Hive.js: Browser-Based Distributed Caching for Adaptive Video Streaming

Roberto Roverso, Mikael Höggqvist

ABSTRACT

Peer-to-peer (P2P) technology has long been considered a natural complement to standard CDN infrastructure for video distribution because it allows for load sharing and improves quality of user experience. However, P2P solutions have traditionally required the installation of additional software or plugins to be deployed, which significantly hinders adoption.

In this paper, we present Hive.js, a browser-based solution for distributed caching that enables a set of browsers to create distributed cache for video streaming. Hive.js is a JavaScript library that enables a set of browsers to create distributed cache for video streaming.

HTTP Streaming

HTTP is the de-facto industry standard for distributing video streams over the internet

- Based on a pull model that utilizes HTTP as transport protocol
- Content available with multiple video and audio qualities (bitrates)
- Player implements complex heuristics to choose which quality to render

Advantages

- Routers and firewalls are more permissive to HTTP
- HTTP caching is straightforward
- Cheaper CDN cost

DASH and WebRTC

DASH and DASH.js

- Interoperable standard for HTTP-based video streaming
- DASH.js implements DASH for the browser using the HTML5 video framework

WebRTC

- Direct browser-to-browser communication with video, voice, or data
- NAT traversal using ICE, STUN and TURN
- Reliable transport layer using SCTP
- DataChannels for arbitrary message exchange between peers

Hive.js Internals

Fragment Loader

- Used to intercept fragment requests from the DASH.js player and redirect to Hive.js

Peer

- The cache stores video fragments in persistent storage or in memory
- The index contains a mapping from video fragment to a list of peers which has cached the fragment

Transport Layer

- Fragments are sent to other peers over the WebRTC data channel using SCTP
- Large fragments are chunked with additional integrity checks

Discovery Service

- Discovery of random peers with data from the video
- Signaling protocol used to setup a direct connection between two peers

Distributed Caching with Hive.js

Hive.js Overlay

Overlay Construction

- Randomized neighbor selection by sampling peers watching the same video from a shared discovery service

Distributed Cache

- Cache hit - retrieve from other peer
- Cache miss - retrieve from CDN

Cache Protocol

- A peer uses the local cache to store retrieved fragments
- When a fragment is retrieved a peer tells neighbors by broadcasting a Hive message
- A peer that wants to download a fragment checks in a local index of Havens to find a peer to get it from

Experiments

Results from initial experiments in a controlled environment with up to 30 peers