Website Hardware Scaling

Asked 16 years, 3 months ago Modified 14 years, 9 months ago

Viewed 854 times

3

So I was listening to the latest Stackoverflow podcast (episode 19), and Jeff and Joel talked a bit about scaling server hardware as a website grows. From what Joel was saying, the first few steps are pretty standard:

1. One server running both the webserver and the database (the current Stackoverflow setup)
2. One webserver and one database server
3. Two load-balanced webservers and one database server

They didn't talk much about what comes next though. Do you add more webservers? Another database server? Replicate this three-machine cluster in a different datacenter for redundancy? Where does a web startup go from here in the hardware department?

hardware scaling

Share

Improve this question

Follow

edited Jan 18, 2021 at 12:38

Community Bot

1 • 1

asked Aug 31, 2008 at 1:11

Chris Upchurch

15.5k ● 6 ● 52 ● 66

6 Answers

Sorted by: Highest score (default)

10

A reasonable setup supporting an "average" web application might evolve as follows:

1. Single combined application/database server
2. Separate database on a different machine
3. Second application server with DNS round-robin (poor man's load balancing) or, e.g. Perlbal
4. Second, replicated database server (for read loads, requires some application logic changes so eligible database reads go to a slave)

At this point, evaluating the current state of affairs would help to determine a better scaling path. For example, if read load is high and content doesn't change too often, it might be better to emphasise caching and introduce dedicated front-end caches, e.g. Squid to avoid un-needed database reads, although you will need to consider how to maintain cache coherency, typically in the application.

On the other hand, if content changes reasonably often, then you will probably prefer a more spread-out solution; introduce a few more application servers and database slaves to help mitigate the effects, and use object

caching, such as memcached to avoid hitting the database for the less volatile content.

For most sites, this is probably enough, although if you do become a global phenomenon, then you'll probably want to start considering having hardware in regional data centres, and using tricks such as geographic load balancing to direct visitors to the closest "cluster". By that point, you'll probably be in a position to hire engineers who can really fine-tune things.

Probably the most valuable scaling advice I can think of would be to avoid worrying about it all far too soon; concentrate on developing a service people are going to want to use, and making the application reasonably robust. Some easy early optimisations are to make sure your database design is fairly solid, and that indexes are set up so you're not doing anything painfully crazy; also, make sure the application emits cache-control headers that direct browsers on how to cache the data. Doing this sort of work early on in the design can yield benefits later, especially when you don't have to rework the entire thing to deal with cache coherency issues.

The second most valuable piece of advice I want to put across is that you shouldn't assume what works for some other web site will work for you; check your logs, run some analysis on your traffic and profile your application - see where your bottlenecks are and resolve them.

 answered Aug 31, 2008 at 1:36

Follow

Rob
48.4k ● 5 ● 75 ● 93

3

[plenty of fish Architecture]

some interesitng videos:

[Youtube scalibility]

[Inteview with Dan Farino, System Architect at Myspace]

Share Improve this answer

Follow

answered Dec 16, 2008 at 23:10

Nikita Ignatov
7,163 ● 2 ● 37 ● 34

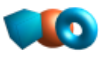
2

Joel mentioned adding a second datacenter, with the same setup, and then assigning your users randomly to each. Changes to the data are logged and sent from one location to the other, so that both locations contain all the data.

Share Improve this answer

Follow

answered Aug 31, 2008 at 1:18



Kibbee
66.1k ● 28 ● 144 ● 184

1

The talk Scalable Web Architectures Common Patterns & Approaches from Cal Henderson (Yahoo) on Web 2.0 Expo was quite interesting. I thought there was an video, but I could not find it. But here are the slides:

http://www.slideshare.net/techdude/scalable-web-architectures-common-patterns-and-approaches

Share Improve this answer

Follow

answered Aug 31, 2008 at 1:20

Peter Hoffmann

58.5k ● 15 ● 77 ● 59

1

A certain next step would be a cluster of webservers (a web farm) and a clustered system of database servers (replication or Oracle RAC etc. etc.)

Share Improve this answer

Follow

answered Aug 31, 2008 at 1:21

Andrei Rînea

20.7k ● 18 ● 121 ● 169

0

If your interested in caching and using .Net, look into the application caching block in enterprise library (of course use this along with the other points above).

Share Improve this answer

Follow

answered Mar 4, 2010 at 10:54

Mr Shoubs

15.4k ● 17 ● 71 ● 109