haute école
neuchâtel berne jura

arc

Hes·so
Haute Ecole Spécialisée
de Suisse occidentale
Fachhochschule Westschweiz
University of Applied Sciences and Arts
Western Switzerland

Bachelor Spring Project

HE-ARC 2016

Overclouds

Author:
Romain Claret

Supervisor:
Marc Schaefer

May 11, 2016

Abstract

Overclouds is a project whose goal is to create an anonymous and decentralized internet data sharing service right through the browser.

1 Description

1.1 English

The initiative behind the project is to create a new generation of internet data sharing tools, suited for today's paranoia for privacy on the internet and the preservation of knowledge for the next humanity generations.

The idea is to give the ability to the user to not rely on corporate servers, or farms of servers anymore. On Over Clouds, everybody and everything are now anonymous nodes, and they connect one to another freely and anonymously.

The network is a democratic mesh of nodes. The data is moving from a node to another across the network via other nodes and is ruled by the consensus of users.

We are aiming that users only need to have a standard Internet connection and a browser with JavaScript capabilities to use the service.

1.2 French

Must to the translation when the English part is validated

Contents

2 Preface

2.1 Introduction

TODO

2.2 The Big Picture

TODO

2.3 Objectives

TODO

2.4 Specifications

TODO

2.5 Management

TODO

2.6 State of the Art

TODO

2.6.1 Similar products (Existing Networks)

TODO

2.6.2 Transfer Protocols

TODO

2.6.3 Protection

TODO

2.6.4 Cryptography

TODO

2.6.5 Hardware

TODO

2.6.6 Block-Chains

So much hype in this, but what is it? Everybody relate it mainly to *Bitcoin*[23], indeed *Bitcoin* introduced this technology (which is its main innovation), but *Bitcoin* is not equivalent to chain-block.

Indeed, the main idea is that no one controls or owns the chain of blocks and forges a system for electronic transactions without relying on trust. A block contains a timestamp and information linking it to a previous block.

Note that a block looks like digital objects that record and confirm when and in what sequence transactions enter in the block chain.

Blocks created by network users with specialized software or specially designed equipment. Block creator are known as "miners" to reference the gold mining. Bitcoin showed us a pretty amazing evolution in the mining procedure; it was designed at the start by Satoshi Nakamoto for CPUs, then it quickly involved into GPU mining then into programmable chips (FPGA) then now it goes into burned circuits (ASIC).

In a crypto-currency system, miners are incentivized to create blocks to collect two types of rewards: a pre-defined per-block award and fees offered within the transactions themselves, payable to any miner who successfully confirms the transaction. Every node in a decentralized system has a copy of the blocks chain; it avoids the need for a trusted authority to timestamp transactions. Decentralized block chains use various timestamping schemes, such as proof-of-work or more recently with PeerCoin the proof-of-participation.

Bitcoin [23] TODO

Ethereum [8] Seen today has the new way to use the block-chains technology. It uses the currency has fuel to execute turing-complete smart contracts. Contracts, see as autonomous agents, are programs that run on the Ethereum Virtual Machine. The EVM is being part of the protocol and runs on each client contributing to the network; they are all doing and storing the same calculations. Note that it's not efficient to compute in parallel redundantly, but it offers a consensus for the computed results.

Others We can find a lot of forked crypto-currencies from Bitcoin; everybody is trying to make the success theirs. However, no major changes have been made to them expect the genesis block (the first block that determines how long will be the chain, etc.). We will just cite some of them that have some special modifications. Note also that they don't provide whitepapers.

Lite Coin[18] The major differences with Bitcoin are the time process focus to generate new blocks of 2.5 mins vs. 10 mins for Bitcoin, the use of the Scrypt[4] library and the maximum cap of coins is 84 million (4 times more than Bitcoin).

Dodge Coin [19] It started as a "Joke Currency" but it got capitalized... Its particularity is to have no limits in coins produced, however, the per block reward decreases.

Peer Coin [15] Based on the paper of Scott Nadal and Sunny King [16] for the Proof-of-Stake Peer Coin was born.

2.6.7 Decentralized applications

TODO

2.6.8 Reputation Management

TODO

2.6.9 Operating Systems

TODO

2.6.10 Technologies

TODO

3 Analyses

3.1 Block-Chains

3.1.1 A future?

It's important to note that with incoming quantum computers (predicted to appear wildly in 20ish years), the mining structure, the security and the anonymity must change from today's perspectives. As for today, the block-chain technology is at its hype, meaning that we see it has the best thing in the world.

However, from now the hype will decrease and maybe a new technology will emerge or/and the block-chains technology will evolve or be modeled to go in a direction we didn't expect yet.

3.1.2 What's next in crypto-currency?

As of today, considering that the technology of block-chains won't change, and it still in use for crypto-currency, it could take two types of path.

One of the paths is the neverending death and birth of crypto-currencies. Indeed, once the mining is no more profitable, the security sharply decrease because miners are verifying the transactions and are playing the role of consensus for validating transactions. Miners are mining as long as the devices allow a profit (power consumption, device rentability, etc.). In the best case, where the hardware technology follows the requirements of computational power to solve the increasing difficulty of mathematical problems. (Note that we are currently brute-forcing the solutions.) And based on the model of crypto-currency of Satoshi Nakamoto, Bitcoin, at some point in time, the maximum amount of coins will be reached, and the network won't generate coins (rewards) anymore. At this point, the only income of the miners will be the transaction fees. If the transactions fees are not high enough to motivate the miners to continue mining (and verifying/validation the operations), the currency will die due to the lack of security. So the miners will move to new

profitable crypto-currency (note that they have a pretty advanced hardware for mining at this point, which will help them to start pretty well).

The second path is the modification of the source code of the actual crypto-currencies to make it compliant with the market evolution. For example, increase the maximum amount of coins, or make public keys quantum proof. Indeed, currently, the **ECDSA** is not quantum proof (however the hashing is at the moment, but SHA3 is ready, just to be safe). The problem is ECDSA, which during a transaction send the public key, and theoretically, a quantum computer can guess the private key from it. However, the address is still secure because it's the hashed public key.

