Scaling the Messages Application Back End

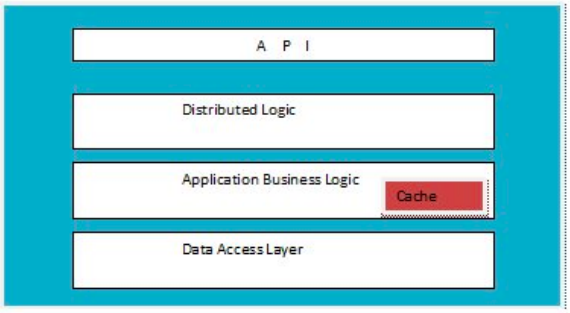
By Prashant Malik on Tuesday, April 12, 2011 at 11:21am

Facebook Messages seamlessly integrates many communication channels: email, SMS, Facebook Chat, and the existing Facebook Inbox. Combining all this functionality and offering a powerful user experience involved building an entirely new infrastructure stack from the ground up.

To simplify the product and present a powerful user experience, integrating and supporting all the above communication channels requires a number of services to run together and interact. The system needs to:

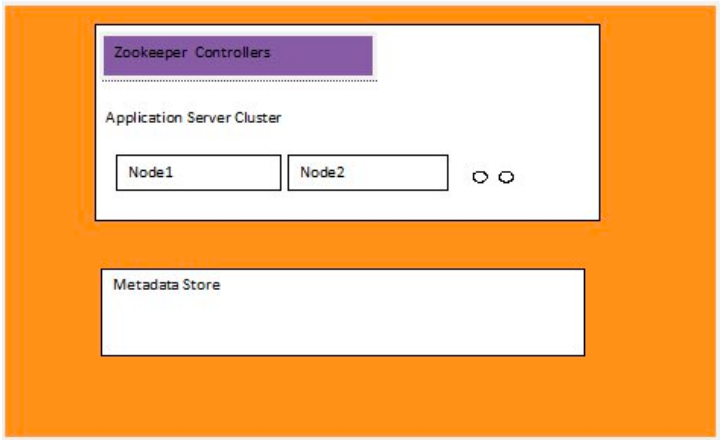
- Scale, as we need to support millions of users with existing message history.
- Operate in real time.
- Be highly available.

To overcome all these challenges, we started laying down a new architecture. At the heart of the application back end are the application servers. Application servers are responsible for answering all queries and take all the writes into the system. They also interact with a number of services to achieve this.



Each application server comprises:

- **API:** The entry point for all get and set operations, which every client calls. An application server is the sole entry point for any given user into the system. Any data written to or read from the system needs to go through this API.
- **Distributed logic:** To understand the distributed logic we need to understand what a cell is. The entire system is divided into cells, and each cell contains only a subset of users. A cell looks like this:



Facebook Engineering

Notes by Facebook Engineering

All Notes

Get Notes via RSS

Embed Post

Understanding Cells

Cells give us many advantages:

- They help us scale incrementally while limiting failure scenarios
- Easy upgrades
- Metadata store failures affect only a few users
- Easy rollout
- Flexibility to host cells in different data centers with multi-homing for disaster recovery

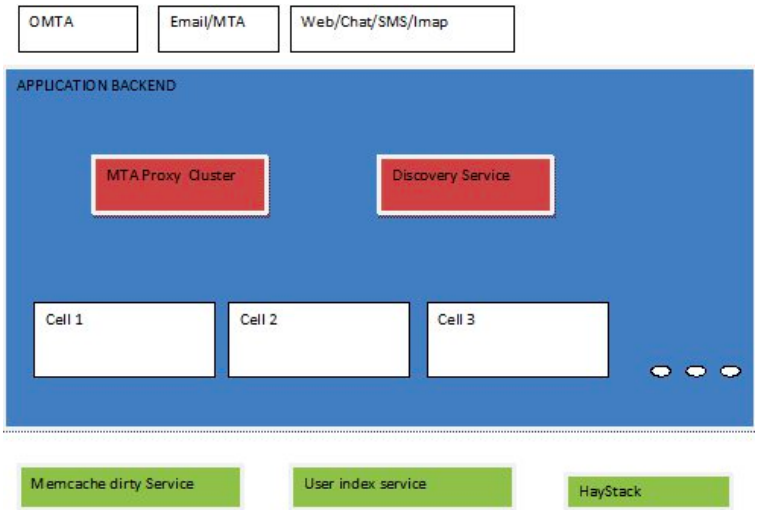
Each cell consists of a single cluster of application servers, and each application server cluster is controlled by a set of ZooKeeper machines.

