http://ansinet.com/itj

ITJ

ISSN 1812-5638

INFORMATION TECHNOLOGY JOURNAL

ANSInet
Asian Network for Scientific Information
308 Lasani Town, Sargodha Road, Faisalabad - Pakistan

Information Technology Journal 11 (12): 1776-1780, 2012
ISSN 1812-5638 / DOI: 10.3923/itj.2012.1776.1780

A Streaming Protocol for Memcached

X. Wang, B. Zhou and W. Li
Department of Computer Science and Technologies, Zhejiang University, Hangzhou, China

Abstract: Memcached is a general-purpose distributed memory caching system that was originally developed by Danga Interactive for Live Journal but is now used by many other sites and It is thought to be one of the most effective solutions to speed up dynamic database-driven websites by caching data and objects in RAM to reduce the number of times that an external data source must be read. Memcached system uses a client-server architecture, it provides its standard memcached protocol to support any types of programming languages. The memcached protocol is simple and very efficient, while it doesn't support big data objects very well. In this study, the memcached systems were firstly illustrated and then a detailed introduction was provided for the memcached protocol and then the issues of the stock protocol were specified. After that, a new streaming protocol was illustrated which was well designed for big data objects and could be easily integrated into original memcached protocols. Finally, some experiments were done to evaluate the new streaming protocol and finally the new protocol was proved to be very efficient for big data objects storage and it explored new application fields for memcached.

Key words: Memcached, protocol, streaming, performance, cache, network

MEMCACHED INTRODUCTION

A recent study (Gulli and Signorini, 2005) shows that over 11.5 billion web pages are indexable in the end of January 2005. As of world wide web size's statistical data (WWWS, 2009), the indexable web pages became at least 25.21 billion in March 2009. In July, 2008, Alpert and Hajaj (2008) from Google announced that one trillion unique URLs had been found. Because of the issues of frustration over congestion in infrastructure and the high latency of browsing the web, a new word "the World Wide Wait" is used to call the World Wide Web. (TechEncyclopedia, 2009).

Cache daemons (Voras and Zagar, 2008) is one of the most effective techniques for decreasing the access latency and increasing the throughput of web sites by caching frequently accessed data in even faster data stores. Most of the web sites always cache the static pages (such as the Java script files, CSS files, images, etc.,) into the memory, by doing that, the latency will be obviously decreased. But due to the profile result, the biggest latency part was the database access time. Memcached is proposed to resolve such performance issues (Fitzpatrick, 2004). Due to a 2010's profile, Wikipedia, Flickr, Twitter, Youtube and many other big web sites were using memcached to accelerate their websites. Actually, Memcached is more than a web cache, it is a common designed data cache, you can also put anything into the cache container and the scalability and flexibility is pretty good with its distributed design. The following figure (Fig. 1) gives a simple introduction of the memcached architecture.

Memcached is a simple key/value storage; you can simply think it be a distributed hash map. All key/value pairs are partitioned by the key, in most cases, the keys will be hashed before storing and retrieving. It supports get, set, append, prepend, delete and many other commands to access the data items and it also has "Check and Set" command to resolve race conditions on updating cache data.

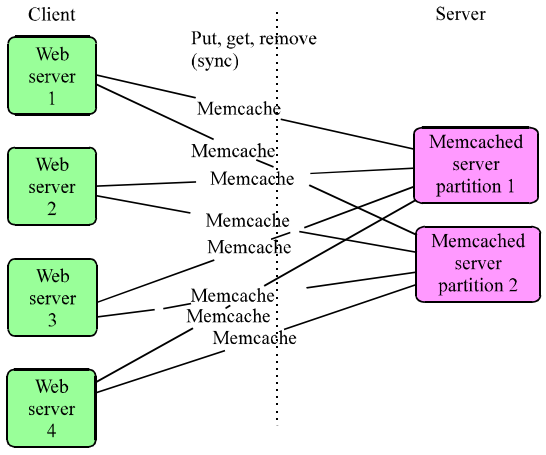

Fig. 1: Memcached architecture

Corresponding Author: Xingen Wang, Department of Computer Science and Technologies, Zhejiang University, Hangzhou, China Tel: +86-571-81022283 Fax: +86-571-81022287