But the second path is **killing** the concept of a stable currency based an expendable raw material stock, and the social and economical results are pretty hard to define. A secondary question would be: What will happen, if tomorrow, we find a new gold mine, which holds the same amount of gold already retrieved (doubling by this mean the maximum quantity of gold available), and with a retrievable difficulty level a lot decreased, so it's again profitable to mine?

3.1.3 Predicted evolution in block-chains?

This subsection will be pretty short because at the moment, this technology is only starting to decedent the hype slope, and the only real evolution that pops out recently is the first version of **Ethereum**[30] (2013a) and more recently (2016) the Homestead version of Ethereum [6].

The particularity of Ethereum is that use the currency as fuel to run smart contracts on the EVM (Ethereum Virtual Machine) using the power of each node on the network to do a calculation, and creating a consensus on the output. This technologies evolution has for example created a startup company named *Slock.it* and an alternative "currency" *DAO* (which is unmineable) that allows the IOT (internet of things) to interact with the crypto-currency Ether. It allows, for example, to control a lock, in a hotel, a door could be locked until a client paid the door to open.

3.1.4 Proof-of-Work

Read as PoW[7]. It's a protocol aiming to reduce the risks of DDOS attacks and family abuses by requiring that the client has done some computational work (processing time) before sending a request. It was a solution developed

mainly for our financial world of transactions.

3.1.5 Proof-of-Stake

What is a stake? It's globally something that holds. In our case it's more like a flag can keep a land (a claimed property).

Proof-of-Stake?[16] Read as PoS. Usually in block-chains PoW, miners validate the transactions that came first depending on their CPU power. Note that the more CPU power you have (GPU, FPGA, ASIC, etc.) larger your influence is.

POS is the same thing but with different paradigms:

In one of them, Stakeholders validate with something they own (raw material like an internal currency). And put simple, everybody has a certain chance (proportional to the account's balance) per amount of time of generating a valid raw material.

In another, we are not working with the amount of raw material owned, but with their age (for example, the raw material is multiplied by the time that it was unused) which gives a weighting factor. However with this paradigm, a collusion attack is pretty important, because we could have a super linearity by accumulating aged raw materials.

There are other different types of approaches but we will not details them all because they are not the best of consensus algorithms. (elitism, identity, excellence, storage, bandwidth, hash power, etc.)

Now, on the security side By using raw material (sort of digital assets) defined by the consensus PoS avoids a Sybil[9] attack. Which is is a technique where the attacker is trying to compromise a system by creating multiple duplicate or false identities. It is resulting into including false information, which then can mislead the system into making not intended decisions in favor to the attacker. By the way, PoW protects itself against a Sybil attack by using computational resources that exist extra-protocol.

However, in PoS' traditional approach, we have two major problems. The first is Nothing-at-Stake, and the second are Long range attacks.

Nothing-at-Stake The dominant problem is that smart nodes have no discouragement from being Byzantine[17]. Indeed, signatures are very easy

to produce and they won't lose any tokens for being Byzantine. Another problem is that nodes with digital assets could never spend.

A solution to this would be to have a security layer on deposits, which would cancel Byzantine deposits. To achieve this, we would need to store information about nodes and their immoral behaviors (which are decided by the consensus) so the consensus would be able to punish them. Now, this works only if the transactions are not hidden (with the proof of malicious actions). Also, we should note that this security layer would ask more power for the consensus during the use of punished accounts. Slowing down the consensus is not acceptable because it acts as the authority and by this mean should be the cheaper to operate in power. Punishing the attackers with power consumption is fair.

Compared to PoW, where attackers aren't receiving compensations for their computational power (which is a disincentive). The PoS' security layer is trying to disincentive attackers by removing their digital assets. It could be an interesting social experiment, however, in our human's economic point of view, attackers should be pretty well disincentivized.

Long Range Attack In this type of attacks, the attacker controls account with no digital assets and is using them to create competing version of transactions. This attack is touching both traditional PoS, and the deposit security layer (as long as authentication ends in the genesis block).

The solution here would be to force nodes (and clients) to authenticate the consensus (for example with its state) by signing with the nodes that have something at stake currently, and nodes must have an updated list of nodes with deposits. It's usually called the *weak subjectivity* method.

Ghost From the full name, Greedy Heaviest-Observed Sub-Tree protocol, introduced by Yonatan Sompolinsky and Aviv Zohar[25], it allows the PoW consensus to work with much lower latency than in the blockchain protocol from Satoshi Nakamoto [23], and of course keeping it secure. Indeed, in blockchain based PoW, a miner is rewarded for each block found so the other miners can continue to mine on top of it. However, when a miner produces an orphaned block (a block that exists in the chain), they are not rewarded for their work, plus the consumed power was in vain because the work is unused by the consensus.

Here comes the solution, Ghost, which includes orphaned blocks. It in-

troduces the notion of rewarding orphaned blocks to miners and increasing the security of the consensus with increased validations of a block in the block-chain.

Casper Now there is a friendly ghost in town, it's Casper[3]. This protocol is based on Ghost, and must be integrated into Ethereum for the Serenity[2] version (final), however the Metropolis version must go out before. We should also note that they released the Homestead version on 14th March 2016 (about a month before this Report was released). It will work on the smart contracts.

Finally In comparison to PoW, PoS is much cheaper to secure, transactions speed is greater and it is maybe the stepping stone for the scaling of the block-chain technology.

3.2 Proof of Activity

The protocol from Bentov, Lee, Mizrahi and Rosenfeld [1] which is implemented into PeerCoin (and its clones), is considered as an hybrid of the PoW and PoS. The nodes are doing PoW work by mining blocks and at the same time with the PoS (meaning that the block-chain includes both types of blocks).

