HAProxy

Step 1: Understand Your Goal HAProxy is usually placed in front of:

- Kubernetes API Server (6443) → if you want HA for multiple masters.
- Kubernetes Services / Ingress → if you want HAProxy as a load balancer for apps exposed on worker nodes.

Since you mentioned only one master (192.168.0.106), I'll assume:

- You want HAProxy (192.168.0.104) to forward external traffic to services running in the cluster (through NodePorts or Ingress).

Step 2: Find a Service to Expose On your cluster, check services:

```
kubectl get svc -A
```

For example, if you have nginx service exposed as NodePort 30080, it'll be reachable at:
http://192.168.0.106:30080

Step 3: Install HAProxy On the HAProxy VM (192.168.0.104):

```
sudo apt update
sudo apt install haproxy -y
```

Step 4: Configure HAProxy

Edit config:

```
sudo nano /etc/haproxy/haproxy.cfg
```

Example for forwarding traffic from port 80 → nginx NodePort 30080:

```
global
  log /dev/log local0
  log /dev/log local1 notice
  daemon
  maxconn 2048

defaults
```

```
  log global
  mode http
  option httplog
  option dontlognull
  timeout connect 5s
  timeout client 50s
  timeout server 50s

frontend http_front
  bind *:80
  default_backend nginx_backend

backend nginx_backend
  balance roundrobin
  server k8s-master 192.168.0.106:30080 check
```

Step 5: Restart HAProxy

```
sudo systemctl restart haproxy
sudo systemctl enable haproxy
```

Now, when you access: http://192.168.0.104/ HAProxy will forward to your Kubernetes service (192.168.0.106:30080). If you later add multiple worker nodes, you can just add them to the backend:

```
backend nginx_backend
  balance roundrobin
  server k8s-worker1 192.168.0.107:30080 check
  server k8s-worker2 192.168.0.108:30080 check
```

5. Optional – Multiple Apps

If you want HAProxy to serve multiple apps (different hostnames or paths), you can extend the config:

```
frontend http_front
  bind *:80
  acl host_app1 hdr(host) -i app1.local
  acl host_app2 hdr(host) -i app2.local
  use_backend app1_backend if host_app1
  use_backend app2_backend if host_app2
```

```
  default_backend app1_backend

backend app1_backend
  server k8s-master 192.168.0.106:30080 check

backend app2_backend
  server k8s-master 192.168.0.106:30081 check
```

This way HAProxy acts like a simple Ingress Controller replacement.

Revision #2
Created 30 August 2025 15:36:46 by Admin
Updated 30 August 2025 15:51:39 by Admin