Proof of Latency Using a Verifiable Delay Function

Department of Future Technologies

Master's Thesis
University of Turku
Department of Future Technologies

2020
Jani Anttonen

UNIVERSITY OF TURKU
Department of Information Technology

JANI ANTTONEN: Proof of Latency Using a Verifiable Delay Function

Master's Thesis, 15 p., 0 app. p.
Department of Future Technologies
November 2020

Tarkempia ohjeita tiivistelmäsivun laadintaan läytyy opiskelijan yleisoppaasta, josta alla lyhyt katkelma.
Bibliografisten tietojen jälkeen kirjoitetaan varsinainen tiivistelmä. Sen on oletettava, että lukijalla on yleiset tiedot aiheesta. Tiivistelmän tulee olla ymmärrettävissä ilman tarvetta perehtyä koko tutkielmaan. Se on kirjoitettava täydellisinä virkkeinä, väliotsakeluettelona. On käytettävä vakiintuneita termejä. Viittauksia ja lainauksia tiivistelmään ei saa sisällyttää, eikä myäskään tietoja tai väitteitä, jotka eivät sisälly itse tutkimukseen. Tiivistelmän on oltava mahdollisimman ytimekäs n. 120 – 250 sanan pituinen itsenäinen kokonaisuus, joka mahtuu ykkäsvälillä kirjoitettuna vaivatta tiivistelmäsivulle. Tiivistelmässä tulisi ilmetä mm. tutkielman aihe tutkimuksen kohde, populaatio, alue ja tarkoitus käytetyt tutkimusmenetelmät (mikäli tutkimus on luonteeltaan teoreettinen ja tiettyyn kirjalliseen materiaaliin, on mainittava tärkeimmät lähdeteokset; mikäli on luonteeltaan empiirinen, on mainittava käytetyt metodit) keskeiset tutkimustulokset tulosten perusteella tehdyt päätelmät ja toimenpidesuositukset asiasanat

Keywords: list, of, keywords

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background and Motivation

Distributed systems are vulnerable to problems related to data integrity, because in many cases there is no single source of truth. Work pioneered by the likes of Leslie Lamport has tried to mitigate these problems with clock synchronization and consensus algorithms, which have fixed some but not all of the underlying issues related to the field.

Proof of Work is the most well-known consensus algorithm, as it is used in most public blockchains today, including Bitcoin, of in which whitepaper it was first described in 2008. New algorithms have been introduced since to battle it's resource intensiveness, including Proof of Stake, which requires network nodes participating in the voting of new blocks to stake a part of their assets as a pawn. If the voters get labeled as malicious by a certain majority, they can get slashed, losing all or a part of the staked asset in the process. This serves as an incentive for honest co-operation, with sufficient computation resources.

Problem is that the block generation votes are not done globally, but by a selected group of peers called the validators, which vote for the contents of proposed blocks by one peer selected as the block generator. The validators are selected randomly. This has generated an increasing demand for verifiable public randomness, that is pre-image resistant, meaning the output of the algorithm generating the randomness cannot be influenced

beforehand. This created a motivation for an algorithm that would prevent multiple malicious actors from being selected to vote at once, effectively creating a more secure public blockchain.

In 2018, two research papers were released independently with similar formalizations of a VDF.[1][2] By definition, a VDF is an algorithm that requires a specified number of sequential steps to evaluate, but produces a unique output that can be efficiently and publicly verified.[3] To achieve pre-image resistance, a VDF is sequential in nature, and cannot be sped up by parallel processing. There are multiple formulations of a VDF, and not all even have a generated proof, instead using parallel processing with graphics processors to check that the calculation is sequential.[**?**] This bars less powerful devices, like embedded devices, from verifying the result efficiently. Thus, generating an efficient proof that requires little time to verify is more ideal.[3]

Using verifiable delay functions, I propose a novel algorithm for producing a publicly verifiable proof of network latency and computation resource difference between two participants in a peer-to-peer network. This proof can be used for dynamic routing in peer-to-peer networks to reduce latency between peers, and for making sybil attacks harder to achieve.