The procedure

- The PoW miner mine.
- Block is found, the network is notified and creates a template. (multiple templates are possible)
- The block hash is used to find random owners by using its hash as numbers to determine owners (nodes from the network).
- Turn by turn each chosen owners sign the key with the key of the block.
  - If a chosen owner is unavailable, the process paused. (it's not a problem this concurrently miners are still mining and generating new templates with different owners)

- At some point in time, blocks will be signed and the reward will be given to the miner and the owners.

Continues data exchange In order to reduce the data traffic, each template does not include a transaction list during the signing process itself; it is the last owner (signer) that is adding it when creating the block.

3.2.1 Attacks on block-chains?

Block-chains is designed to be controlled by the consensus of nodes. It means that it can not be owned not controlled by a third party. Until now this goal has been pretty well achieved. However, experiences showed that the system is not perfect.

The 51% attack It's the most interesting (in a social experiment point of view) and rewarding attack for the attacker. Indeed, if the attacker controls at least 51% of the consensus, it is possible to manipulate transaction by validating malicious transactions. Pools owners can do this. Note that as it is today (for actives crypto-currencies), it's not more possible to mine on your own and be profitable, miners are forced to join pools and distribute the work and rewards between them. Meaning that it creates a vicious circle, the more miners are in a pool, the more power it has. The more power it has, the more reward are generated. Finally, this results in attracting, even more, miners because they also want a bigger and easier reward for mining, which leads to the security risk of malicious pool chief who will control the currency and the transactions. All around the internet people are always saying that it's dangerous, in fact, they are asking others to stop making a profit for the good of others, which is a human self-fish reasoning. Can't wait to see this case of a figure.

Spam attacks The idea here to make a lot of transactions to the victim's wallet (by its addresses) and paralyze the legit transactions and by this way its incomes. Indeed, the network will have to process all the spam transactions as well as the legit transactions, meaning that the a delay is added before receiving the legit transactions. In some cases, like for Wikileaks[29], which is depending on this funds to live, it's pretty bad. Plus, since a lot of transactions appear in a block, its value increase and miners will jump

on it to get the reward, meaning that the legit transactions are but a bit behind because generally they have a lower reward. But usually, the current crypto-currencies have an anti-spam solution. They have a minimum fee and they increase the fee after each new transactions.

DDOS on exchange platforms The profit behind this type of attacks is to either steal wallets or ask a reason to release the servers. The crypto-currency is only affected by the depreciation of its value in "real" money because are not able to trade, and they are more likely to switch to another exchange platform or currency.

Special dedication to Mining malwares It's funny to see that hacking is evolving with the hype. Now instead of having zombie computers doing nothing waiting for DDOS attacks or whatever they are used to. They are now mining coins (generally connected to a pool). How smart is that? I personally think that it's amazing!

3.3 Communication

TODO

3.4 Cryptography

As privacy is being an integral part of Overclouds. A research has begun to find the best type of cryptography done on the client-side, from the browser as the highest priority. We were looking for a fair middle between performance and security.

3.4.1 From Scratch VS Libraries

1st Question Is it possible and does it exists right from the browser?

We started looking at what is done in JavaScript and we found an interesting list of *premium* libraries (maintained by prestigious organizations) such as Stanford Javascript Crypto Library[26], MDN[20], W3C[24], Google Closure[10], or msrCrypto[21].

Then we looked at other crypto libraries such as forge[5], jsHashes[14], crypto-browserify[27], etc...

2nd Question The natural question that followed was: is it worth make it ourselves?

The answer came pretty quick while navigating across numerous forums. It's a pretty bad idea if we don't have a team dedicated to it and a pretty strong community to test it out. Even big companies such as Microsoft or Google are struggling a bit on the last part.

However, for the fun of it, we looked at solutions to start a homemade cryptography library. We found two interesting potential starting points to make it work with the browser, a Symmetric Encryption sample [13], or a procedure for Digital Signatures[12].

Decision Shortly after the second question, it was pretty clear that it won't be possible to create our own cryptography library in the time given. So we decided to find the *best* library out there for our project.

3.4.2 Comparison of some JavaScript Cryptography Libraries

Based on the following tables we can notice that **sjcl**[26], **crypto browserify**[27], and **forge**[5] algorithms have been optimized for defined objectives.

talk about the tables and graphs

See Table 1 Related Figures 1, 2, 3, 4
See Table 2 Related Figures 5, 6, 7, 8
See Table 3 Related Figures 9, 10

3.4.3 A killer

After taking time doing research about the above algorithms, we came across a pretty amazing algorithm: **BLAKE2**[11, 22].

BLAKE2 outperforms MD5, SHA-1, SHA-2, and SHA-3 on recent Intel CPUs and it has **no known** security issues. Plus SHA-1, MD5, and SHA-512 are susceptible to length-extension.

It is a *new* algorithm designed specifically for **performance** and is multifaceted **BLAKE2s** (optimized for 8to32-bit) and **BLAKE2b** (optimized for 64-bit)

3.4.4 Now what

If we look at the graphic 11, BLAKE2 is dominating the two best above. **rusha** is close behind it, and forge's *sha256*. Plus we can note that the curves display a nearly completely linear performance.

Also on at the table 4 with the figure 12, we notice that BLAKE2 is in its own category.

3.4.5 What about OverClouds

We can note that we are not really interested in SHA1, because of potential security flaws. SHA256 is much better for us. However, BLACK2 is pretty amazing. We will try to make it work in the following implementation.

Now, if it doesn't work for whatever reason, we will certainly go with forge, crypto-browserify, or sjcl. The problem with forge and crypto-browserify is that we must trust a company or an individual. With sjcl however, we trust an institution.

