Case Study: Instagram

Abstract. Instagram that is owned by Facebook Inc. since 2012, also known as IG or Insta is a photo and video sharing service. Instagram allows its users to upload their photos and videos which can be further edited by its filters. It also manages tags and location information related with the photos and videos. Other than sharing photos and videos, Instagram also provide different features to its users like IGTV, Instagram Direct, Instagram Stories and Explore. Instagram mainly uses two backend database system that are PostgreSQL and Cassandra. Both the systems have mature replication frameworks and they work well as a globally consistent data store. In 2013, about 200M people were using Instagram and storing 20B photos every month. With no slow-down in sight, Instagram moved from AWS servers to Facebook's infrastructure. In 2015, the figure increased to 400M users and around 40B photos and videos storing every month, serving over a million requests per seconds. After that to keep supporting this growth, and to make sure the community has a reliable experience on Instagram, Instagram decided to scale their infrastructure geographically. To meet their goals Instagram is keep adding new things such as Artificial Intelligence to their application.

Description. Instagram mainly uses two backend database systems: PostgreSQL and Cassandra. Both PostgreSQL and Cassandra have mature replication frameworks that work well as a globally consistent data store. It uses a hybrid of SQL and NoSQL databases.

Global data neatly maps to data stored in these servers. The goal is to have eventual consistency of these data across data centres, but with potential delay. Because there are vastly more read than write operations, having read replica each region avoids cross data centre reads from web servers.

Writing to PostgreSQL, however, still goes across data centres because they always write to the primary.

Technologies Used. Instagram's maximum data lies in PostgreSQL. Instagram found that Amazon's network disk system (EBS) does not support enough disk seeks per second, so having all of their working set in memory is extremely important. To get reasonable IO performance, they set up their EBS drives in a software RAID using mdadm. After that Instagram found vmtouch as a good tool for managing what data is in memory, especially when failing over from one machine to another where there is no active memory profile already. All of their PostgreSQL instances run in a master replica setup using Streaming Replication, and they use EBS snapshotting to take frequent backups of their systems. They use XFS as their file system, which lets them freeze and unfreeze the RAID arrays when snapshotting, in order to guarantee a consistent snapshot. The photos themselves go straight to Amazon S3, which currently stores several terabytes of photo data for us. They use Amazon CloudFront as their CDN, which helps with image load times from users around the world.

Whenever there comes a request to Instagram servers, it goes through load balancing machines. Instagram run 2 nginx machines and DNS Round-Robin between them. The downside approach is the time it takes for DNS to update in case one of the machines need to get decommissioned. Recently, Instagram moved to Amazon's Elastic Load Balancer, with

NGINX instances behind it that can be swapped in and out. Instagram also terminates their SSL at the ELB level, which lessens the CPU load on nginx. Instagram use Amazon's Route53 for DNS, which they have recently added a GUI tool for in the AWS console.

Instagram initially used Elasticsearch for its search feature but later migrated to Unicorn, a social graph aware search engine built by Facebook in-house.

Unicorn powers search at Facebook & has scaled to indexes containing trillions of documents. It allows the application to save locations, users, hashtags etc and the relationship between these entities.

Speaking of the Insta's search infrastructure it has denormalized data stores for users, locations, hashtags, media etc.

These data stores can also be called as documents, which are grouped into sets to be processed by efficient set operations such as AND-OR & NOT

The search infrastructure has a system called Slipstream which breaks the user uploaded data, streams it through a Firehos*e* & adds it to the search indexes.

The data stored by these search indexes is more search-oriented as opposed to the regular persistence of uploaded data to PostgreSQL DB.

Instagram run Djangoon Amazon High-CPU Extra-Large machines, and as their usage grows, they have gone from few of them to 25 of them. Instagram use gunicorn.org as their WSGI server and use mod_wsgi and Apache, but found Gunicorn was much easier to configure, and less CPU-intensive. To run commands on many instances at once like deploying code they use Fabric, which recently added a useful parallel mode so that deploys take a matter of seconds.

On Instagram, when a user decides to share out an Instagram photo on other applications such as Twitter or Facebook, or when Instagram needs to notify one of their Real-time subscribers of a new photo posted, it push that task into German, which is a task queue system. Doing it asynchronously through the task queue means the media uploads can finish quickly, while the 'heavy lifting' can run in the background.

The photos shared by users go straight to Amazon S3, which currently stores several terabytes of photo data. Instagram use Amazon CloudFront as their CDN, which helps with image load times from users around the world

Challenges. There were some challenges in implementing these technologies. A big challenge was caching. The cache layer is the web servers' most frequently accessed tier, and they need to be collocated within a data center to avoid user request latency. This means that updates to cache in one data center are not reflected in another data center, therefore creating a challenge for moving to multiple data centers. For example, suppose a user commented on a newly posted photo. In the one data center case, the web server that served the request can just update the cache with the new comment. A follower will see the new comment from the same cache. In the multi data center scenario, however, if the commenter and the follower are served in different regions, the follower's regional cache will not be updated and the user will not see the comment**.** To overcome this situation Instagram started using PgQ and enhance it to insert

cache invalidation events to the databases that are being modified. Also they reduced computational resources needed for each read by denormalizing counters and reduced the number of reads by using cache leases.

Conclusion. In 2011, Instagram began as a social photo sharing app that let users take photos, apply filters, and share them with friends. While they have evolved over the years, that simple idea is still at the center of what they do today. The rise of the camera phone was crucial to their existence and early success. Their well thought out information architecture and social strategy facilitated their early growth and enabled them to rapidly grow and emerge as a market leader in the photo sharing landscape.

Submitted By:

Hardik Sharma (B)

BETN1CS18033