Voxer Service Architecture

Bunch of Words

Voxer serves a diverse, global user base that depends on the service 24 hours a day. We intend the service to always be up, even during upgrades, machines failures, or maintenance. We are building a new class of telecommunications infrastructure, so it should always be up.

To deliver this kind of service, we must tolerate the loss of any single machine or process. It's possible and likely that certain combinations of failures will degrade service, but the loss of any single machine or process should not be any more serious than any other machine or process. If any machine or process fails, we should try to bring it back as soon as is practical, but the Voxer service must continue to operate fully until the failed component is restored.

To add capacity to the system, we add more hardware. All processes should be able to distribute their work across an increasing pool of machines. This is sometimes referred to as "scaling out, not up".

Technology Choices

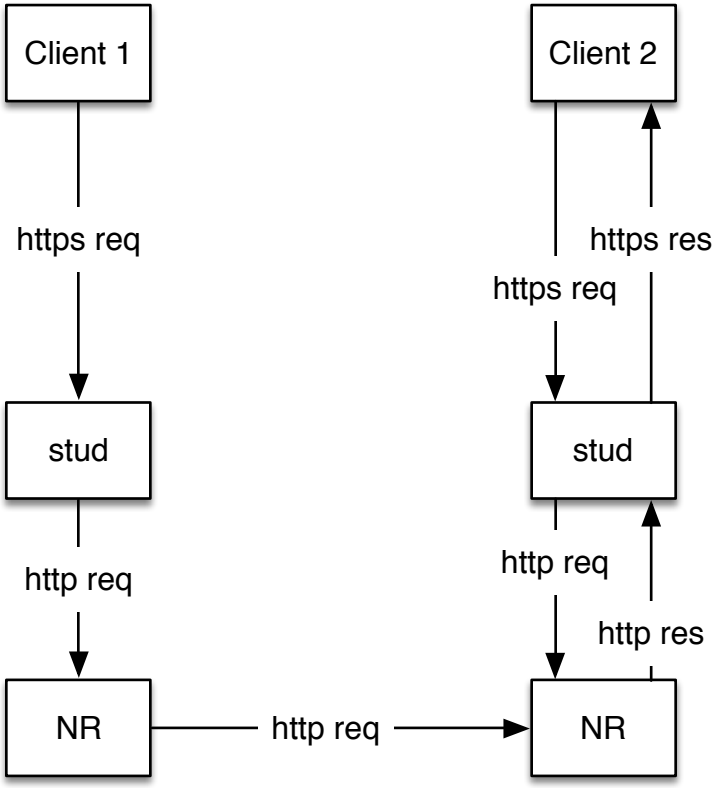
Node.js - JavaScript server runtime using Google's V8 library. Runs all of our code.

Riak - highly scalable and available key/value store. Used for all persistent storage.

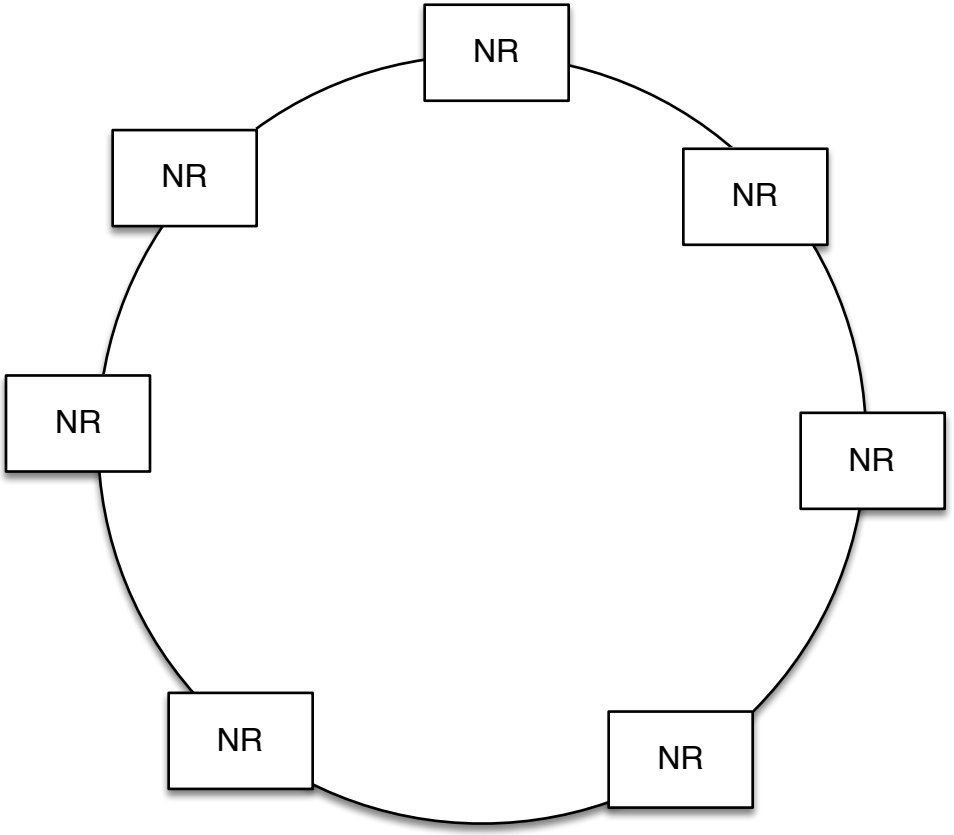
Redis - in-memory database with network API. Used as a cache layer and for ephemeral storage.

