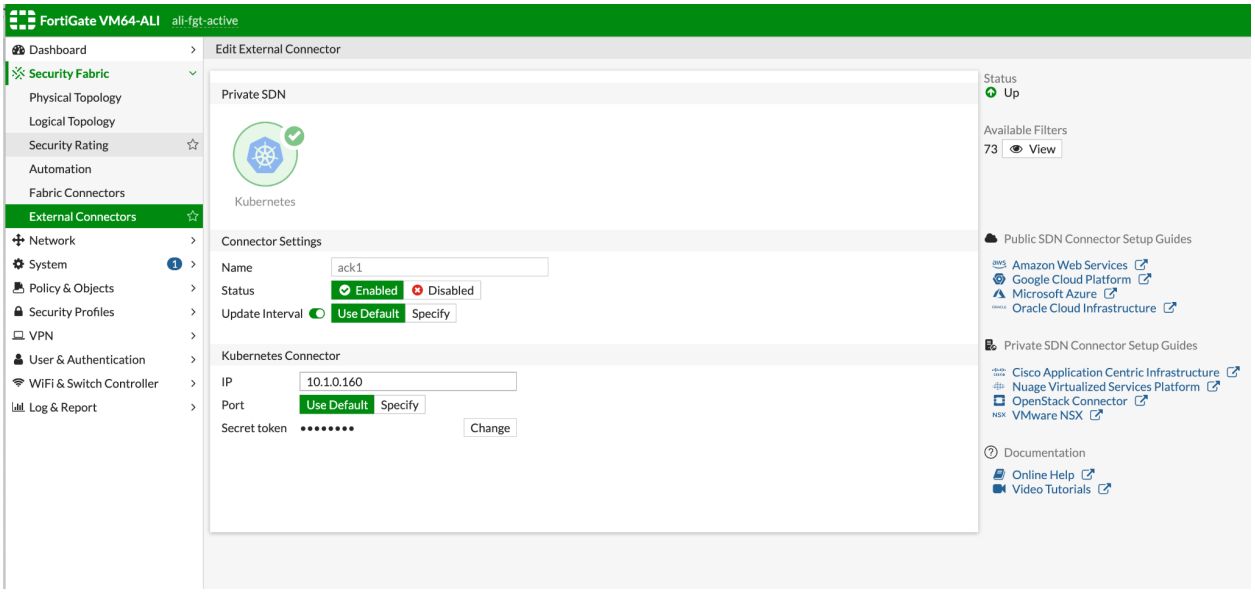
```
HU0N3elFJc28ifQ.eyJpc3MiOiJrdWJlcm5ldGVzL3NlcnZpY2VhY2NvdW50Iiwia3ViZXJuZXR
lcy5pby9zZXJ2aWNlYWNjb3VudC9uYW1lc3BhY2UiOiJrdWJlLXN5c3RlbSIsImt1YmVybmV0ZX
MuaW8vc2VydmljZWFjY291bnQvc2VjcmV0Lm5hbWUiOiJmdG50Y29ubmVjdG9yLXRva2VuLXc3Z
mJyIiwia3ViZXJuZXRlcy5pby9zZXJ2aWNlYWNjb3VudC9zZXJ2aWNlLWFjY291bnQubmFtZSI6
ImZ0bnRjb25uZWN0b3IiLCJrdWJlcm5ldGVzLmlvL3NlcnZpY2VhY2NvdW50L3NlcnZpY2UtYWN
jb3VudC51aWQiOiJjYTQzYzNjZi0zNzg5LTQzYjQtODRiYy1iMjM4MzBhMDNiNTciLCJzdWIiOi
JzeXN0ZW06c2VydmljZWFjY291bnQ6a3ViZS1zeXN0ZW06ZnRudGNvbm5lY3RvciJ9.lLQhahSo
OMTkEqQFX77bep7yxV3hoxEdt6R8Rj3dhdJLSAO5k7OWh9s6GsJ6KU6j7OowYdC2XVJUbJi8v2p
dV8Ok5rS2LHlk4iSNLDVHjQhNdnz-us0G3JAlO9zAseqyG_XBYfhNtebK5LFJOK1zrwL7r5SaIJ
3fRsElgXYFqZCdio3f-6_-QirL_MwQ5Uv6XSj8joc803TGp472deixwzwnMWMHpTa3U1IJtNUqD
Y0xnFYqE56YkQMM7sqaUR85SbHSgFf8kYgU21iFQQltpsMlBLow7IvViZ_xEf1IvSGm8uFrRe0l
43OmJ4vT1VvGVvCXsFAb-o5ew5CIOCtufA

root@iZj6cg0w474i2p8oykt7kqZ:~# kubectl cluster-info
--kubeconfig=/root/.kube/config_ack2
Kubernetes control plane is running at https://10.0.0.247:6443
metrics-server is running at
https://10.0.0.247:6443/api/v1/namespaces/kube-system/services/heapster/pro
xy
KubeDNS is running at
https://10.0.0.247:6443/api/v1/namespaces/kube-system/services/kube-dns:dns
/proxy

To further debug and diagnose cluster problems, use 'kubectl cluster-info
dump'.
root@iZj6cg0w474i2p8oykt7kqZ:~# kubectl1 describe secret ftntconnector -n
kube-system --kubeconfig=/root/.kube/config_ack2
Name:         ftntconnector-token-gzwz4
Namespace:    kube-system
Labels:       <none>
Annotations:  kubernetes.io/service-account.name: ftntconnector
              kubernetes.io/service-account.uid:
20451303-6bf1-450d-a5d6-00a8455e03f6

Type:  kubernetes.io/service-account-token

Data
====
ca.crt:     1180 bytes
namespace:  11 bytes
token:
eyJhbGciOiJSUzI1NiIsImtpZCI6IlBOQ3JuWlhvRU9NVXloVVpzQm15OVEyZjZJSDFwTk1YZXN
ueHJZS2ZRSlEifQ.eyJpc3MiOiJrdWJlcm5ldGVzL3NlcnZpY2VhY2NvdW50Iiwia3ViZXJuZXR
```

