Glide.me Leverages Redis Cloud to Scale Their 200G In-Memory Database

Case Study, May 2014

"It's the quietest part of my infrastructure. It's the part that just works. It keeps all the data and it scales without me having to worry about it."
Max Rabin, Head of Server Development at Glide

redislabs

About Glide

Glide is the world's first mobile video instant messenger. Utilizing a massive real-time network, Glide users are able to send text and video messages via live stream, whilst granting recipients the flexibility to read or watch the messages in real-time or at a later convenience.

Hosted on AWS and Redis Cloud

Leveraging Amazon Web Services' (AWS) broad range of cloud services, Glide heavily relies on the cloud giant for backend hosting. The application maximizes its capabilities by utilizing AWS' Elastic Compute Cloud (EC2), Simple Queue Service (SQS), Route 53, CloudFormation, and CloudWatch. All of the videos are stored in Simple Storage Service (S3) and secure access is managed by Identity and Access Management (IAM). Data is stored in MongoDB and Redis, with the Redis dataset hosted on Redis Cloud, Redis Labs' singular platform. Glide uses Redis Cloud to store sessions, some transient data, and to cache specific types of data. This revolutionary mobile application is available on Android and IOS operating systems.

Background

Glide's platform uses the Redis in-memory database to deliver consistent real-time performance for millions of users. The Redis dataset is hosted and managed by Redis Labs on the **Redis Cloud** platform. Most hosted services offer standard cloud instances pre-loaded with open-source Redis. This approach does not tackle the operational difficulty of running Redis on the cloud and does not provide a great advantage over the do-it-yourself approach. Redis Cloud overcomes these limitations by adding a breakthrough technological layer to open source Redis, while fully managing it. The technology virtualizes multiple cloud servers into an infinite pool of memory, consumed by users according to the actual size of their datasets.

Challenges and Solutions

1 - Challenge: Real-Time Reminders and Analytics

Every time a Glide user receives a message, a reminder is created requiring a push notification to be created in real-time. Consequently, instances are scaled up and down, requiring in-memory storage to be persistent across servers. In addition, aggregated analytics ought to be made in real-time, in efforts to support admin performance reports of the system.

Solution

Redis Labs' platform enables Glide with the ability to store and share the number of unread and missed messages with users right on their homescreen. These message reminders are stored as a number in Redis as opposed to querying the MongoDB database over and over again. In addition, Redis is used to store the list of users who have missed messages along with the actual messages they have missed, facilitating the reminder mechanism and generating efficient database operations.

Aggregated analytics are used to illustrate an overall view of how many times certain actions have occurred: the number of messages created, types of messages (text or video), group chats or one-on-one, registrations made, and logins**.** The activity in the server is incremented to a counter in Redis, which later persists to a backend database for viewing in the admin panel, where charts and graphs are used to illustrate the data. Everything is incremented to Redis so a heavy strain is not placed on the backend database, avoiding the creation of excess write requests.

2 - Challenge: Redis Storage Capacity

One challenge that developed as the result of a client-side bug, had caused session tokens to be corrupted, which resulted in new login requests for every interaction. Due to the fact that every interaction was considered a new session, it manifested as a "memory leak" in Redis. This led to the generation of login requests without the actual sessions, reaching hundreds of gigabytes of storage for the sessions' data.

Nearly reaching the 200GB threshold set for one of their Redis databases, Glide knew they could potentially have a serious problem on their hands. Unnecessary data volume growth at staggering and unproportional rates was certainly not an option.

Solution

The bug was resolved in two stages. As an initial short-term solution, the sessions were moved to a new database in order to isolate the issue. Only then, once the uncontrolled data growth was isolated in a separate database, was the faulty application logic addressed. This took care of the bug, as well as drastically reducing the amount of space needed for session storage. Redis Cloud professionals guided Glide through the process of splitting up the sessions from the rest of the data within the Redis Cloud. By writing a Lua script they managed to migrate data from one database to another, with minimal impact to their application's service.

With Redis Cloud, scaling is performed automatically — the shards autonomously inflate, deflate, multiply or reduce according to the dataset size and the measured performance of each shard. A dataset can continuously grow from a few megabytes to hundreds gigabytes overcoming the Redis scalability limitation. Redis Cloud is completely ‘zero touch’. A Redis database is created in seconds, and from that moment on, all operations are fully-automated.

Glide now has one database for sessions, and one for user data, including missed messages, badges and cached items. While the user database is creeping up on the 200GB mark, by simply requesting additional space, Redis Labs’ team had made sure that the Glide Redis database can scale up to 400GB, providing plenty of room for future growth.

3 - Challenge: Pubnub Message Counter

Glide uses Pubnub to facilitate the real-time instant messaging aspect of the application. Every Glide client is subscribed to Pubnub, which prides itself in delivering messages within a quarter of a second to recipients anywhere in the world. Issues have arisen in regards to the large volume of Glide messages created daily, much to everyone's surprise, due to the fact that the sending limits are obscenely large.

Solution

Pubnub requested to see the number of messages sent per minute in efforts to solve the glitch. Therefore, every time a message is sent to Pubnub, a counter is incremented in Redis in a hash called ‘Pubnub’. Every key contains a value calculated by its time-stamped unit divided by 60, which is incremented. Despite this substantial and unplanned increase in the workload, with Redis Cloud, our Redis is continuing to perform above par, making Glide’s task of tracking messages that much easier for Pubnub metrics.

About Redis Cloud

Redis Labs offers enterprise-class **Redis** and **Memcached** for developers. Our fully-managed cloud services - Redis Cloud and Memcached Cloud - deliver top performance in a highly-available, infinitely scalable, predictable and stable manner. Developers are free of dealing with nodes, clusters, scaling, data-persistence and failure recovery, while providing true auto-scalability and instant failover.

We power tens of thousands of customer apps in multiple cloud environments and enhance use cases such as real-time analytics, social app functionality, job management, and geo-search.

Redis Labs has more than 1,900 paying customers and has raised $13M in venture funding from Bain Capital Ventures, Carmel Ventures, and others.