Using Load Balancers to provide Port Based Destination NAT

- Summary
- Details
  - Terraform / OpenTofu Sample

Summary

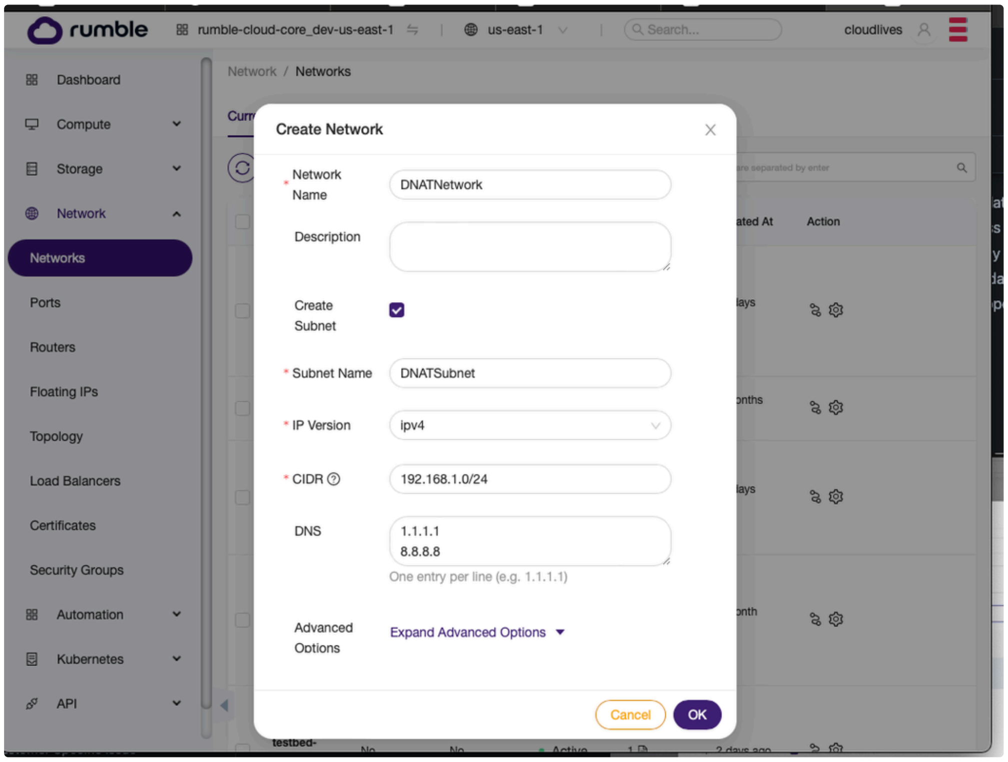
Rumble Cloud support a few different types of address translation. The most common use case is to map a Public IP, referred to as a Floating IP, to an private IP address for a VM on an internal network. However, a floating IP can also be mapped to a Load Balancer. Typically a Load Balancer is used to balance traffic across multiple VM for scaling a service and providing redundancy. However, a Load Balancer can also be used to provide Port Based Destination NAT if configured properly.

To setup up Port Based Destination NAT simply create a load balancer with multiple listeners listening on different ports. Associate those listers with pools that have a single member in them. In this way the port will be specifically dedicated to that server. Note, using this strategy you can provide load balancing and port based destination NAT on the same IP address.

Below, we show step by step how to do this from the Web Interface. However, as this can get complex, we do recommend using Infrastructure as Code via Terraform / OpenTofu for these types of cases. A full terraform template is included.

Details

1. Create a network with subnet

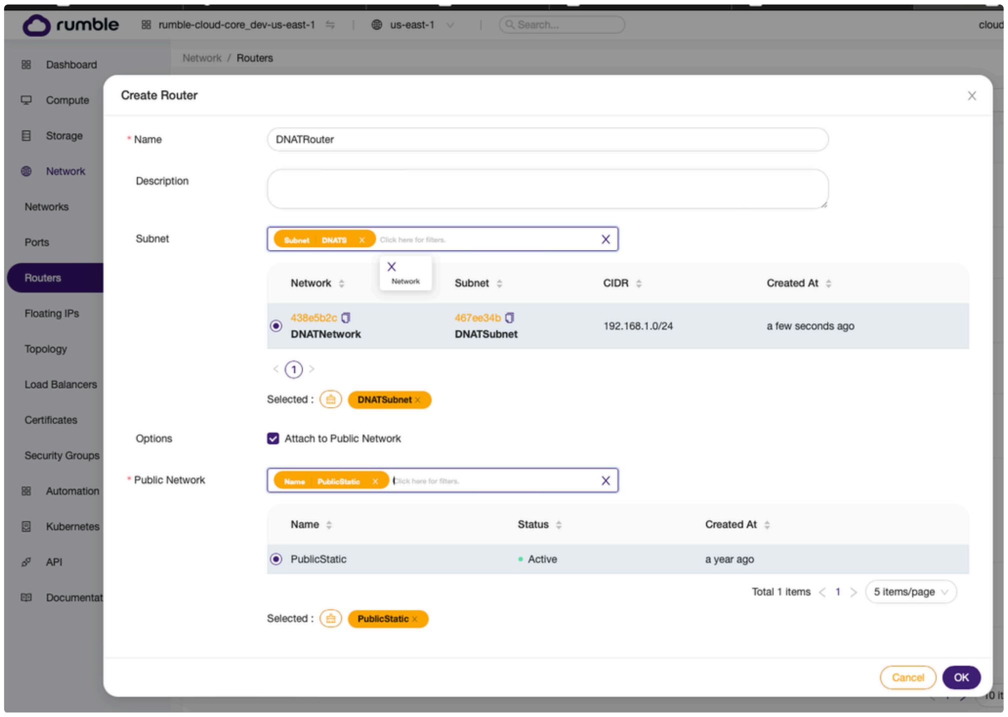


Create Network

* Network Name	DNATNetwork
Description	
Create Subnet	✓
* Subnet Name	DNATSubnet
* IP Version	ipv4
* CIDR	192.168.1.0/24
DNS	1.1.1.1 8.8.8.8
Advanced Options	Expand Advanced Options

One entry per line (e.g. 1.1.1.1)

2. Create a router that uses the network



Create Router

* Name	DNATRouter
Description	
Subnet	Subnet DNATS

	Network	Subnet	CIDR	Created At
◉	438e5b2c DNATNetwork	467ee34b DNATSubnet	192.168.1.0/24	a few seconds ago

