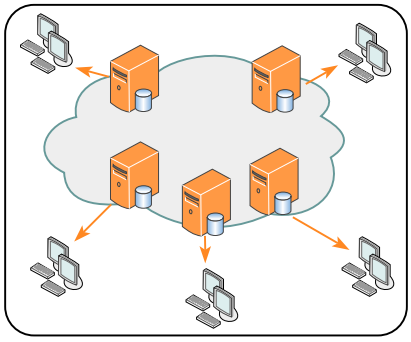
Chair of Network Architectures and Services
Department of Informatics
Technical University of Munich

Thesis M.Sc.

Towards Consistency in Distributed REST API Caching

Motivation



Caching API responses can be challenging. Especially for interactive appliations, users need to see the latest updates to the data they are working with. Moreover, the impact of stale content can be much worse compared to static data (e.g., images). If APIs are authenticated, caching can even open new security vulerabilites as happend to Valve in 2015 [1]. Hence, developers tend to completly disable caching for their APIs or must actively purge them [5]. However, caching could provide major speedups for single page applications that completly rely on navigating APIs. This thesis shall investigate a novel approach for disributed caching of HATAEOAS REST APIs. The challenge relies in the distributed nature of caches. One cache might notice an update to cached data; however, all caches need to invalidate all affected data to supply a consistent view. Consensus concepts [2] like Reliable Broadcast, Bimodal Multicast, CRDTs [4] or ”set-union consensus” [3] are promising for the design a distributed caching solution that handles updates to cached data consistently even in the presence of failures. However, web caches only provide a benefit if they are fast, faster than contacting the origin directly; hence, the used protocols and implementation needs to be as performant as possible. As consequence, the resulting system might just be considered eventual consistent.

Your Task

- Research on suitable consensus approaches
- Implement a simple REST API (json:api [6]) to evaluate caching approaches
- Design and implement a Go cache servers based on the selected consensus protocol and the json:api specification
- Evaluate the performance of the approach on a multi node setup considering node and link failures
- Compare the approach with no caching, time based caching, and active purging

Requirements

Bash, Golang, and Python Django or Java Spring. Familiar with CDNs and distributed computing.

Contact

Markus Sosnowski sosnowski@net.in.tum.de
Richard von Seck seck@net.in.tum.de

References

[1] https://store.steampowered.com/oldnews/19852
[2] George F Coulouris et al. Distributed systems: Concepts and Design. fifth. Pearson Education, 2011.
[3] Dold, F., Grothoff, C. Byzantine set-union consensus using efficient set reconciliation. EURASIP 2017, 14 (2017)
[4] https://en.wikipedia.org/wiki/Conflict-free_replicated_data_type
[5] https://www.fastly.com/blog/building-fast-and-reliable-purging-system
[6] https://jsonapi.org/