Design and implementation of an MPLS based load balancing architecture for Web switching

Radu Dragos, Sanda Dragos and Martin Collier
1. Abstract

WWW - The preferred technology:
web server limitations ⇒
- New architectures - Web clusters
& New technologies - Web Switching

MultiProtocol Label Switching:
The preferred technology for:
- Traffic engineering
- Switching
The next generation of networks

Design & Implementation:
an MPLS web switching architecture for QoS enabled networks
1. Abstract

WWW - The preferred technology:
web server limitations ⇒
- New architectures - Web clusters
& New technologies - Web Switching

MultiProtocol Label Switching:
The preferred technology for:
- Traffic engineering
- Switching
The next generation of networks

Design & Implementation:
an MPLS web switching architecture for QoS enabled networks
1. Abstract

WWW - The preferred technology:
web server limitations ⇒
- New architectures - Web clusters
& New technologies - Web Switching

MultiProtocol Label Switching:
The preferred technology for:
- Traffic engineering
- Switching
The next generation of networks

Design & Implementation:
an MPLS web switching architecture for QoS enabled networks
1. Abstract

WWW - The preferred technology:
web server limitations ⇒
- New architectures - Web clusters
& New technologies - Web Switching

MultiProtocol Label Switching:
The preferred technology for:
- Traffic engineering
- Switching
The next generation of networks

Design & Implementation:
an MPLS web switching architecture for QoS enabled networks
2. Introduction
3. Solutions to the problem of over-congested Web servers

3.1. Web Content Caching

Limitations: static vs dynamic content
3.2. Web Content Mirroring

Drawbacks: *user decision*
3.3. Cluster of web servers

Categories: - replicated content  
- mirrored content
4. Related problems

4.1. Overloading a web server

\[ \lambda \sim 10 \text{ connections/second} \]
\[ 1/\mu \sim 30 \text{ seconds} \]

1 server: \( C_1 \sim 300 \)

2 servers: \( C_2 \sim 150 \)

3 servers: \( C_3 \sim 100 \)
4. Related problems

4.1. Overloading a web server

\[ \lambda \sim 10 \text{connections/second} \]
\[ 1/\mu \sim 30 \text{seconds} \]

\begin{align*}
1 \text{ server: } C_1 & \sim 300 \\
2 \text{ servers: } C_2 & \sim 150 \\
3 \text{ servers: } C_3 & \sim 100
\end{align*}
4. Related problems

4.1. Overloading a web server

\[ \lambda \sim 10 \text{ connections/second} \]
\[ 1/\mu \sim 30 \text{ seconds} \]
4. Related problems

4.1. Overloading a web server

\( \lambda \sim 10 \text{connections/second} \)
\( 1/\mu \sim 30 \text{seconds} \)

1 server: \( C_1 \sim 300 \)

2 servers: \( C_2 \sim 150 \)

3 servers: \( C_3 \sim 100 \)
4.2. The TCP continuity problem
5. Approaches

5.1. Round-Robin DNS

Limitations:  
- DNS caching  
- IP browsing  
- load information
5.2. HTTP redirect

Drawback: - redirection delay
5.3. Load balancing switches

Drawbacks:  
- single point of failure
- expensive solution
6. MPLS Approach

6.1. Introduction to MPLS
6.2. Traffic Engineering objectives for MPLS environment

- traffic oriented
- resource oriented
6.2. Traffic Engineering objectives for MPLS environment

- traffic oriented
- resource oriented
6.2. Traffic Engineering objectives for MPLS environment

- traffic oriented
- resource oriented

Relation with load balancing:
6.2. Traffic Engineering objectives for MPLS environment

- traffic oriented
- resource oriented

Relation with load balancing:

▷ resource oriented objective
  - server over- or under-utilization
6.2. Traffic Engineering objectives for MPLS environment

- traffic oriented
- resource oriented

Relation with load balancing:

▶ resource oriented objective
  - server over- or under-utilization

▶ traffic oriented objective
  - supporting QoS operation
7. Framework

[Diagram showing the MPLS cloud network with clients, LERs, LSRs, and dispatchers.]
7. Framework

![Framework Diagram]
8. Implementation

- IP Client (not MPLS capable)
- Linux Firewall
- Edge MPLS capable Linux router
- Dispatcher
- Host UML MLPS capable Linux router
- UML Edge MPLS capable Linux server
- WebServer
- WebServer 1
- WebServer 2
- WebServer n
- Linux Box
- MPLS cloud
- LER
9. Performance evaluation
9. Performance evaluation