Chapter 2

Cryptography

2.1 RSA

2.2 Asymmetric Cryptography

2.2.1 Diffie-Hellman Key Exchange

2.3 Elliptic Curve Cryptography

Chapter 3

Distributed Hash Tables

Distributed hash tables are a way of pointing content to peers in a distributed network. In addition to indexing content in content-addressed networks like IPFS, they can function as routing tables. A hash table is just a regular key-value store, a mapping from a to b. What makes them distributed is the fact that the data stored is meant to be distributed between peers, with not a single peer keeping all the available data in its DHT, but relaying queries that it can't answer to other peers on the network.

3.1 Kademlia

Kademlia is a DHT designed by Petar Maymounkov and David Mazières in 2002.

3.1.1 Problem with Randomness

A single query in Kademlia has been shown in real-world tests to result in an average of 3 network hops, meaning that the query gets relayed through two peers before reaching the requested resource.[4] Network hops are a necessary evil in distributed systems, and Kademlia does well in requiring on average a log(n) queries in a network of n nodes. There's a problem, though. Since the closeness metric is based on a similarity search

rather than a measurement, the closest peer is only closest by the identifier, not by network latency.

Chapter 4

Verifiable Delay Functions

4.1 History

Verifiable delay functions are based on time-lock puzzles. Time-lock puzzles are computational sequential puzzles that require a certain amount of time to solve.[5] Verifiable delay functions fall under the same definition, but introduce a publicly verifiable proof that is much faster to verify than the puzzle was to solve.

4.2 Applications

4.3 Variations

4.4 Similar Constructs

A VDF can only be calculated sequentially, but even without a proof there is a possibility to make the verification faster through parallellism. A non-verifiable delay function, or time-lock puzzle in short, can be still verified faster than the calculation, because there is no sequential requirement after the puzzle has been calculated, enabling to use multiple CPU cores or highly parallel graphics processing units for verifying the puzzle, like in

Solana.[6]

Chapter 5

Proof of Latency

Proof of Latency, ”the algorithm” or ”PoL”, is a collection of two algorithms that when used in a P2P context, can offer a robust way of reducing network latency between peers on the network by minimizing the number of hops between peers that are close in terms of performance and geographical location.

The first version of the algorithm requires a trusted setting, preferably a trusted computing platform from all participating peers, while the second version is trustless and requires no specific hardware from the participants, while a trusted computing platform would make it fairer and more reliable, by not discriminating based on CPU performance.

5.1 Role of Latency in Distributed Systems

It’s hardly a surprise, but latency is a huge factor in distributed systems, especially trustless, decentralized ones. Latency is mostly constrained by the speed of light, which can not be changed, and thus there are concrete factors that must be taken into account when designing a p2p system with routing. In 2012, the global average round-trip delay time to Google’s servers was around 100ms.[7]

In the new space age the maximum possible latency grows very fast, as there could be peers joining to a distributed network from other planets, space ships or stations. This

might be unnecessary to think about in the distributed P2P context for now, but before all that, we have global satellite mesh internet providers, like Starlink. Elon Musk, the founder of SpaceX, which provides the network of satellites, claims that there's going to be a latency of about 20 milliseconds.[8] But, since speed of light restricts us to about 100ms of minimum global latency[9], those claims cannot be trusted. In legacy satellite internet access, the round-trip time even in perfect conditions is about 550 milliseconds.[9]

5.2 Network Hops Increase Latency

Network hops in P2P systems are introduced when two peers are not directly connected to each other, but rather through one or many relays. There are network hops that cannot be easily avoided, like the hops between network routers in the internet already. Most of the P2P routing protocols used today are oblivious to the problem of introducing large hops to communications between two peers. These DHT-based protocols, like Kademlia, make the assumption that their users have fast internet access, and minimize the average latency by selecting connected peers basically at random.