4 Implementations

4.1 Communication

TODO

4.2 Cryptography

TODO

5 Evaluation

5.1 Tests

TODO

5.2 Results

TODO

5.3 Technologies Recommendations

TODO

6 Conclusion

TODO

7 Bibliography

References

[1] Iddo Bentov, Charles Lee, Alex Mizrahi, and Meni Rosenfeld. Proof of Activity: Extending Bitcoin's Proof of Work via Proof of Stake. 42(240258):1–19, 2013.

[2] Vitalik Buterin. Slasher Ghost, and Other Developments in Proof of Stake, 2014.

[3] Vitalik Buterin. Understanding Serenity, Part 2: Casper, 2015.

[4] COLIN PERCIVAL. STRONGER KEY DERIVATION VIA SEQUENTIAL MEMORY-HARD FUNCTIONS. 2012.

[5] Inc. Digital Bazaar. forge, 2016.

[6] DR. GAVIN WOOD. ETHEREUM: A SECURE DECENTRALISED GENERALISED TRANSACTION LEDGER HOMESTEAD DRAFT. 2015.

[7] Cynthia Dwork and Moni Naor. Pricing via processing or combatting junk mail. *Advances in Cryptology—CRYPTO'92*, pages 139–147, 1993.

[8] Ethereum. Ethereum Homestead Documentation, 2016.

[9] G. Lawrence Paul Sundararaj1 D. R. Anita Sofia Liz2. Anti-Sybil Mechanism against Bogus Identities\nin Social Networks. *International Journal of Advanced Research Trends in Engineering and Technology (IJARTET)*, 01(02):123–127, 2014.

[10] Google. Closure Library, 2015.

[11] Jian Guo, Pierre Karman, Ivica Nikolić, Lei Wang, and Shuang Wu. Analysis of BLAKE2. *Springer International Publishing Switzerland 2014*, 8366(8366):402–423, 2014.

[12] Inc. Info Tech. Digital Signature in the Browser, 2014.

[13] Inc. Info Tech. Symmetric Encryption Sample, 2014.

[14] Paul Johnston. jsHashes, 2015.

[15] Sunny King and Scott Nadal. Peercoin, 2012.

[16] Sunny King and Scott Nadal. PPCoin: Peer-to-Peer Crypto-Currency with Proof-of-Stake. *Ppcoin.Org*, 2012.

[17] Leslie Lamport, Robert Shostak, and Marshall Pease. The Byzantine Generals Problem. *ACM Transactions on Programming Languages and Systems*, 4(3):382–401, 1982.

[18] Litecoin. Litecoin Wiki, 2011.

[19] Max K., Patrick Lodder, and Ross Nicoll. Dogecoin Core, 2013.

[20] MDN. MDN Web API Crypto, 2015.

[21] Microsoft. MSR JavaScript Cryptography Library, 2015.

[22] Ed. Saarinen, M-J. and Jean Philippe Aumasson. The BLAKE2 Cryptographic Hash and Message Authentication Code (MAC), 2015.

[23] Satoshi Nakamoto. Bitcoin: A Peer-to-Peer Electronic Cash System. 2008.

[24] Ryan Sleevi and Mark Watson. Web Cryptography API, 2014.

[25] Yonatan Sompolinsky and a Zohar. Accelerating Bitcoin's Transaction Processing. Fast Money Grows on Trees, Not Chains. *Eprint.Iacr.Org*, pages 1–31, 2014.

[26] Emily Stark, Michael Hamburg, and Dan Boneh. Symmetric Cryptography in Javascript, 2012.

[27] Dominic Tarr. Crypto-Browserify, 2013.

[28] Dominic Tarr. Performance of Hashing in Javascript Crypto Libraries., 2014.

[29] TheBitcoinNews. Bitcoin Spam Attacks, 2015.

[30] Vitalik Buterin. A Next-Generation Smart Contract and Decentralized Application Platform. 2013.

8 Annexes

8.1 JS Cryptography Library Graphs

Add other crypto libraries using **Dominic Tarr**'s benchmark set [28].

The graphs from the following figures have been made by **Dominic Tarr** [28]

