Cluster-based Internet Services: Service Location Lookup

Kai Shen
Dept. of Computer Science, University of Rochester

Basic Services

- Service location lookup
  - locate the nodes providing the desired service on the desired data partition
- Cluster load balancing
  - find the one with light load or lightest load
- Any other basic services?

Requirements

- scalable – what if there are thousands of nodes?
- reliable – what if nodes break down once a while?
- easy to manage – what administration efforts are needed to bootstrap, repair, or upgrade the system?

Tell me your idea ...

How about this approach?

- let each node announce its local service to everybody else once a second
- every node monitors all such announcements and maintains a local service/location mapping table
- any information not being repeated for five seconds is purged out of the mapping table
- service location lookup is merely a local table lookup

Is it

- scalable?
- reliable?
- easy to manage?
Analysis of My Approach

- It is quite reliable and easy to manage.
- Not sure about the scalability?!
- Lineal scale of
  - CPU processing overhead for received service announcements
  -Incoming bandwidth consumption for received service announcements
- How to measure these?

A Measurement

- Methodology:
  - A node receiving service announcements at varying rates
  - Measure the slowdown of a CPU-bound program and a incoming-network-bandwidth program
- My setting:
  - a Linux server with dual 400MHz Pentium II processors; connected with a 100Mbps switched Ethernet link
  - service announcements are in UDP, 1KB payload
- My results:

<table>
<thead>
<tr>
<th># of messages/second</th>
<th>30</th>
<th>100</th>
<th>300</th>
<th>1000</th>
<th>3000</th>
</tr>
</thead>
<tbody>
<tr>
<td>Slowdown of network-bound prog.</td>
<td>0.3%</td>
<td>0.9%</td>
<td>2.7%</td>
<td>9.6%</td>
<td>35.5%</td>
</tr>
<tr>
<td>Slowdown of CPU-bound prog.</td>
<td>0.1%</td>
<td>0.3%</td>
<td>0.8%</td>
<td>2.2%</td>
<td>6.3%</td>
</tr>
</tbody>
</table>

Cluster Load Balancing

- Load index
  - number of active service requests
  - CPU/memory/IO load
- Tell me your idea ...
- How about a similar approach?
  - let each node announce its workload to everybody else once a second
  - every node monitors all such announcements and maintains a local node/workload mapping table
  - load balancing is merely a local table lookup
  - can even be piggybacked with the service announcement messages