StackExchange Architecture Updates - Running Smoothly, Amazon 4x More Expensive

Monday, October 24, 2011 at 9:01AM

Todd Hoff in Example

We've had a few articles on the StackOverlflow Architecture and Stack Overflow Architecture Update - Now at 95 Million Page Views a Month. Time for another update. This time from a podcast. Every week or so Jeff, Joel and guests sit around and converse. The result is a podcast. In a recent podcast they talked about some of their recent architecture issues, problems, and updates. And since I wrote this article before my vacation, they've also published a new architecture update article: The Stack Exchange Architecture – 2011 Edition, Episode 1.

My overall impression is they are in a comfortable place, adding new sites, adding new features, making a house a home.

Notable for their scale-up architecture, you might expect with their growth that they would slam into a wall. Not so. They've been able to scale-up the power of individual servers by adding more CPU and RAM. SSD has been added in some cases. Even their flagship StackOverflow product runs on a single server. New machines have been bought, but very few.

So, the StackOverflow experiment shows the scale-up strategy for even largish sites is a good practice. True, their product naturally separates by topic, much like the early Facebook, but Moore's law and quality engineering are your friends. They estimate Amazon would cost them 4 times much.

Here's what StackExchange has been up to:

- They have a rack in Oregon and two cages at Peer1 in NY
  - The cages are very cramped, especially with lots of cabling for redundant power and networking.
    - Should have bought a rack instead of cages at Peer1. That would give more room.
  - A NY presence moved them closer to their European users.
  - Kept the Oregon location because it was cheap and it's convenient to have another location.
  - Check that power doesn't all failover to a single power source which will fail if a failure actually happens
  - 10 servers in New York. 9 production servers. 1 QA server.
  - Stack Overflow has a big server, SSDs, and more RAM. They've added more processors and more RAM on some machines so they could run Lucene
  - Oregon runs the data explorer and chat. If you can't get to NY you can still chat. The stack exchange blog is run in Oregon and all others are in NY.
- Considered having a failover mode so that all the NY traffic could fail over to Oregon in a read-only mode. You couldn't ask new questions, but since much of the value of the site is in reading, a read-only site has value.
  - They have this read-only mode for:
    - When they moved Stack Overflow to its own database server.
    - When they moved from Oregon to NY.
  - Having an active-active architecture where both sites can handle writes is just way too complicated.
  - You don't know how long the connection to the primary server will be down. It can take a long time to figure out what's

wrong. So going to read-only mode on the backup server is the right thing to do. If you go into read-write mode on the backup server then you'll be more reluctant to go back to the main server until you are sure it's completely dead.
    - Everyplace in the code they write to the database they implement a read-only mode.
- A health dashboard gives the overall status of all their systems. Queried from a SQL database. Having everything stored in SQL makes it much easier to build tools around and scales well enough for now for their traffic loads.
    - Three graphs: CPU, memory, network.
    - The 10 servers that run the bulk of the network are barely loaded. One is peaking 16%. Even if doubled they are massively over provissioned.
    - They weren't over provisioned at one time.
        - The servers have been reconfigured quite a bit.
        - They've added CPUs to the web tier. They were peaking at 60% which was getting uncomfortable.
        - Added SSDs to the web tier so they can:
            - Increase Lucene indexing speed.
            - Speed up boot times for when they cycle the web tier.
            - SSD is a little early in the life cycle to be a no-brainer.
            - SSD shouldn't make your machine run faster because you should be running in RAM already. Doesn't apply when your database is too large to fit in RAM. Hopefully your most active data can fit in RAM.
- Use a commercial Orion network monitor. Essentially paying for time as it's easier to setup and use than Nagios.
    - Keeps all data on SQL server.

    - SQL means it's easy to access and query.
    - Web logs are stored in SQL server too.
- Long term trend towards having all data in SQL Server.
    - Bought a heavy duty server just to hold all this data and be able to perform real-time querying. 2 processors, lots of RAM. Put in 13TB of storage, which is a year of logs files. 20 gigs a day goes into SQL server.
    - Helped because they can tell what is happening in real-time. When they roll out a new feature they can tell how much it is used which tells them if they need to keep the feature or not. They can do this with a simple query instead of looking at logs or going to Google Analytics.
    - They create 1 table a day to store stats.
    - They all know SQL so they can ask the questions they want answered with relative ease.
- Redis Servers
    - Used as a shared state cache between the 10 servers. Requests can be load balanced across the servers. They don't want to hit the database every time so it goes into a cache.
    - Still a network hit. They will do sticky IP stuff so requests can use an in-memory cache on the web server. This removes the network hit, which is much much faster, serve it directly out of memory.
    - Have a network scalability issues when you put everything in cache. It's not an infinitely fast pipe.
- Network Apocalypse
    - Had a problem with microbursting. Burst up to gigabit speeds for very short periods of time which filled up the queues.
    - Never found a root cause.
    - Changed the way the network was architected. All network was centrally routed. The web servers and the database/redis servers were on different networks so they got rid of the router

  - and firewalls between the two sides.
  - They changed the NIC configuration. They were teamed for failover. They changed the configuration to active-active mode and load balanced requests over the NICs. No longer had single networks cards which they hoped would prevent the microbursts.
  - What will drive up costs is if they have run 10 network lines from the switch because you pay per port.
- Surprisingly their traffic mix 40% read and 60% write.
- 60% of their audience is international.
- About 45 people work at StackExchange.
- A CDN gets you 80% of the way for handling international users as apposed to running in multiple zones.
  - Have a CDN server most of the static resources.
  - Live data in two datacenters is too difficult. Would require enough DBAs to be covered 24x7 plus a lot of code changes in support.
- Going to two datacenters might require a client side engine, like Facebook.
  - Client talks to 10 servers and says do you have anything for me.
  - The client side merges the data and shows some of the results.
  - An HTML page is a few divs. Each div independently goes out gets that part of the page. Each part of the page could come off a different server and it's all pulled together client side.
  - StackExchange isn't highly customized, users see roughly the same site, so it's not that big a win.
- Moving to Amazon is not a good idea:
  - Can't trust Amazon with their outages.
  - Amazon would be more than 3x more expensive. Amazon would cost $17K a month and they pay $4K a month now,

after the initial hardware purchases.

- Amazon has old year 2007 technology so they would need a lot more servers than they have now to perform similarly.
- Expect to get 5 years of service from each computer. Older computers will be put onto utility or lesser tasks.
- Would they need another sysadmin if they went to Amazon? Probably not they think. Amazon has a lot of problems with random network latency that can't be tracked down. For a high volume outfit it's not all roses.
- For control freaks it's not the best model. They want to know everything that's going on in the network. They watch performance constantly, tweak NIC configurations, etc. Not being able to control everything wouldn't fit with how they want to work.
- Want the best hardware and care about every little detail. When you are going to Amazon you are saying I don't care at some level.
- The cloud is a redundancy story. It's like buying a thousand hamburgers and not caring if 5 are rancid. That's not the model they've chosen.
- They have a big iron story, especially on the database side. Big iron is not dead. Moore's law is not dead. You don't actually have to spend that much to get a more capable server. That continually marches on.
- The only thing they really worry about is if they run out of RAM on the database. They have separate databases by topic, but Stack Overflow is so big it fills up one machine and they have no way to shard it. They could get a server with 256GB of RAM.

StackExchange is refreshingly open. They consciously try and make their issues public by posting in their blog or asking questions on one of their

sites. Maybe your organization might want to follow their example? It's an awesome way to build awareness and trust.