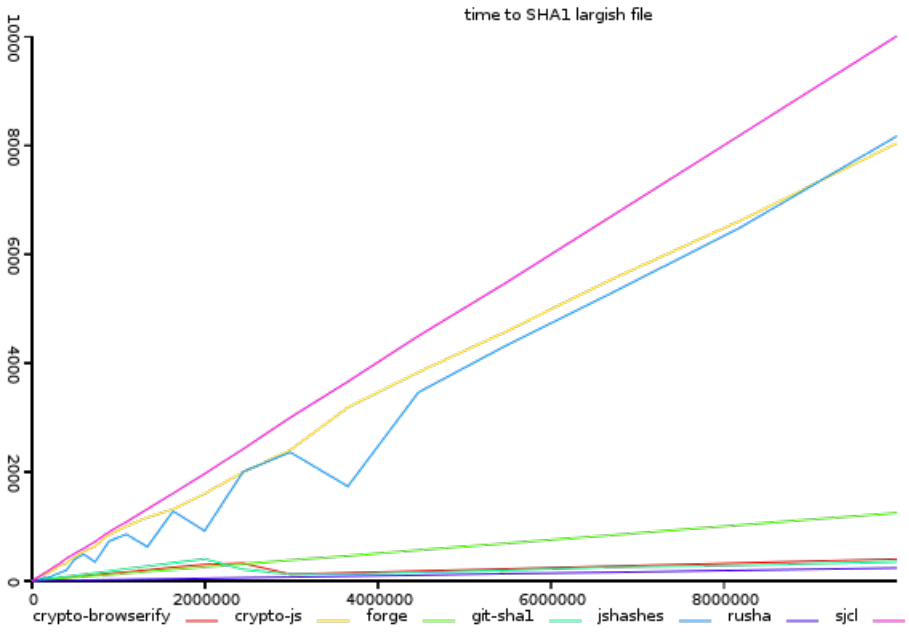

Figure 1: *y-axis shows total time taken, lower is better*

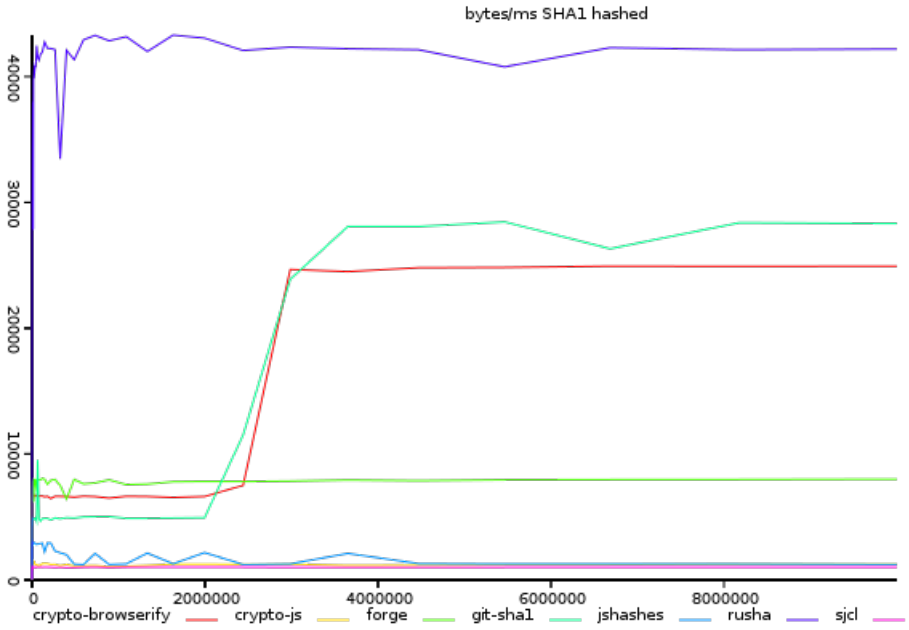

Figure 2: *y-axis shows size/time, higher is better*

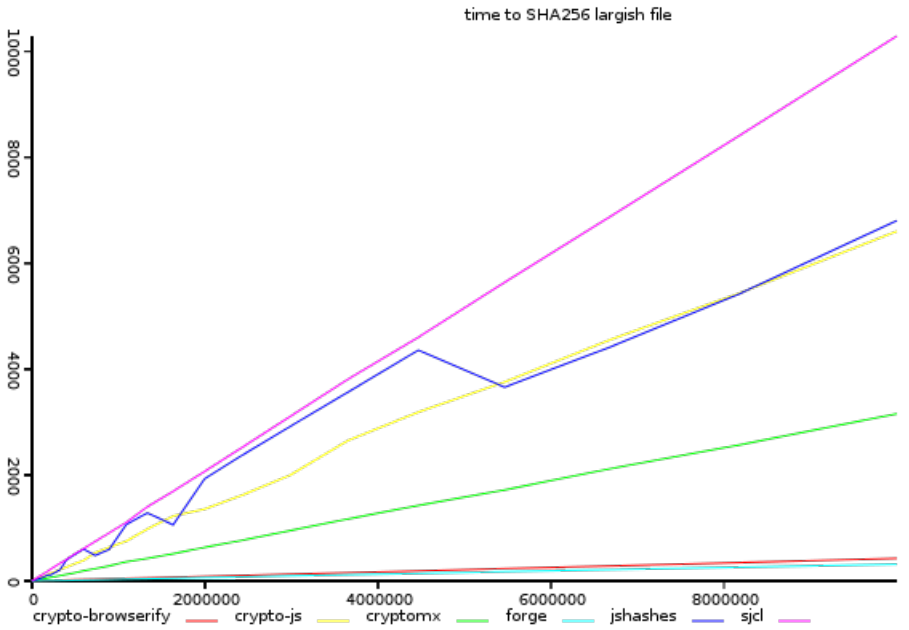

Figure 3: *y-axis shows total time taken, lower is better*

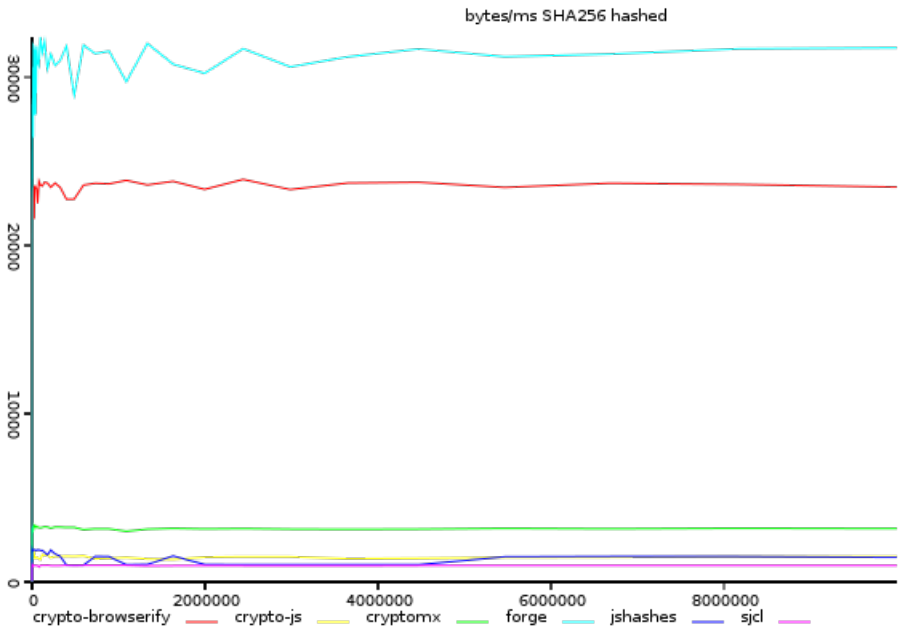

Figure 4: *y-axis shows size/time, higher is better*

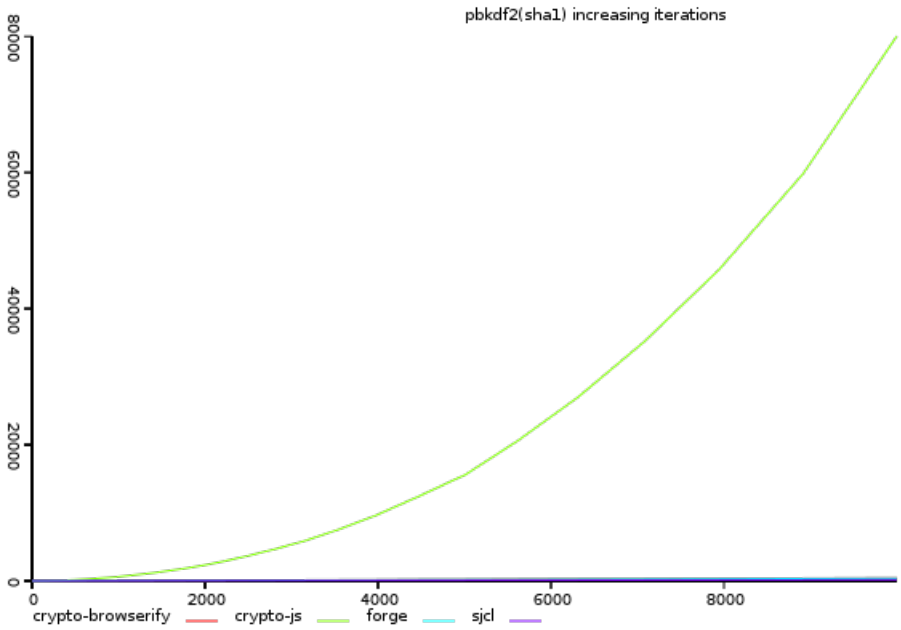

Figure 5: *y-axis shows total time taken, lower is better*

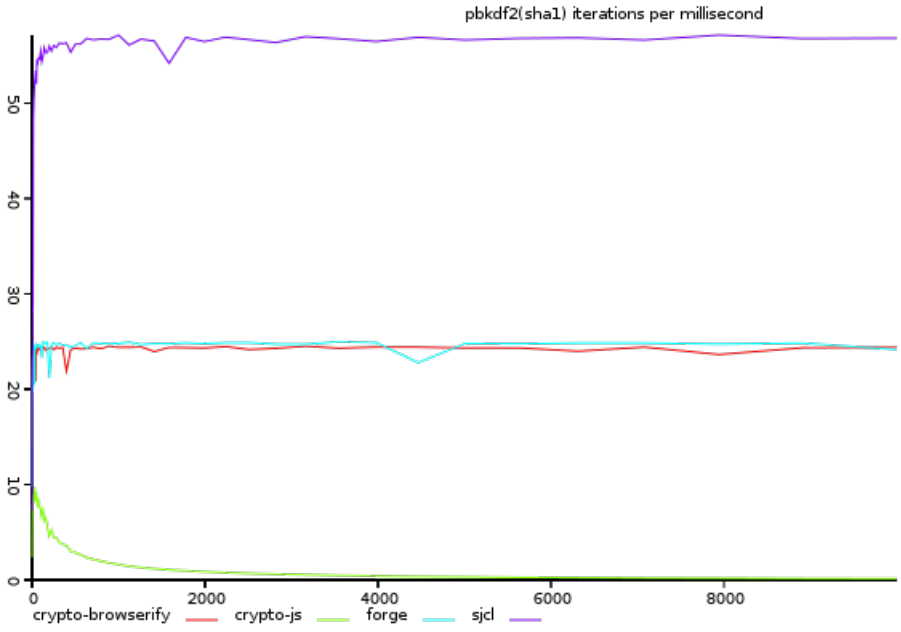

Figure 6: *y-axis shows size/time, higher is better*