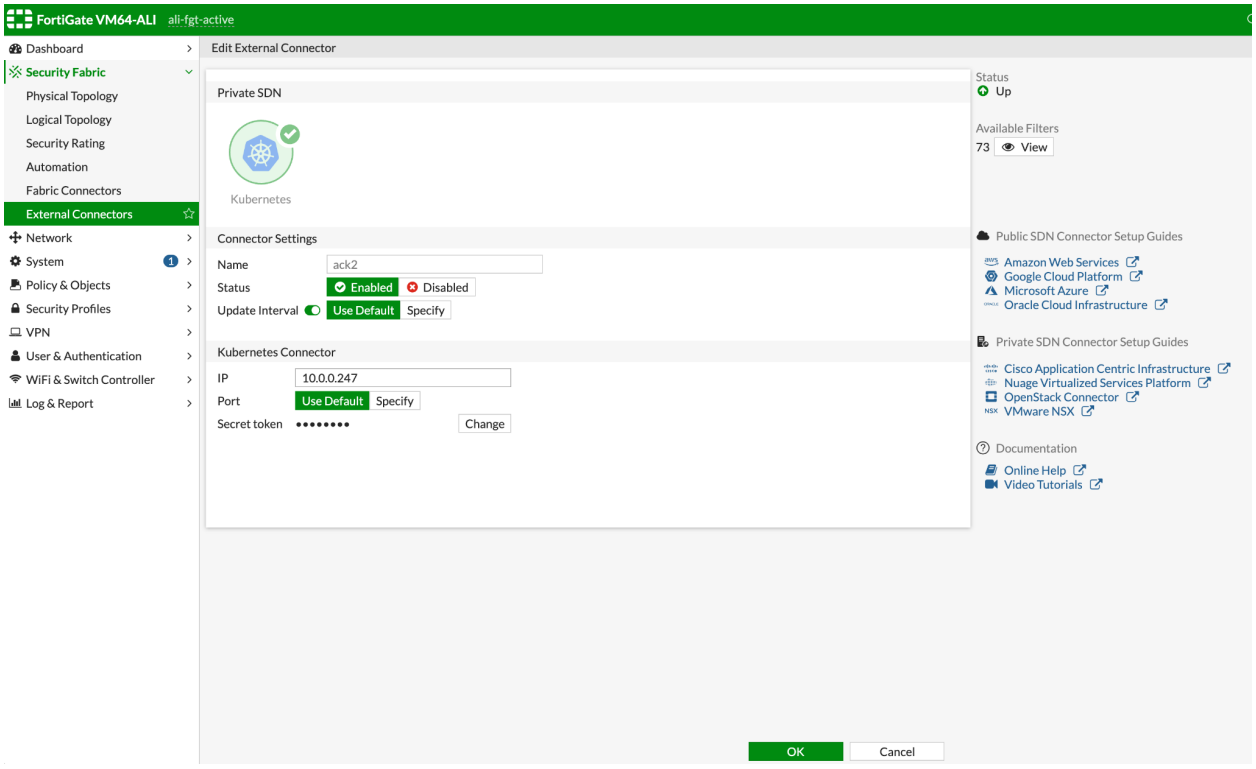
```
lcy5pby9zZXJ2aWNlYWNjb3VudC9uYW1lc3BhY2UiOiJrdWJlLXN5c3RlbSIsImt1YmVybmV0ZX
MuaW8vc2VydmljZWFjY291bnQvc2VjcmV0Lm5hbWUiOiJmdG50Y29ubmVjdG9yLXRva2VuLWd6d
3o0Iiwia3ViZXJuZXRlcy5pby9zZXJ2aWNlYWNjb3VudC9zZXJ2aWNlLWFjY291bnQubmFtZSI6
ImZ0bnRjb25uZWN0b3IiLCJrdWJlcm5ldGVzLmlvL3NlcnZpY2VhY2NvdW50L3NlcnZpY2UtYWN
jb3VudC51aWQiOiIyMDQ1MTMwMy02YmYxLTQ1MGQtYTVkNi0wMGE4NDU1ZTAzZjYiLCJzdWIiOi
JzeXN0ZW06c2VydmljZWFjY291bnQ6a3ViZS1zeXN0ZW06ZnRudGNvbm5lY3RvciJ9.YE48oJo-
cvIFzeUj7kSjzGJprY7aw3o6RLv9npizXGel7SfVb_nJaqc3DBVCge4CdMOfAhd0FA81RkiQi2W
LLFk6ya4ClsjoWYP6wviOpxAWBDMK7qyfrKOJJj9ipyjtRIMWDz4b8MiYuzJxsNyRfXglU1pNDf
LzSrJPKbeSzAN8pUci3KGgiyCZM8G06EEMFOPWyKSUTk0HQDLibMrFi60PJj1D9LW9-CQWfh-FC
yp9zu2husdHBl7itHxhsI17KwVlu_7fu5zkBOn3mnw0XUzT8EY8h2pnlXzsan6sCahIcNkyNCSk
P3Ny7fkXI9MCTsac7M_Y9JxBoSn2m_sW-A
```


root@iZj6cg0w474i2p8oykt7kqZ:~#

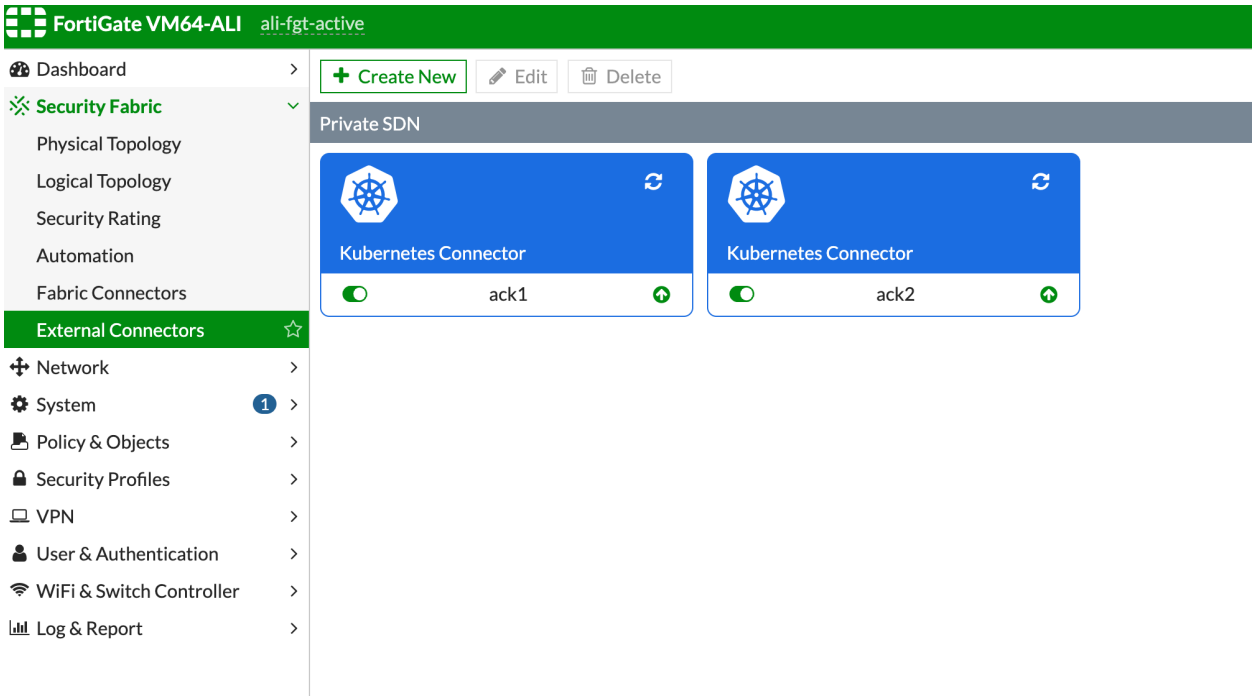
Config k8s connector on fortigate

Fill in ACK1 and ACK2 cluster IP and SA account token.





