Detailed Design of a Distributed Cache

Let's understand the detailed design of a distributed cache.

We'll cover the following

This lesson will identify some shortcomings of the high-level design of a distributed cache and improve the design to cover the gaps. Let’s get started.

Find and remove limitations

Before we get to the detailed design, we need to understand and overcome some challenges:

- There’s no way for the cache client to realize the addition or failure of a cache server.
- The solution will suffer from the problem of single point of failure (SPOF) because we have a single cache server for each set of cache data. Not only that, if some of the data on any of the cache servers is frequently accessed (generally referred to as a hotkey problem), then our performance will also be slow.
- Our solution also didn’t highlight the internals of cache servers. That is, what kind o data structures will it use to store and what eviction policy will it use?

Maintain cache servers list

Let’s start by resolving the first problem. We’ll take incremental steps toward the best possible solution. Let’s look at the following slides to get an idea of each of the solutions described below: