haute école
neuchâtel berne jura

arc

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

Spring Project

HE-ARC 2016

Overclouds

Author:
Romain Claret

Supervisor:
Marc Schaefer

May 13, 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 he or she 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

L'initiative de ce projet est de créer une nouvelle génération d'outils pour le partage de données sur Internet. Lesquels sont adaptés à la paranoïa montée en flèche pour la préservation de la vie privée sur Internet ainsi que la conservation des connaissances pour nos prochaines générations.

L'idée est de donner la possibilité aux utilisateurs de ne plus compter sur les entreprises ou fermes de serveurs capitalisés. Avec Overclouds, tout le monde fait partie du réseau tout en maintenant leurs anonymats. Les utilisateurs se connectent les un aux autres librement avec des règles régies par la communauté.

Le réseau est un maillage démocratique de nœuds. Chaque noeud fait partie du consensus d'utilisateurs et est gouverné par les lois régies par le consensus.

Pour faire partie de la communauté Overclouds, les utilisateurs ont besoin seulement besoin d'une connexion Internet et un navigateur avec le support JavaScript.

Contents

2 Preface

2.1 Introduction

Today The world and more particularly the digital world has an important concern for privacy. Indeed, Edward Snowden's revelations on NSA's massive spying[24] led to a massive media scandal. People started to feel that their digital privacy was at stake by the worldwide "spies", either the government, companies, or even unknown threats yet. Resulting in evolved internet communities, that became an important part of the economic and politic world.

New behaviors emerged Right into the web users, a noticeable split has been made, into two classes of people. Individuals who do not know how to protect their privacy on the Internet, and people who do know how to protect their privacy. For the first group, some are not aware of the global privacy status; some don't care, and some don't know how to protect themselves. Concerning the second panel, we have some personal ethical problems, which led to this project.

The problem It is understood that people want to avoid advertising companies' and governments' tracking. It is also known that some people are scared that their stole private data could be exposed publicly. However, some private data must be sometimes public and some cases forced to be made public. From our humanistic and unpolitical point of view, we believe, that the data from terrorist groups (including their members and partisan) have to be denounced, made public and the locking the data into a secured safely. Why archiving and not just deleting them? We, again, believe that regardless how bad the data is (as seen today), it is still part of our history, and it is our duty to preserve them as a legacy for future generation to understand and learn from what we are mistakes today.

Intellectual Properties Read as IP. Our social and political evolution led to copyright laws, which prohibit the free sharing of any data such as art and knowledge by default. Indeed, our culture specifies that any intellectual discovery or advancement is by default protected by copyrights (we will not discuss the process here either the manipulation of the copyright system).

All this usually leads to knowledge or data retention, which makes us feel that the IP system, is a serious treat for our humanity legacy.

Economy However, it is also understood that with our economic system, people need to make a living out of the IP, which leads into higher protection for their data. Our digital method of protection is to encrypt the data with a digital key provided by the owner (generally companies). The encrypted data is then stored in data centers or on hard-drives (which could sometimes be bad for the long-term preservation). Owners then have the choice of sharing their (copyrighted) material and are usually also choosing to use encrypted manners to distribute them. The transmission protection is to unable people from looking at them without owners' consent (usually money) and redistributing them with consent (not making money out of it). So, the data is protected for the storage, during transmission, and we also have DRMs technology that only decrypts the piece of data a user is looking at on the fly. So, technically, the full data is never in clear anymore.

Game of cat and mouse Some people are protecting data, other are trying to break the protection to retrieve them. Moreover, they both use encryption technologies to secure their transmissions. Meaning that the owners are protecting their data, and the *thief* are also protecting the stolen data, to not by caught. In the end, the same data is being transmitted but never in clear. This procedure makes us believe that something is wrong with the system and that it could be considered as a treat for the humanity global knowledge and culture.

Legacy We are confronted with another ethical problem. We are entering into a fully encrypted data era, which will make all our data appear as a bunch of random noises that for various reasons are bad for us. We would like you to think about how all our global knowledge is and will be sorted shortly. Data such as art or knowledge, how will our next generations of humans be able to retrieve it in a few years? How can we be sure that the unbreakable keys (quantum proof, etc.) that we are making today will not be lost? Now think about the idea of randomizing (with encryption noises) our communication signals that we are sending into outer space? The signals that we are sending out to space are already today much different from what it looked like in the sixties. For example, purely analog signals (TV, FM

radios) are depreciated and got replaced with a digital transmission (DVB, DAB), and of course, most of them are encrypted (pay-to-watch tv, pay-to-listen radios). Moreover, our communication technologies involve into higher frequencies and always fewer consumption. All this is resulting from the fact that today, Earth radiated probably much fewer waves than before and seen as random noises.

Taking the wrong direction It is easily imaginable that soon a company will say: "Hey, we are specialized in transmitting encrypted data, nobody will ever know what you are doing, and when you are doing it. All you have to do is to buy from us, with a crypto-currency, a key every month.". See this has an evolution of the VPN services. We do not have a sharp opinion on neither it is a good or a bad thing. However, it will create a privatization of the data security, and it most likely to destroy any hopes of retrieving the encrypted data one day.

Overclouds The project aims into a different approach for protecting people's privacy and ensure a legacy for future generations. The main idea is that Overclouds is a consensus-driven anonymous network of nodes with storage and computation power. Each node is aware of other nodes, but they are unable by default to identify this owner on the main net (whatever the physical technology of communication at any given point of time). The consensus has limitless power over the network. It can, for example, decide to disclose the physical location of particular nodes, if they considered malicious or wrong for the rest of the network (such as terrorists, scammers, black hats, etc.).

Nice to have We also would like that Overclouds was designed not to act like random noises. Preferably, external and internal listener or watcher should see a nice mathematically driven signal. The mathemagical behind it would allow any source node to predict exactly how its data is will handle by default by the consensus. However, it should be unpredictable for other nodes, but still elegant to look into.

2.2 The Big Picture

After being introduced to the project, let us present our current vision for the outcome of Overclouds. Note that the result is not the bachelor project, allocated man-hours are clearly not enough. Of course, due to the early stage of the project, the vision may evolve. The bachelor work will aim to produce parts of the vision.

Global We are aiming at providing a product easy to use, technophobe-friendly, that does not require any external software. We are focusing on the browser experience; any browser should be able to run the network. The user would be able to use any hardware and main net connections to join the network, very low bandwidth, very low computational power, Hyperboria[41] compatible, or even nonexistent technologies yet. The security allows to for any user to be ISP anonymous and censorship-resistant. Plus, any user could access its personal *desktop* with data right from any browser on the main net.

The consensus A consensus-driven network with unlimited power over the entire network. Putting the digital democracy to the top. It works in harmony with the newly introduced concept of *Mesh-of-Trust* which glues identities and nodes and certifying nodes' hardware and identities. The mesh-of-trust plays the role of data manager and data tribunal.

Network A serverless mesh of self-aware and smart nodes. Every transaction is verified and is mathematically elegant. Intelligent enough for self-preservation, and bypass consensus decisions if they judged as self-destruction. Privacy is being an integrant part of any layer of the network. However, a consensus decision can always bypass any rule. A registry name could be available to access easily the network from browser's URL (.over?).

Storage The network is using cooperative nodes to store and process computational work. Data has many stats, private, public, time to live, or countdown to go public. Meaning that anonymous websites can be hosted.

Crypto-Currency Consensus-driven currency, the genesis block is modifiable. Nodes are automatically contributing to the network wealth. Units of the currency are rewarded by the consensus to identities as a reputation

value. The currency represents how an identity *respect* on the network, which could give him leadership powers like a stronger vote power in specific cases determined by the consensus.

Self updating An ultimate achievement would be a self-programmed network driven by the consensus.[31]

2.3 Original design

During the project bootstrapping, global solutions and concepts emerged as the first overview of the project. Below is the result of the initial concept. However note that during the effective research and analyses, the project has evolved, sometimes resulting in new architectures and new recommendations.

The mesh of trust Acting like a consensus of trusted nodes on a node network.

Certificates Each node has a unique nontransferable certificate, which can be regenerated as a new one. It is used to individualize nodes, and allow nodes to trust each other. The network is storing the certificates, and keep track of its proof-of-activity[5] ranking.

Indeed, the certificate level increases over time by providing proof-of-activity. The level is also influenced by the amount of trust given to other nodes from the network. Note that nodes alone are not aware of others trust interactions, they kept informed of the sum of nodes willing to share their data with it.

Identities Users can use any node from the network. For a user to identify himself, he needs to generate at least one unique identity. Identities are created from an existing node that is already part of the network. Each identity has a public and private key, a parent node (hardware of generation), and optional information (languages, avatar, name, etc.) for internal applications to the network.

An identity cannot be alternated. The user must generate a new one with the previous one was banned, and restart the process of acquiring trust from the network. Depending on situations, they must also need use another node.

Network requirements A node does not need an identity to be able to connect to the network. It connects automatically and integrates the mesh of nodes by giving storage and computational power. However, an identity requires a node to connect to the network and interact with it. So, a node works without an identity, but an identity always needs a node.

Democracy Identities can create votes and of course votes for submitted proposals, which are creations or modification of the rules ruling every transaction on the network. Each identity is weighted by default to the initial vote unit and can vote once. However, the consensus can decide to give more weight or votes to specific identities. By default, nodes can be used only once to give a vote (except if the consensus decides else-wise).

Bans Certificates and identities can be banned from the network after a trial or directly from the consensus. A flag, seen globally by the network, is raised for the banished nodes or identities. During a trial, a random amount of random identities is asked to vote on the event that led to the trial.

Flags Node can flag specific certificates or nodes on the network and apply rules on them, such as filters for minimum certificate level required to be able to connect to them as a relay. The flags are then used by the network, and route communications depending on the nodes' preferences.

For example, in the case of a node receiving data from another node that is not matching its filter, the data would be rejected at entrance. In the event of spam, which assumes that the attacker knows what destination (that he sees as void) to target, the target node will indeed consume power to deny requests. A solution has to be found for a case of a figure and prevent spam.

Another example, the node rerouting. A node can decide not to relay data. In this case, the data is either sent back, which can result in a load problem. Either the data is lost, which is in the event of a UDP-like protocol bad. In the case of a TCP-like protocol, the node would be searching for another node to go through. The second option could on another hand surcharge by searching new paths.

Storage Data is stored across the network. It is spread on the network as encrypted chunks belonging to an identity, the private and public keys for the data is also spread across the network and belongs to the network. The data and its keys have redundancy chunks, also spread across the network.

Data owners The owning rights are given and managed by the network to a specific identity. The owner can give as he wishes the reading, writing, and executing rights to any nodes or identity on the network. He can also set a public access for certificate or trust levels. The ownership is, of course,

revocable or transferable by the consensus. However, the owner can also transfer the ownership to another identity. Both parties must accept the transfer. Note that he cannot transfer blamed data.

Blames A node can anonymously blame chunks or nodes and leave a reason for the blame from the list of multilingual generic reasons. Note that the consensus can also create new categories. Each node can blame a specific data only once. A blaming identity is then linked to the note used to blame, by this mean it cannot vote more than once per data. Plus the first identity to use the node gets the blame validated.

Blames are telling the network that a node or a chunk is not appropriate to the other nodes. Multiple blames on a node may result in a ban of its certificate. Multiples blame on a specific chunk or chunks belonging to the same data resumes into a revocation of all rights given to nodes even the owner.

Data Tribunal Once the required amount of blames reached the network select not related random identities and asks them to rate the blame. The digital judges would be able to read the reasons left during the blaming phase. However, they are not able to read the content of the data itself. Except is the consensus decides otherwise. Owners can ask for a second trial if they think the judgment was unfair and add a generic argument. For the second trial, different unrelated random nodes will have to rate the blame again. The decision from the second trial is definitive. The data is either archived, either the owner and related nodes get their rights back.

Internet Service Providers They should not be able to interpret the network activity. They are only aware of encrypted tunnels made to random nodes. Like the Tor network, the gateway nodes never the same. It allows a censorship protection.

Internal crypto-currency Nodes are automatically retrieving units of the internal crypto-currency with their proof-of-activity. The network owns the currency. Rewards are given to identities for good behavior and participation. The currency is transferable to unbanned identities. However, an identity has no interaction access to its balance during the trial phase. If the identity is banned, its currency balance is returned to the network. The use of the

crypto-currency is not clear for the moment, but it could be used as a fuel like on Ethereum[25].

Backdoor The project is not friendly for third party authorities like governments, police investigations, companies, etc. However, the consensus has virtually unlimited power over the network and can model its democracy as it is pleased. We can easily imagine that the consensus decides to disclose all the pedophiles from the network with the goal that they would be punished by the *real world*. The consensus decides what is right or wrong, and morality.

