Kronenstr. 22A
70173 Stuttgart
Germany

Tel: +49 711 89 66 56 - 0
Fax: +49 711 89 66 56 -10

Web: nextcloud.com
E-Mail: sales@nextcloud.com

Global Scale

Nextcloud Global Scale

Architecture Whitepaper

Credits:
Frederik Orellana (DeiC)
Frank Karlitschek (Nextcloud)

Summary

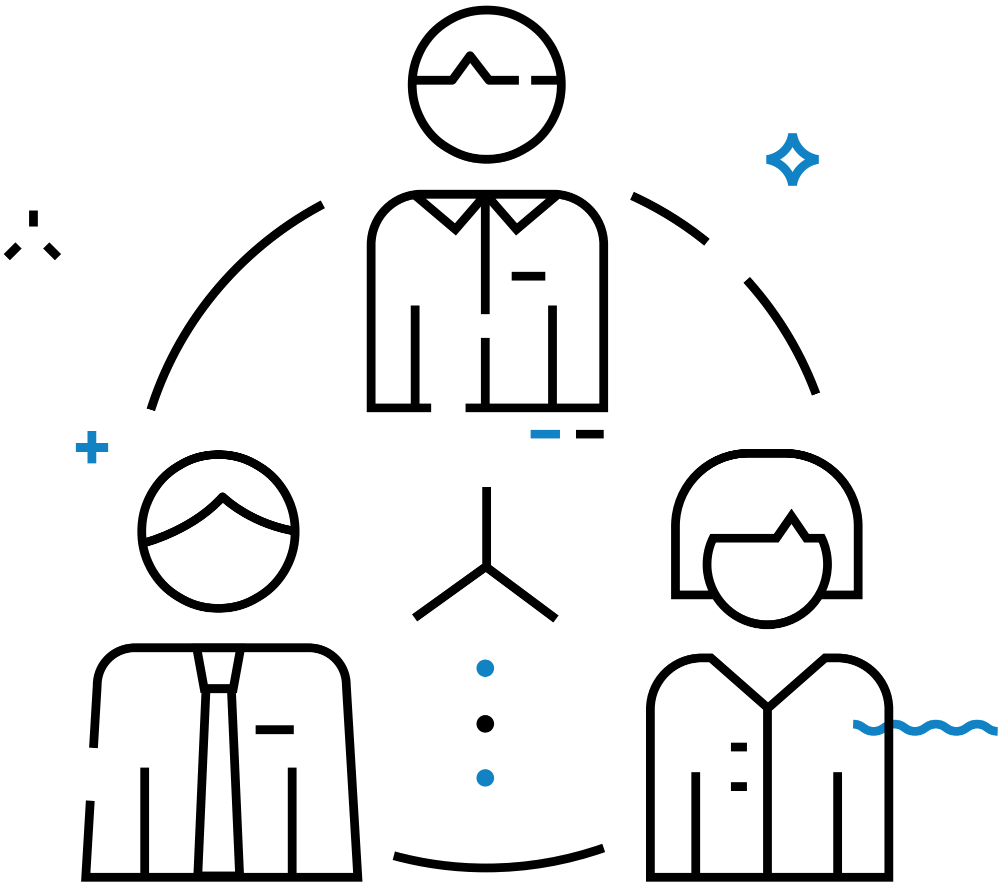
In this whitepaper we describe the current web application architecture Nextcloud uses and its limitations; and introduce Global Scale (GS), an architecture designed to lift these limitations. GS spreads users and data over indepedent Nodes and introduces 3 components to manage them: the Global Site selector which enables users to log in from one place; the Lookup Server which mediates sharing and stores certain metadata; and the Balancer which monitors and manages the Nodes. We address some other elements that need to be considered (sharing, comments and other metadata and more) and conclude with limitations and side benefits.