Selected : DNATSubnet

Options ☑ Attach to Public Network

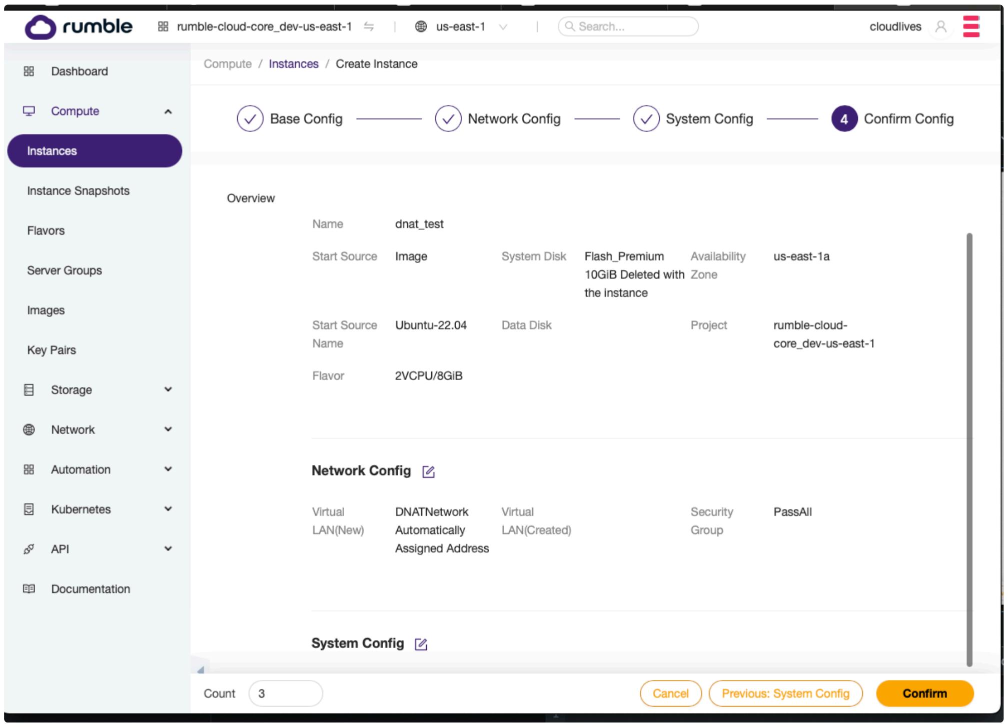
* Public Network Name PublicStatic

	Name	Status	Created At
◉	PublicStatic	Active	a year ago

Total 1 items 1 5 items/page

Selected : PublicStatic

3. Create 2 or more VM's on the network



Overview

Name	dnat_test				
Start Source	Image	System Disk	Flash_Premium 10GiB Deleted with the instance	Availability Zone	us-east-1a
Start Source Name	Ubuntu-22.04	Data Disk		Project	rumble-cloud-core_dev-us-east-1
Flavor	2VCPU/8GiB				

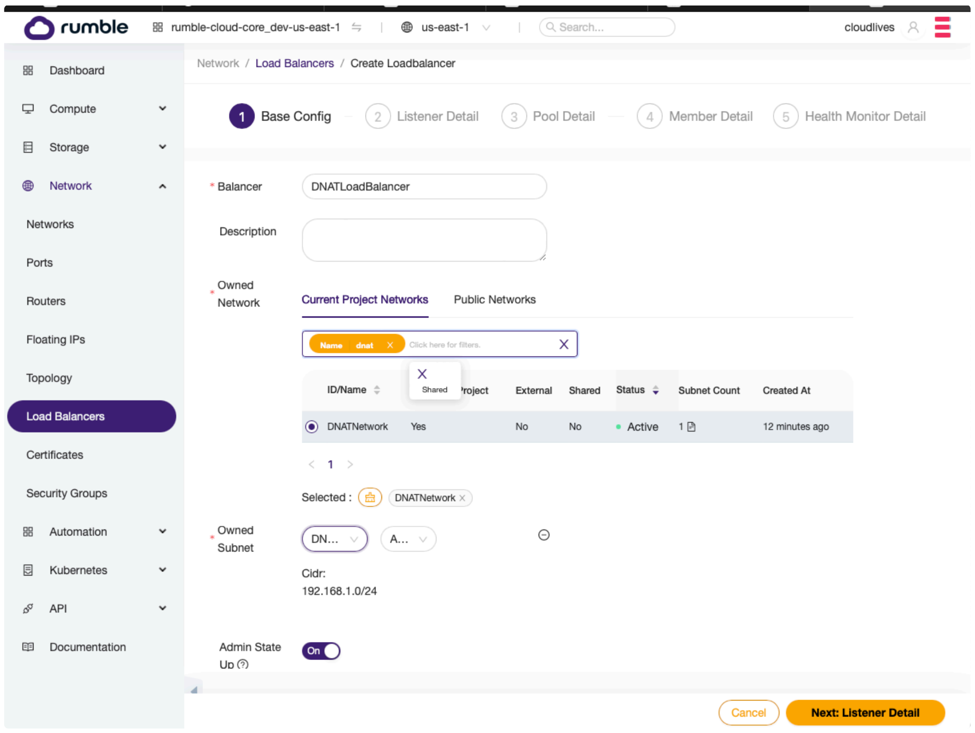
Network Config

Virtual LAN(New)	DNATNetwork Automatically Assigned Address	Virtual LAN(Created)		Security Group	PassAll

System Config

Count 3

4. Create a load balancer that provides destination NAT for SSH, as well as provides HTTP/HTTPS load balancing
   a. Create Loadbalancer


rumble | rumble-cloud-core_dev-us-east-1 | us-east-1 | Search... | cloudlives

Dashboard
Compute
Storage
Network
Networks
Ports
Routers
Floating IPs
Topology
Load Balancers
Certificates
Security Groups
Automation
Kubernetes
API
Documentation

Network / Load Balancers / Create Loadbalancer

1 Base Config — 2 Listener Detail — 3 Pool Detail — 4 Member Detail — 5 Health Monitor Detail

* Balancer: DNATLoadBalancer

Description:

* Owned Network: Current Project Networks | Public Networks

Name dnat × Click here for filters.

ID/Name	Shared	Project	External	Shared	Status	Subnet Count	Created At
◉ DNATNetwork	Yes		No	No	● Active	1	12 minutes ago

