

# VPC Inline LB Plugin (VPC Public Load Balancing)

## Bug Reference

[CLOUDSTACK-9282](#)

## Branch

4.9.0

## Introduction

CloudStack supports a default VPC Virtual Router provider for offering Public Load Balancing within Virtual Private Clouds (VPC's). In such deployments, the VPC Virtual Router is provisioned to actively load-balance public LB rules towards private real-server-VM's deployed inside the Public Tier using the HA Proxy implementation of the VPC Virtual Router.

In SDN backed CloudStack deployments, this may not be the desired deployment, mostly because in SDN backed CloudStack deployments, the Virtual Router may not be present at all.

When deploying CloudStack with a SDN platform (e.g. Nuage Networks Virtualized Services Platform), all routing, DHCP/DNS services and security features may be realized by the SDN platform, typically realized in a distributed manner, without further relying on the Virtual Router VM (which is a centralized solution).

In order to generically support Public Load Balancing within SDN backed CloudStack deployments, a new Load Balancer Provider/Plugin is proposed : VPC Inline LB Provider. When this provider is selected for Public Load Balancing, the Load Balancing functionality is realized by an appliance VM (VPC Inline LB VM) which is deployed in the VPC Public Tier guest network itself (i.e. as a guest VM). This appliance by default is based on a VR appliance but which could be generalized to any type of appliance, which could be more lightweight than System VR template or reversely could be a commercial appliance. This flexibility is not implemented today but could be easily added when this plugin feature gets wider traction. The VPC Inline LB Provider takes care of orchestrating the deployment of the appliance and its provisioning upon the first public load balancer rule being configured with server vms, and similarly takes care of the resource clean-up upon the last public load balancer rule being unconfigured. As mentioned, unlike the VPC Virtual Router implementation case, in this case Load Balancer appliance is a guest VM inside the VPC Public tier, and no longer has a NIC in every single VPC tier.

The design and implementation of this new type of Public Load Balancing solution is generic and can be deployed with any VPC Network provider.

## Purpose

This is the functional specification for a new network plugin called 'VPC Inline LB VM'

## Document History

Author	Description	Date
Kris Sterckx	Added clarification about the LB appliance being a regular System VM	10 Apr 2016
Nick Livens	Small modifications	25 Feb 2016
Nick Livens	Uploaded design document to CWiki	23 Feb 2016

## Use Cases

### VPC Public Load Balancing

- Create a VPC selecting a VPC offering with LB support
- Add a Tier to the VPC, selecting a Network offering with Public LB support
- Acquire a new Public IP for the VPC
- Configure LB Rules on the public IP to load balance servers in the public tier.
- Clean up of VPC Public LB

## Architecture and Design description

We will introduce a new CloudStack network plugin "VpcInlineLbVm" which is based on the Internal LoadBalancer plugin and which just like the Internal LB plugin is implementing load balancing based on at-run-time deployed appliances based on the VR (Router VM) template (which defaults to the System VM template), but the LB solution now extended with static NAT to secondary IP's.

The VPC Inline LB appliance therefore is a regular System VM, exactly the same as the Internal LB appliance today. Meaning it has 1 guest nic and 1 control (link-local / management) nic.

With the new proposed VpcInlineLbVm set as the Public LB provider of a VPC, when a Public IP is acquired for this VPC and LB rules are configured on this public IP, a VPC Inline LB appliance is deployed if not yet existing, and an additional guest IP is allocated and set as secondary IP on the appliance guest nic, upon which static NAT is configured from the Public IP to the secondary guest IP. (See below outline for the detailed algorithm.)

**In summary**, the VPC Inline LB appliance is reusing the Internal LB appliance but its solution now extended with Static NAT from Public IP's to secondary (load balanced) IP's at the LB appliance guest nic.

The following outline depicts the detailed flows :