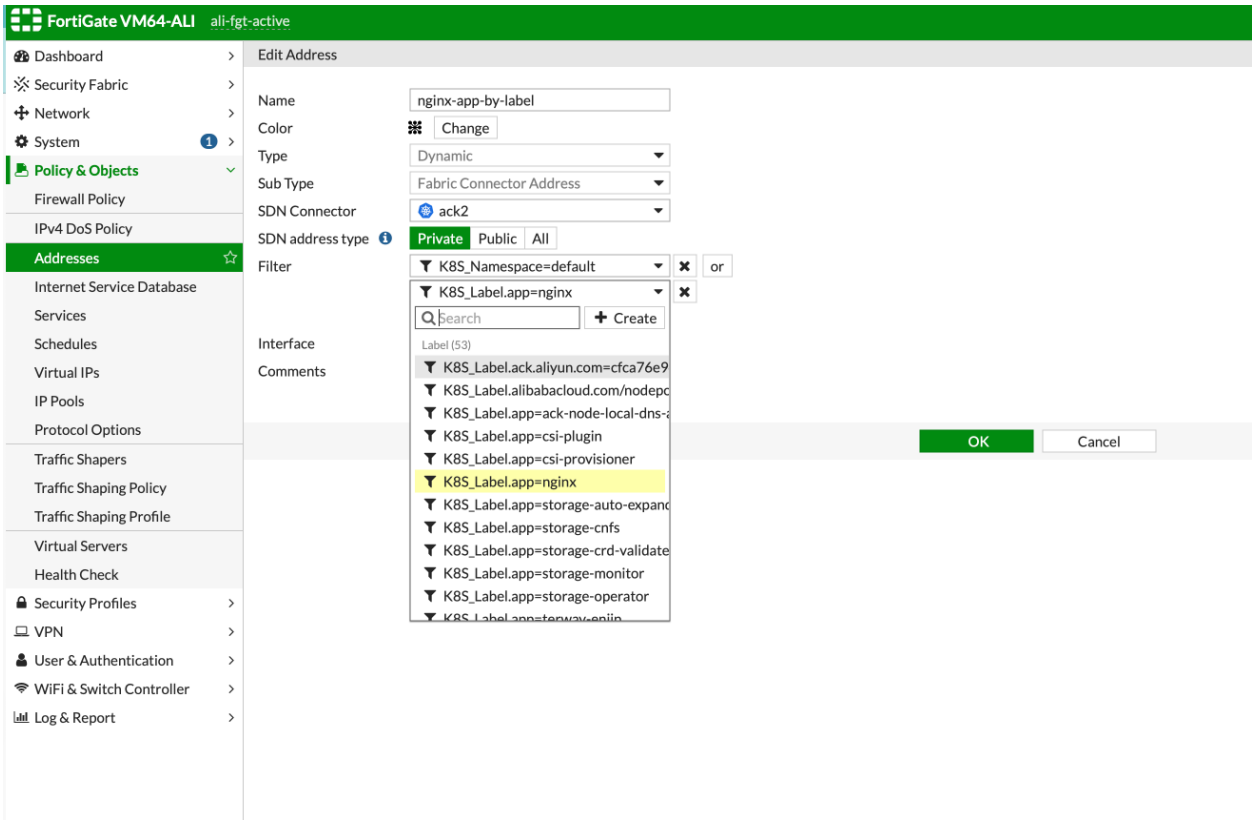
once successful, you shall see



Create Firewall Policy based on dynamic object

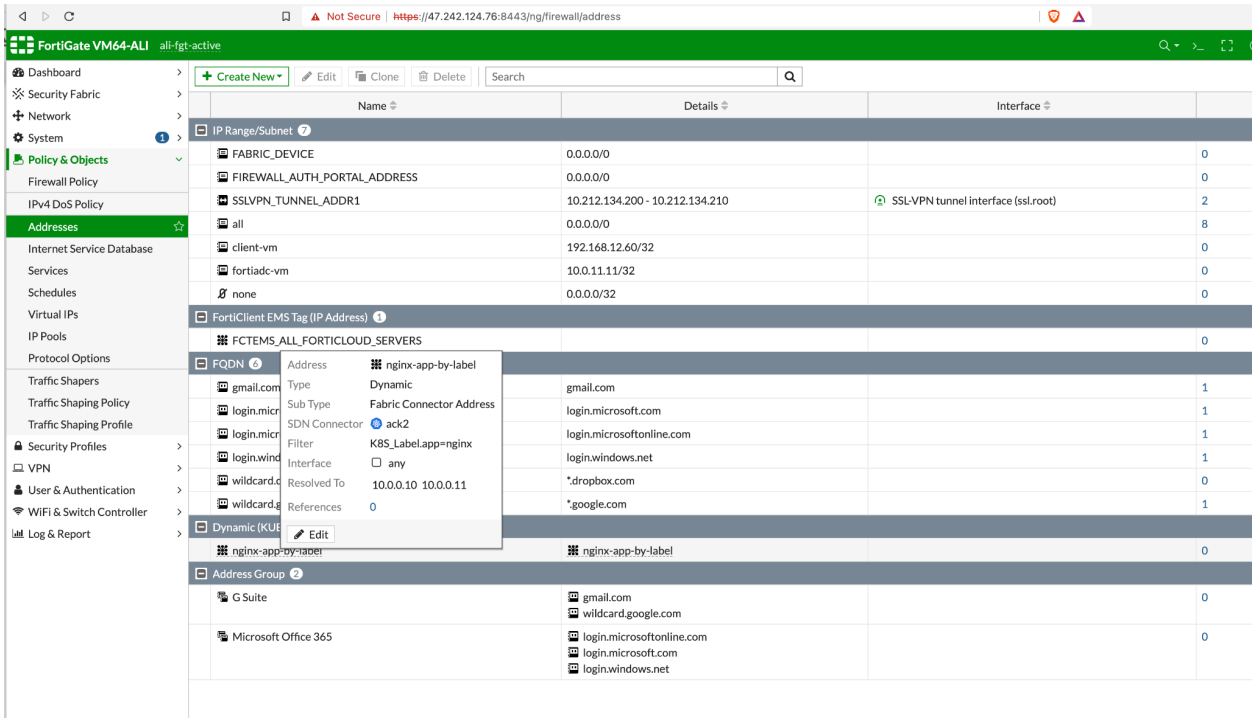
Now we have the k8s connector configured, the fortigate can dynamically pull the k8s object and use it in the firewall policy.