< 1 >

Selected : DNATNetwork ×

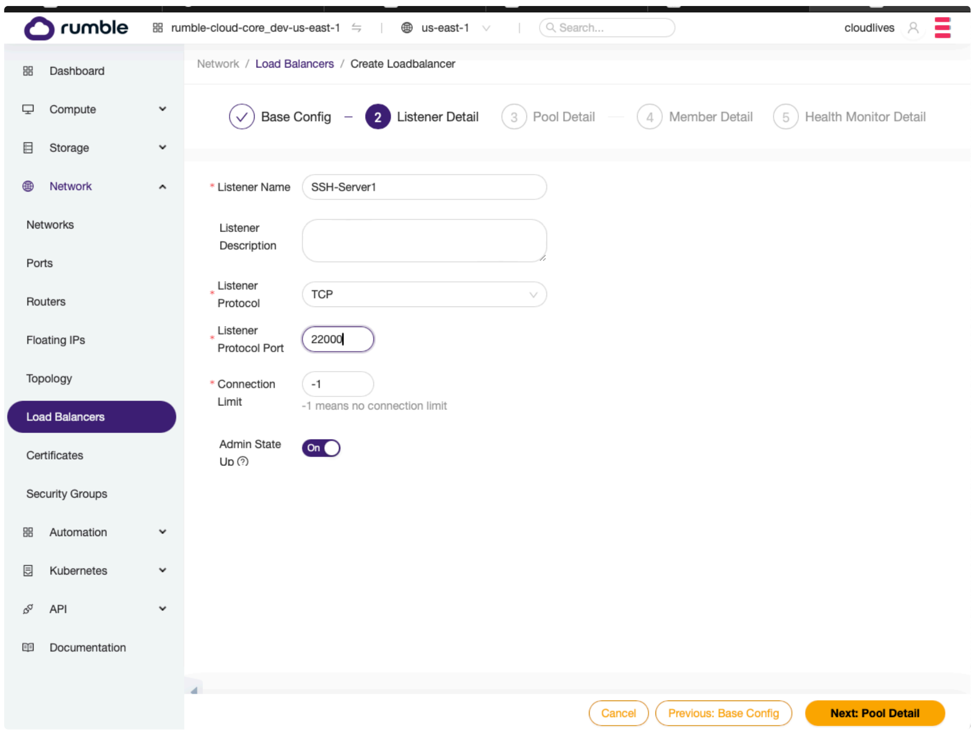
* Owned Subnet: DN... A...

Cidr:
192.168.1.0/24

Admin State Up: On

Cancel | Next: Listener Detail

b. Setup First Listener


rumble | rumble-cloud-core_dev-us-east-1 | us-east-1 | Search... | cloudlives

Dashboard
Compute
Storage
Network
Networks
Ports
Routers
Floating IPs
Topology
Load Balancers
Certificates
Security Groups
Automation
Kubernetes
API
Documentation

Network / Load Balancers / Create Loadbalancer

✓ Base Config — 2 Listener Detail — 3 Pool Detail — 4 Member Detail — 5 Health Monitor Detail

* Listener Name: SSH-Server1

Listener Description:

* Listener Protocol: TCP

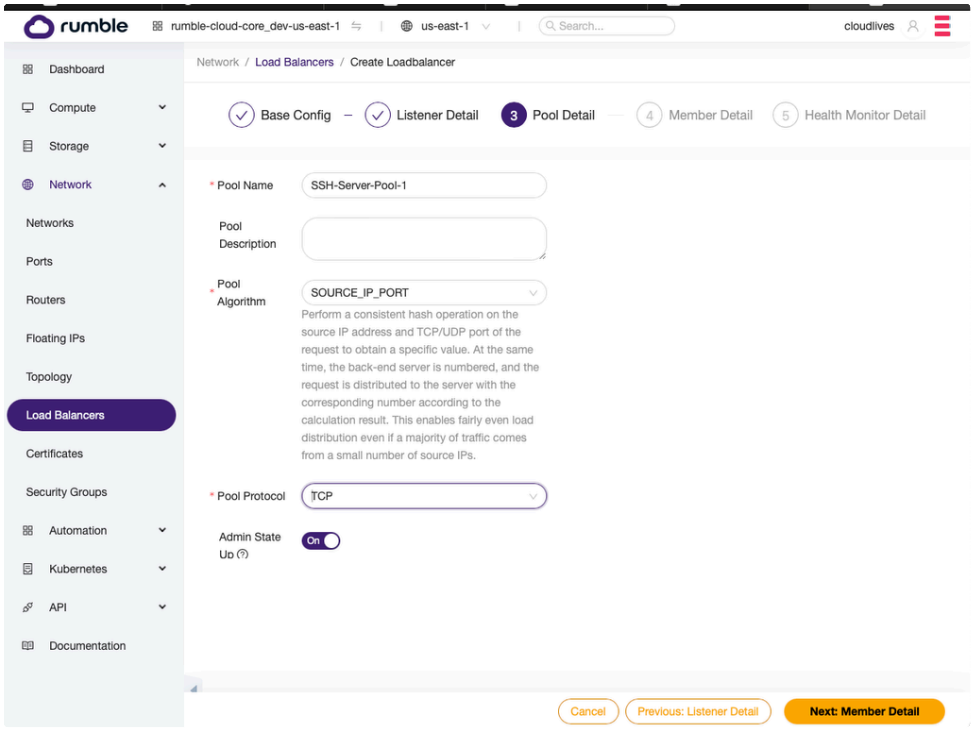
* Listener Protocol Port: 22000

* Connection Limit: -1
-1 means no connection limit

Admin State Up: On

Cancel | Previous: Base Config | Next: Pool Detail

c. Setup First Pool



* Pool Name: SSH-Server-Pool-1

Pool Description:

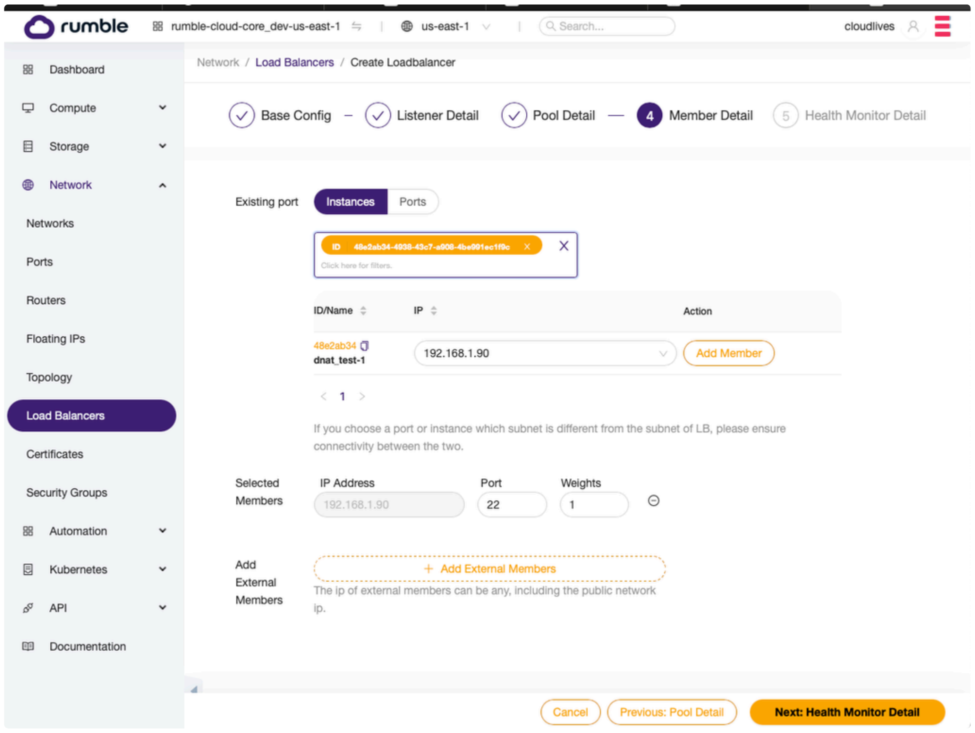
* Pool Algorithm: SOURCE_IP_PORT

Perform a consistent hash operation on the source IP address and TCP/UDP port of the request to obtain a specific value. At the same time, the back-end server is numbered, and the request is distributed to the server with the corresponding number according to the calculation result. This enables fairly even load distribution even if a majority of traffic comes from a small number of source IPs.