Table of Contents

Standard Nextcloud architecture........3
Limitations of the standard architecture:........4
1. Scalability........4
2. Storage Cost........4
3. Global distribution........5
Goals:........5
Approach........6
Architecture........6
Components........7
Node........7
Global Site Selector........7
Lookup Server........8
Balancer........8
Other elements........9
Migration........9
Public sharing links........9
Federated comments and activities........9
Backup........9
Conclusion........10
Limitations........10
Side benefits........10

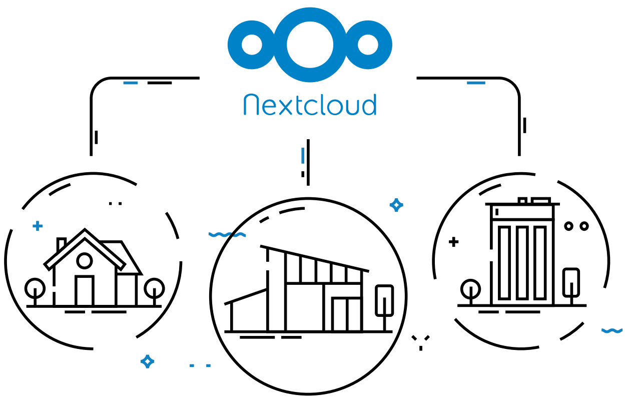
Standard Nextcloud architecture

Currently, Nextcloud is using a standard web application architecture. The incoming user traffic is distributed through load balancers to a number of application servers that run independent Nextcloud instances. This application servers access a shares storage, a shared database and a shares cache. This architecture scales well up to a 6 figure number of users. Especially the application servers are easy to scale because doubling the number of server doubles the performance. But scalability limitations can be found in the shared components. This are the loadbalancers, the hosting center uplink, the database, the storage and the cache.



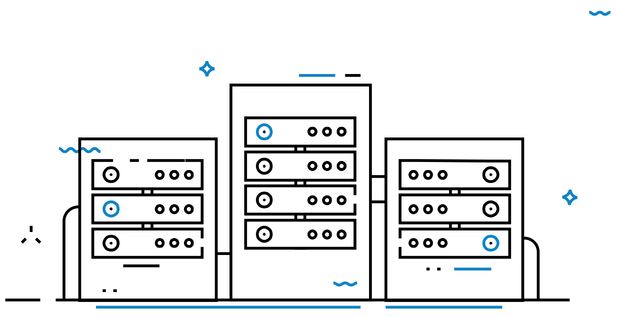
Limitations of the standard architecture:

The standard Nextcloud architecture works well for a lot of use-cases but has several limitation. The 3 main ones are the following:



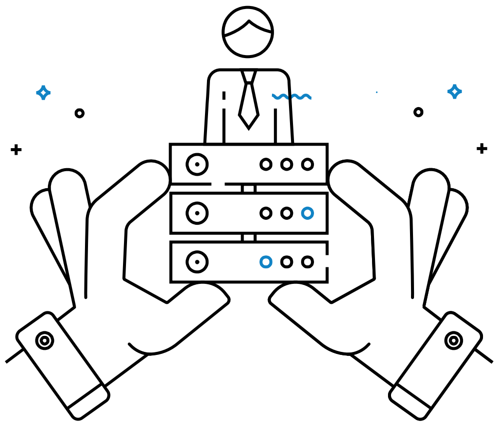
1. Scalability

It is hard to scale the standard architecture to instances with 7, 8 or 9 digit number of users. The shares components loadbalancers, hosting center uplink, database, storage and cache will sooner or later become bottlenecks. Especially the database is hard to scale bejond a 4 node Galera Cluster which limits the number of users and files.



2. Storage Cost

Several big Nextcloud users raised the issue that scaling the storage becomes exponentially more expensive when dealing with a high number of petabytes. Software defined storage and object stores are unfortunately not solution in reality. Dr. Hildmann from the Technical University of Berlin estimated that 60% to 80% of the cost of running a file sync and share service is caused by the storage sub system alone. It would significantly lower the total cost of ownership if the storafe cost could be reduced.



3. Global distribution

Some Nextcloud users have the need to run a Nextcloud instance that is distributed over several hosting centers, countries or continents. The reasons might be the need for data locality for legal reasons, the need to leverage fast local networking speed or the fact that some parts of the instance have to be operated by a different group of administrators or that different auditing or security rules apply. This requirements are hard to implement with the standard Nextcloud architecture. There are solutions to replicate storage across continents and the same is true for some databases or caches. But there is no technology available that can replicate this in a syncronous way so that filesystem, database and cache changes are bind together in replication transactions. And even if thi could be done then it would be really slow.