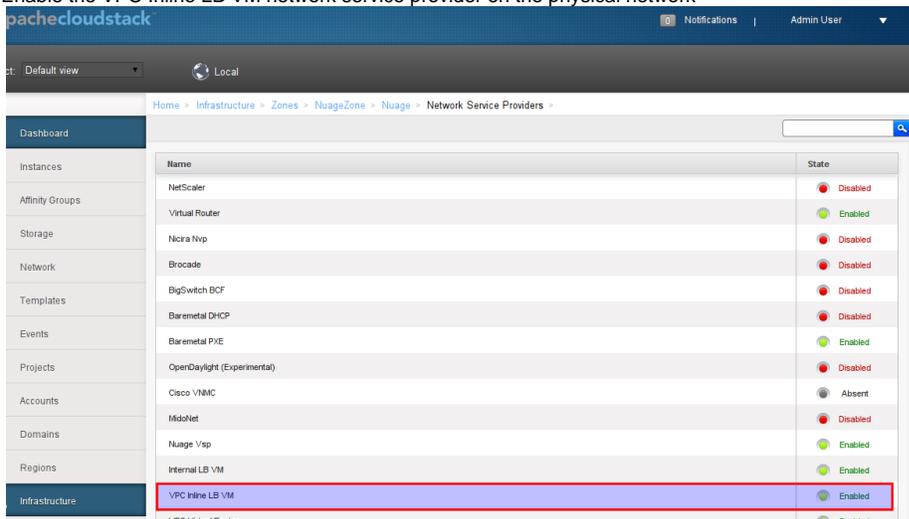
- Apply LB rules:
  - Check if a LB Appliance exists
  - If not, deploy a new one.
    - During VM orchestrate start, VM Guru is called to finalize the VM profile, and the deployment, where it will setup the required nics. (link-local + guest).
  - Group rules by public IP, ignoring rules without destination VM's
  - For each public IP:
    - Check if the Public IP-Guest secondary IP mapping exists for the LB Appliance. This mapping will be defined as follows:
      - VpcInlineLoadBalancerMapping
      - Guest Nic will have secondary IP's
      - Public IP will hold the exact Guest secondary IP in the vmlp field.
    - If not
      - allocate a secondary Guest IP and save mapping
      - Configure appliance to listen on new secondary IP
      - Enable Static NAT (delegate to Network Plugin), passing the secondary IP as destination
    - Translate rules to use Guest secondary IP
    - Send the translated rules to the Hypervisor Agent, which will configure HAProxy.
- Restart Network of LB Tier:
  - Shutdown (only in case of cleanup=True)
    - Destroy the LB Appliance in the network
  - Implement
    - If LB rules exist in the network:
      - Check if a LB Appliance exists for the public IP
      - If not, deploy a new one.

## Web Services APIs

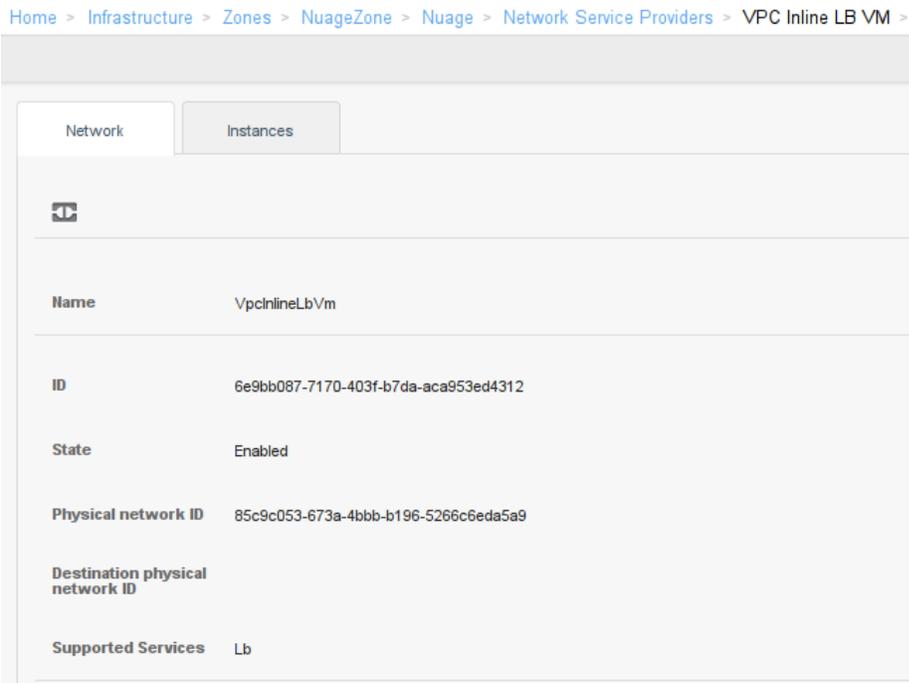
API	Parameters	Description
listVpcInlineLoadBalancerVMs	/	Lists all the VPC Inline LB VMs
startVpcInlineLoadBalancerVM	<i>id</i> : The UUID of the VPC Inline LB VM	Start a VPC Inline LB VM
stopVpcInlineLoadBalancerVM	<i>id</i> : The UUID of the VPC Inline LB VM	Stop a VPC Inline LB VM
configureVpcInlineLoadBalancerElement	<i>id</i> : The UUID of the VPC Inline LB element <i>nspid</i> : The UUID of the network service provider <i>enabled</i> : True to enable, false to disable	Configure the VPC Inline LB element
createVpcInlineLoadBalancerElement	<i>nspid</i> : The UUID of the network service provider	Create a VPC Inline LB element
listVpcInlineLoadBalancerElements	<i>id</i> : The UUID of the VPC Inline LB element <i>nspid</i> : The UUID of the network service provider <i>enabled</i> : True to list enabled, false to list disabled	List the configured VPC Inline LB elements