* Pool Protocol: TCP

Admin State Up: On

d. Add the first server to the pool



Existing port: Instances

ID 48e2ab34-4938-43c7-a908-4be991ec1f9c

ID/Name	IP	Action
48e2ab34 dnat_test-1	192.168.1.90	Add Member

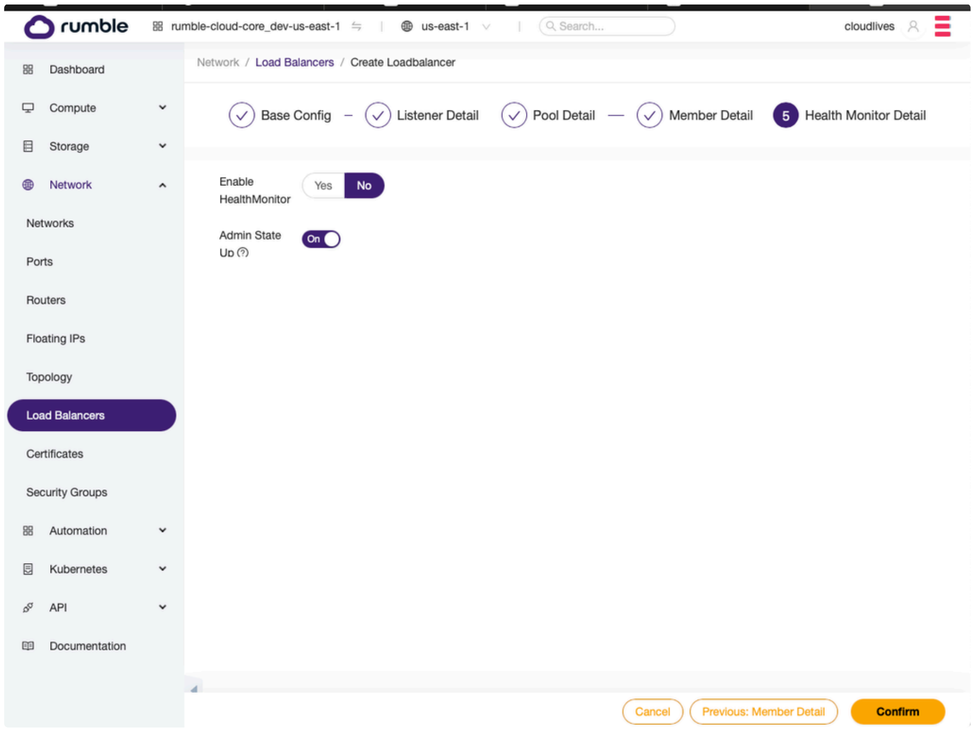
If you choose a port or instance which subnet is different from the subnet of LB, please ensure connectivity between the two.

Selected Members:

IP Address	Port	Weights
192.168.1.90	22	1

Add External Members: The ip of external members can be any, including the public network ip.

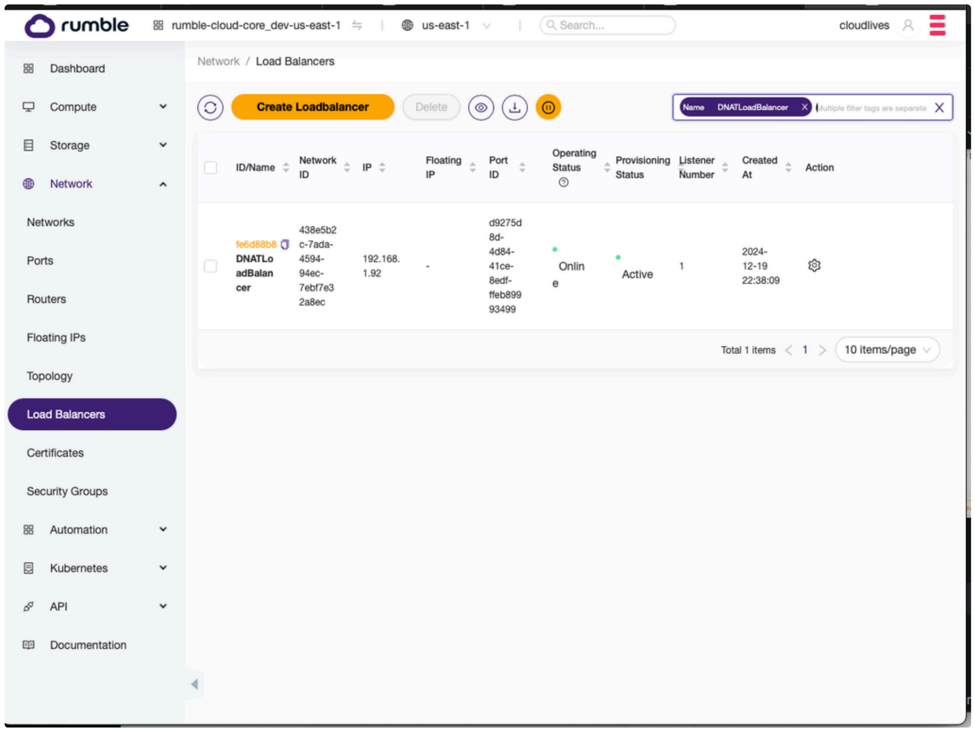
e. You can skip the health monitor for any DNAT scenarios since you have exactly one member in the pool. However, if the “Confirm” button doesn’t work you may have to put valid information in the Health Monitor form first, to allow the Confirm button to work (this will be fixed in the interface soon)



Enable HealthMonitor: Yes [✓ No]

