btcproxyd/OmniProxy Architecture

Table of Contents

Overview

btcproxyd (also known as **OmniProxy** when OmniLayer functionality is enabled) is a multi-function proxy server that communicates with **Bitcoin Core** (or **Omni Core** which is an extended version of **Bitcoin Core**) and can be used for the following purposes:

- Create a public API to a Bitcoin Core node
- Provide a relatively lightweight backend for a non-custodial wallet
- Use as a framework for creating Bitcoin-based or Omni-based applications
- Create Bitcoin or Omni Lightning applications (requires Lightning Node)
- Create Blockchain viewer applications
- Create real-time Blockchain analysis applications

Functionality

Current and planned core functionality includes

- JSON-RPC request proxying
- JSON-RPC request filtering ("allow list" for permitted methods)
- JSON-RPC request caching
- JSON-RPC request aggregation and value-added computation
- Efficiently processing ZeroMQ push-based messages from Bitcoin Core

- WebSocket gateway/proxy for ZeroMQ messages (under development)
- REST API for Bitcoin Core functionality (under development)

Implementation Technology

btcproxyd is written in *modern* Java. What do we mean by *modern*?

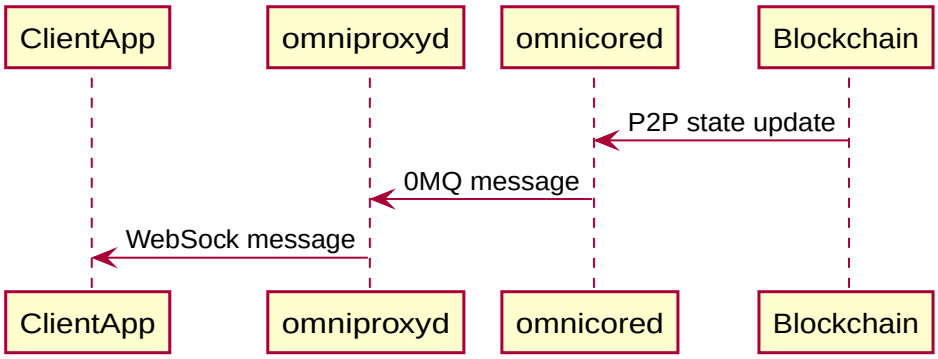
- Implemented using OpenJDK 11 language features and API (may migrate to JDK 17 in the future)
- **bitcoinj** data types are used, to provide access to Bitcoin data validation, cryptographic operations, formatting, etc.
- Major interfaces are asynchronous (`CompletableFuture`) and/or reactive (**RxJava** 3)
- Written in functional-style java with minimal mutable state
- Supports ahead-of-time compilation to native executable using GraalVM `native-image`
- Can also be deployed on latest HotSpot VMs for compatible use-cases
- High-performance, event-driven I/O using modern concurrency constructs
- Uses the modern, cloud-native **Micronaut Framework**
- Docker Images available (both native-image and OpenJDK runtime)

Use Cases and Sequence Diagrams

This section provides explanation of current and future user cases along with sequence diagrams showing the data-flow for that use case.

End-to-end Push (e.g. new Transaction)

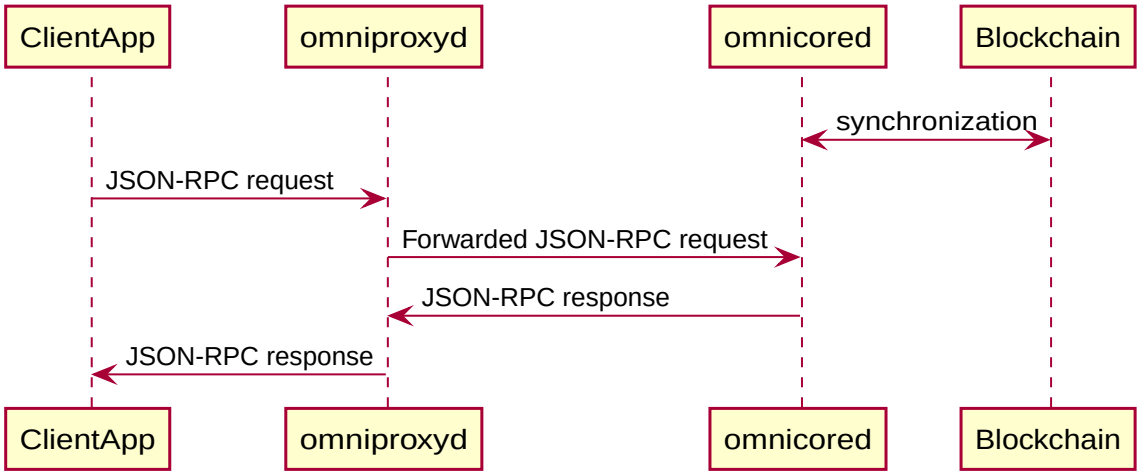
The core architecture supports receiving updated blockchain data via low-latency/push operations. (Currently, ZeroMQ messages from Bitcoin/Omni Core are supported, but the WebSocket client interface is still under development.)