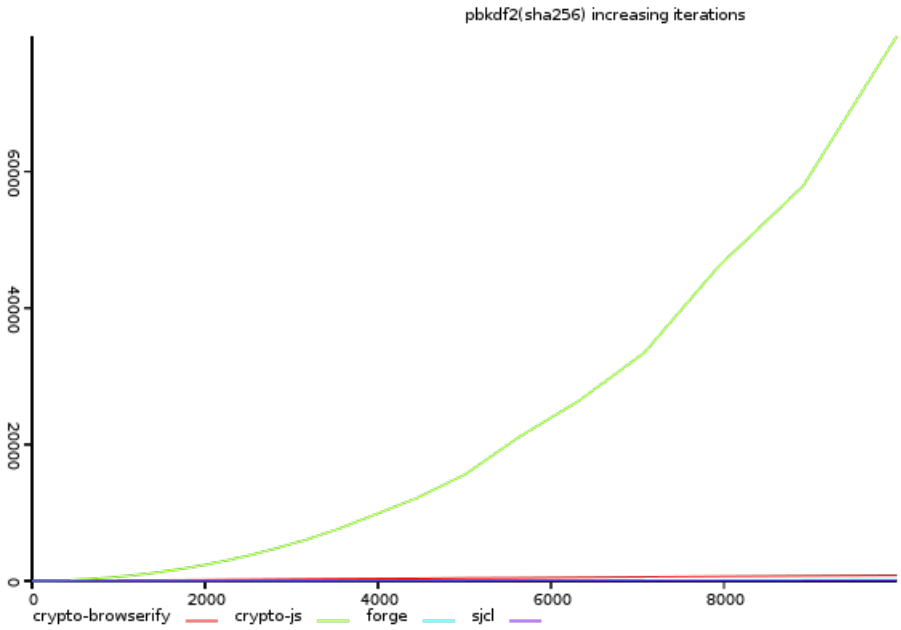

Figure 7: *y-axis shows total time taken, lower is better*

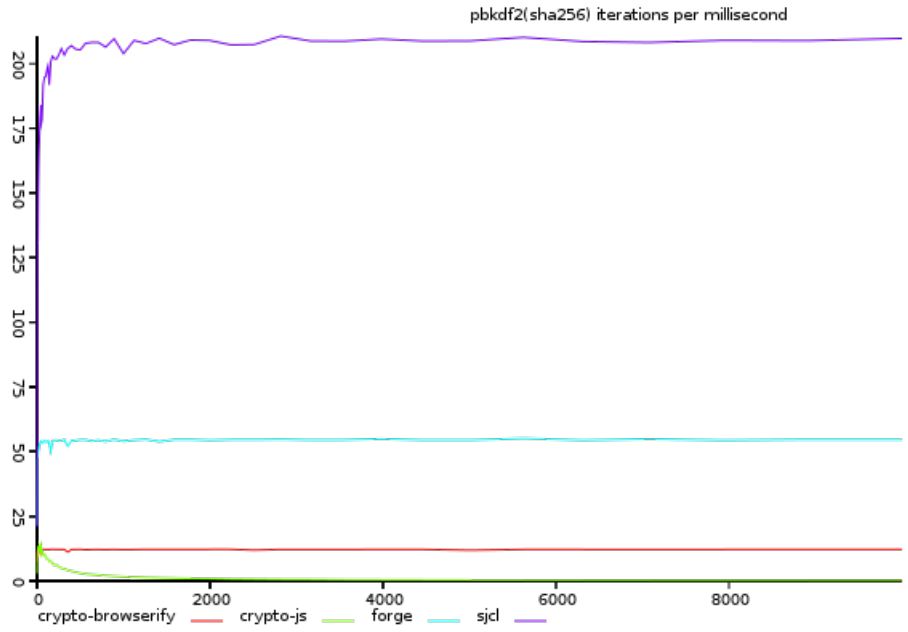

Figure 8: *y-axis shows size/time, higher is better*

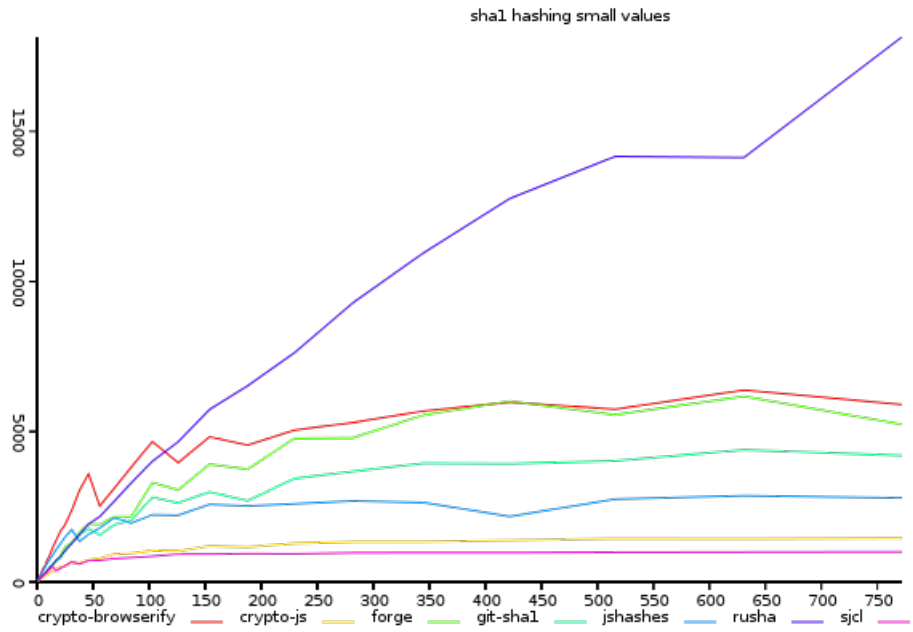

Figure 9: *y-axis shows size/time, higher is better*

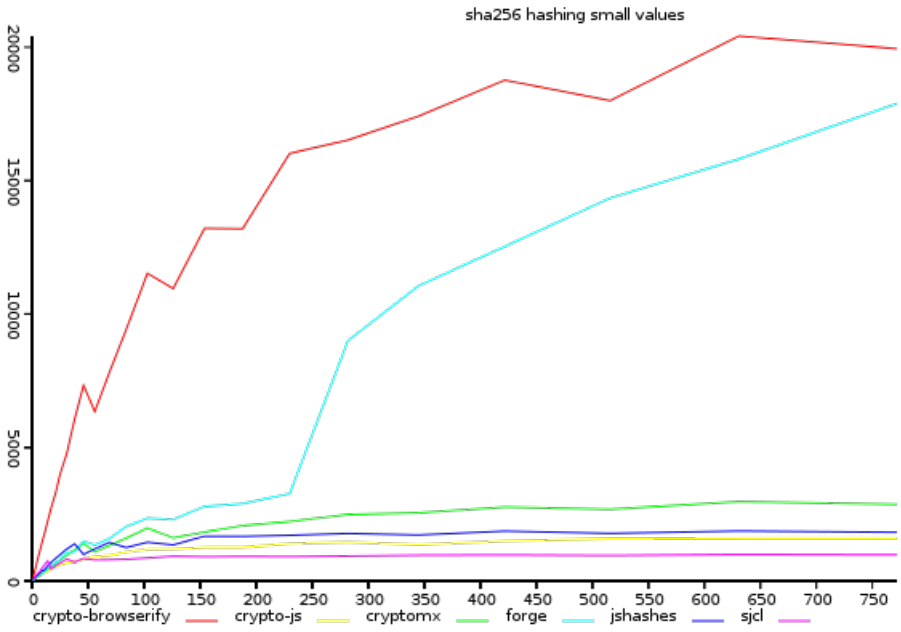

Figure 10: *y-axis shows size/time, higher is better*

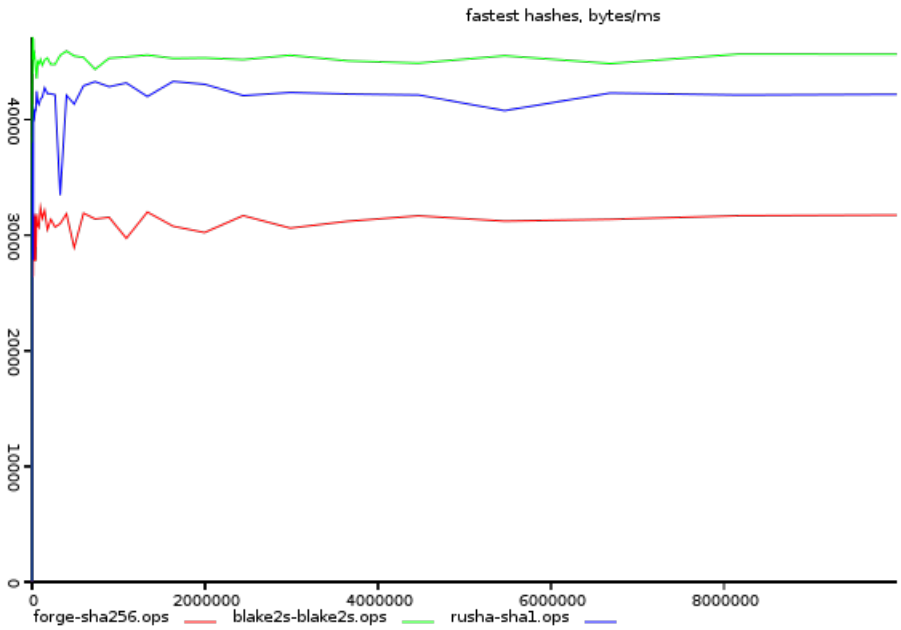

Figure 11: *y-axis size/time, higher is better*

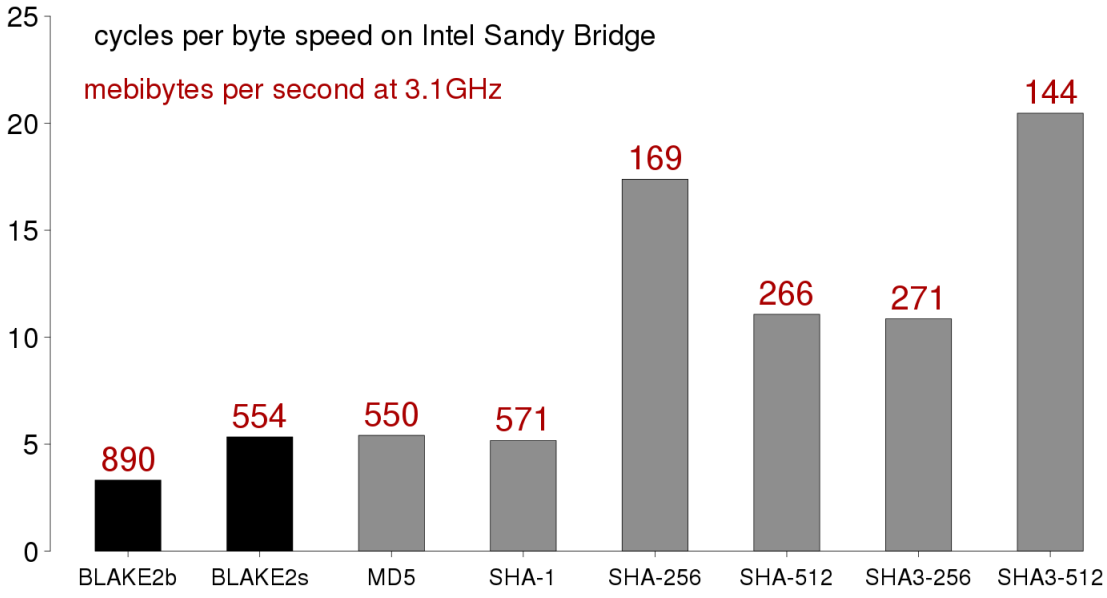

Figure 12: *lower is better*

8.2 JS Cryptography Library Tables