Admin State Up: On

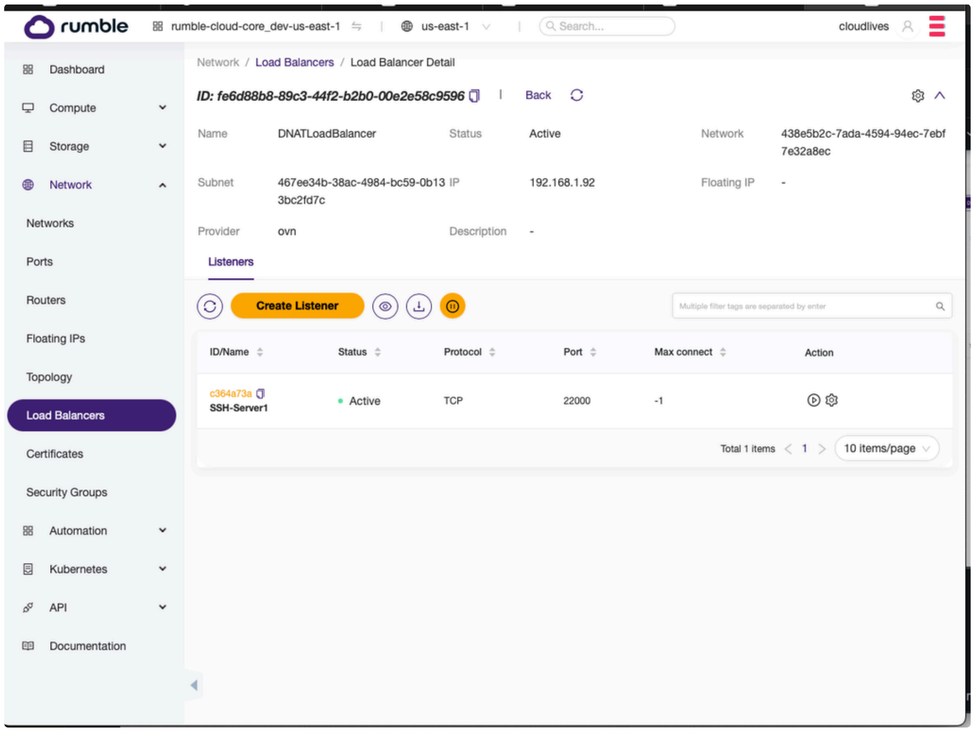
5. The Load Balancer should have been created with a single listener for the first server. Now, find the load balancer and click on it to open its details, and allow you to add more listeners.



ID/Name	Network ID	IP	Floating IP	Port ID	Operating Status	Provisioning Status	Listener Number	Created At	Action
fe6d88b8 **DNATLoadBalancer**	438e5b2c-7ada-4594-94ec-7ebf7e32a8ec	192.168.1.92	-	d9275d8d-4d84-41ce-8edf-ffeb89993499	● Online	● Active	1	2024-12-19 22:38:09	⚙

Total 1 items < 1 > 10 items/page

a. Select Create Listener

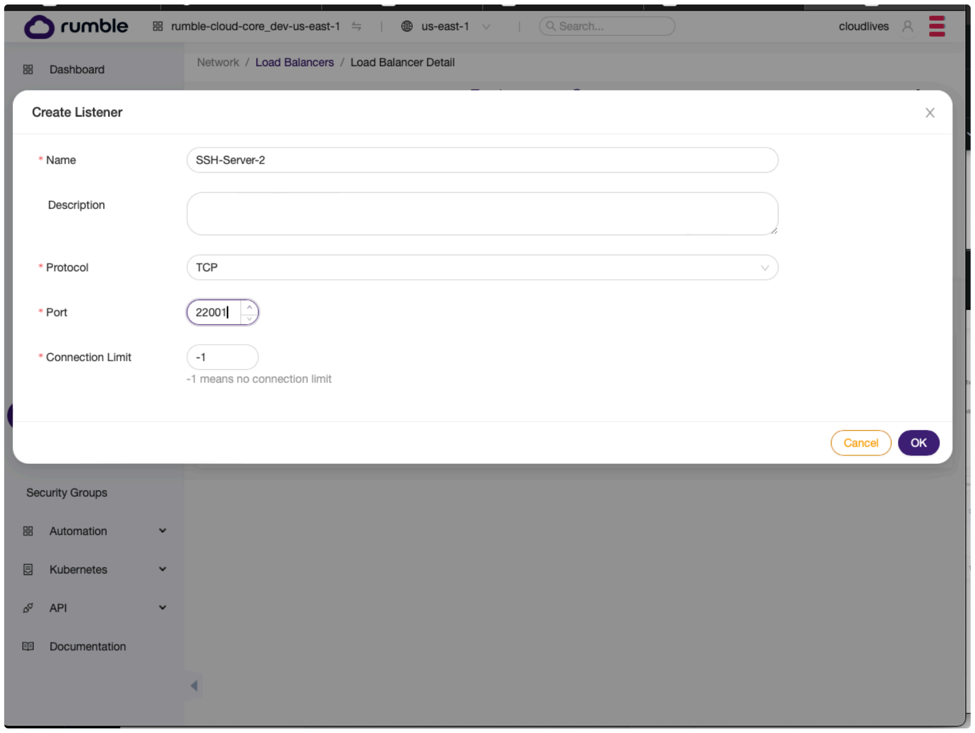


ID: fe6d88b8-89c3-44f2-b2b0-00e2e58c9596

Name	DNATLoadBalancer	Status	Active	Network	438e5b2c-7ada-4594-94ec-7ebf7e32a8ec
Subnet	467ee34b-38ac-4984-bc59-0b133bc2fd7c	IP	192.168.1.92	Floating IP	-
Provider	ovn	Description	-		

ID/Name	Status	Protocol	Port	Max connect	Action
c364a73a SSH-Server1	Active	TCP	22000	-1	

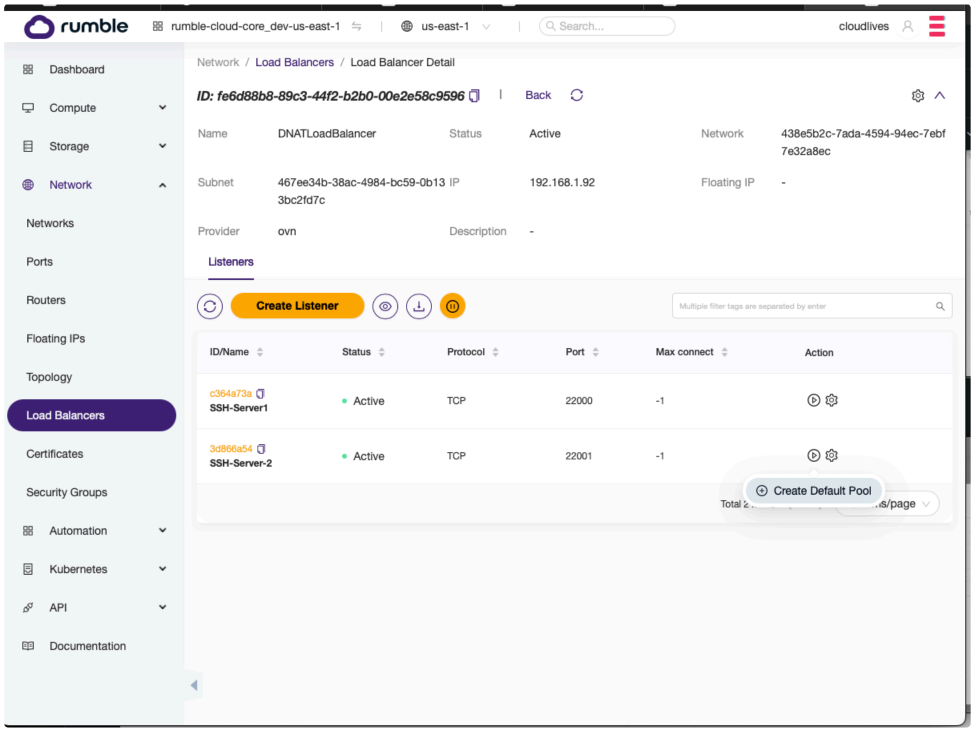
b. Fill in the next port for the new listener



