Today

1. Review of Day 6 (Please answer Top Hat Questions)
2. Modulo L/B
3. Consistent hashing

Global LB => DNS/BOC:
+ latency
+ data location
+ policies

DNS (none => IP mappings)

Internet (BGP)

Think BIG (Cache)

Small Data Center
(CDN/POP 400 servers)

Large Data Center
(Clusters 100K servers)

Today

TCP Startup

TCP Authentication

Clients

Login

geticnues

LB

Servers

Servers

Servers

Servers

Listeners
<table>
<thead>
<tr>
<th>LB techniques</th>
<th>eBuy's Neutrino</th>
<th>Google's Mogul</th>
<th>FB's Katran</th>
<th>AWS ELB</th>
<th>GitHub LB</th>
<th>Mostly</th>
</tr>
</thead>
<tbody>
<tr>
<td>least connection</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
</tr>
<tr>
<td>round robin</td>
<td>✔️</td>
<td></td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
</tr>
<tr>
<td>Hash (consistent hashing)</td>
<td>✔️</td>
<td></td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
</tr>
<tr>
<td>Hash (rehashing)</td>
<td></td>
<td></td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
</tr>
<tr>
<td>Modulo LB</td>
<td></td>
<td></td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
</tr>
</tbody>
</table>

**Modulo LB**

\[
\text{(Client IP)} \mod N = \text{Server ID}
\]

- \(0\) \(1\) \(2\) \(3\) \(4\) \(N-1\)
- \(N: 6 \Rightarrow 7 \quad \left(\frac{28}{42}\right)\)

Key reasons why number of servers in data center changes:
- failures
- add new servers
How can you extend LB(random/Modulo) to provide consistent mapping of client to servers?

1. LB should hold cache & client → server mapping
2. Store the mapping in an external location (disk)
3. Random of Cache
Extending Round Robin

- Add state
  - Client \(\rightarrow\) Server
  - Map

- Periodically state

- Maintaining \& sharing state is complex
  - Introduces issues during LB failure/restart
  - Hard to share state fast enough

- On LB failure \(\Rightarrow\) find way to replicate state
  - (external \(\Rightarrow\) S3 or persistent)

- If many LB \(\Rightarrow\) find way to share state
  1. External K/V
  2. RPC calls to exchange state

How can you be stateless \& provide consistent LB?
MD5 \Rightarrow 64 \text{ bits} \xrightarrow{\text{Consistent Hashing}} 32 \text{ bits} \xrightarrow{\text{Keyspace}} \frac{128}{256}

1. Server ID = \text{hash (IP address)} \mod N

2. Client ID = \text{hash (IP address)} \mod N

\text{using the equations} \Rightarrow \text{every LB can calculate server ID without state}

① How to deal w/ hash collisions?
② Client \Rightarrow this okay: the clients are assigned to the same server
③ Server \Rightarrow change IP address of the server
serverlist = sort server ID

foreach server in serverlist
  if (server > client ID)
    myServer = server
    break
  
if (myServer == nil)
  myServer = serverlist[0]

if no server w/ ID greater than client then you are in wraparound edge case
  then pick first server (which turns out to be next server when in "ring")