Artificial Intelligence Based on today's technology it is unlikely to see a very smart AI emerge inside the project. However, we could imagine applications like self-preservation, meaning that the network could learn and decide by itself with the consensus is right or wrong.

Communication The speed and size of the transactions should be compliant with any bandwidth. For example, the network should be able to work on a network of ALIX node bidden to a Xbee connection. Alternatively, networks from the emergent 3rd world.

3rd parties Depending on the technologies used on the project, we could, in a first time, use third party services like Tor, Github, Twitter, BitTorrent trackers, etc. (note the public status and censorship of the example). The security could be compromised during the bootstrap phase in some cases because the ISP and others services could target and track nodes' activities.

Compromised users In the case of malicious software presence on a system used by the node solutions can be taken into account. Assuming that there is less than X% (exhausting value to estimate at this phase of the project) of malicious software designed for the network. It is indeed difficult to protect people from malicious people. We could add security layers, like a Pin or a passphrase, but if the user is running a keylogger, it will always be insecure. However, since the node can only be accessible from a unique hardware (certificate linked to the hardware), the stolen key could not be as useful as planned. However, the system could also be compromised by someone or something physical like the RUBBER USB[37], in which case the

hardware protection could be bypassed. Now we could think of security that only shows a virtual keyboard, but a specific program could sniff the mouse positions and actions. A solution could be to use a USB-key as key, but it could also be replicated if the key is writable. We could use an external hardware, but it would impact Overclouds' public attractivity. Moreover, in the end, how to be sure that the company making the encrypting hardware will not be hacked, resulting in the release of the algorithms for generating the keys? Another solution is to control the hardware and the software, assuming that there is no way for an external or internal entity to know what's the key. This last option would mean that Overclouds would have to be privatized somehow, and it would impact nature and vision of the project, by making part of the project proprietary.

Certificate or Identity Clones The network does not allow clones using the network at the same time. The consensus will decide which one is the real one, and the clone will not be ignored by the network, no peers would accept a connection from it.

In the case an identity is stolen, in the current state of the concept, the attacker would have full access to the identity's assets. However, a solution to avoid this concern would be to link an identity to a unique node. Forcing the user to have a different identity for each node it connects to. The attacker would have to have a clone of the identity key and a clone of the node certificate, which increase the difficulty of the attack. However, it is still possible if the attacker has access to the original hardware and have the technical knowledge to emulate the hardware running the clone of victim's node certificate. To increase the security again and solve this problem, we would have to add an extra layer of security with an secure additional hardware such as a NitroKey[64]. Now, those solutions are considered extreme, in general, going this far into security is not necessary. However, if those solutions enter into consideration, we could predict a substantial impact on new users willing to join the network.

Note that the consensus could be able to revoke a key or certificate. Protocol for asking the consensus to revoke an identity could be implemented. Alternatively, simply asking many people to blame the stolen identity or certificate.

Unanswered questions

- Should we take into account that the wired Internet speed only improves? ... Probably ...
- Should we start from a programmable network such as Ethereum or MaidSafe? Alternatively, should we start from scratch? ... Make, buy or adapt study ...
- Would it be possible to allow nodes to run programs on the network like on Ethereum or with an API like on Maidsafe[56]. ... Probably, it could be interesting to make bots for content sharing, selling, etc. ...
- Is it possible to pledge that a node will be able to connect to the network with a random node? What about firewalls? Would connections only be based on TOR hidden service help? Alternatively, I2P? or Freenet? Alternatively, services as such? ... The idea of only using a browser could be in jeopardy. ...

2.4 Objectives

The goal in for this Spring Project is to prepare the material and planning for the Bachelor Project. The work done during this project is mainly doing literature research and prototyping by iteration. Indeed, we are looking to help the start of the bachelor project with various defined aspects, concepts, and prototypes.

Can we provide a distributed data storage solution that prevents espionage and that simultaneously doesn't open a Pandora box of illegal and ethical data exchange? It is the question we will be trying to answer during the bachelor project.

As it is today, the data is centralized into specific operators like Google, Facebook, Dropbox, Tor hidden services, and much more (expect of services like Freenet[14]). They are usually using encrypted protocol to transmit data to their client but are most of the time stored in clear on their cloud servers. Sometimes the data is stored encrypted, but the operators must comply with the local laws of the data centers, which could compromise the security. The user could encrypt the data himself. However, technical knowledge is required and sometimes the rules of the storage operators does not allow homemade security due to laws they are submitted to. Note that during the data transfer, even by using SSL or TLS protocols, the IP address are visible. Solutions to IP anonymity are commonly provided by (trusted) VPNs, TOR[21], or darknet[18] services like Freenet, I2P[18], etc. Plus encryption keys could be stolen if vulnerabilities are found like heartbleed[60]. Indeed, advance security for anonymity requires some specific software to run, and could be discouraging or even dangerous for uneducated people. Plus, the more protection via encryption is used, more data is *lost* for the human legacy, we cannot archive or index encrypted data.

2.5 Specifications

The spring project aims to study the state of the art, provide solutions and achieve incremental prototypes that meet part or all of the following objectives. It is of course not required to reinvent the wheel.

- Study of the state of the art (for example https://www.usenix.org/conference/atc14/technical-sessions/presentation/bessani and Marcelo Pasin)
- Sharing and transfer of data ownership
- Data decentralization (distributed storage), which is encrypted by the network via cooperating nodes
- Pseudo or full anonymity from the ISP
- Rating and aging notion of the trust level from network's nodes via a mesh-of-trust.
- Using the mesh-of-trust to decision of accepting partial data storage
- Banishment of data and node certificates from the network (via a Data Tribunal)
- Private keys theft resistance, even if a heartbleed[60] type vulnerability is discovered on the nodes later on[79]

2.6 Management

An iterative approach has been used during the progress of the project. So, the SCRUM methodology was chosen. The Backlog and the Sprint advances can be found in the annexes.

Originally six sprints were planned. However, the status of the project was going as fast as intended. Finally, we ended up with four sprints and two replannings.

Indeed, an error was made by the student during the initial planning; he planned the scrums based on results and not man-hours (86h) which were in fact about 8.6h per week during ten weeks. However, the required work per week to produce the scrum results was largely above. We noticed at week three that something was not right, resulting into forcing to control that the 8.6 hours were not exceeded. We noticed that 43 hours were made instead of 34.4, and decide on week 5 to do a new planning with one week increase on the current sprint.

The planning was remade for the second time in week 6, 46 hours was produced at that point in time for 43 required. The number of scrum schedule got reduced into four sprints, and by this means moving two research themes into the bachelor backlog.

At the end of the 10th week, the student spent 131h for 81h required hours.

It is important to point out that the problems did not only come from the unrealistic first planning, expect if the research was done very lightly, but the prototype would not have been something to rely on. The student was a bit too much interested about the new world is was discovering and started to do maybe to deep research into each treated subjects, times flew much faster than expected, and time was missing in the end for the planning.

The last two weeks got very expensive in time with the goal to deliver the report with the research done, even planned research.

For the Bachelor planning, the man-hours will be more carefully looked at, and adjust the planning right away if something is not going has planned.

2.7 State of the Art

For the project initiation, it was important to do research with the goal of targeting the needs for existing knowledge and technologies. That information could potentially be used to help achieve this project. This subsection will expose the research done.

2.7.1 Similar products

To be straight forward. A comparable project working right from the browser without the help of any third party or background software is nonexistent. At least not from the public knowledge available with our research keywords.

2.7.2 Existing Networks

The most common form of networks approaching our project's vision are called *Darknets*[6] and they started to emerge during the years 2000ish[18]. They are all aiming to encrypt data transmissions and protect network's users from being spied on and bypass censorship.

Freenet[13, 14] Description from the official website [29]

> Freenet is free software which lets you anonymously share files, browse and publish *freesites* (web sites accessible only through Freenet) and chat on forums, without fear of censorship. Freenet is decentralised to make it less vulnerable to attack, and if used in *darknet* mode, where users only connect to trusted nodes (real life friends, etc.), is very difficult to detect.
>
> Communications on Freenet nodes are encrypted and are routed through other nodes to make it extremely difficult to determine who is requesting the information and what its content is.
>
> Each user contributes to the network by giving bandwidth and a portion of their hard drive for storing files. Files are encrypted, so generally the user cannot quickly discover what is in his *datastore*. Chat forums, websites, and search functionality, are all built on top of this distributed data store.

I2P[27] Description from the official website [43] Originally IIP[44].

> The Invisible Internet Project is an anonymous network, exposing a simple layer that applications can use to anonymously and securely send messages to each other. The network itself is strictly message based, but there is a library available to allow reliable streaming communication on top of it (a la TCP). All communication is end to end encrypted (in total there are four layers of encryption used when sending a message), and even the end points (*destinations*) are cryptographic identifiers (essentially a pair of public keys).

MaidSafe[56] Description from the official website [55]

> The SAFE (Secure Access For Everyone) Network is made up of the unused hard drive space, processing power and data connection of its users. It offers a level of security and privacy not currently available on the existing Internet and turns the tables on companies, putting users in control of their data, rather than trusting it to organizations.

Tor[21] Description from the official website [85] "Tor is free software and an open network that helps you defend against traffic analysis, a form of network surveillance that threatens personal freedom and privacy, confidential business activities and relationships, and state security."

ZeroNet[94] Description from the official website [93]

> Real-time updated, P2P websites using Bitcoin cryptography and the BitTorrent network. Zeronet is decentralized, open source software in Python aimed to build an Internet-like computer network of peer-to-peer users. It is not anonymous by default, but users can hide their IP address by using Tor which uses Bitcoin cryptography and the BitTorrent network.

Project Maelstrom Description from the official website [8]

> BitTorrent wants your help creating a P2P-powered web with Project Maelstrom. (Beta on Windows only) The Maelstrom Network - currently under hectic development - is gearing up to be able to provide users and developers a joined interface to each other. Unlike some distinct offerings out there, this isn't a social network, and it never will be. All your information is for your eyes only until you allow an application to use it. We're trying to get web applications to a first class status on the web, similar to how an application on your computer is tightly integrated into the rest of the computer.

Diaspora* Description from the official website [19] " The community-run, distributed social network. diaspora* is based on three key philosophies: Decentralization, Freedom, and Privacy. "

FlowingMail Description from the official website [26]

> FlowingMail is the name of a new decentralized, secure and encrypted email protocol. The most used email systems rely on a central server that receives, stores and forward the messages: FlowingMail is decentralized and does not rely on a central server to deliver the encrypted emails. The scope of the FlowingMail protocol is to hide the information being transmitted and the parties involved in the communication. The main component of the FlowingMail protocol is a Kademlia Distributed Hash Table (DHT), which is responsible for storing the encrypted emails while they are in transit and the certificates of the participants in the FlowingMail network.

Bitmessage[89] Description from the official website [7]

> Bitmessage is a P2P communications protocol used to send encrypted messages to another person or to many subscribers. It is decentralized and trustless, meaning that you need-not inherently trust any entities like root certificate authorities.

Retroshare Description from the official website [70]

RetroShare is free software for encrypted filesharing, serverless email, instant messaging, chatrooms, and BBS, based on a friend-to-friend network built on GPG. It is not strictly a darknet since optionally, peers may communicate certificates and IP addresses from and to their friends.

MediaGoblin[90] Description from the official website [59] "MediaGoblin is a free software media publishing platform that anyone can run. You can think of it as a decentralized alternative to Flickr, YouTube, SoundCloud, etc."

Movim Description from the official website [62]

My Open Virtual Identity Manager is a distributed social network built on top of XMPP, a popular open standards communication protocol. Movim is a free and open source software licensed under the AGPL. It can be accessed using existing XMPP clients and Jabber accounts.

The IPFS Project[4] Description from the official website [47]

The InterPlanetary File System is a new hypermedia distribution protocol, addressed by content and identities. IPFS enables the creation of completely distributed applications. It aims to make the web faster, safer, and more open. IPFS claims to be a Permanent Web with a new peer-to-peer hypermedia protocol.

Serval Project[33] Description from the official website [73] "Serval is a telecommunications system comprised of at least two mobile phones that are able to work outside of regular mobile phone tower range due thanks to the Serval App and Serval Mesh."

Open Garden Description from the official website [32]

Open Garden's technology creates a software-based network, also known as a peer-to-peer wireless mesh network, among participating mobile devices using WiFi, Bluetooth LE, and other technologies. Open Garden's innovations include: seamless device discovery and pairing, offline identity, a proprietary network protocol

for addressing and routing messages off-the-grid, distributed algorithms for managing mesh networks, advanced traffic management (multi-hop, store and forward), and battery use reduction.

