Load Balancing for Web Socket Connections

Paul Oldridge
Department of Computer Science
Faculty of Graduate Studies
University of Victoria

April 12, 2015

Intro

Load balancing is a well established best practice for scaling to handle large amounts of traffic in the client/server model. Both TCP and HTTP load balancing are common in production environments. Typically load balancing is used for short-lived connections, but Websockets are long-lived. In this paper we will explore how to gain the advantages of load balancing, for long-lived websocket connections.

Background

Load Balancing

Load balancing is simply any means of balancing traffic amongst multiple servers. It is used to enable horizontal scaling, which improves both scalability and fault tolerance. The simplest, and cheapest, form of load balancing is round robin DNS, which occurs during DNS lookup and therefore requires no additional hardware. Round robin DNS, works by configuring the nameserver to respond to each DNS request it receives with the addres of the next server in the pool. This distributes load to all the servers, but not necessarily in a fair way; one machine could receive several requests which take large amounts of processing, while another receives small requests which require almost no processing. The negative effects of this disappear as the number of servers increases.

Round Robin DNS. AWS Route 53 health checks.

Bryhni, Klovning, and Kure [4] stated that application layer load balancers are not scalable, since they can easily become the bottleneck. However, since 2000, there has been a major trend from web sites to web applications, where the main purpose of the server is no longer to server static files. Algorithm for balancing based on load.
Most load balancers are transparent to the client, so the client doesn't know whether they are connected to a load balancer or an actual server.

Advantages include SSL termination, better balancing algorithms, and application layer balancing configurations, such as sticky sessions.

Software Load Balancer (HAProxy, Nginx, Apache, Pound, Balance, node-proxy, etc.)

Physical Load Balancer (Cisco, Barracuda, RadWare, etc.)

ELB balances based on load.

Concerns have been raised about the design of the web socket protocol, in particular with regards to how they can achieve enterprise-grade scalability [3, 8].

Many production environments have scaled Redis 2.x [7]. Redis 3.0.0, which was just released ten days ago (April 1, 2015) [1], as of the time of this writing, includes Redis Cluster. A

distributed implementation of redis, that allows one to transparently horizontaly scale Redis.

Amazon Web Services Elastic Load Balancer, is a cloud based fully managed load balancing solution by Amazon, which uses physical load balancers managed by Amazon to scale to handle any amount of traffic (Amazon has the infrastructure in place to add capacity well before it is needed).

HTTP vs TCP

- HTTP is Stateless, and short live (longest keep-alive is usually 60 seconds).
  - Websockets are long lived TCP connections

Implementation

- node.js load balancer, balance individual messages, with extra auth mechanism / protocol, maybe even talking to same redis cluster as back-end nodes. Increase number of file descriptors

IP hashing is very fast[5].
socket.io and others recommend nginx IP hash mode. socket.io uses HTTP long polling as a SHIM for missing client-side websocket

Comparison

Direct Server Return: Load balancer is only used to pick the machine to form a connection to, then a direct connection is made from the client to the chosen server.

haProxy least connections mdoe (works for websockets but not socket.io)

websocket-benchmark

Future Work

Tri-tier, health-checks, connection

Bibliography

[1] Redis 3.0 release notes. `https://raw.githubusercontent.com/antirez/redis/3.0/00-RELEASENOTES`. Accessed: April 11, 2015.

[2] Socket.IO: Using multiple nodes. `http://socket.io/docs/using-multiple-nodes/`. Accessed: April 12, 2015.

[3] Amir Almasi and Yohanes Kuma. Evaluation of websocket communication in enterprise architecture, 2013.

[4] Haakon Bryhni, Espen Klovning, and Oivind Kure. A comparison of load balancing techniques for scalable web servers. *Network, IEEE*, 14(4):58–64, 2000.

[5] Zhiruo Cao, Zheng Wang, and Ellen Zegura. Performance of hashing-based schemes for internet load balancing. In *INFOCOM 2000. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, volume 1, pages 332–341. IEEE, 2000.

[6] Ian Fette and Alexey Melnikov. The websocket protocol. 2011.

[7] Tilmann Rabl, Sergio Gómez-Villamor, Mohammad Sadoghi, Victor Muntés-Mulero, Hans-Arno Jacobsen, and Serge Mankovskii. Solving big data challenges for enterprise application performance management. *Proceedings of the VLDB Endowment*, 5(12):1724–1735, 2012.

[8] Vanessa Wang, Frank Salim, and Peter Moskovits. *The definitive guide to HTML5 WebSocket*, volume 1. Springer, 2013.