* Name	SSH-Server-2
Description	
* Protocol	TCP
* Port	22001
* Connection Limit	-1

-1 means no connection limit

c. Once, the listener is created, click on the first icon next to the listener and select “Create Default Pool”



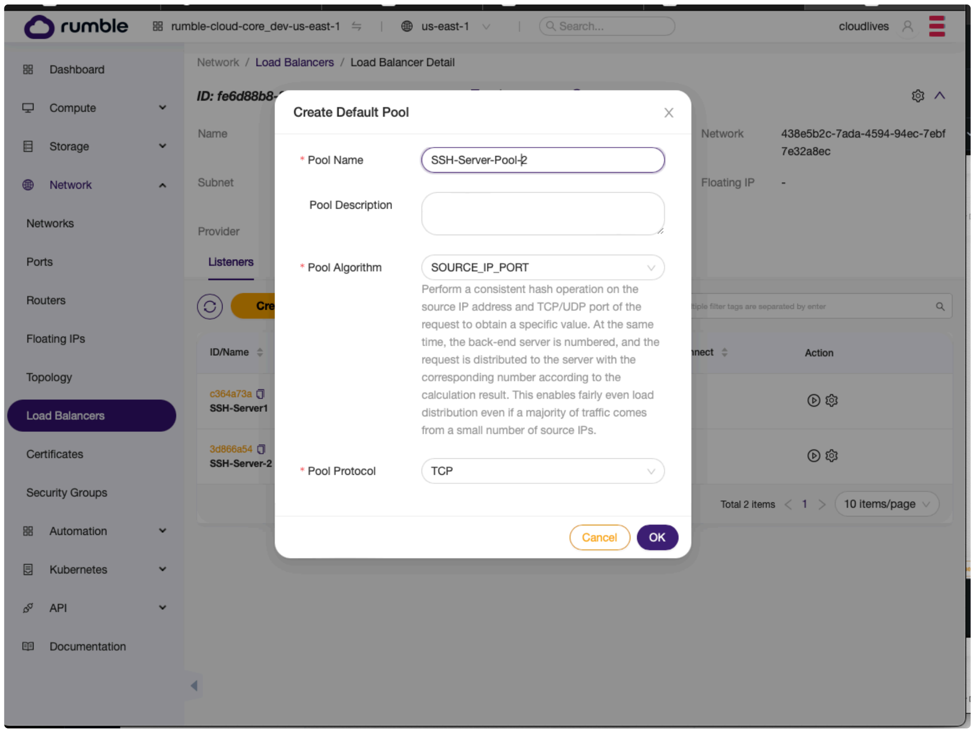
Network / Load Balancers / Load Balancer Detail

ID: fe6d88b8-89c3-44f2-b2b0-00e2e58c9596

Name	DNATLoadBalancer	Status	Active	Network	438e5b2c-7ada-4594-94ec-7ebf7e32a8ec
Subnet	467ee34b-38ac-4984-bc59-0b133bc2fd7c	IP	192.168.1.92	Floating IP	-
Provider	ovn	Description	-		

ID/Name	Status	Protocol	Port	Max connect	Action
c364a73a SSH-Server1	Active	TCP	22000	-1	
3d866a54 SSH-Server-2	Active	TCP	22001	-1	

d. Populate the information for the pool



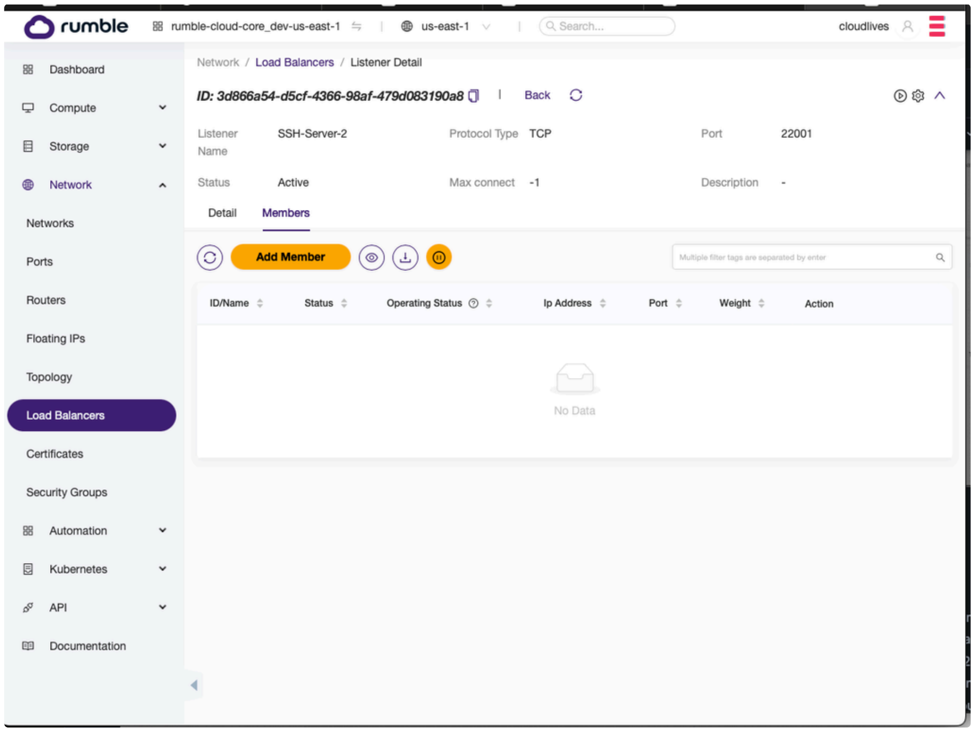
Create Default Pool

- * Pool Name: SSH-Server-Pool-2
- Pool Description:
- * Pool Algorithm: SOURCE_IP_PORT

  Perform a consistent hash operation on the source IP address and TCP/UDP port of the request to obtain a specific value. At the same time, the back-end server is numbered, and the request is distributed to the server with the corresponding number according to the calculation result. This enables fairly even load distribution even if a majority of traffic comes from a small number of source IPs.
- * Pool Protocol: TCP

Total 2 items 1 10 items/page

e. Select the listener and go to the “Members” Tab;