While the randomness is great for preventing eclipse attacks, they can introduce unnecessary geographical hops between two peers. If two peers are in the same WAN, for example, in Kademlia they might still connect to each other through a network hop going through another continent. This makes individual connections less efficient. Now, if we were to rely on IP address geolocation, we could more efficiently connect to peers that are close-by. This is unfortunately impossible in privacy-oriented P2P networks, like mixnets, which aim to hide as much of the packet routing information as possible, by routing individual packets through different peers and hiding IP addresses of two connected peers from each other.[10]

Proof of Latency is made to improve the performance of current P2P networking so-

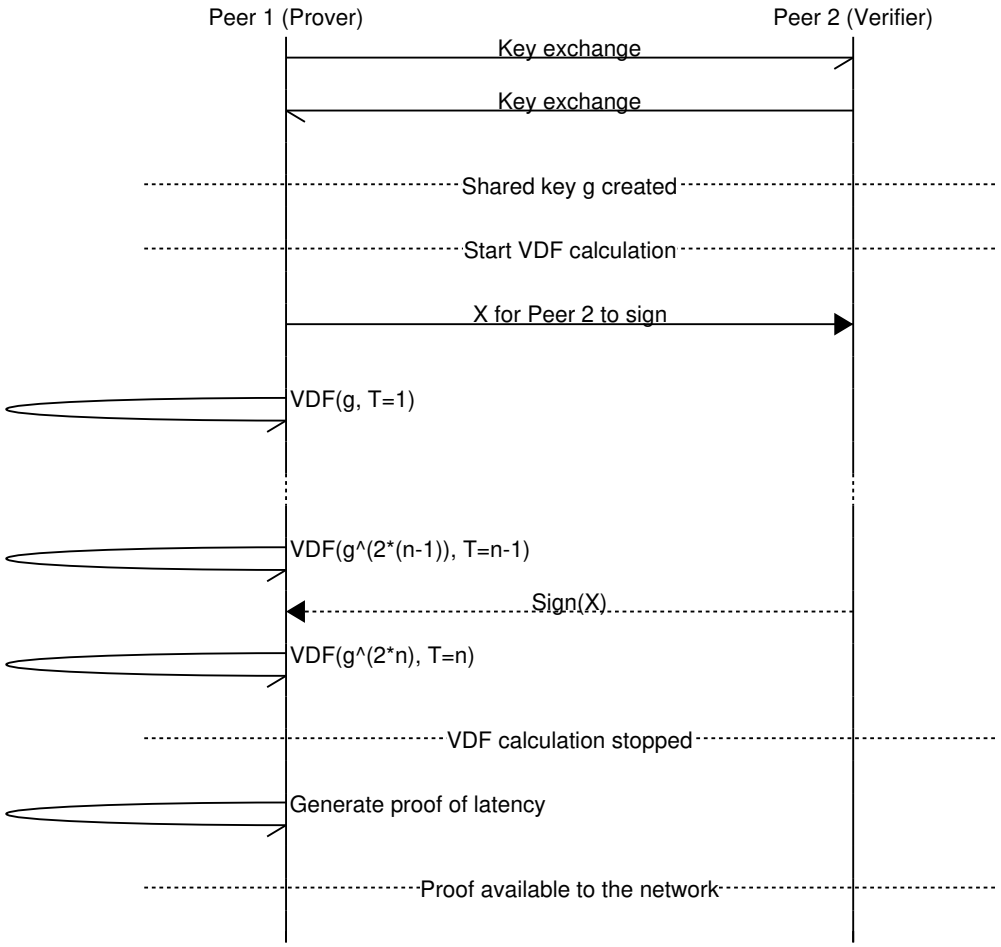

Figure 5.1: Protocol diagram of Proof of Latency

lutions and make them future-proof, even for hops between planets, while still being compatible with some privacy-preserving P2P protocols.

5.3 First Version of PoL

The first iteration of the algorithm is based on an attempt to simplify the initial Proof of Latency algorithm. After some thought, I came to the conclusion that the initial iteration, which ended up as the second and final version of the algorithm, is actually the most resistant to attacks and requires less trust from the participants.