Hyperboria[41] Description from the official website [42] "A community of local Wifi initiatives, programmers, and enthusiasts. They are running a peer-to-peer IPv6 network with automatic end-to-end encryption, distributed IP address allocation, and DHT-based Source Routing."

2.7.3 Transfer Protocols

Route-based VPN [68]

> The use of VPN Tunnel Interfaces (VTI) introduces a new method of configuring VPNs called Route Based VPN. This method is based on the notion that setting up a VTI between peer Security Gateways is much like connecting them directly.

Commotion Description from the official website [16] "Commotion is an open-source communication tool that uses wireless devices to create decentralized mesh networks."

Hubzilla Description from the official website [40] "Hubzilla is a powerful platform for creating interconnected websites featuring a decentralized identity, communications, and permissions framework built using common webserver technology."

Tent Description from the official website [83]

> Tent lets you control your data instead of handing it over to service and app providers. Just like email, you choose a provider or can set up your own Tent server. Tent data and relationships are portable so you can change Tent providers easily at any time.

cjdns Description from the official website [12]

> Networking Reinvented. Cjdns implements an encrypted IPv6 network using public-key cryptography for address allocation and

a distributed hash table for routing. This provides near-zero-configuration networking and prevents many of the security and scalability issues that plague existing networks.

WebRTC Description from the official website [34] "WebRTC is a free, open project that provides browsers and mobile applications with Real-Time Communications (RTC) capabilities via simple APIs. The WebRTC components have been optimized to best serve this purpose." http://iswebrtcreadyyet.com

Open Peer Description from the official website [39]

> Real-time Communications Protocol & Specification. What further reduces the proliferation of peer-to-peer is a lack of standardization, openness and ubiquity of the technology. The standards bodies have been working for years on firewall traversal techniques and standardization of the approaches and a new joint effort called WebRTC between the W3C and IETF on how browsers can directly communication between browsers to move media. This joint effort does not specify how signalling happens between peers so it's not a complete solutions on its own.

freedom.js Description from the official website [28]

> It's a framework for building peer-to-peer (P2P) web apps. Easily create social applications that work in modern web browsers, Chrome packaged apps, Firefox extensions, node.js, and native mobile apps. freedom.js apps are just JavaScript, so they can be distributed as packages on an app store or hosted on static web servers.

2.7.4 Cryptography

Stanford Javascript Crypto Library[78] [87] "The Stanford Javascript Crypto Library (hosted here on GitHub) is a project by the Stanford Computer Security Lab to build a secure, powerful, fast, small, easy-to-use, cross-browser library for cryptography in Javascript."

MSR JavaScript Cryptography Library Description from the official website [61] "The Microsoft Research JavaScript Cryptography Library has been developed for use with cloud services in an HTML5 compliant and forward-looking manner."

MDN Description from the official website [58]

> The Crypto interface represents basic cryptography features available in the current context. It allows access to a cryptographically strong random number generator and to cryptographic primitives.
>
> An object with this interface is available on Web context via the Window.crypto property.

W3C Description from the official website [76]

> This specification describes a JavaScript API for performing basic cryptographic operations in web applications, such as hashing, signature generation and verification, and encryption and decryption. Additionally, it describes an API for applications to generate and/or manage the keying material necessary to perform these operations. Uses for this API range from user or service authentication, document or code signing, and the confidentiality and integrity of communications.

Google Closure Description from the official website [35]

> The Closure Library is a broad, well-tested, modular, and cross-browser JavaScript library. You can pull just what you need from a large set of reusable UI widgets and controls, and from lower-level utilities for DOM manipulation, server communication, animation, data structures, unit testing, rich-text editing, and more.

forge Description from the official website [20] "The Forge software is a fully native implementation of the TLS protocol in JavaScript as well as a set of tools for developing Web Apps that utilize many network resources."

jsHashes Description from the official website [49] "Fast and dependency-free cryptographic hashing library for node.js and browsers (supports MD5, SHA1, SHA256, SHA512, RIPEMD, HMAC)"

crypto-broserify Description from the official website [81] "The goal of this module is to reimplement node's crypto module, in pure javascript so that it can run in the browser."

BLAKE2[71] Description from the paper Analysis of BLAKE2 [36] "Recently proposed and already in use tweaked version of the SHA-3 finalist BLAKE."

2.7.5 Block-Chains

So much hype in this, but what is it? Everybody relate it mainly to *Bitcoin*[72], 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 [72] Why did everybody already hear the word *Bitcoin*? It is the first successful implementation of a distributed crypto-currency as described partially by Wei Dai in 1998[91]. The foundation of Bitcoin is the assumption that money could be anything (object, record, stake, etc.) accepted as payment for goods or services by a consensus (country, social, economical, etc.). Designed with the idea of using cryptography as proof of existence and transfer of virtual assets (proved to exist). Rather than relying on a central authority (trusted third party), Bitcoin is decentralized, meaning that it works with the network consensus. It uses the peer-to-peer technology to operate with the transaction management and verifying that the virtual assets is carried out collectively by the network.

Ethereum [25] 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[54] 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[15] library and the maximum cap of coins is 84 million (4 times more than Bitcoin).

Dodge Coin [57] 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 [50] Based on the paper of Scott Nadal and Sunny King [51] for the Proof-of-Stake Peer Coin was born.

2.7.6 Decentralized applications

YaCy Description from the official website [92] "YaCy is a free distributed search engine, built on principles of peer-to-peer networks. Its core is a computer program written in Java distributed on several hundred computers, so-called YaCy-peers."

2.7.7 Technologies

QEMU JS [3] "It is a generic and open source machine emulator and virtualizer."

PeerJS Description from the official website [80]

> PeerJS is a service which makes it easier to build a chat room using the present WebRTC's PeerConnection API. The PeerConnection API proposes to be able to send data, video etc from one user-agent to another without the need for it going through a server. PeerJS handles this handshake with a simple Socket.IO backend server.

PeerShare Description from the official website [17]

> PeerShare is a P2P file sharing website that uses WebRTC technologies to allow uses to send and receive files without going through any servers. PeerShare is mainly built on Javascript and jQuery and uses PeerJS as a WebRTC API.

PeerSurf Description from the official website [38] "PeerSurf is a demo (and kind of a library) of P2P websites powered by WebTorrent"

OpenWebRTC Description from the official website [65]

> With OpenWebRTC you can build native WebRTC apps that communicate with browsers that supports the WebRTC standard, such as Chrome, Firefox and Bowser. OpenWebRTC is especially focused on mobile platforms, with powerful features such as hardware accelerated video coding and OpenGL-based video rendering.

PeerServer Description from the official website [74]

> This system allows you to quickly create a decentralized, short-lived web application where all the content lives within your browser. The traditional server only performs the initial handshake between the client-browsers and the client-server; your browser serves all other content peer-to-peer.

3 Analyses

3.1 Block-Chains

3.1.1 Worth it?

It is 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 did not expect yet.

Yes, from today's perspective, the timeframe is pretty significant, it is worth the interest.

3.1.2 What's next in crypto-currency?

As of today, considering that the technology of block-chains will not 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.). Best case scenario, the hardware technology continues to involve, as well as the required computational power. (Note that we are currently brute-forcing the solutions.) Moreover, 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 will not 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 ex-

ample, 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 is the hashed public key.

However, 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 is again profitable to mine?

So, we do not know if it will be a next big crypto-currency. In my opinion, I would bet on Ethereum. However, again, it is personal.

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**[88] (2013a) and more recently (2016) the Homestead version of Ethereum [22].

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[23, 48]. It is 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 is globally something that holds. In our case it is more like a flag can keep a land (a claimed property).

Proof-of-Stake?[51] 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). Moreover, 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[30] 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[53]. Indeed, signatures are very easy to produce, and they will not 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 are not 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 is usually called the *weak subjectivity* method.

Ghost From the full name, Greedy Heaviest-Observed Sub-Tree protocol, introduced by Yonatan Sompolinsky and Aviv Zohar[77], it allows the PoW consensus to work with much lower latency than in the blockchain protocol from Satoshi Nakamoto [72], 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 introduces 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 is Casper[11]. This protocol is based on Ghost, and must be integrated into Ethereum for the Serenity[10] 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 release). 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 into scaling the block-chain technology.

3.2 Proof of Activity

The protocol from Bentov, Lee, Mizrahi and Rosenfeld [5] which is implemented into PeerCoin (and its clones), is considered as a 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 is 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 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 is 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 is 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 is 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 many 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[84], which is depending on this kind of funding to live, it is pretty bad. Plus, since many 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 they have a lower reward. However, 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 is 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? We think that it is amazing!

3.3 Consensus

Read this subsection as a teaser from the final project analysis on the consensus concept. Indeed, the time allocated for this research and analyze was null. However, knowledge grows over time even if the research was targeting something else.

3.3.1 Consensus != Block-chains

The block-chains technology being very popular nowadays, it sometimes can put eye cups on our field of decisions.

Block-chains have indeed proved that it works as a consensus. However, a consensus with highly fault tolerant networks and overcome the Byzantine (Two Generals Problem) is not something that only block-chains have.

Just for taking one example, Maidsafe[56] uses another technology to obtain a consensus[63]. Instead of using the whole network to validate a transaction, they give the consensus role to a random group of nodes.

Comparing They both have pros and cons of course.

- Block-chains
  - Pros: Shared global record of all transactions.
  - Cons: The chain can be gigantic (Bitcoin more than 60GB at the moment), and the file must be synced between all network's 6000 plus. Network speed. nodes[2].
- MaidSafe
  - Pros: Bandwidth speed limitations only. Low data storage consumption.
  - Cons: Small groups of nodes are playing the role of consensus for transactions. Nodes could never be aware of transactions that happened elsewhere if they are not related to them at any point in time.

3.4 Architecture

While doing research, the concept and based had to be pretty clear to help us focus on the right track. We then made an architecture for the project. Of course, this architecture is not definitive but contributes to the understanding of what we are trying to do. The latest global architecture version is found on figure 8 in the annexes.

The main point of this architecture is to create a consensus-driven network. The nodes running on machines connected to the network are acting not more than clients.

paragraphConsensus-driven Working as a black-box, the goal of the is to applying the rules made by the identities. The security between the nodes is controlled by Overclouds, indeed, for client's point of view, they are talking with the network and not a particular node from the network. The network work is split into encrypted and distributed chunks to nodes. Note that the chunks are encrypted for the specific node as the keys are stored and owned by the consensus. Each node is participating by default to the network storage and computation power.

Storage Controlled by the consensus and being the second part of Overclouds' black-box, storage allows the identities to save data into the distributed nodes of the network. Integrity checks are done by the consensus as multiple nodes are doing the same computation work and chunks storage (including redundancy).

Nodes Considered as clients to the network, they have a trust level, which is stored on the network. This level increases over time with proof-of-participation and correct integrity checks. Identities' trust level is rising on the other hand for good behavior. This last point is described in the Big Picture chapter.

Figure 1: Latest schematic architecture version seen from a User.

