ABSTRACT

The development of social network applications had been grown rapidly with a wide range of applications supported by smart devices as a platform to run them. One of the social network applications uses the chat application. When the number of clientaccess chat services increase then the server can not handle it so it will be an issue that resulted in the server service stopped due to excess load received by a single server. So, we need a method of load balancing to maintain QoS chat service and it can increase the scalability of the system to distribute the load in multiple servers. Load balancing method is using to keep the communication that occurs can continue interwoven with due regard to the existing session. The goal to improve services for every request it will send by the user. To support load balancing system become strong, it is using schedule algorithm testing between the round robin and least connection. After doing evaluation before and after condition load balancing implemantation. The comparison showed the average value of the least connection has a response time of 10.64 ms is smaller than the round robin which has a response time of 13,84 ms. for throughput, least connection also has a larger value ie 148.12 Kb / sec compared to the round robin which has a value of 147.98 Kb / sec. As the result the least connection algorithm is better than the round robin for the speed of response and the number of packages can be passing. So for the system to be able to handle the connection load balancing more than 5000 times with 500 requests/sec it means without load balancing is only able can handle the connection more than 3000 times with 300 requests/sec. So the load balancing system scalability can be improved. On the other hand the system fault tolerance has been success to implement. When the master load balancer fails then the role can be replaced by a load balancer slave in 3 seconds, while the load balancer master recover their role has taken back from the slave load balancer in 2 seconds, so the system availability can be maintain. As a result the system built load balancing and fault tolerance, the performance of the service chat application can

improve and the percentage of failures caused a failure of one server load balancing can minimize.

Keywords - load balancer, fault tolerance, server chat system.

INTISARI

Perkembangan aplikasi *social network* telah tumbuh begitu cepat dengan berbagai aplikasi yang didukung oleh piranti cerdas sebagai *platform* untuk menjalankanya. Salah satu aplikasi *social network* yang digunakan adalah aplikasi *chat*. Ketika jumlah pengguna yang mengakses layanan *chat* meningkat dan *server* tidak dapat mengatasinya, tentu ini dapat menjadi masalah yang mengakibatkan layanan *server* terhenti disebabkan adanya beban berlebih yang diterima oleh *server* tunggal. Sehingga diperlukan metode *load balancing* yang dapat menjaga QoS aplikasi *chatting* serta mampu meningkatkan skalabilitas dari sistem tersebut dengan mendistribusikan beban kedalam beberapa *server*. QoS terkait dengan pengujian parameter *throughput* dan waktu respon yang akan digunakan sebagai acuan dalam menilai kinerja layanan aplikasi *chatting*. Metode *load balancing* yang digunakan harus dapat menjaga agar komunikasi yang terjadi dapat terus terjalin dengan memperhatikan sesi yang ada. Hal ini bertujuan untuk meningkatkan pelayanan untuk setiap permintaan yang dikirim oleh pengguna. Untuk mendukung sistem *load balancing* yang kuat, maka dilakukan pengujian algoritma penjadwalan antara *round robin* dan *least connection*. Setelah itu dilakukan evaluasi kondisi sebelum dan sesudah *load balancing* diimplemantasikan. Dari hasil total rata-rata pengujian yang dilakukan *least connection* memiliki waktu respon lebih kecil yaitu 10,64 ms dibandingkan *round robin* yang memiliki waktu respon 13,84 ms. Kemudian untuk *throughput*, *least connection* memiliki nilai yang lebih besar yaitu 148,12 Kb/sec dibandingkan *round robin* yang memiliki nilai 147,98 Kb/sec. Sehingga algoritma *least connection* lebih baik dibandingkan dengan *round robin* untuk kecepatan respon dan jumlah besaran paket yang dapat dilewatkan. Dengan *load balancing*, sistem mampu menangani koneksi sebanyak 5000 kali dengan 500 *request*/detik. Sedangkan tanpa *load balancing* sistem hanya mampu menangani koneksi sebanyak 3000 kali dengan 300 *request*/detik. Artinya dengan *load balancing* skalabilitas sistem dapat ditingkatkan. Pada kondisi yang lain sistem yang *fault tolerance* telah berhasil diterapkan. Ketika *load balancer master* mengalami

kegagalan maka perannya dapat digantikan oleh *load balancer slave* dalam waktu 3 detik dan saat *load balancer master* pulih kembali maka perannya dapat diambil kembali dari *load balancer slave* dalam waktu 2 detik sehingga ketersediaan sistem dapat terus terjaga. Alhasil dengan dibangunnya sistem *load balancing* dan *fault tolerance* maka kinerja layanan dari aplikasi *chatting* dapat terus ditingkatkan serta persentase kegagalan yang diakibatkan kesalahan pada salah satu *server load balancing* dapat diperkecil.

Kata kunci -- *load balancer*, *fault tolerance*, sistem *server chatting*.