Round-robin load balancing for large files:

<table>
<thead>
<tr>
<th></th>
<th>Server 1</th>
<th></th>
<th>Server 2</th>
<th></th>
<th>Server 3</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>no.</td>
<td>%</td>
<td>no.</td>
<td>%</td>
<td>no.</td>
<td>%</td>
</tr>
<tr>
<td>20 con</td>
<td>7</td>
<td>35%</td>
<td>5</td>
<td>25%</td>
<td>8</td>
<td>40%</td>
</tr>
<tr>
<td>30 con</td>
<td>11</td>
<td>36.66%</td>
<td>12</td>
<td>40%</td>
<td>7</td>
<td>23.33%</td>
</tr>
<tr>
<td>50 con</td>
<td>16</td>
<td>32%</td>
<td>16</td>
<td>32%</td>
<td>18</td>
<td>36%</td>
</tr>
</tbody>
</table>
9. Performance evaluation

Round-robin load balancing for large files:

<table>
<thead>
<tr>
<th></th>
<th>Server 1</th>
<th>Server 2</th>
<th>Server3</th>
</tr>
</thead>
<tbody>
<tr>
<td>no.</td>
<td>%</td>
<td>no.</td>
<td>%</td>
</tr>
<tr>
<td>20 con</td>
<td>7 35%</td>
<td>5 25%</td>
<td>8 40%</td>
</tr>
<tr>
<td>30 con</td>
<td>11 36.66%</td>
<td>12 40%</td>
<td>7 23.33%</td>
</tr>
<tr>
<td>50 con</td>
<td>16 32%</td>
<td>16 32%</td>
<td>18 36%</td>
</tr>
</tbody>
</table>

Round-robin load balancing for small files:

<table>
<thead>
<tr>
<th></th>
<th>Server 1</th>
<th>Server 2</th>
<th>Server3</th>
</tr>
</thead>
<tbody>
<tr>
<td>no.</td>
<td>%</td>
<td>no.</td>
<td>%</td>
</tr>
<tr>
<td>300 con λ = 3</td>
<td>102 34%</td>
<td>99 33%</td>
<td>99 33%</td>
</tr>
<tr>
<td>900 con λ = 9</td>
<td>295 32.77%</td>
<td>303 33.66%</td>
<td>302 33.55%</td>
</tr>
<tr>
<td>300 con λ = 12</td>
<td>94 31.33%</td>
<td>96 32%</td>
<td>110 36.66%</td>
</tr>
</tbody>
</table>
10. Conclusions

- exploiting the QoS capabilities of underlaying network
- a novel technique for the next generation switching protocols (e.g. MPLS)
- a working, cost effective architecture in an open source environment
- MPLS based solution performed well even for highly loaded Web switches (12 connections/second ∼ 1 million hits/day)
- porting to hardware can increase the performance
10. Conclusions

- exploiting the QoS capabilities of underlaying network

- a novel technique for the next generation switching protocols (e.g. MPLS)

- a working, cost effective architecture in an open source environment

- MPLS based solution performed well even for highly loaded Web switches (12 connections/second ~ 1 million hits/day)

- porting to hardware can increase the performance
10. Conclusions

- exploiting the QoS capabilities of underlaying network
- a novel technique for the next generation switching protocols (e.g. MPLS)
- a working, cost effective architecture in an open source environment
- MPLS based solution performed well even for highly loaded Web switches (12 connections/second ~ 1 million hits/day)
- porting to hardware can increase the performance
10. Conclusions

- exploiting the QoS capabilities of underlaying network

- a novel technique for the next generation switching protocols (e.g. MPLS)

- a working, cost effective architecture in an open source environment

- MPLS based solution performed well even for highly loaded Web switches (12 connections/second ∼ 1 million hits/day)

- porting to hardware can increase the performance
10. Conclusions

- exploiting the QoS capabilities of underlaying network
- a novel technique for the next generation switching protocols (e.g. MPLS)
- a working, cost effective architecture in an open source environment
- MPLS based solution performed well even for highly loaded Web switches (12 connections/second ∼ 1 million hits/day)
- porting to hardware can increase the performance
10. Conclusions

- exploiting the QoS capabilities of underlaying network

- a novel technique for the next generation switching protocols (e.g. MPLS)

- a working, cost effective architecture in an open source environment

- MPLS based solution performed well even for highly loaded Web switches (12 connections/second ~ 1 million hits/day)

- porting to hardware can increase the performance