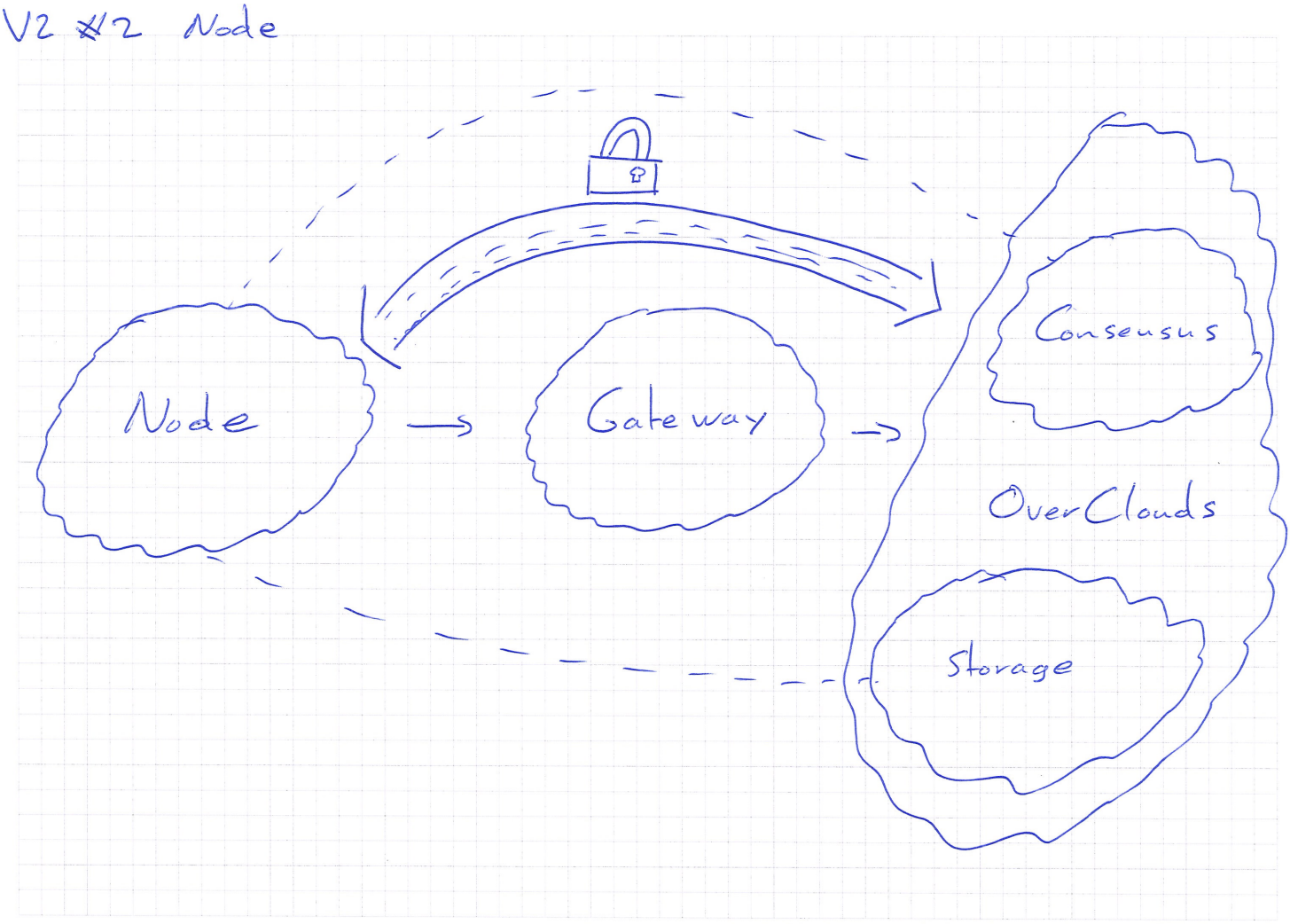
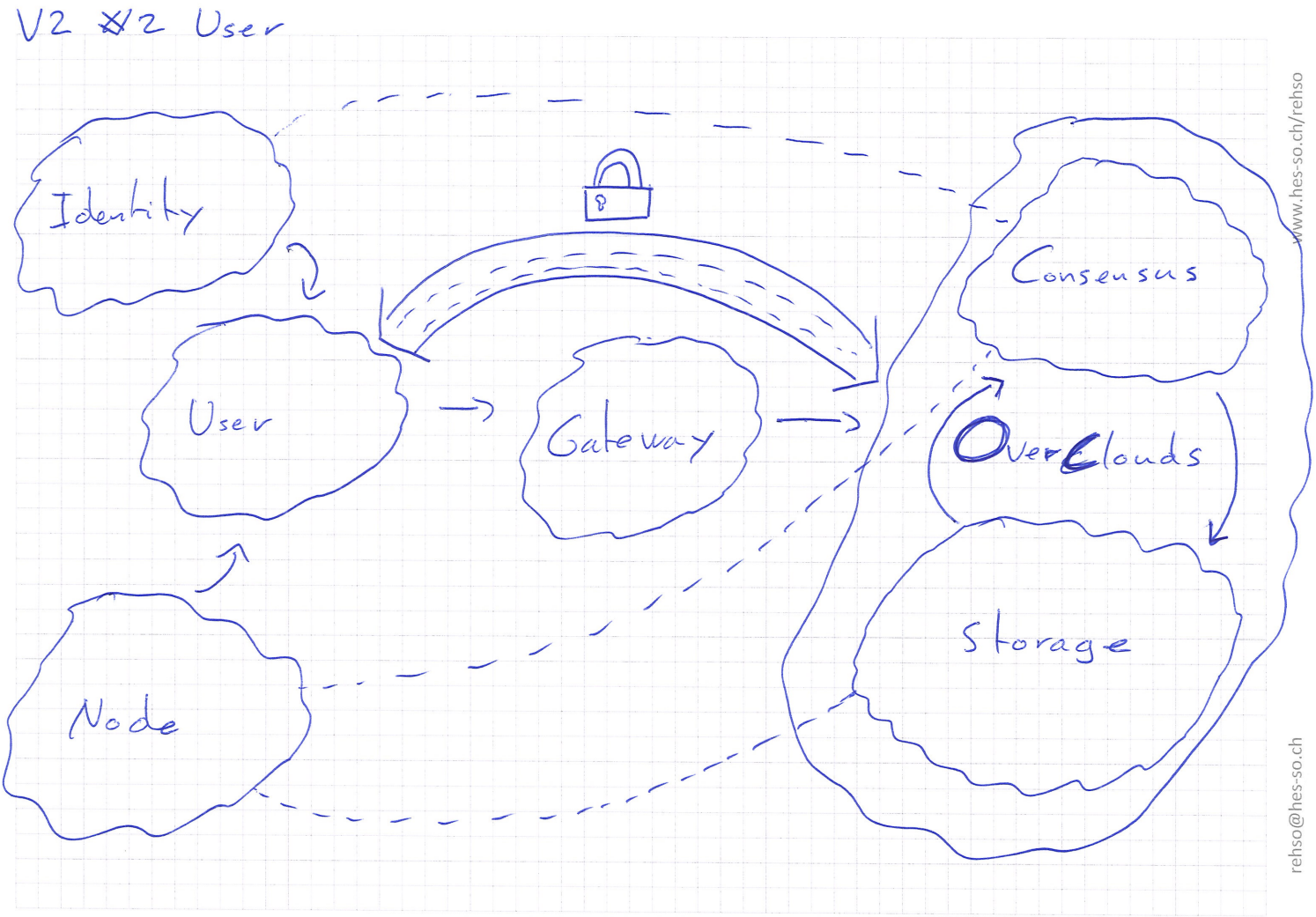


Figure 2: Latest schematic architecture version seen from a Node.



3.5 Communication

Considered to be the most important of the project, without internodes communication, Overclouds has no meaning. We are also aiming at a browser only system, leading into looking at modern technologies written in Web-driven languages.

Propagation We started to look at homemade solutions. In the cases of communication by data propagation, the path optimization will not be possible, and the global status of the network will be tough to maintain. Plus, it will not be easy to be sure that the chunk arrived at the destination. However, it is easy to push data into the void. Also, note, that a protocol must be made to avoid to the propagation of the source address and preserve anonymity.

3rd parties The most common way to do peer-to-peer from the browser is to relay on a 3rd party service to do the signaling and put peers into relation. To preserve the anonymity and security integrity, we must, in this case, maintain a list of *trusted* nodes (gateways) by the consensus to join the network. We could use TOR technology, Figure 9 in the annexes, but a browser only integration is not done yet. The 3rd party services could be either servers running custom code, for example, wrote in NodeJS, either companies specialized into signaling services like peer5[66], openpeer[39], peerjs[80], etc. In the second case, we must give even more trust into their services (hosting and signaling), we are hardly favorable on this option, how can we ensure their services will not be shut down or will not censor our project because they do not like it or if they are asked to do so?

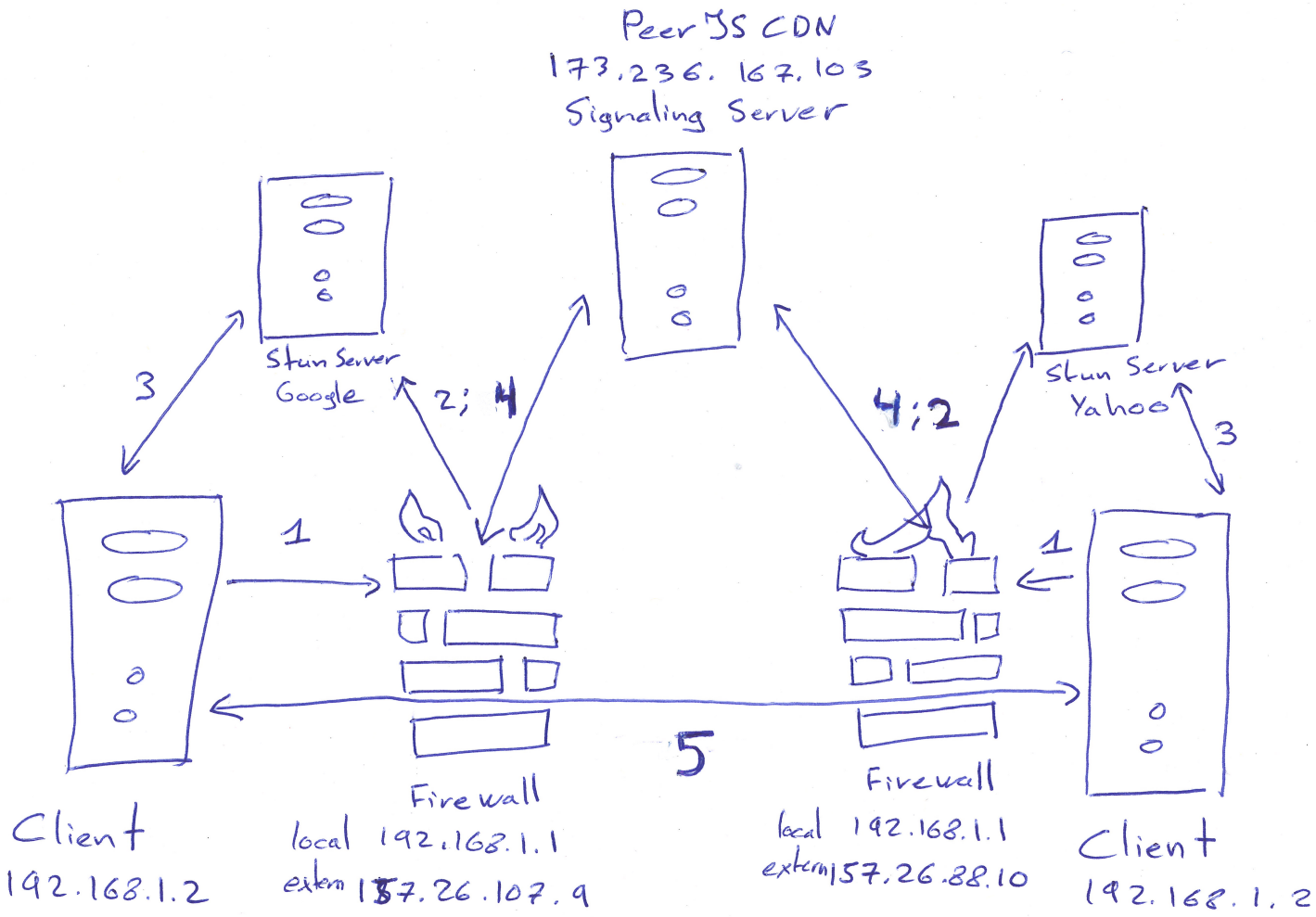
Organization's Server If the 3rd party servers solution is selected. It could be interesting to create a protocol that gives the consensus the control over the servers and manage the costs. Indeed, a consensus-driven payment via crypto-currencies directly to the 3rd party host could be interesting, but it would mean that the consensus owns capitalized crypto-currencies. We could imagine that the network would mine crypto-currencies.

3.5.1 WebRTC

Pushed by HTML5 standards and companies like Google, Mozilla, Opera, etc. WebRTC[34] provides a clear API, optimized for Real-Time Communications (RTC), in an always growing developer community, and the major browsers supported (Chrome, Firefox, and Opera). Note the Chrome and Firefox have interoperability. However, a server is needed for the signaling (sort of tracker).

Datachannel[69] Based on top of the basic WebRTC, and almost real-time, the main limitation is peers' speed connection and Javascript itself. It has two modes, **reliable mode**, which provides TCP-like guaranteed packets' order and no loss. Moreover, the **unreliable mode** which provides like UDP no guarantees for the data transfer. It allows encrypted communication and data transfer between compatible browsers (Chrome and Firefox). However, a threat still exists, it is people. Indeed, that can send malicious data like a virus to other peers. Note that the transfer is said to be secure at the moment, but it is not infallible, it is recommended to encrypted the data before the transfer. In Overclouds, we would encrypt each chunk. Like the basic WebRTC, linking peers is done by signaling servers. The figure 3 schematize an advanced procedure to make a data channel between two clients, which should prevent censorship or freaky firewalls.

