DISTRIBUTED CACHING SYSTEM  TD5-02

PARTICIPANTS
Yeung Sin Yan  Lam Tik Ka
Leung Hoi Yi  Cheung Yan Kwong

SUPERVISOR
Professor Danny Tsang
Use of the Internet has been increasing tremendously in the past decade which leads to the rapid growth of Internet’s traffic. As a result, a need of deploying proxy caches arises as it is a technology aimed at improving the utilization of the Internet.

A proxy cache sits between web servers and clients, and watches requests for HTML pages, images and files (commonly called as web objects) come by, saving a copy for itself. If there is another request for the same object, it will send the copy that it has to the client, instead of asking the web server for it again. Reduction in the usage of Internet’s bandwidth can then be achieved.

This project aimed at building a distributed caching system that is composed of numerous proxy caches, called WebCaches and a WebController. Features including multi-session fetching for large web objects, content pushing and replication were implemented. A centralized administration tool was built to control, configure and monitor all the WebCaches within a cluster. The ultimate objective of this project is to apply some new algorithms and techniques in order to improve the performance of current implementation of the distributed caching system.

Content pushing refers to the process of sending updated web objects to a proxy cache before they are requested by the client. When the client requests these objects, the response time therefore can be reduced. Content replication refers to the process of replicating popular web objects from one WebCache to others. This can prevent the WebCache which have stored those objects from overloading.

Multi-session fetching makes use of two or more HTTP connections for downloading web resources simultaneously. This technique increases the speed in downloading large objects as clients can take more bandwidth in the network.

Cache Digest is a summary of the content of a proxy cache, which is made available to other caches. It allows proxy caches to determine whether a particular web object was cached in the other WebCaches. WebCaches exchange their Cache Digest with each other on demand.

Administrative page is a graphical user interface for viewing WebCaches’ status, statistics and configuration. It is written in dynamic web page, which is PHP in our case, as the displayed information changes from time to time. This tool can also be used to modify the configuration files of WebCaches.
Explanations of line colors:

- Clients’ browsers send HTTP requests to and receive requested web objects from the WebController.
- WebController send SNMP requests to all WebCaches in the system for load balancing.
- WebCaches fetch large web objects from any web servers by multi-session.
- Web servers together with WebPush pushes content to any of the WebCaches.
- 1. WebCaches exchange Cache Digest on demand or periodically
- 2. Content replication from one WebCache to the others.
- Administrator access Administrative Page by a browser.
WebController plays an important role in forming our cooperative distributed caching system with cluster architecture. It is also responsible for balancing workload within the cluster and administering WebCache remotely through a graphical user interface. These systems can be connected by Inter-cluster WebCache Server Y. Therefore, larger system would be formed and scalability of the system would be increased. WebCaches would exchange their cache digest in order to reduce the response time of fetching object.

A new protocol, content_push was implemented. The format is similar that of HTTP. It is aimed at facilitating content pushing and replication.

<table>
<thead>
<tr>
<th>Received Message</th>
<th>Corresponding Action taken by WebCache</th>
</tr>
</thead>
<tbody>
<tr>
<td>Get content_push://object_url</td>
<td>Content Pushing: Fetch the fresh copy of web object, object_url from web server and replace the old one in local cache.</td>
</tr>
<tr>
<td>Cache-Control: no_cache</td>
<td></td>
</tr>
<tr>
<td>Purge http://object_url http1.0 (Normal http request)</td>
<td>Replication: Remove old copy of web object, object_url from local cache and fetch fresh one from sender of the content_push request (usually another WebCache).</td>
</tr>
<tr>
<td>Followed by Get content_push://object_url</td>
<td></td>
</tr>
</tbody>
</table>

This graph shows the difference in downloading time according to file size between using single-session and multi-session fetching technique. After calculation, it is found that downloading time through multi-session is saved by 30% than through single-session.