

Set Amazon's Servers on Fire, Not Yours



Parks Hall Fire, July 3, 2002 - <http://www.acadweb.wvu.edu/dbrunner/>

Why trust us?

- ☺ Bootstrapped.
- ☺ Profitable.
- ☺ No debt.
- ☺ 140M photos.
- ☺ 192TB at S3.
- ☺ Doubling yearly.



Why trust us?

- ☺ Bootstrapped.
- ☺ Profitable.
- ☺ No debt.
- ☺ 140M photos.
- ☺ 192TB at S3.
- ☺ Doubling yearly.
- ☺ Super Heroes.





SmugMug's Founders

Our Love Affair with S3



It's no secret that we love S3. But, like all good love affairs, it has it's ups and downs. :)

Our Love Affair with S3

- ☺ Always on, global, infinite storage.
- ☺ Inexpensive. \$0.15/GB/month w/replicas.
- ☺ Easy. REST API. (SOAP too, but...)
- ☺ Fast. Not 15K-SCSI fast, but Internet fast.
- ☺ Game changer.

Amazon? Infrastructure?

SmugMug 



Photo by Bob Knight - <http://bobknight.smugmug.com/>

Amazon? Infrastructure?

- ☺ Started with books.
- ☺ Soon added CDs & DVDs.
- ☺ Toys R Us, Borders, Target.
- ☺ zShops, Marketplace, E-Commerce API
- ☺ People building their businesses on Amazon is cool.
- ☺ What else do we have lurking in the corners?

Why use them?

- ☺ Not a lot of web-scale expertise on Planet Earth.
- ☺ Reputation for systems.
- ☺ Once competed with Amazon - fatbrain {✱}
- ☺ They eat their own dogfood. Dozens of products.
- ☺ Focus on the app, not the muck.

Show me the money!



Photo by Kirk Tanner - <http://kirktanner.smugmug.com/>

Show me the money!

- ☺ Guesstimate: ~\$500K saved per year.
- ☺ Actual:
 - ☺ Growth: 64M photos -> 140M photos
 - ☺ Disks would cost: \$40K -> \$100K/month.
 - ☺ \$922K would have been spent.
 - ☺ \$230K spent instead.
 - ☺ **\$692K in cold, hard savings.**
- ☺ Nasty taxes! \$295K 'saved' in cash flow. Bonus!
- ☺ Reselling disks - recouping sunk cost.

- ☺ Perfect for startups & small companies.
- ☺ Ideal for ‘store lots, serve little’ businesses of all sizes.
- ☺ Not so great (yet?) for serving lots if you’re a medium or large sized business. Transfer costs high if you can buy bandwidth in 1 Gbps+ chunks.
- ☺ We’re a ‘store lots, serve lots’ company. What to do?

5 of my employees.

Me with my NeXT gear on.



- ☺ Architecture remarkably similar to SmugFS.
- ☺ Similar to lots of startups.
- ☺ Stupid we're all building the same thing.
- ☺ Easy to drop-in.
- ☺ Started on Monday, live in production on Friday.

Our S3 evolution

- ☺ Started just doing secondary storage. Too cold!
- ☺ Tried out as Primary. Too hot!
- ☺ Finally, hot & cold model = Just right!
- ☺ Amazon gets 100% of the data.
- ☺ SmugMug keeps “hot” data local.
- ☺ 95% reduction in # of disks bought.

Sample Request

- ☺ Client 'Smuggy' -> www.smugmug.com
 - ☺ “Hey, gimme photo 31337”
- ☺ www.smugmug.com -> SmugFS
 - ☺ “Hey, you got photo 31337?”
 - ☺ If YES, send to Smuggy.
 - ☺ If NO:
 - ☺ Log that it wasn't in SmugFS for analysis.
- ☺ www.smugmug.com -> Amazon S3
 - ☺ “Hey, you got photo 31337?”
 - ☺ If YES, send to Smuggy.
 - ☺ If NO:
 - ☺ PANIC! :)

Proxy vs Redirect vs Direct Links

- Built SmugMug->S3 with multiple modes.
- Can flip a switch to change.
- Nearly 100% served are proxy reads.
- Sometimes HTTP redirects.
- Rarely direct S3 links.

- ☺ We have complicated permissions.
- ☺ Passwords, privacy, external links, oh my!
- ☺ Proxying allows strong protection.

REST vs SOAP

- ☺ Love REST, hate SOAP.
- ☺ Lightweight.
- ☺ Nothing useful added with SOAP's complexity.