## UI Flow

1. Enable the VPC Inline LB VM network service provider on the physical network



2. Overview of the VPC Inline LB VM network service provider



3. Add a VPC offering with VpcInlineLbVm as Load Balancer Provider

 Add VPC Offering

\* Name:

\* Description:

Supported Services:

DHCP:	<input checked="" type="checkbox"/>
DHCP Provider:	<input type="text" value="NuageVsp"/>
DNS:	<input type="checkbox"/>
Load Balancer:	<input checked="" type="checkbox"/>
Load Balancer Provider:	<input type="text" value="VpcInline"/>
Gateway:	<input type="checkbox"/>
Static NAT:	<input checked="" type="checkbox"/>
Static NAT Provider:	<input type="text" value="NuageVsp"/>

Redundant router capability:

4. Add a network offering with VpcInlineLbVm as Load Balancer Provider

 Add network offering

\* Name:

\* Description:

Network Rate (Mb/s):

Guest Type:

Persistent:

Specify VLAN:

VPC:

Load Balancer Type:

Supported Services:

VPN:	<input type="checkbox"/>
DHCP:	<input checked="" type="checkbox"/>
DHCP Provider:	<input type="text" value="NuageVsp"/>
DNS:	<input type="checkbox"/>
Firewall:	<input type="checkbox"/>
Load Balancer:	<input checked="" type="checkbox"/>
Load Balancer Provider:	<input type="text" value="VpcInlineL"/>
User Data:	<input type="checkbox"/>

Redundant router capability:

Supported Source NAT type:

Supports Stretched L2 Subnet:

Conserve mode:

Tags:

5. Create a VPC with the previously created VPC offering

 Add VPC

\* Name:

\* Description:

\* Zone:

\* Super CIDR for Guest Networks:

DNS domain for Guest Networks:

Public Load Balancer Provider:

\* VPC Offering:

6. Create a tier with the previously created network offering

 Add new tier

\* Name:

\* Network Offering:

\* Gateway:

\* Netmask:

ACL:

7. Spin a VM in the newly created tier

### 8. Associate a public IP to a VPC

Home > Network - VPC > lbVpc > lbVpc > Router - Public IP addresses >

IPs	Zone	Network Name	State	Quickview
10.30.21.246	NuageZone	serverTier	<span style="color: green;">●</span> Allocated	+
10.30.21.245 [Source NAT]	NuageZone		<span style="color: green;">●</span> Allocated	+

### 9. Configure LB Rules on the public IP and associate them with the spinned VM

Home > Network - VPC > lbVpc > lbVpc > Router - Public IP addresses > 10.30.21.246 > Load Balancing >

Load Balancing

Name	Public Port	Private Port	Algorithm	Stickiness	Add VMs	State	Actions
<input type="text"/>	<input type="text"/>	<input type="text"/>	Round-robin	<input type="button" value="Configure"/>	<input type="button" value="Add"/>		
+ http	80	80	Round-robin	<input type="button" value="Configure"/>	<input type="button" value="Add"/>	Active	<input type="button" value="edit"/> <input type="button" value="delete"/>
+ https	8443	8443	Round-robin	<input type="button" value="Configure"/>	<input type="button" value="Add"/>	Active	<input type="button" value="edit"/> <input type="button" value="delete"/>
+ http...	443	443	Round-robin	<input type="button" value="Configure"/>	<input type="button" value="Add"/>	Active	<input type="button" value="edit"/> <input type="button" value="delete"/>