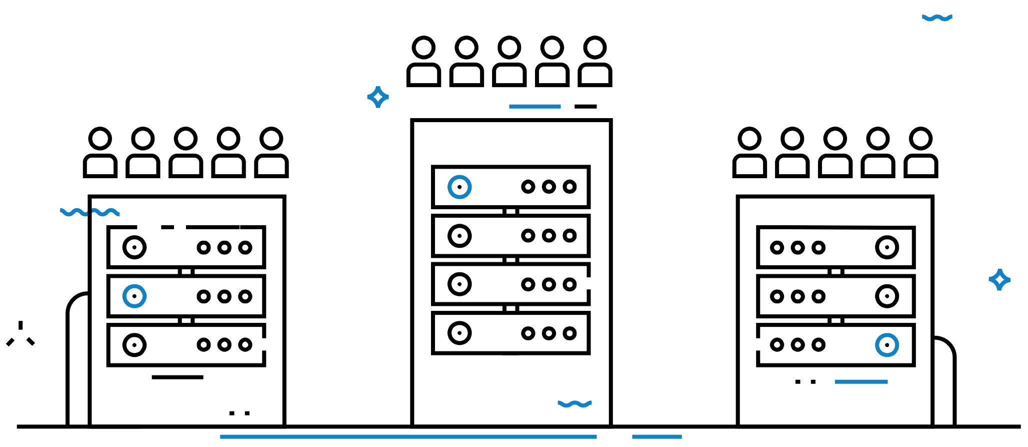
Goals:

The goal of the Nextcloud GS architectue is to solve these three architecture limitations. Specifically, the goals are to build an architecture which is::

- High scalable up to a 9 figure number of users.
- Makes it possible to distribute an instance over different hosting centers and continents.
- Lowers hardware costs by leveraging comodity hardware
- Lowers maintainance costs.
- Has no significant software license costs beside Nextcloud
- Gives the possibility to start with a small installation and scale up over time.

Approach

The approach is to remove the shared components of the standard architecture and move the logic into the pplication server level. Shared components are the loadbalancers, the hosting center uplink, the database, the storage and the cache. It has to be possible to run the applicartion servers on standard, inexpensive commodity hardware to save costs and increase flexibility. Storage, database and cache are running local on the application servers.



Architecture

An Nextcloud GS instance consists of a lot of independent application servers called nodes. There are no central database, storage or caching instances. A node is a tuple of two machines that share a cluster FS and a cluster DB and a shared cache. This guarantees the high availabvility of the nodes. If one machine of a node goes down then the other machines takes over. A node could optionally contain of more then two machines. It could even be a full standard clustered Nextcloud architecture instance if it makes sense. For the context of this document we look into the extreme case where a node is as small as possible which is a touple of two machines.

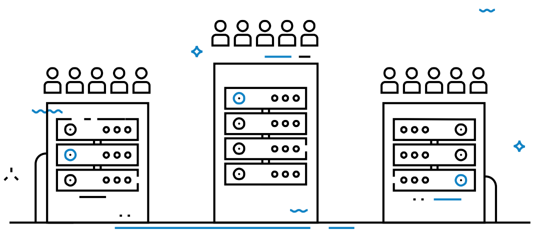
These nodes can be located in different hosting centers. No fast interconnect between the sites is necessary.

Every user is local to a node so everything so all the user data exists only on this one location. All sharing is done via federated sharing even with users on the same node. This is importnt because the location of users might change if the user is moved to a different node.

Components

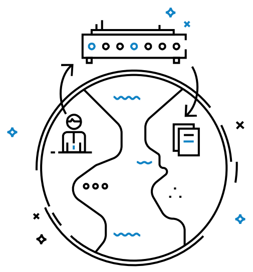
Global Scale consists of Nodes while interaction between users, nodes, shares and the nodes themselves are managed by three systems: the Global Site Selector (GSS), the Lookup Server (LS) and the Balancer.

Node



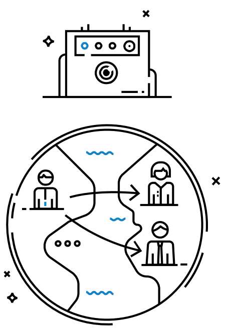
This is a tuple of machines which acts as one logical Nextcloud instance. This are Linux servers using commodity hardware. It's running a webserver including TLS, Nextcloud, local storage, local database and local cache. This components are shared between the machines of the node. Possible software candidates are GlusterFS, MySQL Cluster or Galera and Redis. This nodes use central remote logging and central authentication like for example LDAP. The software on this nodes is packaged and distributed via Docker containers to enable easy deployment of new nodes.