Figure 3: *Advanced Data Channel*



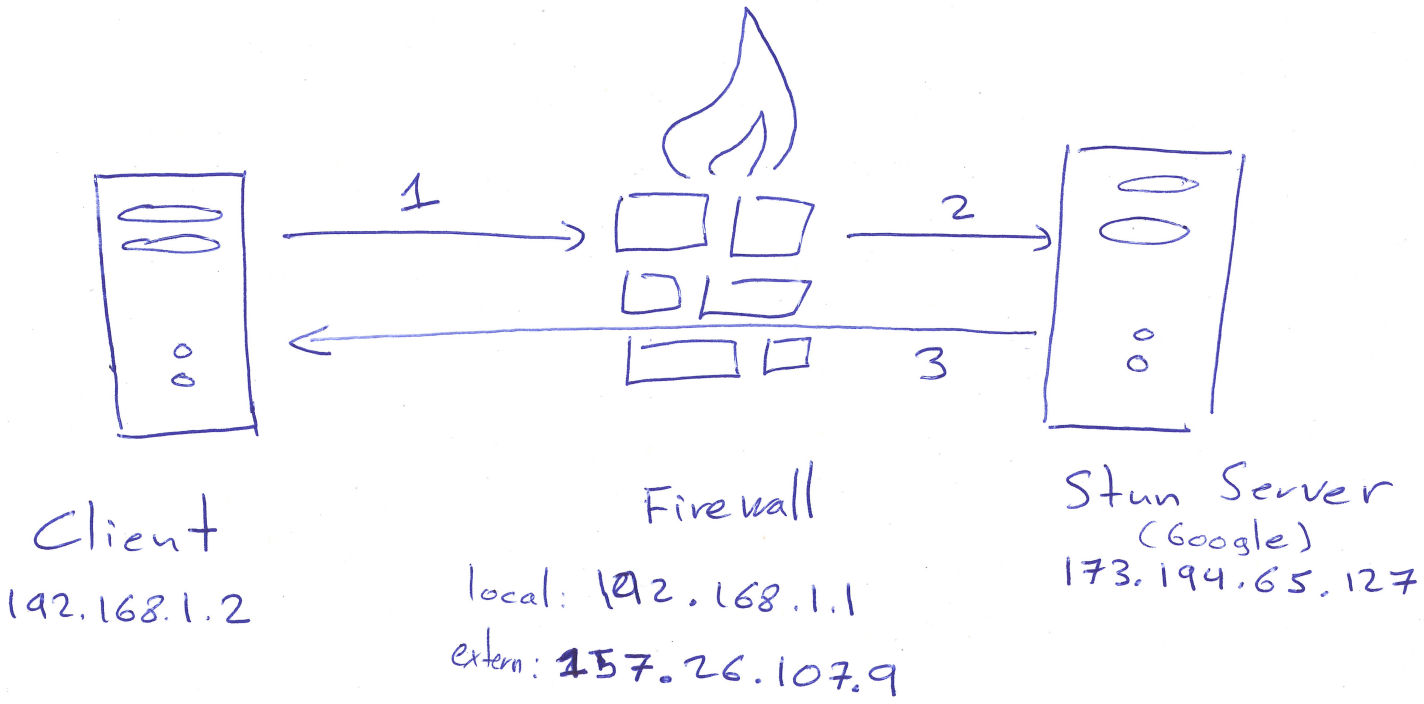
ORTC On the other end, Object real-time communications technology is pushed by W3C & Microsoft. It is younger than WebRTC, 2011 vs. 2014, the main difference[75] between the two protocols is the supported media protocols and also the fact that ORTC does not have a standard for its signaling protocol. It is not clear today, what major differences will be between them in the future.

3.5.2 Overclouds

By taking the pros and the cons of the previous communication solutions. Our choice goes to **WebRTC Datachannel** is the close technology to our needs. However, the signaling requirement was challenging. The solutions would be to have a trusted signaling node list controlled by the consensus.

Serverless The main problem we were confronted with while doing research on WebRTC, was the incompatibility with Overclouds' vision of browser only need, read as the user should not install any software expect a browser to be able to connect to the network. We finally found a solution to use the very promising WebRTC, however, the downside is the signaling procedure, which centralizes the tracking of nodes on the network, but it could be accepted if the server is trusted and maintained by the consensus. To our greatest pleasure, during the implementation, we found a solution to make the signaling procedure serverless. Users will still need to communicate their secret peer address somehow, but the addresses will not be managed via a 3rd party. The next step will be to find a procedure to transmit the peer address securely. The figure 4 schematize how the hole is made in the firewall with STUN servers.

Figure 4: *Serverless STUN Scheme*



3.6 Cryptography

Privacy Being an integral part of Overclouds, research has been made to find the best type of client-side cryptography (right from the browser as the highest priority). We were looking for a fair middle ground between performance and security.

3.6.1 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[78], MDN[58], W3C[76], Google Closure[35], or msrCrypto[61].

Then we looked at other crypto libraries such as forge[20] (a native implementation of TLS in Javascript and tools designed for crypto-based and network-heavy web apps), jsHashes[49], crypto-browserify[81], 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 is a pretty bad idea if we do not have a team dedicated to it and a pretty strong community to test it out. Even large enterprises 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 [46], or a procedure for Digital Signatures[45].

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

3.6.2 Comparison of some JavaScript Cryptography Libraries

Based on the table 1 below and the following referenced tables we can notice that **sjcl**[78], **crypto browserify**[81], and **forge**[20] algorithms have been optimized for defined objectives.

Table 1: Key Derivation (pbkdf2) based on figures 14, 15, 16, 17

Libraries	Sha1 (time)	Sha1 (size)	Sha256 (time)	Sha256 (size)
sjcl	**Amazing**	**Amazing**	**Amazing**	**Amazing**
crypto-js	Awful	Awful	Awful	Awful
forge	**Amazing**	Nice	**Amazing**	Okay
crypto-browserify	**Amazing**	Nice	Good	Ugly

Also see Tables 3 and 4 and their related Figures 10, 11, 12, 13, 18, 19

Read the table as going from Amazing to Awful.

3.6.3 A killer

After taking time doing research about the above algorithms, we came across a pretty amazing algorithm based on Sha3: **BLAKE2**[36, 71].

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)

3.6.4 Now what

If we look at the figure 5, 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 2 with the figure 6, we notice that BLAKE2 is in its own category.

3.6.5 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 does not work for whatever reason, we will certainly go with forge, crypto-browserify, or sjcl. The problem with forge and crypto-browserify

Figure 5: *y-axis size/time, higher is better*

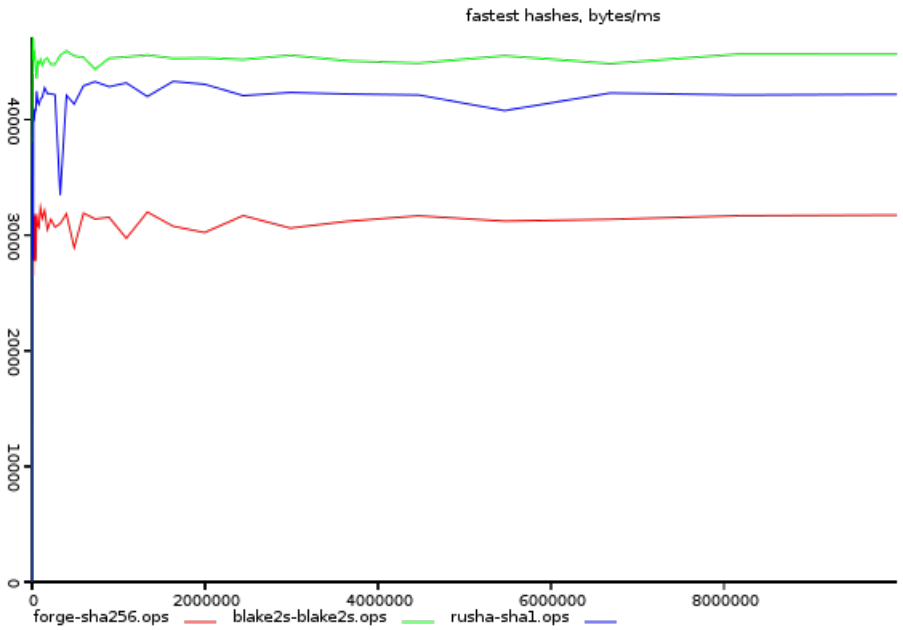


Figure 6: *lower is better*

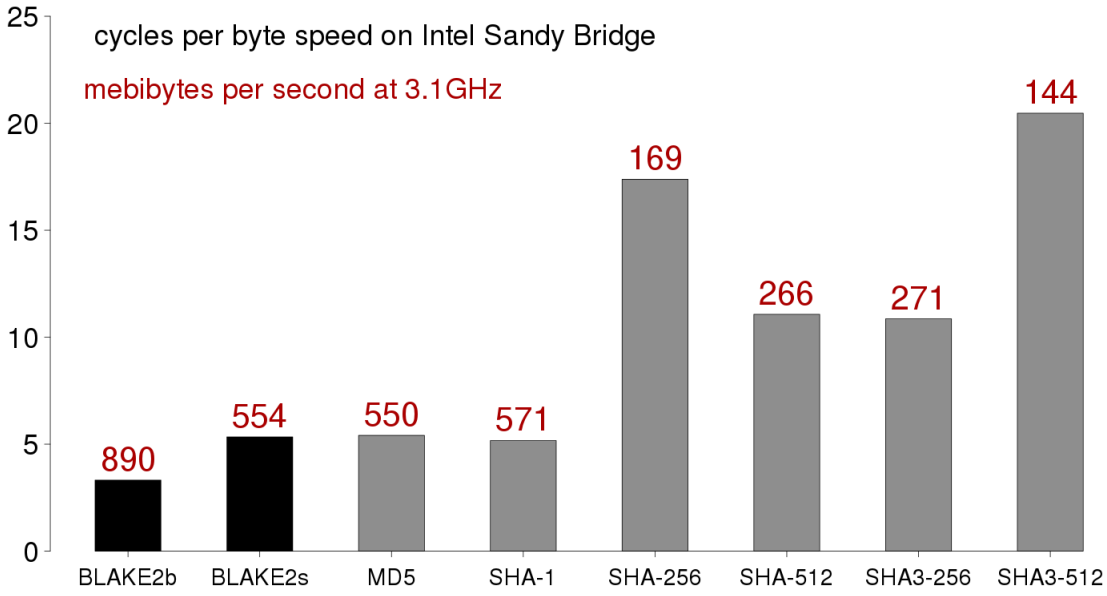


Table 2: Fastest Hashes /milliseconds based on figures 5, 6

Libraries	Sha1 (size)	Sha256 (size)	Sha3 (size)
rusha	**Amazing**	Neutral	Neutral
forge	Neutral	**Amazing**	Neutral
blake2s	Neutral	Neutral	**Amazing**

is that we must trust a company or an individual. With sjcl however, we are confident into a scholar institution.

3.6.6 Special Mention

We researched also PGP[67] and GPG[52] technology. We conclude that it is still a good encryption protocol, but too heavy for Overclouds.

4 Implementations

4.1 Communication

During the analyze, it was concluded that the best solution would be to have a consensus-driven list of nodes. We focused during this project on the base of the communication between node. We then started with the PeerJS' Open Source technology that allows us to run in the cloud or on personal servers a signaling server with simple APIs. During the prototyping and testing, we had the pleasure to find a serverless procedure for signaling.

4.1.1 PeerJS

The first version of the prototype used PeerJS' APIs and its free server for developers (maximum of 50 concurrent connections). Note that developers are encouraged to run their own signaling servers. The figure 7 schematize what was implemented.

