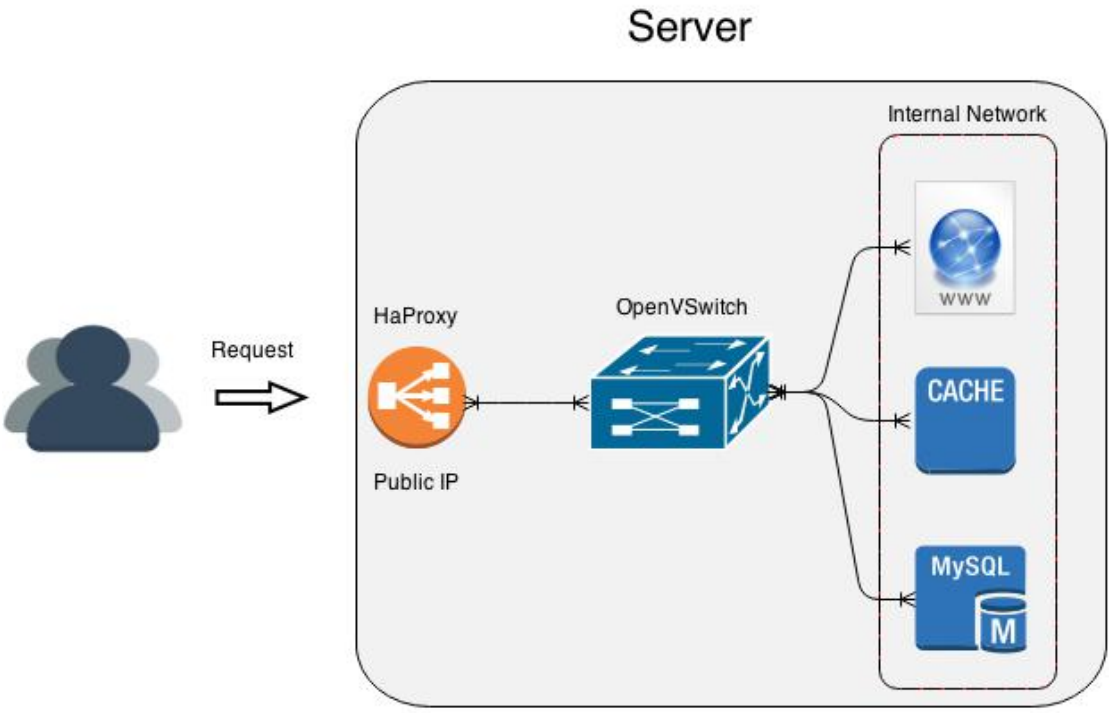
Shipmn - with openvswitch!

You are a sysadmin or developer, and you wanna make some tests in databases, webservers, haproxys, etc, keeping you system “clean” without using virtualbox, vmware or docker.

Our goal:



System requirements:

- Kernel > 3.14
- pip
- unshare
- nsenter
- openvswitch

Install the lasted kernel (reboot after the installation):

```
$echo 'deb http://ftp.debian.org/debian/ wheezy-backports main non-free
contrib' >> /etc/apt/sources.list
$apt-get update
$apt-get -t wheezy-backports install linux-image-amd64 linux-headers-amd64
```

Packages:

```
$apt-get install build-essential python-dev python-pkg-resources python-
setuptools  python-ipaddr curl unzip vim openvswitch-switch
$easy_install pip
$pip install ipcalc netaddr sh config
```

Configure openvswitch:

Change the /etc/network/interfaces to:

```
auto eth0
iface eth0 inet static
address 0.0.0.0
auto ovsbr
iface ovsbr inet static
        pre-up /etc/init.d/openvswitch-switch start
        ovs_type OVSBridge
        ovs_ports eth0 ovsbr
        address 101.137.113.24
        netmask 255.255.255.0
        broadcast 101.137.113.255
        gateway 101.137.113.1
        dns-nameservers 8.8.8.8
        up /sbin/ifconfig $IFACE up
```

Add bridge and port to openvswitch:

```
$ovs-vsctl add-br ovsbr &&  ovs-vsctl add-port ovsbr eth0 &&  ifdown eth0
&& ifup eth0 && ifup ovsbr
```

Configure virtual network interface:

```
$ifconfig ovsbr:1 10.10.10.1
$echo 1 > /proc/sys/net/ipv4/ip_forward
$iptables -t nat -A POSTROUTING -o ovsbr -j MASQUERADE
$iptables -A FORWARD -i ovsbr -o ovsbr:1 -m state --state
RELATED,ESTABLISHED -j ACCEPT
$iptables -A FORWARD -i ovsbr:1 -o ovsbr -j ACCEPT
```

Download ship:

```
$wget https://github.com/bsurfer/ship/archive/master.zip
$unzip master.zip
$cd ship-master/
```

Install ship:

```
$sh install_me_ovs
```

Create webserver:

```
$shipmn create web01 10.10.10.10
```

Create db01:

```
$shipmn create db01 10.10.10.20
```

Create cache01:

```
$shipmn create cache01 10.10.10.30
```

Ship status:

```
$shipmn status
Status all
Ship: web01 – STOPED
Ship: db01 – STOPED
Ship: cache01 – STOPED
```

Start all Ship:

```
$shipmn start all
```

Enter in webserver:

```
$shipmn con web01
```

All ship's are started and ready for receive your configuration

Now you will be able to install packages, configure services, etc, etc.

After that, you only need configure haproxy in Server to route the request to web01, pe.