Global Site Selector



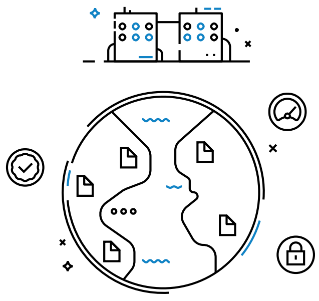
The global site selector acts as a central instance that is accessed by the user during the first login. The main url of the services is mapped to the global site selector as a public visible endpoint. A user accesses the global site selector via Web, WebDAV or REST. The gss authenticated the user via the central usermanagement like for example LDAP. Then it looks up the mode where the user is located in the lookup server and redirects the user to the right hostname. The following calls during the same session are done directly from client to the node. This has the benefit that the load on the global site selector is low and that the clients and leverage fast direct network speed if the node is located near the client.
The global site selector component is implemented as an Nextcloud App

Lookup Server



The lookup server is a component that stores the physical location of a user. It can be queried using a valid userid to fetch the federated sharing id of a user. In some setups it is importnt to limit queries to a certain IP space to avoid data leakes. For the purpose of migrating users and sharing links between instances it is also necessary to store a list of old federated sharing ids. The lookup server can also store additional data of the users like for example the required required QoS. This contains settings for: storage/quota settings, speed class, reliability class, location of users on the earth.

Balancer



This is a component that is implemented as Nextcloud app. It runs once per service as dedicated machine to and monitors the storage usage, cpu and ram load, network utilisation and uptime of all nodes. It can mark nodes as online or offline and can initiate the migration of user accounts to different nodes.

The decision to move user between nodes is done based on User QoS settings and balancer data. The SLA of the user, physical location and proximity of nodes is also considered.

Other elements

A number of other factors needs to be adjusted or taken into account when implementing Nextcloud Global Scale. We need a strategy for migration; sharing; exchanging metadata like comments and activities; and backup.

Migration

Moving users is done as following:

1. Switch user offline at server 1
2. Export user at server 1 (For better performance can be prepared earlier)
3. Import user at server 2
4. Update lookup server
5. Switch user online at server 2
6. Delete user at server 1

These migration steps can be triggered with OCS calls from remote.

Public sharing links

Public sharing links are always handled by the nodes directly and are not redirected by the global site selector. Because of that a system is required to keep sharing links working if a user is migrated to a different node. Because of that a new Nextcloud app is required that stores old public links and redirects to the new node if they are accessed. The lookup server is queried to get the old and current federated sharing id.

Federated comments and activities

To make it possible to see activities from other nodes and enable federated comments a new API needs to be implemented. The idea is to use a standard API for this for example the ActivityPub API recommended by W3C. Nextcloud 12 introduces federated activities using this API already.

Backup

If a node fails completely no data should be lost. Because of that a full backup strategy is important. Nodes should be backuped automatically in the background to orther nodes. This backup can happen asyncroniously to prevent andy performance penalty. The number of redundant copies should be configurable. A backup is the full storage of the instance and a database clone. It has to be possible to switch on a backup node without significant downtime.

Conclusion

With the above architecture, a single Nextcloud instance should scale from tens of thousands to hundreds of millions of users. It achieves the goals we have set: dramatically lowering costs by reducing complexity; and enabling administrators to control where user data is located in a fine grained way.

Limitations

Auditing, config and logging is distributed which might make operations more difficult. On the other hand specialized high load Galera Cluster and Storage Subsystems know how is no longer needed and high performance, scalable systems to handle configuration and logging data exist.

All operations and sysadmin tasks have to be highly automated to avoid overhead while managing the high number of comodity servers. Luckily there are plenty good orchestration tools for this purpose available...

Side benefits

Most of the GS features are also useful for 'normal' homeusers. They can collaborate with others but also move their Nextcloud instance to anothe provider or host or device without loosing shares or connections to other instances. The federation of activities, comments and more

Some of the GS features like federated activities are also a big step towards a federated social network.