JSON-RPC Client API

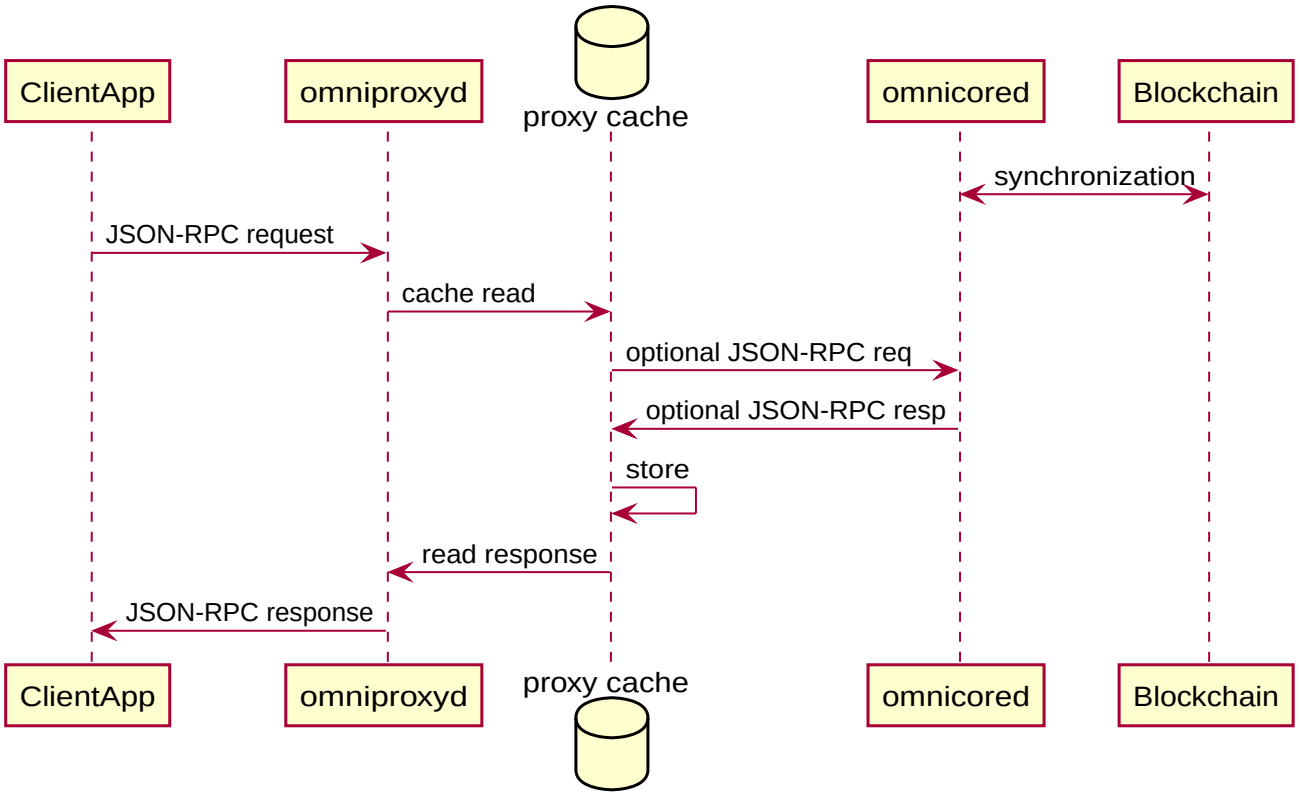
Simple JSON-RPC Proxying

This is the default behavior for any RPC method that is on the allow-list.



JSON-RPC Proxy w/Cache

For RPC methods that are frequently requested and/or computationally expensive (such as `gettxoutsetinfo`) it is essential that the responses be cached. For many methods it makes sense to cache the response and invalidate and refresh whenever a new block is mined. This approach works well for `gettxoutsetinfo`.