Add other crypto libraries using **Dominic Tarr**'s benchmark set [28].

The following tables have been made based on **Dominic Tarr**'s[28] benchmarks.

Table 1: Hashing 0-10MB Files /milliseconds based on [28]

Libraries	Sha1 (size)	Sha1 (hash)	Sha256 (size)	Sha256 (hash)
sjcl	- - - -	- - - -	- - - -	- - - -
crypto-js	- - -	- -	-	- - -
forge	+	+	+ + + +	+ + + +
crypto-browserify	+ +	+ +	+ + +	+ + +
crypto-mx	null	null	+	- -
git-sha1	+ + +	+ + +	null	null
jshashes	-	-	- -	- - -
rusha	+ + + +	+ + + +	null	null

Table 2: Key Derivation (pbkdf2) based on [28]

Libraries	Sha1 (time)	Sha1 (size)	Sha256 (time)	Sha256 (size)
sjcl	+ + + +	+ + + +	+ + + +	+ + + +
crypto-js	- - - -	- - - -	- - - -	- - - -
forge	+ + + +	+ +	+ + + +	+
crypto-browserify	+ + + +	+ +	+ + +	- -

Table 3: Hashing Small Files /milliseconds based on [28]

Libraries	Sha1 (size)	Sha256 (size)
sjcl	- - -	- - -
crypto-js	- -	- -
forge	+ +	+ + +
crypto-browserify	+ + +	+ + + +
crypto-mx	null	+
git-sha1	+	null
jshashes	-	-
rusha	+ + + +	null

Table 4: Fastest Hashes /milliseconds based on [28]

Libraries	Sha1 (size)	Sha256 (size)	blake2s (size)
rusha	+ + + +	null	null
forge	null	+ + + +	null
blake2s	null	null	+ + + +