Define the Address object



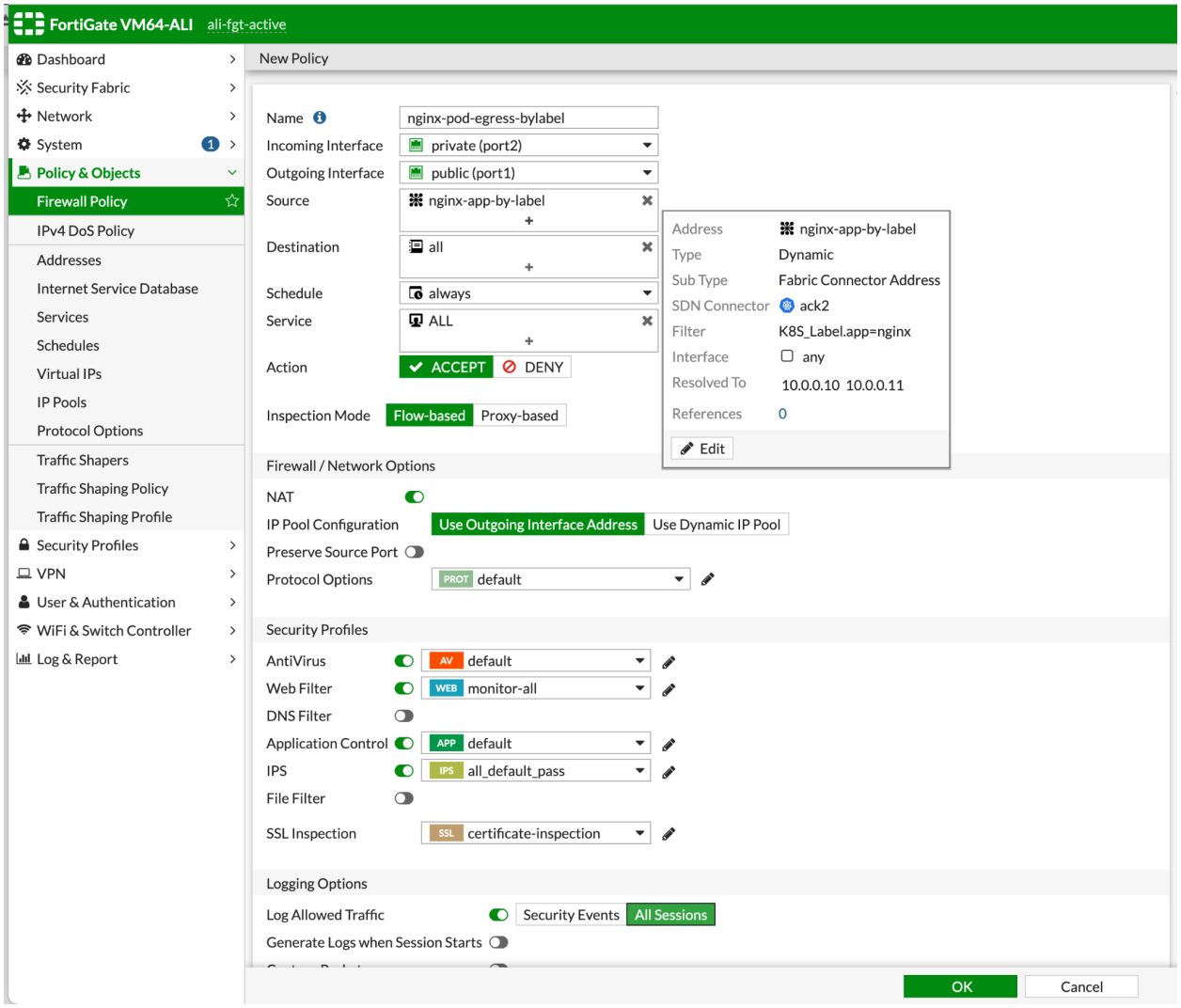
On the Filter Dropbox, we can see K8s Object ,can be filtered by use namespace, Label, Service,Node, PODname etc.,