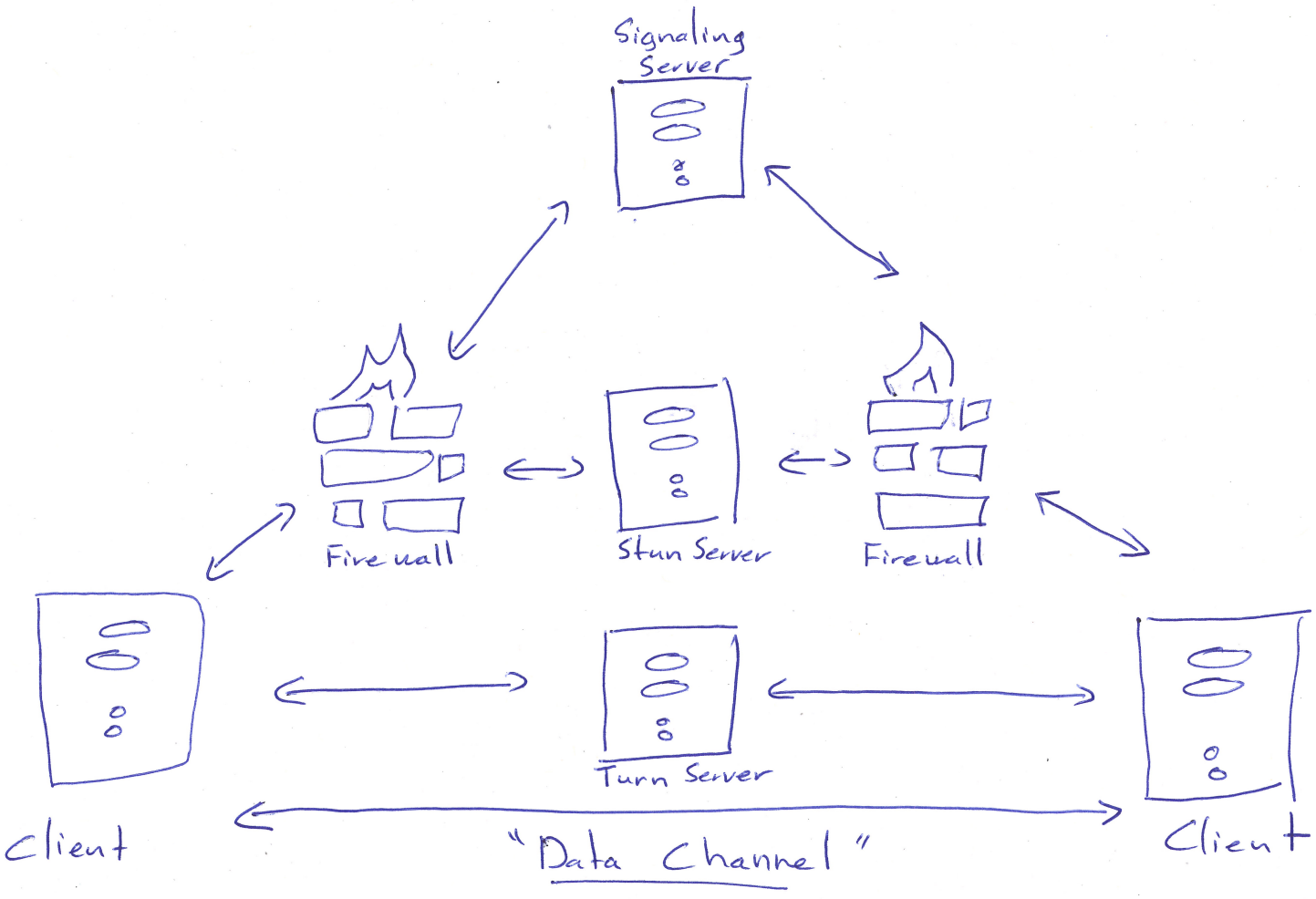
4.1.2 Tracker

In the goal to avoid a centralization of the signaling server, we tried looked into solutions for homemade trackers. The results theoretically solutions were to use GitHub or Twitter has a listing. However, research led us to an already existing similar technology WebTorrent[86]. We tested this solution, the client (browser) is connecting into a known tracker, which could be virtually run by anyone, and the client retrieves the list of connected users ids. It is the same concept as peerJS (managing the peer ids to initialize the connection between peers).

4.1.3 Serverless

The main idea here it to use STUN servers from vast and public companies like Google to open a port that goes through the firewall to talk to the STUN server and opens a bidirectional flux. This flux is then matched with another client who did the same thing and the magic happens. Google's STUN used is `stun.l.google.com:19302`.

Figure 7: *Basic Signaling Scheme*



4.1.4 Live

It's possible to try the PeerJS version from this public URL address `http://rocla.github.io/OverClouds-public` Note that we chose not to release the serverless solution to the public yet.

4.2 Cryptography

Sadly not much has been down for the implementation of this part. We have however made and tested an encryption app in the browser with the **Creating a File Encryption App with JavaScript** tutorial[1] by Martin Angelov.

We, of course, used the CryptoJS library[9], which is part of the tutorial. However, we also tested to replace this it with SJCL[78], it worked, but didn't investigate further for the gain is speed and security. However, if we rely on the Table 1 in the Analysis Chapter on Cryptography, we should be able to see significant differences.

4.3 Results & Technologies Recommendations

The Spring Project is now finished, and lot of information has been gathered. This chapter will target our recommendations based on the results for the following into the Bachelor Project.

4.3.1 Communication

Serverless The analysis concluded that Overclouds can work without relying on centralized servers. We recommend to look at a WebTorrent[86] protocol combination with the serverless protocol already in use. This could fix the question of how users are giving their private address to each other.

4.3.2 Encryption

Blake2 Based on the Table 2 and Figure 5 we recommend the use of Blake2, however it wasn't tested yet with our data encryption. From the tested cryptography libraries, Stanford Javascript Crypto Library is recommended.

5 Conclusion

Overclouds project is fascinating and challenging, there is much material to research and learn. It was said in the specification of this Spring Project that the wheel should not be reinvented, it is true, much work and progress has been made in this field for the past six years. As it is until now, for the communication and cryptography part, it is just a matter of fusing the knowledge to innovate in the field.

Planning Mistakes have been made during the initial planning. The student has bitten off more than he could chew, which led into rescheduling and reducing considerably the research that wanted to be done.

Architecture The architecture evolved three times, and it more than probable that it will evolve again. Currently, it looks more like a mind map, figure 8. However, we believe that the hot spots have been drafted and that Overclouds' big picture is respected.

Communication We proved that communications can be done from the browser with the need of dedicated third party servers. Provided results from the prototype are very promising.

Encryption Javascript is capable of providing client side data encryption required for the transactions and storage, which continues to confirm that a browser-only solution for Overclouds is conceivable.

Consensus This subject has not been treated yet, it is, however, in the Bachelor Project backlog. The general concept starts to get out of the fog. Indeed, looking at the block-chains technology, proof-of-participation, proof-of-activity, the Maidsafe-like concept of small groups of random nodes, and Ethereum fuel concept makes us believe that something could emerge our Consensus concept based on what we consider good ideas.

Bachelor Project Learning from is still the first result at school, for the BP, the planning will carefully reschedule during the first week, then controlled every week. Indeed, the student has a bad habit to be scattered

because everything is interesting, and needs to be done at the moment. Man-hours should be watched as well.

6 Bachelor Project

We recommend taking the first week of the Bachelor Project to take the time to re-plan the project. Indeed, unfortunately the following planning was made in rush, it mainly contains the global structure of the project.

6.1 Specifications

The Bachelor Project aims to continue the Spring Project in a more intensively. The studying of the state of the art, providing incremental solutions and prototypes are still the heart of the project. The succeeding objectives must be fully meet or partially. We keep our previous motto: *do to reinvent the wheel*.

- Study of the state of the art
- Sharing and transfer of data ownership
- Distributed storage
- Encrypted storage
- Pseudo or full anonymity from the ISP
- Trust ranking via the mesh-of-trust
- Notion of aging for trusts mesh-of-trust
- Applying consensus decisions
- Banishment of data and node certificates from the network via a Data Tribunal
- Private keys theft resistance, even if a heartbleed[60] type vulnerability is discovered on the nodes later on[79]

6.2 Planning

The AGILE methodology is still applied, the project is incremental and uses Sprints of a defined man-hours length.

Sprint number: X Sprint Theme: Trust

- Update Architecture
- State of the art
  - EigenTrust `https://en.wikipedia.org/wiki/EigenTrust`
  - TeamSpeak `http://www.teamspeak.com`
- Global Rules/Laws of the network
  - Apply global rules
  - Verify transactions
  - Democratic (community decisions)
  - Data Tribunal
  - Sharing data
- Mesh-of-Trust
  - Certificate on hardware
  - Each user has its own identity keys
  - Signing node+users transactions
  - Levels of trust (Certificates and Identity)
  - Trust giving (Manual or Automatic)
  - Bans
- Node certificates
  - Resetable
- Identities
  - Linked to a "home" node

  - PKI
  - Data owner
  - Public profile
- Deliverable:
  - Mesh-of-Trust
  - Node/Identities
  - Global Rules & Data tribunal

Sprint number: X Sprint Theme: Storage

- Update Architecture
- State of the art
  - Read `https://www.usenix.org/conference/atc14/technical-sessions/presentation/bessani`
- Storing on the network
  - Data
  - Rules
  - Time to live for data
  - Countdown to go publicly
  - Public certificates and identities
  - Flags
  - No data is sent over the network, only the location of the data and the rights
- Deliverable:
  - Demo in the browser of the Storage

7 Bibliography

References

[1] Martin Angelov. Creating a File Encryption App with JavaScript, 2013.

[2] Ayeowch. GLOBAL BITCOIN NODES DISTRIBUTION.

[3] Fabrice Bellard. Javascript PC Emulator, 2015.

[4] Juan Benet. IPFS-Content Addressed, Versioned, P2P File System. *arXiv preprint arXiv:1407.3561*, (Draft 3), 2014.

[5] 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.

[6] Peter Biddle, Paul England, Marcus Peinado, and Bryan Willman. The Darknet and the Future of Content Protection. *Digital Rights Management Technological, Economic, Legal and Political Aspects*, pages 344–365, 2003.

[7] Bitmessage. Bitmessage.

[8] BitTorrent. Project Maelstrom.

[9] Brix. CryptoJS, 2013.

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

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

[12] Cjdns. cjdns.

[13] I Clarke and Et Al. A distributed decentralised information storage and retrieval system. *Undergraduate Thesis*, 1999.

[14] Ian Clarke, Oskar Sandberg, Brandon Wiley, and Tw Hong. Freenet: A distributed anonymous information storage and retrieval system. *Designing Privacy Enhancing ...*, 23:46–66, 2001.

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

[16] Commotion. Community Wireless Networks.

[17] Cuonic. PeerShare.

[18] Laurie Delmer. *L'emergence au sein d'internet de communautes virtuelles et anonymes, Freenet et i2p*. PhD thesis, Universite catholique de Louvain - Departement des sciences politiques et sociales, 2009.

[19] Diaspora. Diaspora*.

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

[21] Roger Dingledine, Nick Mathewson, and Paul Syverson. Tor: The second-generation onion router. *SSYM'04 Proceedings of the 13th conference on USENIX Security Symposium*, 13:21, 2004.

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

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

[24] Snowden Edward. Snowden Digital Surveillance Archive.

[25] Ethereum. Ethereum Homestead Documentation, 2016.

[26] FlowingMail. FlowingMail.

[27] Real Foodists. The Underground Internet. 2003.

[28] Freedom.js. freedom.js, 2012.

[29] Freenet. Freenet project.

[30] 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.

[31] Mark Gabel, Christopher Brigham, Adam Cheyer, and Joshua Levy. Dynamically evolving cognitive architecture system based on third-party developers. *United States Patent Application*, 1(20150100943), 2008.

[32] Open Garden. Open Garden.

[33] Paul Gardner-stephen. The Serval Project : Practical Wireless Ad-Hoc Mobile Telecommunications. (June):1–29, 2011.

[34] Google. Web RTC.

[35] Google. Closure Library, 2015.

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

[37] Hakshop. Rubber Ducky USB.

[38] Matt Hayward. PeerSurf.

[39] Hookflash. Open Peer, 2014.

[40] Hubzilla. Hubzilla.

[41] Hype. Hyperboria Whitepaper.

[42] Hyperboria. Hyperboria.

[43] I2P. The Invisible Internet Project.

[44] IIP. Invisible IRC Project, 2003.

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

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

[47] IPFS. The IPFS Project.

[48] Markus Jakobsson and Ari Juels. Proofs of work and bread pudding protocols (extended abstract). *Secure Information Networks*, pages 258–272, 1999.

[49] Paul Johnston. jsHashes, 2015.

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

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

[52] Werner Koch. Using the GNU Privacy Guard. *Notes*, (March), 2008.

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

[54] Litecoin. Litecoin Wiki, 2011.

[55] MaidSafe. MaidSafe.

[56] MaidSafe. MaidSafe.net announces project SAFE to the community, 2014.

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

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

[59] MediaGoblin. MediaGoblin.

[60] Neel Mehta. The Heartbleed Bug, 2014.

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

[62] Movim. Movim.

[63] Lambert Nick. CONSENSUS WITHOUT A BLOCKCHAIN, 2015.

[64] Nitrokey. Nitokey.

[65] OpenWebRTC. OpenWebRTC.

[66] Peer5. Peer5, 2015.

[67] PGP. PGP ® White Paper PGP ® Global Directory. (August), 2005.

[68] Check Point and Software Technologies. VPN Administration Guide R76. (August), 2014.

[69] Ed. R. Stewart. Stream Control Transmission Protocol. *Network Working Group*, 2007.

[70] Retroshare. Retroshare.

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

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

[73] Serval. Serval Project.

[74] Peer Server. Peer Server, 2013.