Network / Load Balancers / Listener Detail

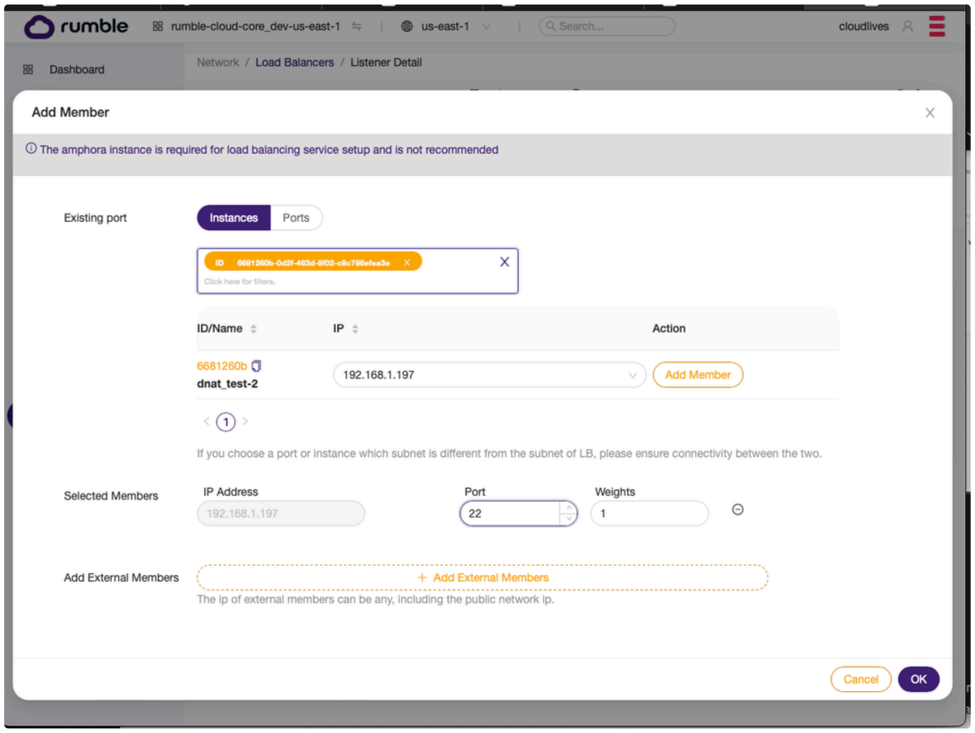
ID: 3d866a54-d5cf-4366-98af-479d083190a8

Listener Name	SSH-Server-2	Protocol Type	TCP	Port	22001
Status	Active	Max connect	-1	Description	-

ID/Name	Status	Operating Status	Ip Address	Port	Weight	Action

No Data

f. Hit Add members and populate the second SSH host



Add Member

The amphora instance is required for load balancing service setup and is not recommended

Existing port: Instances | Ports

ID 6681260b-0d2f-463d-8f02-c8c786efea3e

ID/Name	IP	Action
6681260b dnat_test-2	192.168.1.197	Add Member

If you choose a port or instance which subnet is different from the subnet of LB, please ensure connectivity between the two.

Selected Members

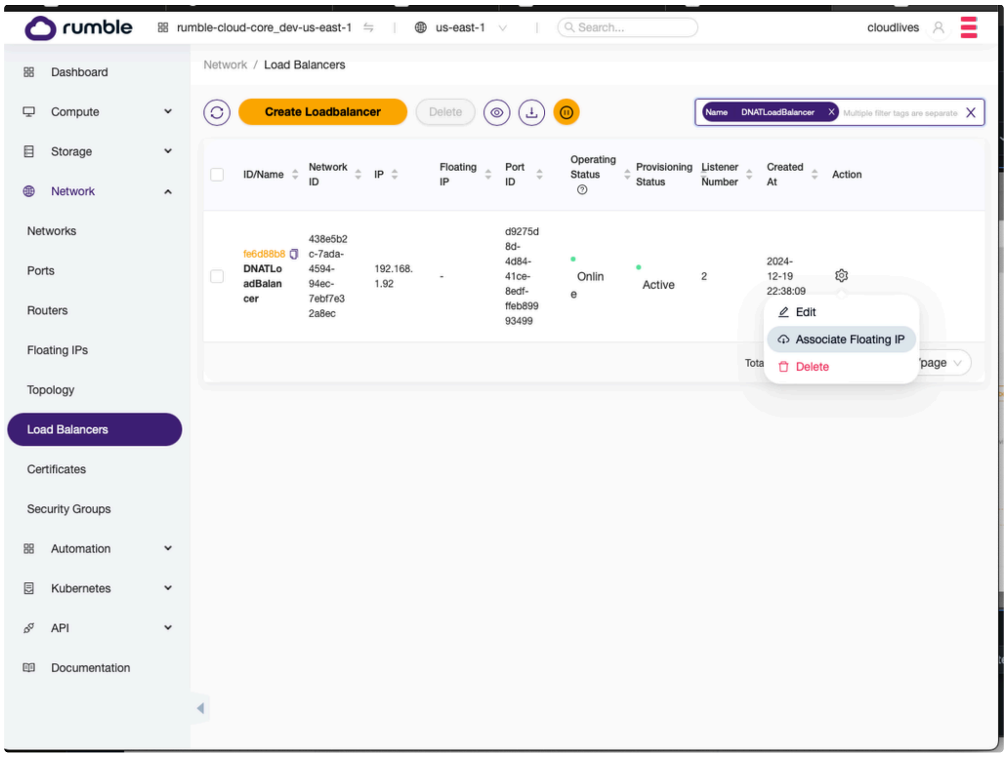
IP Address	Port	Weights
192.168.1.197	22	1

Add External Members

The ip of external members can be any, including the public network ip.

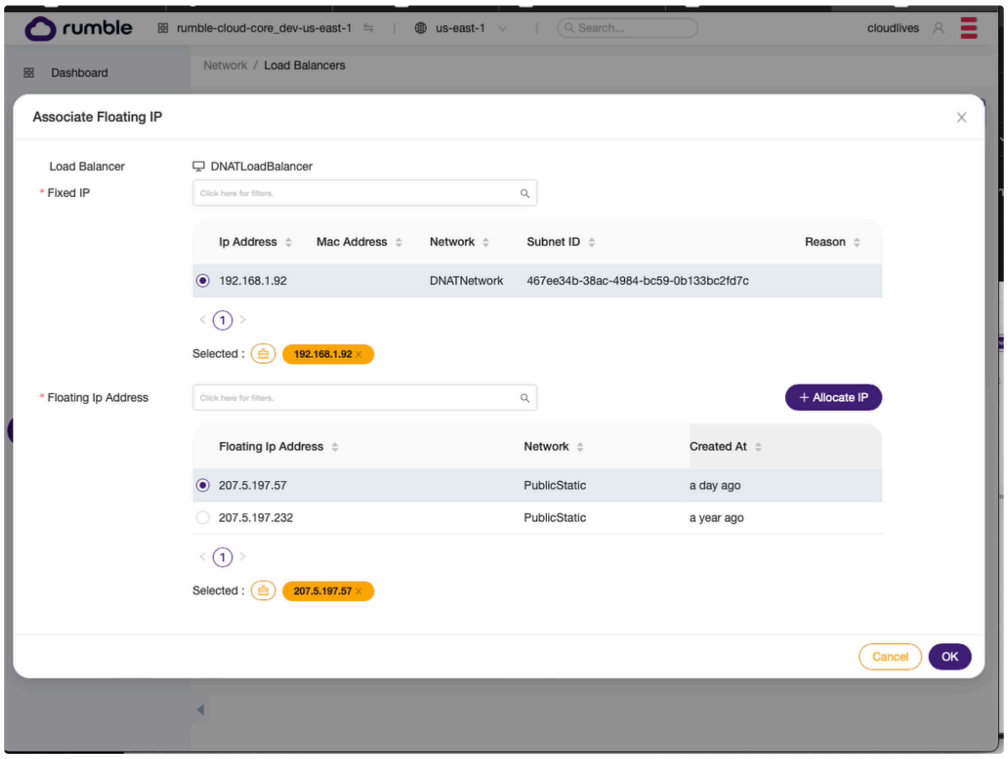
g. Repeat the above steps to add more single member listeners and pools on other ports OR add full fledged pools with multiple members for load balanced services.