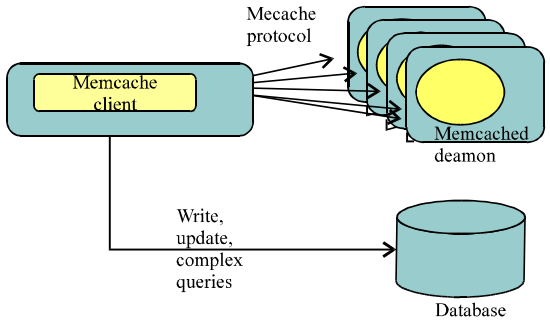

Fig. 2: Typical use of memcache for scaling read mostly applications

Figure 2 shows how it is typically used in industry (Shalom, 2010). Memcached has grown in popularity in latest years, both because many Web sites now need to scale and memcached offers a method to do so that is both easier and cheaper than clustering (Lerner, 2008). Wikipedia is using memcached as its cache engine and actually there are many websites are using it, include Slashdot, LiveJournal, SourceForge, etc. (Voras and Zagar, 2008).

Though memcached is widely used in caching dynamic data objects, while the memcached protocol itself is limited in supporting small size of data objects. The objective of this study is to propose an improved memcached protocol so that memcached can easily support big size of data objects and improve the efficiency for storing and retrieving data objects for memcached.

MEMCACHED PROTOCOL AND PROBLEMS

Memcached protocol supports both TCP and UDP protocol and it provides two types of application level protocols in TCP and UDP. They are "text protocol" and "binary protocol". "Text protocol" comes together with the first version, while "binary protocol" comes with the version 1.4+.

Memcached protocol: Both text protocol and binary protocol provides following types of commands; the only difference between text and binary is that binary protocol eliminates the parsing time in both client and server side, it is said to be have lower latency in performance and better extensibility (Maesaka, 2008); while text protocol also has its advantage, such as "easy debugging".

Storage commands: There are six storage commands: "set", "add", "replace", "append", "prepend" and "cas". They ask the server to store some data identified by a key. The client sends a command line and then a data block; after that the client expects one line of response which will indicate success or failure. Since you can easily get the full version of text protocol from memcached official website, only set command is illustrated as following in this article:

set <key> <flags> <exptime> <bytes> [noreply]\r\n
<data block>\r\n

where, <key> is the key under which the client asks to store the data; <flags> is an arbitrary 16-bit unsigned integer (written out in decimal) that the server stores along with the data and sends back when the item is retrieved; <exptime> is expiration time; <bytes> is the number of bytes in the data block to follow; [noreply] is an optional parameter instructs the server to not send the reply; <data block> is a chunk of arbitrary 8-bit data of length <bytes> from the previous line.

The response from server is very simple, it can be one of STORED, NOT_STORED, EXISTS.

Retrieval commands: "get" and "gets" commands operates like this:

get <key>*\r\n
gets <key>*\r\n

where, <key>* means one or more key strings separated by white space.

The response for each item is something like this:

VALUE <key> <flags> <bytes> [<cas unique>]\r\n
<data block>\r\n

And it will get an "END\r\n" when terminates. All parameters here has the same meaning as above.

Other commands: These commands don't involve unstructured data. In all of them, the client sends one command like and expects either one line of response, or several lines of response ending with "END" on the last line. These commands include "delete", "incr/decr", "stats", "STAT", "version", "flush_all" and "quit".

Problems in the old memcached protocol: Let's get started with "set" command which is a representative memcached storage command. In most cases, data objects were stored into memcached server through memcached client. The procedure can be easily divided into 4 major steps, they are "Serialize Object", "Transfer Data over network", "Store Data on the server" and "send Response Back" (Fig. 3).