[75] Sinch. ORTC VS WEBRTC — WHAT'S THE DIFFERENCE, 2015.

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

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

[78] Emily Stark, Michael Hamburg, and Dan Boneh. Symmetric cryptography in javascript. In *Proceedings - Annual Computer Security Applications Conference, ACSAC*, pages 373–381, 2009.

[79] Nick Sullivan. Staying ahead of OpenSSL vulnerabilities, 2014.

[80] Mr Switch. PeerJS.

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

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

[83] Tent.is. Tent.io, 2013.

[84] TheBitcoinNews. Bitcoin Spam Attacks, 2015.

[85] Tor. Tor.

[86] Web Torrent. Web Torrent, 2015.

[87] Stanford University. Stanford Javascript Crypto Library, 2009.

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

[89] Jonathan Warren. Bitmessage: A Peer-to-Peer Message Authentication and Delivery System. page 5, 2012.

[90] Chris Webber. GNU MediaGoblin Documentation. 2016.

[91] Dai Wei. B-Money, 1998.

[92] YaCy. Web Search by the people, for the people.

[93] ZeroNet. ZeroNet.

[94] Zeronet. ZeroNet, 2016.

8 Annexes

8.1 Overclouds latest architecture

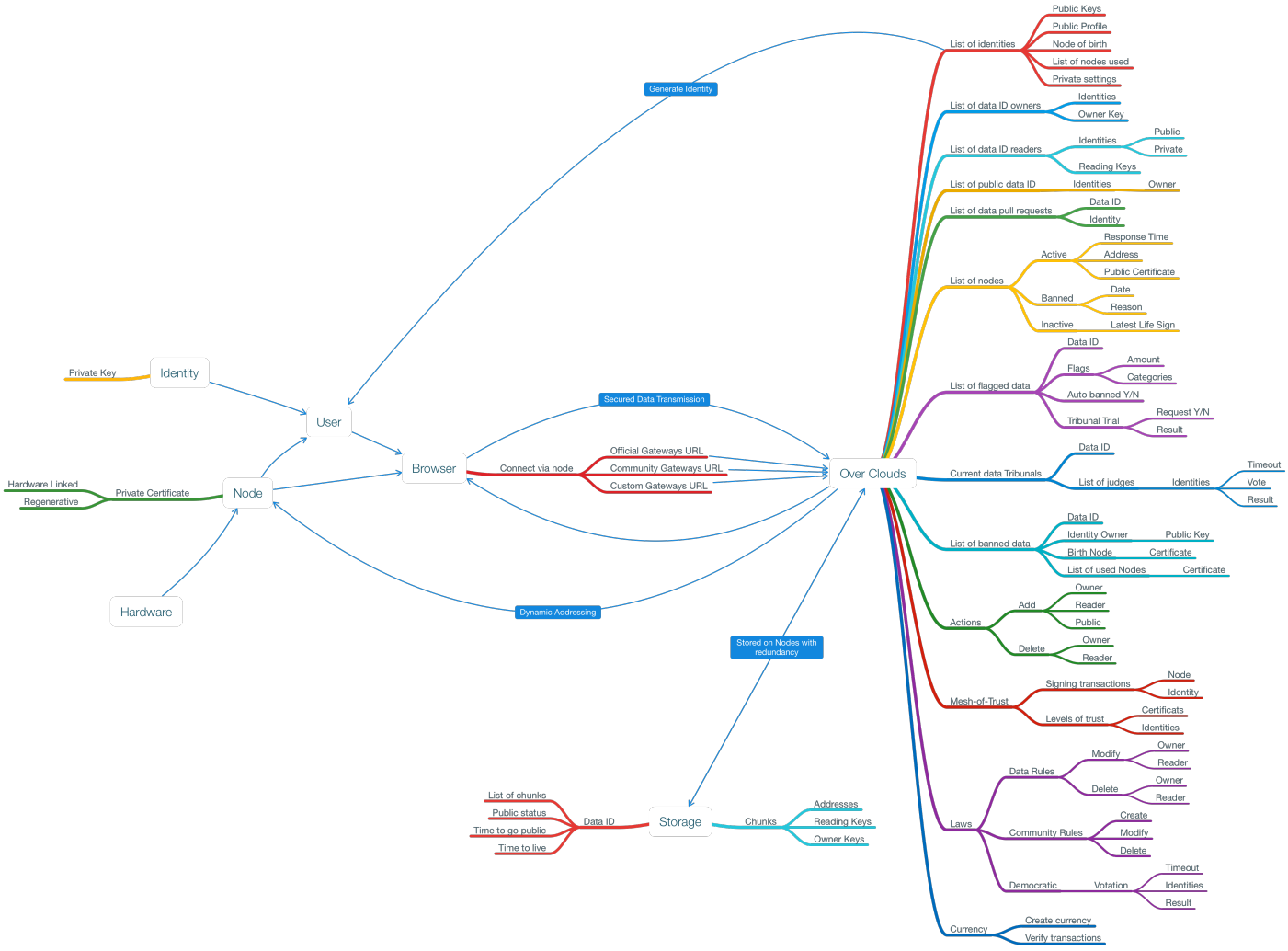
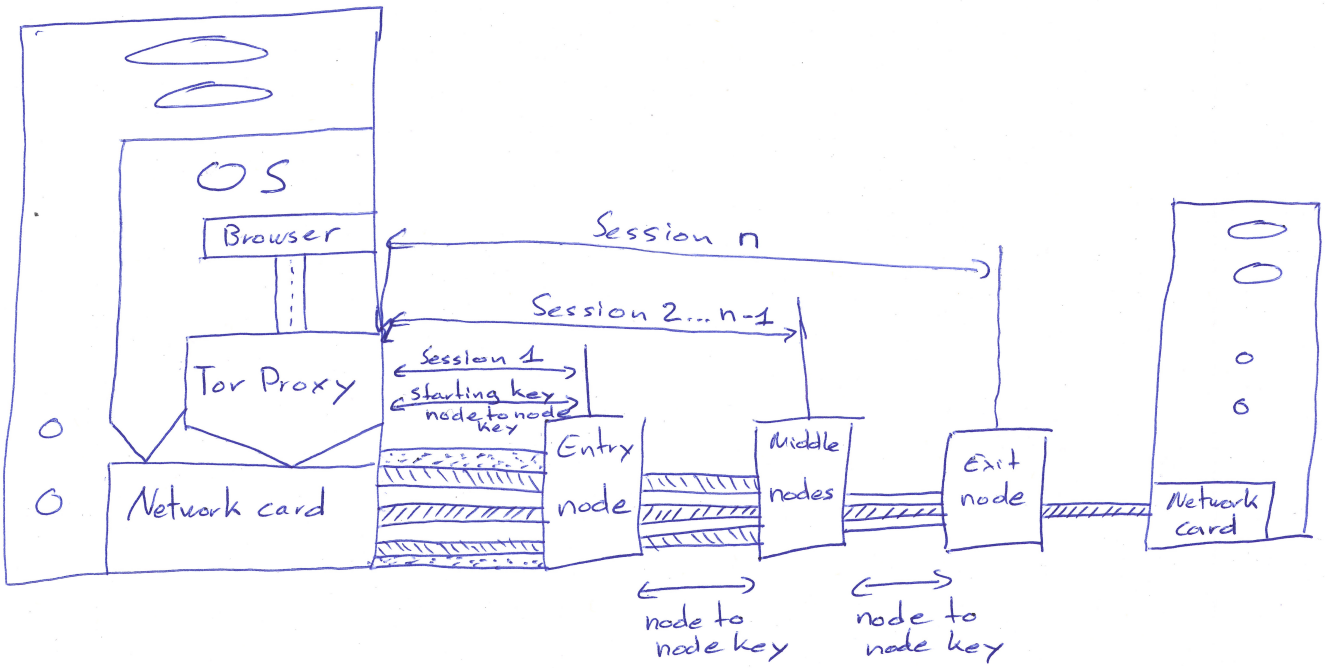


Figure 8: Latest global architecture version

8.2 TOR

Figure 9: *TOR Routage Schematic*



8.3 JS Cryptography Library Graphs

The graphs from the following figures have been made by **Dominic Tarr** [82]