### 10. Overview of the configured HA Proxy rules on the VPC Inline LB VM

```
root@b-99-DEV:~# less /etc/haproxy/haproxy.cfg
global
    log 127.0.0.1:3914    local0 warning
    maxconn 4096
    maxpipes 1024
    chroot /var/lib/haproxy
    user haproxy
    group haproxy
    daemon

defaults
    log      global
    mode    tcp
    option  dontlognull
    retries 3
    option  redispatch
    option  forwardfor
    option  forceclose
    timeout connect    5000
    timeout client     50000
    timeout server     50000

listen stats_on_public 10.10.10.36:8081
    mode http
    option httpclose
    stats enable
    stats uri    /admin?stats
    stats realm  Haproxy\ Statistics
    stats auth   admin1:AdMiN123

listen 10_10_10_108-80 10.10.10.108:80
    balance roundrobin
    server 10_10_10_108-80_0 10.10.10.240:80 check
    mode http
    option httpclose

listen 10_10_10_108-8443 10.10.10.108:8443
    balance roundrobin
    server 10_10_10_108-8443_0 10.10.10.240:8443 check

listen 10_10_10_108-443 10.10.10.108:443
    balance roundrobin
    server 10_10_10_108-443_0 10.10.10.240:443 check
```

### 11. Overview of the IP table rules associates with these rules

```
root@b-99-DEV:~# iptables-save
# Generated by iptables-save v1.4.14 on Tue Feb 23 10:17:35 2016
*mangle
:PREROUTING ACCEPT [439:36206]
:INPUT ACCEPT [439:36206]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [628:38308]
:POSTROUTING ACCEPT [628:38308]
-A PREROUTING -m state --state RELATED,ESTABLISHED -j CONNMARK --restore-mark --nfmask 0xffffffff --ctmask 0xffffffff
-A PREROUTING -i eth0 -m state --state NEW -j CONNMARK --set-xmark 0x0/0xffffffff
COMMIT
# Completed on Tue Feb 23 10:17:35 2016
# Generated by iptables-save v1.4.14 on Tue Feb 23 10:17:35 2016
*filter
:INPUT DROP [0:0]
:FORWARD DROP [0:0]
:OUTPUT ACCEPT [559:29503]
:FW_OUTBOUND - [0:0]
:NETWORK_STATS - [0:0]
-A INPUT -j NETWORK_STATS
-A INPUT -i eth0 -m state --state RELATED,ESTABLISHED -j ACCEPT
-A INPUT -i eth1 -m state --state RELATED,ESTABLISHED -j ACCEPT
-A INPUT -p icmp -j ACCEPT
-A INPUT -i lo -j ACCEPT
-A INPUT -i eth1 -p tcp -m state --state NEW -m tcp --dport 3922 -j ACCEPT
-A INPUT -d 224.0.0.18/32 -j ACCEPT
-A INPUT -d 225.0.0.50/32 -j ACCEPT
-A INPUT -i eth0 -p udp -m udp --dport 67 -j ACCEPT
-A INPUT -i eth0 -p udp -m udp --dport 53 -j ACCEPT
-A INPUT -i eth0 -p tcp -m tcp --dport 53 -j ACCEPT
-A INPUT -i eth0 -p tcp -m tcp --dport 80 -m state --state NEW -j ACCEPT
-A INPUT -i eth0 -p tcp -m tcp --dport 8080 -m state --state NEW -j ACCEPT
-A INPUT -i eth1 -p tcp -m tcp --dport 3922 -m state --state NEW,ESTABLISHED -j ACCEPT
-A INPUT -d 10.10.10.108/32 -p tcp -m tcp --dport 443 -m state --state NEW -j ACCEPT
-A INPUT -d 10.10.10.108/32 -p tcp -m tcp --dport 80 -m state --state NEW -j ACCEPT
-A INPUT -d 10.10.10.108/32 -p tcp -m tcp --dport 8443 -m state --state NEW -j ACCEPT
-A INPUT -d 10.10.10.36/32 -p tcp -m tcp --dport 8081 -m state --state NEW -j ACCEPT
```

### 12. Overview of the VPC Inline LB VMs

Home > Infrastructure > Zones > NuageZone > Nuage > Network Service Providers > VPC Inline LB VM >

Refresh

Network Instances

Name	Zone	Type	Status	Quickview
b-64-DEV	NuageZone	VPC	Running	+

### 13. Overview of a VPC Inline LB VM

Home > Infrastructure > Zones > NuageZone > Nuage > Network Service Providers > VPC Inline LB VM > b-90-DEV >

Refresh

Details NICs

Name	b-90-DEV
ID	49da41b4-d300-46ba-bab4-72196ed0a7f0
State	Running
Network ID	5d69fe3a-0e53-4c11-9649-3d601257c24
Public IP Address	10.30.21.246
Guest IP Address	10.10.10.87
Link Local IP Address	169.254.3.30
Host	Nick-VRS