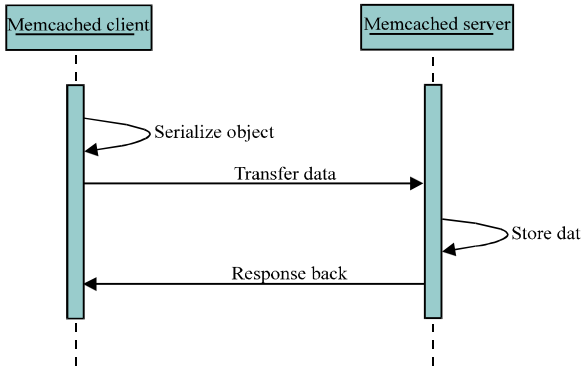

Fig. 3: Steps of memcached storage commands

Fig. 4: Sequence storage of data objects in memcached

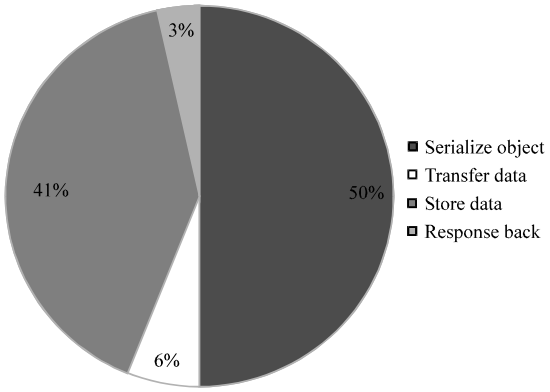

Fig. 5: Latency composition for traditional memcached storage command

Because memcached storage commands require a "<bytes>" field in the protocol which means you have to specify the chunk size of the value here before filling out other fields; while in most cases you don't know the exact bytes of the value before it is fully serialized. Since for that, you have to fully serialize or Marshall the value object before send out the whole key/value pair. Well, in this case, all above steps must occur in strict sequences which means that the total latency for this operation is the sum totals of all the latencies of these steps. You can find the result in Fig. 4.

Since in some object oriented programming languages, the serialization time will take about 25-50% of the time (Philippsen and Haumacher, 1999). Some measurements were done in this study to measure the latencies for each step and they were summarized in following pie chart (Fig. 5).

In our test environment, 2 I BM×3650 m^2 servers were used to hold the memcached client and server and they are connected with 1GB network; A representative data object was chosen in this test case, the data object is an address book with 100 contacts or about 10 kb in size.

Actually, in such situation, the latency of "Transfer Data" from the total latency could be easily cut with an improved memcached protocol. And this new improved memcached protocol is illustrated in following section.

MEMCACHED STREAMING PROTOCOL

"Transfer data while serializing the object" is an efficient solution to minimize the latency for a remote method invocation. Actually, Memcached Streaming Protocol is exactly the protocol based on this strategy.

Definition: The "<bytes>" mentioned in stock memcached protocol is the key point why "serialization" and "data transfer" should be in strict order. Since for that, in Memcached Streaming Protocol, "<bytes>" field was simply removed and "<data block>" was re-organized into "<data frame>". A general Memcached Streaming Storage Command is defined as following; it is so called "Streaming Set" or "sset" for short:

sset <key> <flags> <exptime> [noreply]\r\n
<data frame>\r\n

where, <key>, <flags>, <exptime> and [noreply] have the same meaning as stock memcached storage commands. <data frame> is something like a linked list and all data frames are linked together. A sample data frame is defined as following:

|<bytes>|<body>|

where, <bytes> is two bytes field which indicates the real length for this frame, the maximum length for a data frame is 65535, it will be 0 if this frame is the tail; while <body> is a chunk of arbitrary 8-bit data of length <bytes> in the header of this data frame. Thus, a sample "sset" command should have at least one data frame and can reach up to tens of thousands of data frames if the data object is too much big. Figure 6 is schematic diagram for these data frames.