6. Assign a Public IP (i.e, Floating IP) to the load balancer.
   a. First find the LoadBalancer and select “Associate Floating IP”



ID/Name	Network ID	IP	Floating IP	Port ID	Operating Status	Provisioning Status	Listener Number	Created At	Action
fe6d88b8 DNATLoadBalancer	438e5b2c-7ada-4594-94ec-7ebf7e32a8ec	192.168.1.92	-	d9275d8d-4d84-41ce-8edf-ffeb89993499	Online	Active	2	2024-12-19 22:38:09	

b. Pick a Public IP and assign it.



Load Balancer: DNATLoadBalancer

* Fixed IP

Ip Address	Mac Address	Network	Subnet ID	Reason
(●) 192.168.1.92		DNATNetwork	467ee34b-38ac-4984-bc59-0b133bc2fd7c	

Selected : 192.168.1.92

* Floating Ip Address

Floating Ip Address	Network	Created At
(●) 207.5.197.57	PublicStatic	a day ago
( ) 207.5.197.232	PublicStatic	a year ago

Selected : 207.5.197.57

Terraform / OpenTofu Sample

This is a sample of creating a load balancer that provides a load balance HTTP and HTTPS service on ports 80 and 443, but using the same Load Balancer (and therefore same IP) is also able to provide SSH access to all the VM in the Application VM Pool with ssh access starting at port 22000 and increasing by 1 for each VM in the pool. Note that the key difference is when setting a load balanced service, you put multiple servers in the same pool. In the case of doing port based natting, you put one server in the pool to accomplish the same thing.

Here is a full zip providing to create 3 Ubuntu VM's with a Load Balancer in front that provides load balanced HTTP and HTTPS on ports 80 and 443, and that also provides SSH access to the 3 VM's via ports 22000, 22001, 22002

terraform-dnat.zip

Here is the key terraform just around creating the load balancer:

```
resource "openstack_lb_loadbalancer_v2" "loadbalancer_app" {
  name           = "${var.system_name}-loadbalancer-app"
  vip_network_id = openstack_networking_network_v2.network_internal_app.id
}

# HTTP Setup

resource "openstack_lb_listener_v2" "listener_http_app" {
  name            = "${var.system_name}-app_listener_http"
  protocol        = "TCP"
  protocol_port   = 80
  loadbalancer_id = openstack_lb_loadbalancer_v2.loadbalancer_app.id
  default_pool_id = openstack_lb_pool_v2.pool_http_app.id
}

resource "openstack_lb_pool_v2" "pool_http_app" {
  name            = "${var.system_name}-pool_http_app"
  protocol        = "TCP"
  lb_method       = "SOURCE_IP_PORT"
  loadbalancer_id = openstack_lb_loadbalancer_v2.loadbalancer_app.id
}

resource "openstack_lb_member_v2" "member_http_app" {
  name        = "${var.system_name}-member_http_app-${count.index}"
  count = length(openstack_compute_instance_v2.server_app)
  pool_id       = openstack_lb_pool_v2.pool_http_app.id
  address       = openstack_compute_instance_v2.server_app.*.access_ip_v4[count.index]
  protocol_port = 80
}

resource "openstack_lb_monitor_v2" "monitor_http_app" {
  name        = "${var.system_name}-monitor_http_app"
  pool_id     = openstack_lb_pool_v2.pool_http_app.id
  type        = "TCP"
  delay       = 10
  timeout     = 5
  max_retries = 3
}

# HTTPS Setup

```

```
resource "openstack_lb_listener_v2" "listener_https_app" {
  name            = "${var.system_name}-listener_https_app"
  protocol        = "TCP"
  protocol_port   = 443
  loadbalancer_id = openstack_lb_loadbalancer_v2.loadbalancer_app.id
  default_pool_id = openstack_lb_pool_v2.pool_https_app.id
}

resource "openstack_lb_pool_v2" "pool_https_app" {
  name            = "${var.system_name}-pool_https_app"
  protocol        = "TCP"
  lb_method       = "SOURCE_IP_PORT"
  loadbalancer_id = openstack_lb_loadbalancer_v2.loadbalancer_app.id
}

resource "openstack_lb_member_v2" "member_https_app" {
  name        = "${var.system_name}-member_https_app-${count.index}"
  count         = length(openstack_compute_instance_v2.server_app)
  pool_id       = openstack_lb_pool_v2.pool_https_app.id
  address       = openstack_compute_instance_v2.server_app.*.access_ip_v4[count.index]
  protocol_port = 443
}

resource "openstack_lb_monitor_v2" "monitor_https_app" {
  name        = "${var.system_name}-monitor_https_app"
  pool_id     = openstack_lb_pool_v2.pool_https_app.id
  type        = "TCP"
  delay       = 10
  timeout     = 5
  max_retries = 3
}

# SSH ACCESS

resource "openstack_lb_listener_v2" "listener_ssh_app" {
  count           = length(openstack_compute_instance_v2.server_app)
  name            = "${var.system_name}-listener_ssh_app-${count.index}"
  protocol        = "TCP"
  protocol_port   = 22000 + count.index
  loadbalancer_id = openstack_lb_loadbalancer_v2.loadbalancer_app.id
  default_pool_id = openstack_lb_pool_v2.pool_ssh_app.*.id[count.index]
}

resource "openstack_lb_pool_v2" "pool_ssh_app" {
  count           = length(openstack_compute_instance_v2.server_app)
  name            = "${var.system_name}-pool_ssh_app-${count.index}"
  protocol        = "TCP"
  lb_method       = "SOURCE_IP_PORT"
  loadbalancer_id = openstack_lb_loadbalancer_v2.loadbalancer_app.id
}

resource "openstack_lb_member_v2" "member_ssh_app" {
  count         = length(openstack_compute_instance_v2.server_app)
  name          = "${var.system_name}-member_ssh_app-${count.index}"
  pool_id       = openstack_lb_pool_v2.pool_ssh_app.*.id[count.index]
  address       = openstack_compute_instance_v2.server_app.*.access_ip_v4[count.index]
  protocol_port = 22
}
```

```

resource "openstack_lb_monitor_v2" "monitor_ssh_app" {
  count         = length(openstack_compute_instance_v2.server_app)
  name        = "${var.system_name}-monitor_ssh_app-${count.index}"
  pool_id     = openstack_lb_pool_v2.pool_ssh_app.*.id[count.index]
  type        = "TCP"
  delay       = 10
  timeout     = 5
  max_retries = 3
}
```