SmartOS - Modern Solaris-based operating system with unique tooling for node.js and overall performance optimization.

Live Messaging

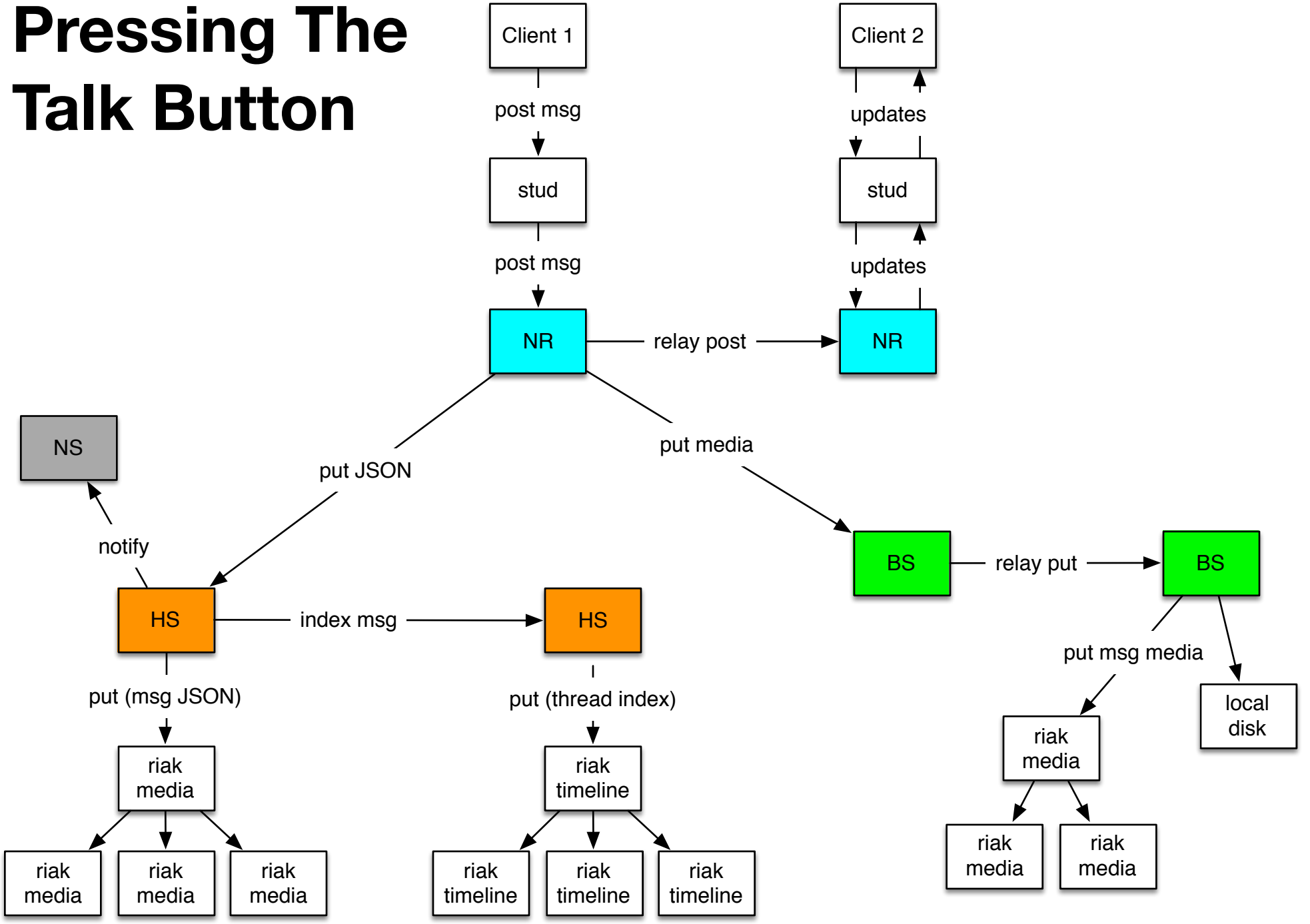


Consistent Hashing



See also: http://en.wikipedia.org/wiki/Consistent_hashing

Pressing The Talk Button



All Services

NR - Node Router
HS - Header Store
BS - Body Store
NS - Notification Sender
NMN - New Message Notification
PBR - Purchase Broker
PM - People Matcher
PS - Profile Search
WWW - public web
Auth - Authentication
Admin - admin tool
MS - Metrics Server
Voxerbot - auto responder / welcome chat