Advantages: The advantages of the Memcached Streaming Protocol are obvious compared to the stock memcached protocol.

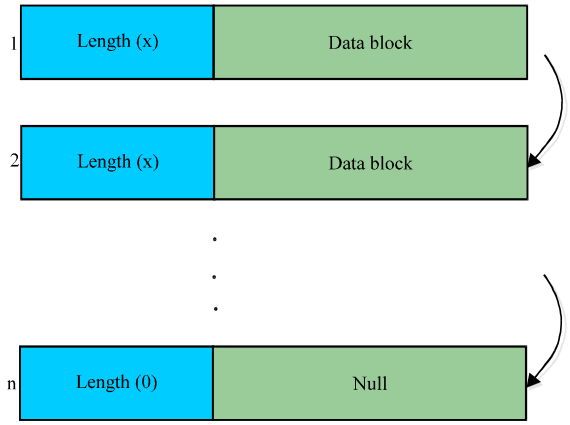

Fig. 6: Data frames in memcached streaming protocol

Fig. 7: Parallel storage of data objects with streaming protocol

Improve the latency: Since a memcached command can easily be divided into 4 major steps as in Fig. 3 and these 4 steps must be executed in strict order, while in this new protocol, the first 2 steps which were to be executed in client side, can be executed in parallel (Fig. 7). Because of that, the latency shall be decreased and in theory, if TSO represents for the latency of "Serialize Object" and TTD represents for the latency of "Transmit Data", min (TSO, TTD) will be cut off from the whole latency compared to the stock protocol.

Utilize memory efficiently: The second advantage for the Streaming Protocol is that it can help minimize the memory usage in the client side. Just imagine that you have a very big object, you have to create enough temp memory to hold all serialized bytes in the original memcached protocol; while with this new Streaming Protocol, you just need to malloc a small memory to hold the small data frame since the data frames will be sent out one by one as a stream. And of course, it saves the garbage collection time in more or less.

Besides that, this new protocol is easy to be integrated into original protocol because it is a new memcached command and is an extension of original one. And because of its low-latency and well-support for big data objects, memcached can easily be explored to more application fields, such as caching of the multi-media objects, etc.

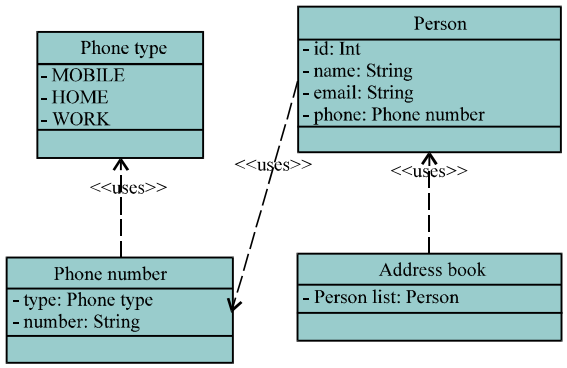

Fig. 8: Class diagram of address book

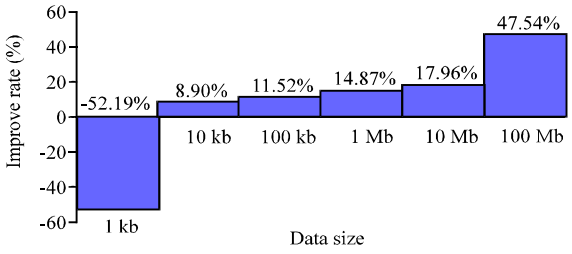

Fig. 9: Improvements of memcached streaming protocol

Table 1: Latency comparison for memcached protocols

	Latency (μsec)	
Data size	SET	SSET
1 kb	71.77	109.23
10 kb	679.96	619.41
100 kb	6135.3	5428.4
1 Mb	63479	54040
10 Mb	662670	543650
100 Mb	10857133.33	5696166.667

EXPERIMENTS AND RESULTS

Some experiments were completed to demonstrate the improvements of the Memcached Streaming Protocol.

