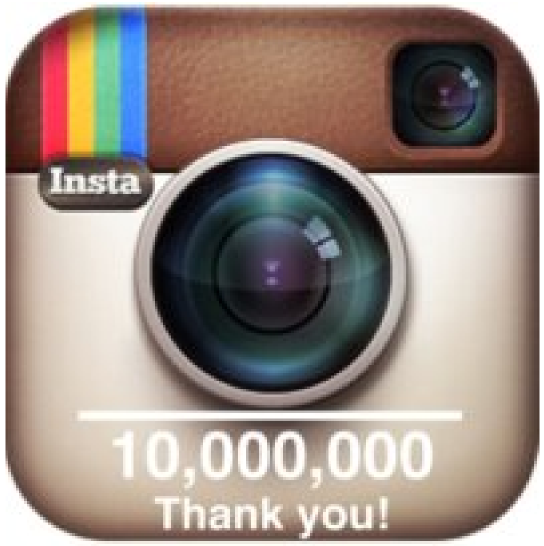
The Instagram Architecture Facebook Bought for a Cool Billion Dollars

Monday, April 9, 2012 at 12:49PM

Todd Hoff



It's been a well kept secret, but you may have heard Facebook will Buy Photo-Sharing Service Instagram for $1 Billion. Just what is Facebook buying? Here's a quick gloss I did a little over a year ago on a presentation Instagram gave on their architecture. In that article I called Instagram's architecture the "canonical description of an early stage startup in this era." Little did we know how true that would turn out to be. If you want to learn how they did it then don't take a picture, just keep on reading...

Instagram is a free photo sharing and social networking service for your iPhone that has been an instant success. Growing to 14 million users in just over a year (now 30 million users), they reached 150 million photos in August while amassing several terabytes of photos, and they did this with just 3 Instaneers, all on the Amazon stack.

The Instagram team has written up what can be considered the canonical description of an early stage startup in this era: What Powers Instagram: Hundreds of Instances, Dozens of Technologies.

Instagram uses a pastiche of different technologies and strategies. The team is small yet has experienced rapid growth riding the crest of a rising social and mobile wave, it uses a hybrid of SQL and NoSQL, it uses a ton of open source projects, they chose the cloud over colo, Amazon services are highly leveraged rather than building their own, reliability is through availability zones, async work scheduling links components together, the system is composed as much as possible of services exposing an API and external services they don't have to build, data is stored in-memory and in the cloud, most code is in a dynamic language, custom bits have been coded to link everything together, and they have gone fast and kept small. A very modern construction.

We'll just tl;dr the article here, it's very well written and to the point. Definitely worth reading. Here are the essentials:

- Lessons learned: 1) Keep it very simple 2) Don’t re-invent the wheel 3) Go with proven and solid technologies when you can.
- 3 Engineers. (They now reportedly have 13 employees, remember this was awhile back)
- Amazon shop. They use many of Amazon's services. With only 3 engineers so don’t have the time to look at self hosting.
- 100+ EC2 instances total for various purposes.
- Ubuntu Linux 11.04 (“Natty Narwhal”). Solid, other Ubuntu versions froze on them.
- Amazon’s Elastic Load Balancer routes requests and 3 nginx instances sit behind the ELB.
- SSL terminates at the ELB, which lessens the CPU load on nginx.
- Amazon’s Route53 for the DNS.
- 25+ Django application servers on High-CPU Extra-Large machines.
- Traffic is CPU-bound rather than memory-bound, so High-CPU Extra-Large machines are a good balance of memory and CPU.
- Gunicorn as their WSGI server. Apache harder to configure and

more CPU intensive.

- Fabric is used to execute commands in parallel on all machines. A deploy takes only seconds.
- PostgreSQL (users, photo metadata, tags, etc) runs on 12 Quadruple Extra-Large memory instances.
- Twelve PostgreSQL replicas run in a different availability zone.
- PostgreSQL instances run in a master-replica setup using Streaming Replication. EBS is used for snapshotting, to take frequent backups.
- EBS is deployed in a software RAID configuration. Uses mdadm to get decent IO.
- All of their working set is stored memory. EBS doesn’t support enough disk seeks per second.
- Vmtouch (portable file system cache diagnostics) is used to manage what data is in memory, especially when failing over from one machine to another, where there is no active memory profile already.
- XFS as the file system. Used to get consistent snapshots by freezing and unfreezing the RAID arrays when snapshotting.
- Pgbouncer is used pool connections to PostgreSQL.
- Several terabytes of photos are stored on Amazon S3.
- Amazon CloudFront as the CDN.
- Redis powers their main feed, activity feed, sessions system, and other services.
- Redis runs on several Quadruple Extra-Large Memory instances. Occasionally shard across instances.
- Redis runs in a master-replica setup. Replicas constantly save to disk. EBS snapshots backup the DB dumps. Dumping on the DB on the master was too taxing.
- Apache Solr powers the geo-search API. Like the simple JSON interface.
- 6 memcached instances for caching. Connect using pylibmc & libmemcached. Amazon Elastic Cache service isn't any cheaper.

- Gearman is used to: asynchronously share photos to Twitter, Facebook, etc; notifying real-time subscribers of a new photo posted; feed fan-out.
- 200 Python workers consume tasks off the Gearman task queue.
- Pyapns (Apple Push Notification Service) handles over a billion push notifications. Rock solid.
- Munin to graph metrics across the system and alert on problems. Write many custom plugins using Python-Munin to graph, signups per minute, photos posted per second, etc.
- Pingdom for external monitoring of the service.
- PagerDuty for handling notifications and incidents.
- Sentry for Python error reporting.

And now you know the secret to getting bought for a billion dollars... getting your architecture written up on HighScalability!

Related Articles

- Scaling Instagram - AirBnB Tech Talk 2012 by Mike Krieger of Instagram
- Storing hundreds of millions of simple key-value pairs in Redis
- Simplifying EC2 SSH Connections
- Sharding & IDs at Instagram
- Membase Cluster on EC2 or Amazon ElastiCache? by Alex Popescu

---

Article originally appeared on High Scalability (http://highscalability.com/).

http://highscalability.com/blog/2012/4/9/the-instagram-architecture-facebook-bought-for-a-cool-billio.html

See website for complete article licensing information.