ABSTRACT

The development of social network applications today has grown so fast with various intelligent media tools as a platform to run the application. One of the social network applications used by many users is the chat app. When the number of users accessing the chat service increases and the database server system can not cope with the amount of data so increased, this can certainly be a problem resulting in database server service disconnected data communications and excessive loads received by a single database server. So we need a load balancing method that can keep QoS chat service and can increase the scalability of the system by distributing the load into several database servers. The load balancing method used must be able to keep the communication going on by keeping in mind the existing session. It aims to improve the service for every request sent by users and servers. To support a strong load balancing system, it is tested by using scheduling algorithm, which is round robin algorithm and least connection algorithm. After that performed data replication on both database systems and performed a data backup system. Next is an evaluation of conditions before and after load balancing that has been implemented. The result of evaluation of load balancing method using least connection algorithm has average response time 10.797552 ms smaller than round robin algorithm which has average response time 12.546064 ms. Then for throughput, least connection algorithm also has bigger value that is 135.420512 kb / sec compared with round robin algorithm 134.282112 kb / sec. So in the results of testing and evaluation of least connection algorithms is better than round robin algorithm for the speed of response and the number of packets that can be passed. Then for a system with load balancing able to handle the connection as much as 5000 times with 500 requests / second, while without using the load balancing method is only able to handle the connection as much as 3000 times with 300 requests / second. This means that the load balancing method can improve the scalability of database server systems. In other conditions the system that implements fault tolerance has

been successfully applied. This is evident when the master balancer load fails then its role can be replaced by the slave load balancer within 2 seconds and when the balancer master load has recovered its role can be recovered from the slave balancer load within 2 seconds, so the system's availability can be maintained. Next is the database server system has managed to replicate data throughout the database and on the communication system database server has succeeded in performing data backup system on the third database server. As a result of the system development of load balancing and fault tolerance methods, the performance of services from social network applications can be improved and percentage to a system failure risk caused by the large number and overflow of data on one more database server can be minimized.

Keywords : Load balancer, fault tolerance, data replication, database server system.

INTISARI

Perkembangan aplikasi *social network* saat ini telah tumbuh begitu cepat dengan berbagai media piranti cerdas sebagai *platform* untuk menjalankan aplikasi tersebut. Salah satu aplikasi *social network* yang digunakan banyak *user* adalah aplikasi *chat*. Ketika jumlah pengguna yang mengakses layanan *chat* meningkat dan sistem *server* basis data tidak dapat mengatasi jumlah data yang begitu meningkat, tentu ini dapat menjadi masalah yang mengakibatkan layanan *server* basis data terputus komunikasi data dan beban berlebihan yang diterima oleh *server* basis data tunggal. Sehingga diperlukan metode *load balancing* yang dapat menjaga QoS layanan *chat* serta mampu meningkatkan skalabilitas dari sistem tersebut dengan mendistribusikan beban kedalam beberapa *server* basis data. Metode *load balancing* yang digunakan harus dapat menjaga agar komunikasi terjadi dapat terus terjalin dengan memperhatikan sesi yang ada. Hal ini bertujuan untuk meningkatkan pelayanan untuk setiap permintaan yang dikirim oleh *user* dan *server*. Untuk mendukung sistem *load balancing* yang kuat, maka dilakukan pengujian dengan menggunakan algoritme penjadwalan, yaitu algoritme *round robin* dan algoritme *least connection*. Setelah itu dilakukan replikasi data pada kedua sistem basis data dan dilakukan sebuah sistem *backup* data. Selanjutnya adalah dilakukan evaluasi kondisi sebelum dan sesudah *load balancing* yang telah diimplementasikan. Hasil dari evaluasi metode *load balancing* dengan menggunakan algoritme *least connection* memiliki rata-rata waktu respon 10.7975 ms lebih kecil dibandingkan dengan algoritme *round robin* yang memiliki rata-rata waktu respon 12.546 ms. Kemudian untuk *throughput*, algoritme *least connection* juga memiliki nilai yang lebih besar yaitu 135.420512 kb/sec dibandingkan dengan algoritme *round robin* 134.282112 kb/sec. Sehingga dalam hasil dari pengujian dan evaluasi algoritme *least connection* lebih baik dibandingkan dengan algoritme *round robin* untuk kecepatan respon dan jumlah paket yang dapat dilewatkan. Kemudian untuk sistem dengan *load balancing* mampu menangani koneksi sebanyak 5000 kali dengan 500 *request*/detik, sedangkan tanpa menggunakan metode *load balancing* hanya mampu menangani koneksi sebanyak 3000 kali dengan 300 *request*/detik. Artinya dengan metode *load balancing* dapat meningkatkan skalabilitas sistem *server* basis data. Pada kondisi yang lain sistem yang mengimplementasikan *fault tolerance* telah berhasil di terapkan. Hal ini terbukti ketika

load balancer master mengalami kegagalan maka perannya dapat digantikan oleh *load balancer slave* dalam waktu 2 detik dan saat *load balancer master* telah pulih kembali maka perannya dapat diambil kembali dari *load balancer slave* dalam waktu 2 detik, sehingga ketersediaan sistem dapat terus terjaga. Selanjutnya adalah sistem *server* basis data telah berhasil melakukan replikasi data seluruh basis data dan pada komunikasi sistem *server* basis data telah berhasil melakukan sistem *backup* data pada *server* basis data ke tiga. Alhasil dengan dibangunnya sistem metode *load balancing* dan *fault tolerance* maka kinerja layanan dari aplikasi *social network* dapat terus ditingkatkan serta persentase untuk sebuah resiko kegagalan sistem yang diakibatkan banyaknya dan meluapnya data pada *server* basis data yang lebih satu dapat diperkecil.

Kata kunci – *Load balancer, fault tolerance,* replikasi data, sistem *server* basis data.