ZooKeeper is open source software that we use mainly for two purposes: as the controller for implementing sharding and failover of application servers, and as a store for our discovery service. Since ZooKeeper provides us with a highly available repository and notification mechanism, it goes a long way towards helping us build a highly available service.

Each application server registers itself in ZooKeeper by generating N tokens. The server uses these tokens to take N virtual positions on a consistent hash ring. This is used to shard users across these nodes. In case of failures, the neighboring nodes take over the load for those users, hence distributing the load evenly. This also allows for easy addition and removal of nodes into and from the application server cluster.

- **Application business logic:** This is where the magic happens. The business logic is responsible for making sense of all user data, storing and retrieving it, and applying all the complex product operations to it to perform various functions. It also has a dedicated cache that acts as a write-through cache, since the application servers are the only entry points to read/write data for any given user. This cache stores the entire recent image for the user and gives us a very high cache hit rate. The business logic also interacts with the Web servers to respect user privacy and also apply any policies.
- **Data access layer:** The data access layer is the schema used to store the user's metadata. It consists mainly of a time sequenced log, which is the absolute source of truth for the user's data, and is used to back up, retrieve, and regenerate user data. The schema also consists of snapshots that represent the serialized user objects understood by the business logic. This layer is designed to present a generic interface to the application servers while making the underlying store pluggable.
- **Metadata store:** Each cell also has a dedicated metadata store. We use HBase as our metadata store. The data access layer interacts with HBase to provide storage functionality. Late last year we talked about our Messages storage infrastructure, which is built on top of Apache HBase.

Finally, the whole system has a number of cells, and looks like this:



Other Messages Services

The Messages application back end needs to parse email messages and attachments, and also provide discovery of the right application servers for the given user. This is achieved with the following services:

- **MTA proxy:** This service receives all incoming email messages and is responsible for parsing the email RFCs, attachments, large bodies of email, and so forth. These parsed out values are stored in a dedicated Haystack cluster (which is the same key/value store that we use for photos). Once the proxy has created a lightweight email object, it talks to the appropriate application server and delivers the message. But talking to the appropriate application server involves figuring out the cell and machine a particular user resides on, which brings us to the discovery service.
- **Discovery service:** This consists of a map of user-to-cell mappings. Every client needs to talk to the discovery service before it can contact an application server for any request. Given the stringent requirements, this service needs to be very highly available, scalable, and performant.
- **Distributed logic client:** These clients listen for ZooKeeper notifications and watch for any changes in the application server cluster state. Each application server cluster or cell has a dedicated client. These clients live in the discovery service process, and once the discovery service has mapped the user's cell, it queries that cell's client, which executes the consistent hash algorithm to figure out the correct application server node for the user.

The Messages application back end also relies on the following services:

- **Memcache dirty service:** The application servers query message counts from the home page very frequently to accurately display the message notification jewels. These counts are cached in memcache in order to display the home page as quickly as possible. As new messages arrive, these entries need to be dirtied from the application servers. Thus, this dedicated service runs to dirty these caches in every data center.
- **User index service:** This provides the social information for each user, like friends, friends of friends, and so forth. This information is used to implement the social features of messaging. For example, on every message that is added to the system, the application server node queries this service to determine if this message is from a friend or a friend of friend and directs it to the appropriate folder.

The clients of the application back end system include MTAs for email traffic, IMAP, Web servers, SMS client, and Web Chat clients. Apart from the MTAs, which talk to the MTA proxy, all other clients talk directly to the application servers.

Given that we built this services infrastructure from scratch, one of the most important things was to have the appropriate tools and monitoring in place to push this software on almost a daily basis without any service disruption. So we ended up building a number of useful tools that can give us a view of the various cells, enable/disable cells, manage addition and removal of hardware, do rolling deployments without disrupting service, and give us a view of the performance and bottlenecks in various parts of the system.

All these services need to work in tandem and be available and reliable for messaging to work. We are in the process of importing millions of users into this system every day. Very soon every Facebook user will have access to the new Messages product.

At Facebook, taking on big challenges is the norm. Building this infrastructure and getting it up and running is a prime example of this. A lot of sweat has gone into bringing this to production. I would like to thank every individual who has contributed to this effort and are continuing to do so. This effort involved not only the Messages team but also a number of interns and various teams across the company.

Prashant is a software engineer at Facebook. He leads the Messages back end team.