- ☺ Not 100%. Close, though.
- ☺ More reliable than SmugFS which is quite reliable.
- ☺ Lots of failure points:
 - ☺ SmugMug's datacenter
 - ☺ Internet backbones
 - ☺ Amazon's datacenter
- ☺ No other software, hardware, or service we use is 100%, either.

Handling failure

- ☺ Build from day one with failure in mind.
- ☺ Stuff breaks - try again.
- ☺ Writes fail? Write locally, sync later.
- ☺ Reads fail? Handle intelligently. Alerts?

- ☺ Fast for reads and writes. (XX Mbps)
- ☺ Mostly speed-of-light limited. (20-80ms)
- ☺ Parallel i/o for massive throughput. (XXX Mbps)
- ☺ Machine measurable, human indistinguishable.

- ☺ S3 isn't a Content Delivery Network.
- ☺ It's storage.
- ☺ No global locations (yet?).
- ☺ Limited edge caching.
- ☺ Future Amazon Web Service?

Store-and-forward vs Stream

☺ Two ways to serve your content.

☺ Store-and-forward

☺ Great resiliency.

☺ Poor performance (TTFB).

☺ Stream

☺ Poor resiliency.

☺ Great performance (TTFB)

☺ Do a quick HEAD first to verify.

The Speed of Light Problem

- ☺ Amazon hasn't solved faster-than-light data transmission. Yet.
- ☺ Unavoidable - make sure your app can deal.
- ☺ Parallelized i/o can mask problem.
- ☺ Caching can help.
- ☺ Streaming can help.

Outages & Problems

- ☺ Not perfect. 5 major issues.
- ☺ 3 outages (15-30 mins). 2 core switch failures and one DNS problem. Amazon.com affected.
- ☺ 2 performance degradations. One, our customer noticed. Second, they didn't.
- ☺ Not a big deal - everything fails. Expect it.

- ☺ We don't care about SLA, but you may.
- ☺ Service Support: One area where Amazon is weak.
 - ☺ This is a utility.
 - ☺ They need a service status dashboard.
 - ☺ Pro-active customer notifications.
 - ☺ Ability to get ahold of a human.
- ☺ Amazon.com's customer service is good, AWS will likely catch up.

Saving our butts

- ☺ Knocked power out of ~70TB of storage. Oops!
- ☺ Moved datacenters during normal business hours, customers not affected.
- ☺ Stupid bugs.

- ☺ Use cURL
 - ☺ Faster.
 - ☺ More reliable.
 - ☺ Storing vs Streaming is simple.
- ☺ Make stuff as async as possible
 - ☺ Hides speed-of-light issue
 - ☺ Hides or masks problems
 - ☺ Fast customer response

Flirting with the other services.



Elastic Compute Cloud (EC2)

- ☺ Like S3, only for compute.
- ☺ Scale up or down via API.
- ☺ Web servers, processing boxes, development test beds, build servers, etc. You name it.
- ☺ Launching large EC2 implementation “soon”
 - ☺ Image processing.
 - ☺ 500K-1M photos/day.
 - ☺ 10-20 Terapixels/day processed
 - ☺ Peak traffic on weekends, holidays
 - ☺ Ridiculously parallel

Simple Queue Service (SQS)

- ☺ Simple, reliable queuing.
- ☺ Mates well with EC2 & S3
 - ☺ Stick jobs in SQS
 - ☺ Retrieve jobs with EC2 instances using S3 data
 - ☺ Run jobs, report status to SQS.
- ☺ \$0.10/1000 items
 - ☺ Priced well for small projects.
 - ☺ Gets costly for huge ones (millions+).

- ☺ Database API or DB grade EC2 instances.
 - ☺ Fast (lots of local spindles, lots of RAM)
 - ☺ Persistent.
- ☺ Load balancer API.
 - ☺ Single IP in front of lots of EC2 instances.
 - ☺ Programmable to add/remove/change clusters.
 - ☺ Can be done with software on an EC2 instance, but painful.
- ☺ CDN

Questions?

SmugMug ☺



☺ **Blog:** <http://blogs.smugmug.com/onethumb>

☺ **Slides:** See the blog. Posting soon.

☺ **Email:** don AT smugmug

☺ **Twitter:** <http://twitter.com/DonMacAskill>

☺ **Photo sharing:** <http://www.smugmug.com/>

☺ **Thanks!**