Once done, the ip address of the dynamic object will be obtained.

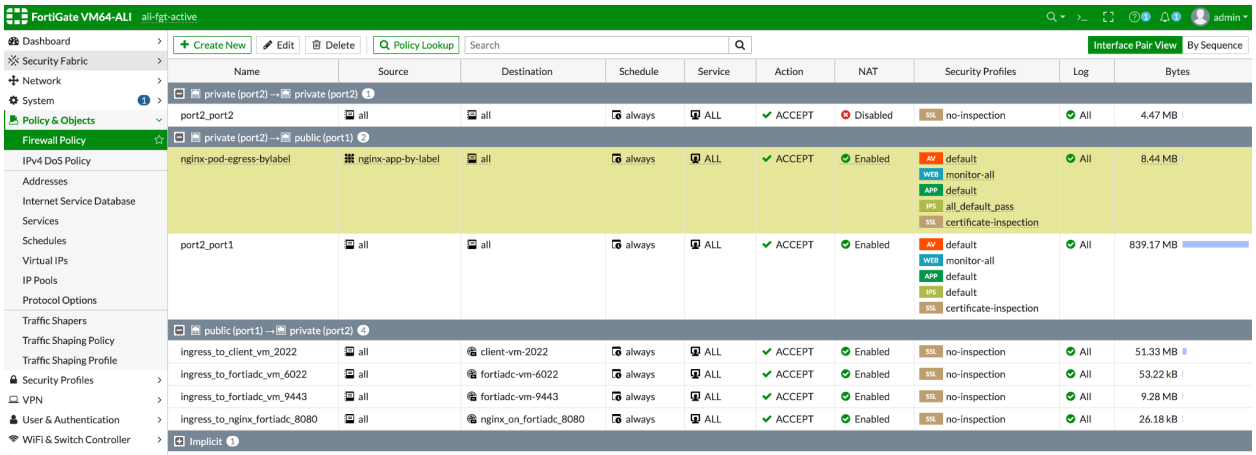


then We can use the address object in firewall policy. Once the object changes on k8s , for example, due to scale out, as long as the filter does not change, the new ip will be dynamically updated on the fortigate side.

create firewall policy



then a more fine grained policy for POD that with label nginx is created.



This gives you the grainer control to the k8s object.