Super/Ultra-Basic Load-Balancing

Introduction

For AFNOG 2012
Joel Jaeggli
What is Load-balancing

- The act of dividing a workload between \( N > 1 \) devices capable for performing a task.
- Multiple contexts in internet services where this concept occurs.
  - DNS
  - MX records
  - Multiple links (L2 trunks, L3 ECMP)
  - Multiple servers
Goals

- Greater scalability
- Higher availability
- Reduced cost
Examples – L2 trunk or L3 ECMP

- Stateless per-flow-load balancing
- Per-packet causes reordering so...
- XOR 5-tuple
L3+L4  L4 or L7 Load-balancing

- IP+TCP or Application layer (http(s) imap etc)
Applications

• L2/L3 Switches
  • LACP
  • L3ECMP

• L4+
  • Haproxy (L4, L7)
  • NGINX (L7)
  • F5 LTM
  • A10
  • Netscalar
So what does an L4 load Balancer do.

- Looks and the Destination IP and Port to determine which pool of servers a connection is mount for.
- Forwards the incoming connection to one pool member on the basis of policy.
- Could be one-sided e.g. Direct-Server-Return
- Or Source-NAT
- Keep the connection pinned to the particular pool member by tracking the connection.
- How do you track?
5-tuple

- What is a 5-tuple
- XOR hash of source/dest ip, source/dest port, protocol number.
- IP header

<table>
<thead>
<tr>
<th>32-bit IPv4 source address</th>
</tr>
</thead>
<tbody>
<tr>
<td>32-bit IPv4 destination address</td>
</tr>
<tr>
<td>8-bit protocol</td>
</tr>
<tr>
<td>16-bit UDP source port</td>
</tr>
<tr>
<td>16-bit UDP length</td>
</tr>
<tr>
<td>Data (and possible pad byte)</td>
</tr>
</tbody>
</table>
5-tuple continued

- TCP Headers

<table>
<thead>
<tr>
<th>16 bit Source Port</th>
<th>16 bit Destination Port</th>
</tr>
</thead>
<tbody>
<tr>
<td>32 bit Sequence Number</td>
<td></td>
</tr>
<tr>
<td>32 bit Acknowledgement Number</td>
<td></td>
</tr>
<tr>
<td>4 bit header length</td>
<td>6 bit reserved</td>
</tr>
<tr>
<td>16 bit Checksum</td>
<td>16 bit Urgent Pointer</td>
</tr>
<tr>
<td>Options (if any)</td>
<td></td>
</tr>
</tbody>
</table>
What does an L7 load balancer do?

- An L7 load balancer answers incoming connection requests.
- It understands the protocol being spoken across the connection (e.g. HTTP IMAP FTP etc).
- On the basis of either 5-tuple hash or some higher layer value, (example a URI or a cookie or both) the request is directed to a member of the appropriate pool.
- L7 is another word for proxy or ALG (Application Layer Gateway).
Isn't L7 going to be slower than L4?

- Probably but not always.
- Importantly there are optimizations that can reduce the expense.
  - TCP syn-cookies
  - Connection pooling
  - Consider 3-way handshake
Applications - Cont

• Open source
  • Apache mod_proxy_balance
  • Haproxy
  • NGNIX
  • LVS
Applications Commercial

• Commercial
  • F5
  • Netscalar
  • A10

• Benefits of a commercial approach
  • Coordination of supporting elements
    - Routing
    - DNS
    - Complex health checks
    - HA
  • Can have ASIC based acceleration.
High Availability Approaches

- Active-Passive
  - VRRP
  - State replication
- Active-Active
  - State-replication considerations
- Horizontally scaled
  - GTM – DNS based approach
  - L3ECMP (routed)
HA – active/passive
HA – active/passive - failover
HA – active/passive failover with replication
Active / Passive

- Active-passive failover requires a mechanism
- Could use:
  - VRRP (Virtual Router Redundancy Protocol)
  - CARP (Common Address Redundancy Protocol)
- If failover is not coordinated with load-balancer-health, a failed load-balancer may remain active (coordination problem).
- If state is not replicated between load balancers, failover will not account for existing connections (not a problem for short-lived connections with no affinity)
Active / Passive Cont

- Affinity can be preserved with a Cookie
- LVS (linux virtual server) can do state-replication (using a kernel module)
- State-replication doesn't help with scaling performance-wise (at all)
Active/Active
Active/Active – How?

• Need a mechanism to distribute requests to multiple front end load-balancers. In effect, a load balancer for your load balancers.

• HOW?
  • DNS e.g. each LB has a separate ip address associated with resources it's load-balancing
    - Return one or more resource records either randomly or on some externally instrumented basis.
    - Fail load balancers in or out using health check or manually
  • L2 or L3 stateless plus sticky mechanism.
Active/Active – Stateful vs Not

- Stateful is typically done by clusters of commercial load-balancers. State replication can be expensive and imperfect.
  - At scale, can be extremely expensive
  - Memory on cluster members and bandwidth/cpu for replication is the limiting factor for state and connections per section.
- Stateless
  - In the DNS case resource records for a failed LB have to time out of caches before that LB stops being used.
  - In the L3-ECMP case a failure will cause some fraction of connections to rehash across other load-balancers anywhere from a quarter to half (they will then be rendered out of state and lost).
Our Exercise - HAProxy

- We're going to deploy HAProxy to load-balance connections to two http servers.
- HAProxy can do L4 (any TCP) or L7 (HTTP) load balancing
- We're going to do L7, this allows us to access http related features, including for example including a cookie.
HAProxy vs NGINX

- **L4 vs L7**
  - HAProxy can load balance anything over TCP or do L7.
  - NGINX is L7 only (HTTP(s) and IMAP/POP3).

- **SSL**
  - HAProxy doesn't support (can't only treat as TCP)
  - NGINX does, so cookies for example can be parsed, can be used for SSL offload etc.

- **Model**
  - HAProxy is threaded, effectively allowing it to engage multiple cpus in the activity.
  - NGINX uses an event driven single threaded model.
  - Both have merit, HAProxy is probably more scalable.
Goals

1) Install and perform a basic configuration of HAProxy.

2) Configure two additional webserver instances on alternate ports in Apache.

3) Demonstrate load-balanced-http connections between them.

4) Log X-Forwarded-For.

5) Bonus: use a cookie to pin a requesting host to one server or another.
Bibliography

- HAProxy - http://haproxy.1wt.eu/
- NGNIX - http://wiki.nginx.org/Main