Verkle Trees

John Kuszmaul

Abstract

We present Verkle Trees, a bandwidth-e cient alternative to Merkle Trees. Merkle Trees are currently employed in a variety of applications in which membership proofs are sent across a network, including consensus protocols, public-key directories, cryptocurrencies such as Bitcoin, and Secure File Systems. A Merkle Tree with n leaves has $O(\log_2 n)$-sized proofs. In large trees, sending the proofs can dominate bandwidth consumption. Vector Commitments (VCs) pose a potential alternative to Merkle Trees, with constant-sized proofs. Unfortunately, VC construction time is $O(n^2)$, which is too large for many applications. We present Verkle Trees, which are constructed similarly to Merkle Trees, but using Vector Commitments rather than cryptographic hash functions. In a Merkle Tree, a parent node is the hash of its children. In a Verkle Tree, a parent node is the Vector Commitment of its children. A Verkle Tree with branching factor k achieves $O(kn)$ construction time and $O(\log_k n)$ membership proof-size. This means that the branching factor, k, o ers a tradeo between computational power and bandwidth. The bandwidth reduction is independent of the depth of the tree; it depends only on the branching factor. We nd experimentally that with a branching factor of k = 1024, which provides a factor of 10 reduction in bandwidth, it takes 110.1 milliseconds on average per leaf to construct a Verkle Tree with 2^{14} leaves. A branching factor of k = 32, which provides a bandwidth reduction factor of 5, yields a construction time of 8.4 milliseconds on average per leaf for a tree with 2^{14} leaves. (The performance on a tree with 2^{14} leaves is representative of larger trees because the asymptotics already dominate the computation costs.) My role in this research project has been proving the time complexities of Verkle Trees, implementing Verkle Trees, and testing and benchmarking the implementation.