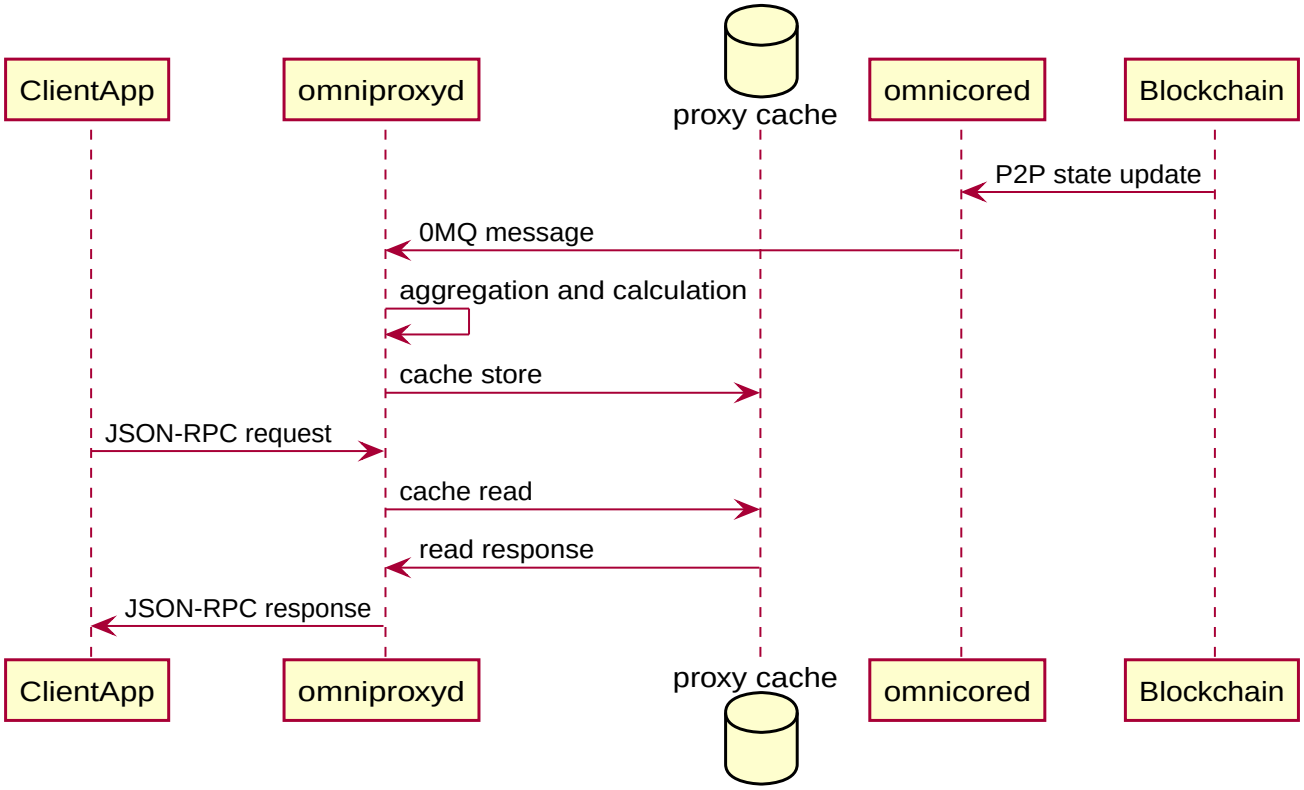


Aggregation, Calculation, and Caching

It is also possible to create new RPC methods like `omniproxy.listproperties` that can return the detailed information for each Omni token (smart property) (as returned for a single property by `omni_getproperty`) but for all current Omni tokens (like the less detailed information provided by

`omni_listproperties`.)

The server will be able to be extended via plugins (plugin mechanism TBD) that can add new, reactive aggregations and computations. (You can look at the existing Omni Layer operations for examples and Pull Requests are welcome.)



Example Applications

OmniMarketCap

OmniMarketCap is a relatively simple application that lets you view the entire list of Omni Tokens (Smart Properties) using live data from the OmniLayer on the Bitcoin blockchain. Using price feeds from centralized exchanges (for `BTC`, `OMNI`, `USDT`, `MAID`, and a few others) it is able to rank Omni tokens by a dynamically updated market capitalization (as new tokens are created or granted "total tokens" is updated, and the market cap is "total tokens" multiplied by the exchange rate.) It also provides dynamically-updated rich lists showing the distribution of market-cap by Bitcoin address.

OmniMarketCap is an OmniProxy client app and works according to the principals illustrated in the sequence diagrams above — it is represented by the **ClientApp** box. In addition, it uses reactive user-interface design patterns so all displayed data updates dynamically the instant new data arrives from the network.

OmniPortfolio

The last release of OmniPortfolio used the OmniAPI. A new version is under development that uses OmniProxy directly.