SyncFree Technology White Paper

Legion: Enriching Internet Services with Peer-to-Peer Interactions

Albert van der Linde, Pedro Fouto, João Leitão, Nuno Preguiça
Universidade Nova de Lisboa

Santiago Castiñeira, Annette Bieniusa
University of Kaiserslautern

12 September 2016

Peer-to-Peer Web Interactions

A large number of web applications mediate interactions among users. Examples include collaborative applications, social networks, and multi-user games. Many of these applications manage a set of shared objects, the application state, and each user reads and writes on a subset of these objects. For example, in a collaborative text editor, users share the document being edited.

Unfortunately, many of these applications still rely on centralized infrastructures to maintain the shared state and mediate all interactions among users. Such solutions can incur in scalability bottlenecks. Although this can be partially mitigated by relying on cloud computing elasticity, using such approach leads to potential high monetary costs, making it unsuitable for small startups and companies launching their web applications. Additionally, if the centralized infrastructure becomes unavailable, clients become unable to interact, and even access the application. Additionally, mediating all interactions between clients through a centralized infrastructure can expose clients to high latency in their interactions.

One alternative to overcome these drawbacks is to rely on direct interactions among clients, making the system less dependent on the centralized infrastructure. Besides avoiding the scalability bottleneck and availability issues of typical web application architectures, such approach can also improve user experience by lowering the latency of interactions among clients. Additionally, such an approach has the potential to reduce the load imposed on centralized components, minimizing the infrastructure cost.

The Legion Design Principles

Legion, is a novel framework for designing web applications that exploits recent advances such as Web Real Time Communication (WebRTC) [2] that enables direct and real-time communication across browsers; STUN and TURN [6] that provide simple mechanisms to circumvent firewalls and NAT boxes; as well as new mechanisms that are made available by HTML5 which, for instance, allow applications to store persistent data on browsers, allowing developers to enrich their web applications with seamlessly peer-to-peer interactions, while still leveraging a centralized infrastructure (for instance, to ensure durability of application state). Legion introduces and demonstrates the practicality of a novel hybrid model for designing web applications, that seamlessly combines centralized/cloud-based architectures with direct browser-to-browser interactions among clients.

In Legion, each client maintains a local data storage with replicas of a subset of the shared application objects. Legion adopts an eventual consistency model and exploits synchronization-free replications mechanisms where each client can modify its local replicas without coordination, while updates are propagated asynchronously to other replicas. To guarantee that all replicas converge to the same state despite concurrent updates, Legion relies on Conflict-free Replicated Data Types (CRDTs) [8]. CRDTs are replicated data types designed to provide eventual convergence without resorting to strong coordination, that have been extensively explored within the scope of the SyncFree project.

Unlike systems such as Simba [7], SwiftCloud [9], or Parse [1] that cache objects at the client, besides synchronizing with the servers, Legion clients can also synchronize directly among them, using a peer-to-peer interaction model. To support these interactions, (subsets of) clients establish overlay networks to propagate objects and updates among them. This leads to low latency for propagating updates and objects between nearby clients.

Unlike many overlay networks (*e.g*, HyParView [5], Scamp [3], among others) where all clients operate in a similar fashion, Legion relies on a non-uniform design, where a few nodes are elected to act as bridges between the client decentralized infrastructure and servers that store data persistently (and serve as access points to legacy clients or clients that are unable to establish direct connections with other clients). These bridges upload updates executed by clients in the network and download new updates executed by clients that have not joined the overlay network. This approach reduces the load on the centralized component, which no longer needs to broadcast every update to all clients (nor track these clients).

While leveraging direct client interactions brings significant advantages, it also creates security challenges. We address these challenges by extending our design to make it impossible to users that are not allowed to access some objects to either observe those objects or interfere with operations issued by authorized clients. Our design uses lightweight cryptography and builds on the access control mechanism of the central infrastructure to securely distribute keys among clients.

Enriching Systems with Legion

To simplify the integration of Legion on existing applications, Legion supports the inclusion of extensions, that enable the framework to interact and leverage existing web infrastructures/cloud-based storage systems. We have implemented an extension for Google Drive Realtime (GDR) [4], a Google collaboration as a service platform, which is commonly used to support web applications similar to Google Docs. This extension allows Legion to store data in the GDR centralized storage, while exposing an API and data model fully compatible with GDR.

In fact, this extension enables developers to adapt their existing GDR applications to take advantage of the mechanisms made available by Legion, by simply modifying two javascript lines of code in their original GDR application.

We are also integrating Legion with the Antidote system that was developed in the Syncfree project, making Legion the natural solution for extending the storage and replication provided by Antidote towards the end-clients.

References

[1] Parse. http://parse.com.

[2] WebRTC. http://www.webrtc.org/.

[3] A. Ganesh, A.-M. Kermarrec, and L. Massoulié. Scamp: Peer-to-peer lightweight membership service for large-scale group communication. In *Networked Group Communication*, pages 44–55. 2001.

[4] Google Inc. Google Drive Realtime API. https://developers.google.com/google-apps/realtime/overview.

[5] J. Leitão, J. Pereira, and L. Rodrigues. Hyparview: A membership protocol for reliable gossip-based broadcast. In *Dependable Systems and Networks, 2007*, pages 419–429. IEEE, 2007.

[6] R. Mahy, P. Matthews, and J. Rosenberg. Traversal Using Relays around NAT (TURN): Relay Extensions to Session Traversal Utilities for NAT (STUN), RFC 5766. Technical report, IETF, Apr. 2010.

[7] D. Perkins, N. Agrawal, A. Aranya, C. Yu, Y. Go, H. V. Madhyastha, and C. Ungureanu. Simba: Tunable End-to-end Data Consistency for Mobile Apps. In *Proceedings EuroSys '15*, pages 7:1–7:16, 2015.

[8] M. Shapiro, N. Preguiça, C. Baquero, and M. Zawirski. Conflict-free Replicated Data Types. In *Proceedings of the 13th SSS'11*, pages 386–400, 2011.

[9] M. Zawirski, N. Preguiça, S. Duarte, A. Bieniusa, V. Balegas, and M. Shapiro. Write Fast, Read in the Past: Causal Consistency for Client-side Applications. In *Proceedings of the Annual ACM/IFIP/USENIX Middleware Conference*. ACM/IFIP/Usenix, Dec. 2015.