Address book is selected as the data object in our performance benchmark, since Google Protocol Buffer did the similar tests with such benchmark. Figure 8 shows the class diagram of the data object.

The experiments were done in following environments: 2 IBM×3650 M^2 machines, one is the Server and the other is the client; these 2 machines are connected with 1 GB low latency network.

In our experiments, the tests were run for thousands of times and the average latency for each request was caculated and the result can be found in following table (Table 1).

From the results, it shows that the latency improved obviously in most cases and it improved more and more when the size of data objects continue increasing (Fig. 9).

The effectiveness of the new Memcached Streaming Protocol is negative in small data object, that's because it contains 2 frames (with an additional tail frame) compared with 1 frame of stock Memcached Protocol in this case. The Latency keeps improving when the size of the data object increases in our experiments and the major reason for these improvements is of its efficiency in memory utilization (refer to the 100 Mb test case in our experiments).

CONCLUSIONS

The new Memcached Streaming Protocol improves the latency for memcached commands and also improves the memory utilization in the client side. Besides that, it is easy-integrated and is 100% compatible with stock memcached protocol. You can simply set data objects with "sset" command and get them with "get" or "sget" command. And for the other stock memcached storage commands, such as "add", "append", etc., you can simply create related streaming memcached commands like "sadd", "sappend", etc., to make them support the new streaming protocol.

Due to our statistics, the new Memcached Streaming Protocol will have about 5-10% latency improvements in the storage of data objects and over 50% improvement in the extreme case.

Because of its well-support of big data object, memcached can now be used to store even big data objects, such as the multi-media files (flash, big picture, mp3, video clips, etc.).

ACKNOWLEDGMENTS

As advisors, sincerely thanks Prof. Shanping Li and Prof. Bo Zhou for their constructive suggestions and timely encourages. And also thanks very much for Wei Li's utility to generate the analysis report.

REFERENCES

Alpert, J. and N. Hajaj, 2008. We knew the web was big. The Official Google Blog, http:// googleblog. blogspot.com/2008/07/we-knew-web-was-big.html#!/2008/07/we-knew-web-was-big.html

Fitzpatrick, B., 2004. Distributed caching with memcached. Linux J.,

Gulli, A. and A. Signorini, 2005. The indexable web is more than 11.5 billion pages. Proceedings of the 14th International Conference on World Wide Web, Special Interest Tracks and Posters. MAy 10-14, 2005, ACM, New York, USA., pp: 902-903.

Lerner, R., 2008. Speed up your site with Memcached. http://ostatic.com/blog/speed-up-your-site-with-memcached

Maesaka, T., 2008. Memcached binary protocol in a nutshell. http://www.slideshare.net/ tmaesaka/ memcached-binary-protocol-in-a-nutshell-presentation

Philippsen, M. and B. Haumacher, 1999. More efficient object serialization. Proceedings of the 11th IPPS/SPDP'99 Workshops with 13th International Parallel Processing Symposium and 10th Symposium on Parallel and Distributed Processing, April 12-16, 1999, San Juan, Puerto Rico, USA., pp: 718-732.

Shalom, N., 2010. Marrying memcached and NoSQL. http://natishalom.typepad.com/nati_shaloms_blog/ 2010/10/marrying-memcache-and-nosql.html

TechEncyclopedia, 2009. World wide wait. United Business Media. http://www.techweb.com/ encyclopedia/defineterm.jhtml

Voras, I. and M. Zagar, 2008. Web-enabling cache daemon for complex data. Proceedings of the ITI 2008 30th International Conference on Information Technology Interfaces, June 23-26, 2008, Dubrovnik, pp: 911-916.

WWWS, 2009. The size of the world wide web (The internet). http://www.worldwidewebsize.com/