Ahmet Murati nice, keep it up
April 12, 2011 at 11:24am


Mustafa Kzkz

بارد

April 12, 2011 at 11:24am


Frantz Romain built with C++ or Python
April 12, 2011 at 11:25am


Dave Case this is quite the schematic...
April 12, 2011 at 11:25am


Wagner Skellington Nice!
April 12, 2011 at 11:25am


Frantz Romain ?
April 12, 2011 at 11:26am


Gowrish Bhaskar very interesting...Prashant .looking fwd to use the new messaging product
April 12, 2011 at 11:28am · 1


Duncan Washington Dan I AM SO INTERESTED INTO fb API
April 12, 2011 at 11:29am


Ivan Christopher Great!!!!!!!!
April 12, 2011 at 11:30am


Arif Skygear nice implementations.
April 12, 2011 at 11:30am


Jay Tee @Frantzdy : C++, HipHop for PHP
April 12, 2011 at 11:33am


Sigit Nurseto Thanks for sharing this
April 12, 2011 at 11:33am


Kevin O'Malley How do I turn it off?!?!?! I'm sick of having my chats cluttering up my inbox! It's impossible to find anything in all that mess! You've made the inbox useless! Chat is one thing, messages are another - seperate them, or give me a way to undo this mess! BAD IDEA.
April 12, 2011 at 11:36am · 4


Shaan M. Khan great
April 12, 2011 at 11:37am


Brandon Hilk I'm convinced that Facebook Engineers can pretty much do anything.
April 12, 2011 at 11:50am · 2


Syd Pao Really huh..
April 12, 2011 at 11:51am


Syd Pao I wonder if we can have a @facebook.com email add...
April 12, 2011 at 11:51am


Oliver Staunton @Janos - your FB email address is syddo@facebook.com under this new system
April 12, 2011 at 11:55am


Monis 'Mendax' Ahmad @janos we already have
April 12, 2011 at 11:57am


Ahmed Khalid awsom i hope i get the new system soon ..
April 12, 2011 at 11:57am


Marko Mihelčić Tnx for sharing the overview
April 12, 2011 at 12:02pm


Kevin O'Malley The new system impresses only Facebook engineers. Once you see the mess your inbox becomes you'll sing a different tune. Not one person I know who tried this out likes it. Not one. And there is no way to undo it. And it breaks the Android app notifications. Perpetually getting notifications of new messages on my phone when there are none. This blows big time.
April 12, 2011 at 12:06pm · 5


Mohnish Harisinganey Prashant - which app server are you using?
April 12, 2011 at 12:21pm


NavyaSrikanth Metlapalli interestin ..
April 12, 2011 at 6:59pm


Vaibhav Agrawal Amazing..
April 12, 2011 at 11:01pm


Hiufai Wong Kool
April 13, 2011 at 12:50am


Like Chris good job!!!
April 13, 2011 at 5:32am


Beverly McDowell cutting out the disabled is unwothy enough. Yet,
...messages run rampant and cannot appear in zoom.
April 13, 2011 at 1:46pm


Waleed Ahmed I Love the way Facebook's engineers works.
April 13, 2011 at 2:17pm


Jodelyn Marcial Susano-Maur how does it works ?
April 14, 2011 at 7:33am

Olson Wong Thanks for sharing. Keep it up.
April 14, 2011 at 1:24pm

Craig Bennett Facebook seems to have recently changed the filter settings for email notifications, which means that MANY "new" options have been selected to "yes" without your permission.

If, like me, you use your mobile phone to receive Facebook notifications fo... See More
April 20, 2011 at 5:16am

Zahid Javed i am also a engineer i want to a new look in engineering
April 20, 2011 at 7:30pm

Sharun Kumar The new features are cool!
April 22, 2011 at 11:59pm

Omar Kassem is there a way to disable chat messages pls! its is cluttering up my inbox n i really dont want a chat history saved in my inbox!
April 26, 2011 at 2:19am

Rishi Baldawa could you describe more about tools and monitors used and the performance metrics & thresholds . also, you could probably use user index to figure out at connectors for two people and use that as a factor while sending new friend requests i.e. asking someone to introduce A to B.
April 28, 2011 at 8:06pm

Salma Nehme Get 10 fbml table & All code with video learn,Join now this page = > http://www.facebook.com/Static.FBMLs
May 16, 2011 at 5:05pm

Igor Khomyakov I think you mean another Haystack http://www.facebook.com/note.php?note_id=76191543919
May 18, 2011 at 12:36am · 1

Régis Eduardo good job!
August 18, 2011 at 5:58pm

Tian Pan Thanks for sharing! This article keeps saying how important the cells are and how they are sharded, but you still do not tell us what a cell is... Pictures are helpful but not helpful enough. What is the relationship between those concepts in those boxes (and how do they interact)?
January 31 at 3:58pm