Distributed load balancing
Real case example using open source on commodity hardware

Booking.com

Pavlos Parissis | LinuxConf Berlin 2016
The traditional way

- Scales only vertically
- Single point of failure
- Choke point for (D)DOS
- Very expensive

Active Node

Standby Node
A better way
How to get there

- Equal-Cost Multi-Pathing routing
- Anycast network address scheme
- Bird Internet Routing Daemon
- A healthchecker for Anycasted services
- HAProxy Layer4-7 load balancer
Equal-Cost Multi-Pathing routing

Nodes are distributed across multiple networks
- Preserves source and destination addresses
- Cheapest form of balancing
- Load balancing at wire-speed
- Adding/removing a path reshuffles flows

<table>
<thead>
<tr>
<th>Destination IP</th>
<th>Next hop</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.56.17.220/32</td>
<td>node1</td>
</tr>
<tr>
<td>5.56.17.220/32</td>
<td>node2</td>
</tr>
<tr>
<td>5.56.17.220/32</td>
<td>node3</td>
</tr>
<tr>
<td>5.56.17.220/32</td>
<td>node4</td>
</tr>
</tbody>
</table>
Equal-Cost Multi-Pathing

Tier 1 Load balancer

Layer 3

Layer 7
Layer 7
Layer 7

Tier 2 Load balancer
2-Tier setup in production

- **Users**
  - **Fabric Layer**
    - Layer 3
    - Tier 1 Load balancer
      - Layer 3
      - ToR Layer
        - Layer 3
        - Tier 2 Load balancer
          - Layer 7
Benefits of 2-Tier setup

- Horizontally scalable
- Scaling and managing each tier independently
- Single device becomes less critical
Anycast network address scheme

sender

receiver A

receiver B

receiver C

distance in number of hops
Anycast in production

Data-center A

LB platform

local users

Data-center B

LB platform

local users

transition time ~20ms
Benefits of Anycast in production

- Network detect failures within 1.2 secs (BFD protocol helps a lot)
- Switches traffic to other location within 1 sec
- Reduces network distance which lowers response time
- Provides a very fast and without manual intervention fail-over which improves service reliability

- **Works** for TCP protocol
Dive into details

- Bird Internet Routing daemon
- A healthchecker for anycasted services
- HAProxy Layer4-7 load balancer
How it works

- **Load balancer node**
  - HAProxy
  - anycast healthchecker
  - Bird

- **ToR switch**
- **Fabric switch**

- **Users**

- **Apps**
How Bird advertise routes

Bird daemon
Load balancer node: 10.1.1.1

BGP protocol

1.2.3.1/32  dev lo [direct1 2016-09-19] * (240)
1.2.3.2/32   dev lo [direct1 2016-09-19] * (240)

BGP peer

import routes

export routes
Filtering routes for unhealthy services

import routes

BGP peer

route in LIST

filter

exported routes: 1.2.3.1/32

LIST= [1.2.3.1/32]

anycast-healthchecker

service

1.12.3.1/32  dev lo [direct1 2016-09-19] * (240)
1.12.3.2/32  dev lo [direct1 2016-09-19] * (240)

direct protocol

loopback interface
1.2.3.1/32
1.2.3.2/32

Booking.com
HAProxy load balancer

- Highly configurable
- Rock solid
- Excellent support
- Supports Lua
- Faster than Nginx in our setup, benchmark yours
HAProxy load balancer performance

**HTTPS thousands of req/sec**
- HAProxy: 350
- NGINX: 210

**Latency (ms)**
- HAProxy: 0.75
- NGINX: 1.15

**CPU User level**
- HAProxy: 30%
- NGINX: 45%
Software and Hardware we use

- Arista switches
- 2 x 10GbE interfaces on servers and 160GbE (4 x 40GbE) on switches
- Bird Internet Routing Daemon http://bird.network.cz
- HAProxy load balancer http://www.haproxy.org
- https://github.com/unixsurfer/anycast_healthchecker
- https://github.com/unixsurfer/haproxystats
- https://github.com/unixsurfer/haproxyadmin
- HP discrete/blade servers
We are hiring
Site Reliability Engineers
https://workingatbooking.com