5.3.1 Results

Since the network peers are trying to compete with each other for the least latency, they win the game by calculating the least amount of VDF iterations. In the first version of the algorithm, this soon becomes a problem. There's no way to tell how fast another computer is, so in this case spoofed results of 1 iteration would win every time.

This version only works if the peers can trust each other, but in that case we could also resort to using ping. Ways of introducing trust would be using a trusted computing platform, but even that could be vulnerable to undervolting or other physical tampering. This problem drove me to revert back to my original idea, which I'm calling the version two of Proof of Latency.

5.4 Second Version of PoL

The second version of PoL introduces a race between the two peers. Like in the first version, there's a prover and a verifier. The difference here is that they both calculate a VDF and the verifier calculates the difference in iterations between the two.

First they do a Diffie-Hellman key exchange to construct a previously unknown key. Then, they both start calculating a VDF in parallel. The prover only calculates the VDF up to a predefined threshold, and then sends the proof with a signature back to the verifier. The verifier then stops its own calculation, generates a proof of it, and then calculates the absolute difference between the amount of iterations between its own VDF and the prover's.

Since calculating a VDF is relatively easy for modern processors, a VDF over as little as a few milliseconds of time can be a valid way of measuring latency. Still, without an ASIC chip for calculating VDFs faster than any other available processor, these protocols are also a measurement of processing performance. This might introduce an unfortunate barrier for entry for mobile and IoT devices. The second version of PoL is meant to tackle

this problem by creating a performance and latency gradient to the network. The network topology results in a gradient that is defined by geographical location and the similarity in performance. This means that connectedness between mobile and IoT devices is going to be better than between devices that have a huge performance difference.

Chapter 6

Conclusion

Since calculating a VDF is relatively easy for modern processors, a VDF over as little as a few milliseconds of time can be a valid way of measuring latency. Still, without an ASIC chip for calculating VDFs faster than any other available processor, these protocols are also a measurement of processing performance. This might introduce an unfortunate barrier for entry for mobile and IoT devices. The second version of PoL is meant to tackle this problem by creating a performance and latency gradient to the network. The network topology results in a gradient that is defined by geographical location and the similarity in performance. This means that connectedness between mobile and IoT devices is going to be better than between devices that have a huge performance difference.

References

[1] Benjamin Wesolowski. Efficient verifiable delay functions. Technical Report 623, École Polytechnique Fédérale de Lausanne, 2018.

[2] Krzysztof Pietrzak. Simple Verifiable Delay Functions. Technical Report 627, IST Austria, 2018.

[3] Dan Boneh, Joseph Bonneau, Benedikt Bünz, and Ben Fisch. Verifiable Delay Functions. In Hovav Shacham and Alexandra Boldyreva, editors, *Advances in Cryptology – CRYPTO 2018*, Lecture Notes in Computer Science, pages 757–788, Cham, 2018. Springer International Publishing.

[4] Stefanie Roos, Hani Salah, and Thorsten Strufe. Comprehending Kademlia Routing - A Theoretical Framework for the Hop Count Distribution. *arXiv:1307.7000 [cs]*, July 2013. arXiv: 1307.7000.

[5] Ronald L Rivest, Adi Shamir, and David A Wagner. Time-lock puzzles and timed-release Crypto. page 9.

[6] Anatoly Yakovenko. Solana: A new architecture for a high performance blockchain, October 2018.

[7] Ilya Grigorik. Latency: The New Web Performance Bottleneck - igvita.com. Library Catalog: www.igvita.com.

[8] Liam Tung. Elon Musk: SpaceX's internet from space should be good enough for online gaming. Library Catalog: www.zdnet.com.

[9] Satellite Internet Latency - VSAT Systems Broadband Satellite Internet Latency - Broadband Satellite Internet Service Latency.

[10] Nym. Sphinx: The anonymous data format behind Lightning and Nym, February 2020. Library Catalog: medium.com.