Figure 10: *y-axis shows total time taken, lower is better*

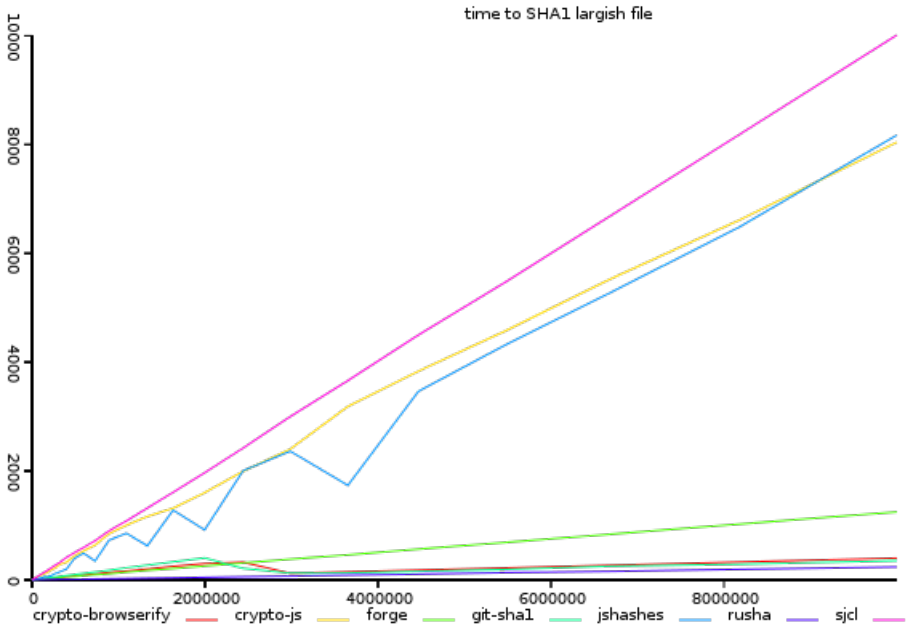


Figure 11: *y-axis shows size/time, higher is better*

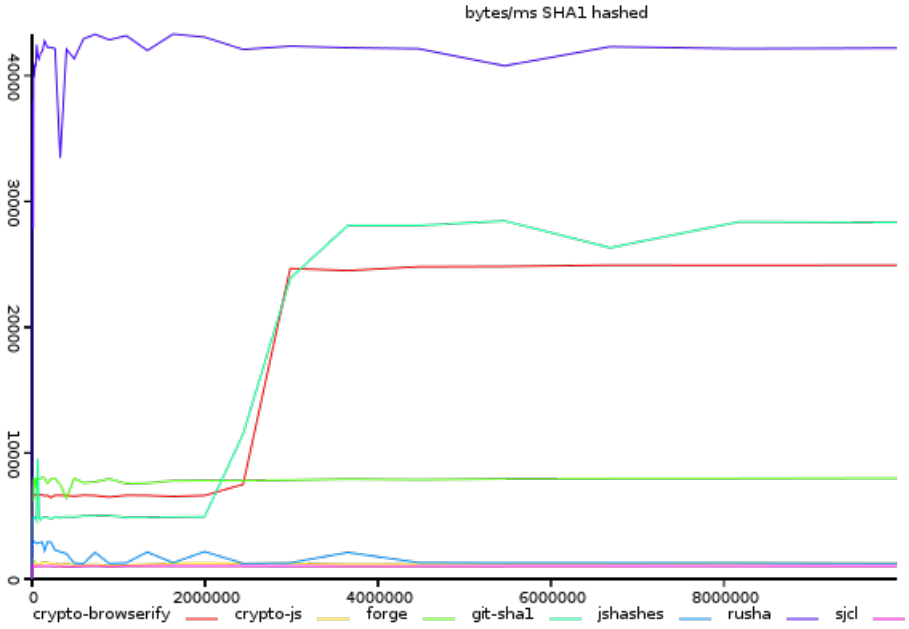


Figure 12: *y-axis shows total time taken, lower is better*

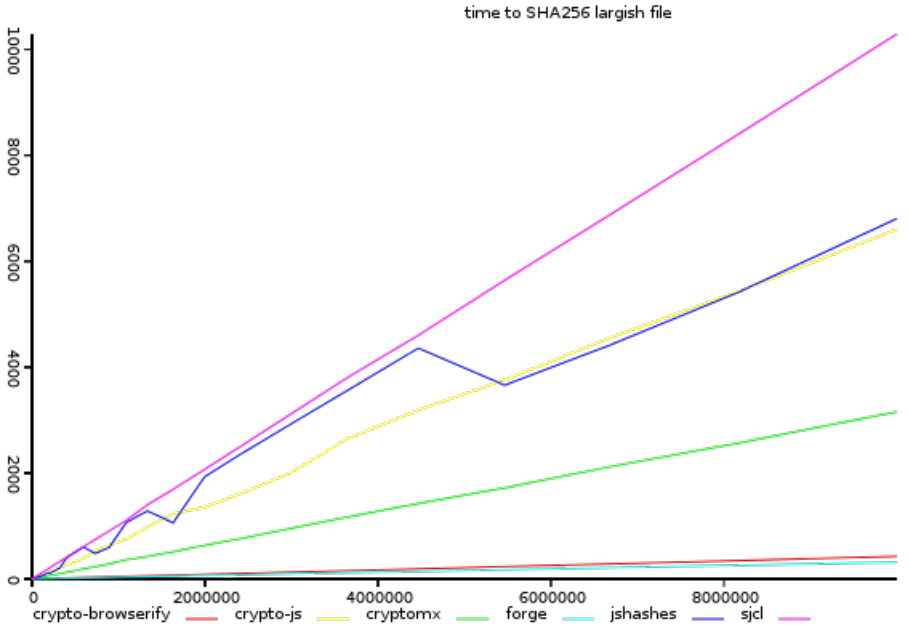


Figure 13: *y-axis shows size/time, higher is better*

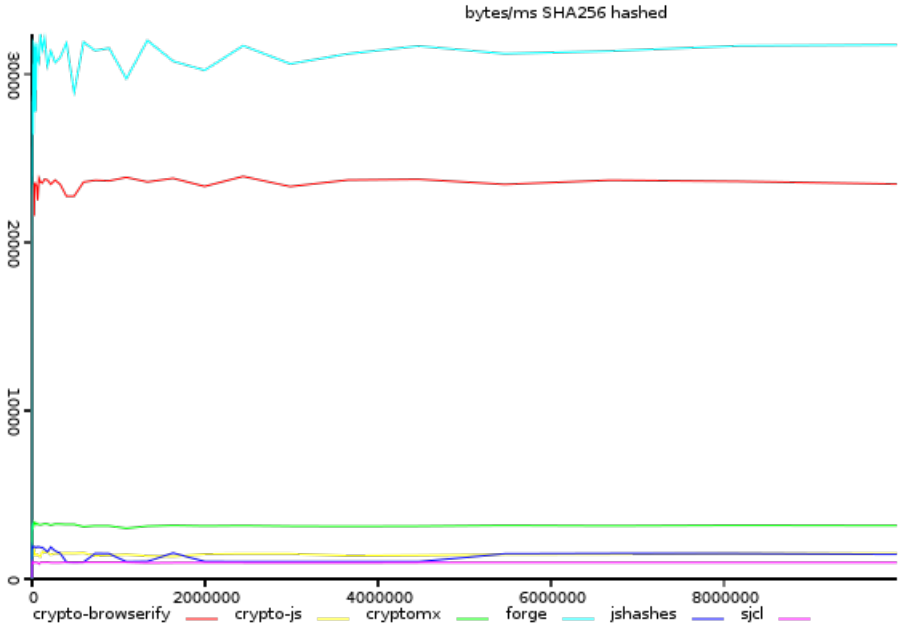


Figure 14: *y-axis shows total time taken, lower is better*

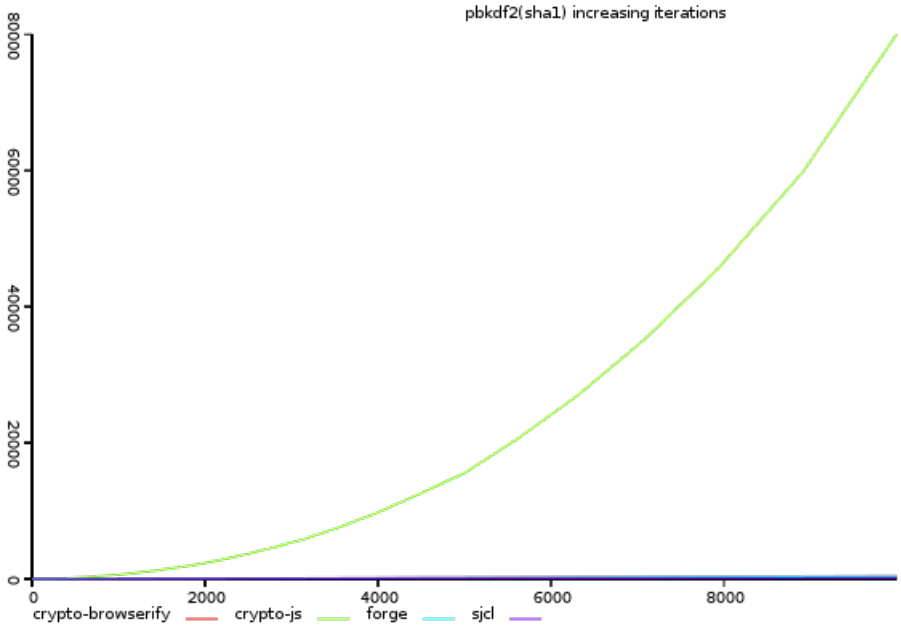


Figure 15: *y-axis shows size/time, higher is better*

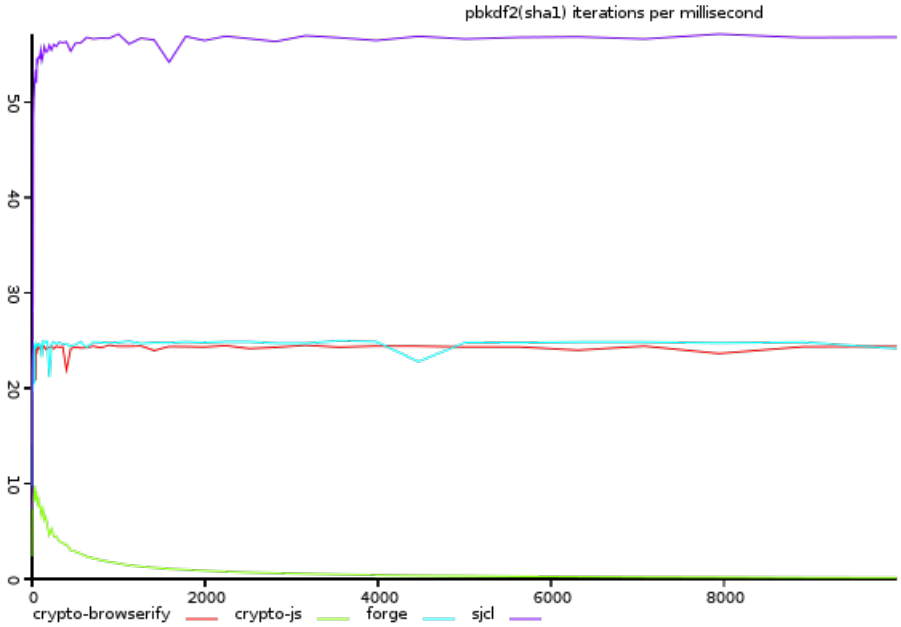


Figure 16: *y-axis shows total time taken, lower is better*

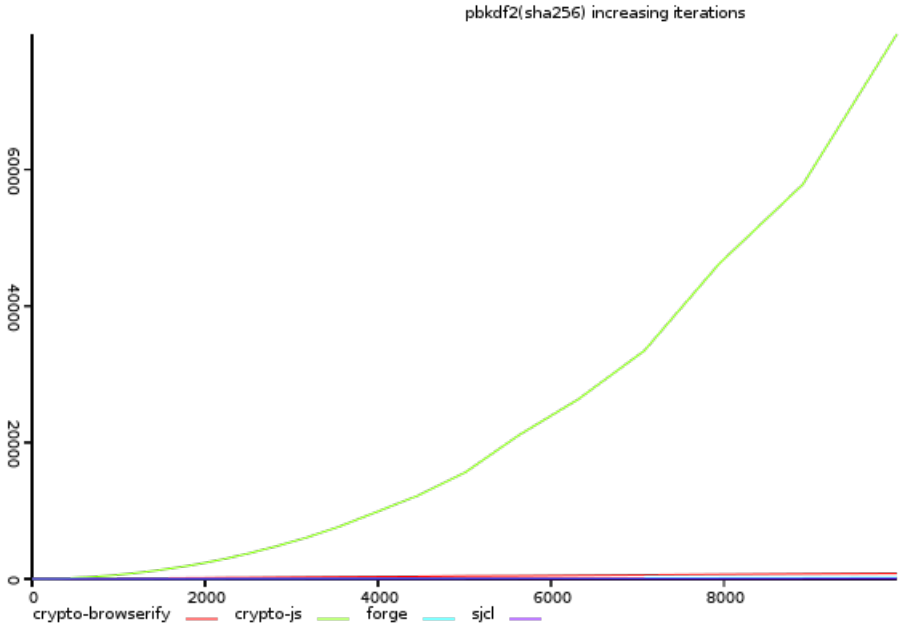
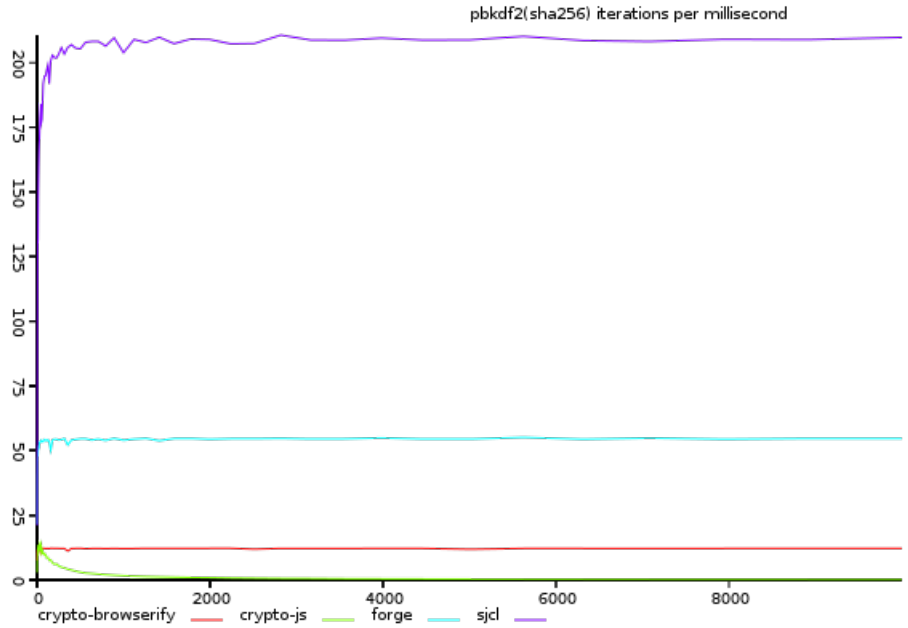


Figure 17: *y-axis shows size/time, higher is better*



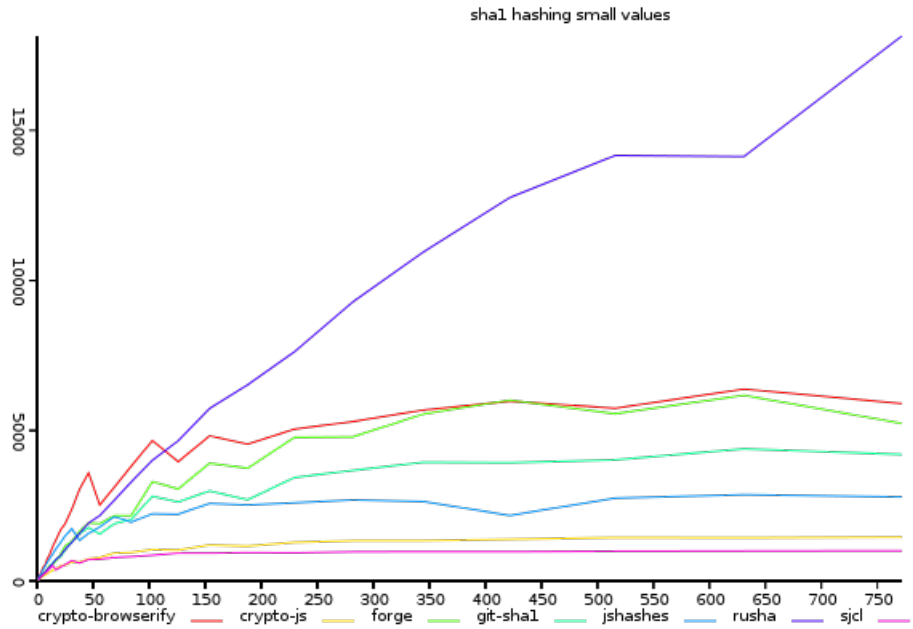
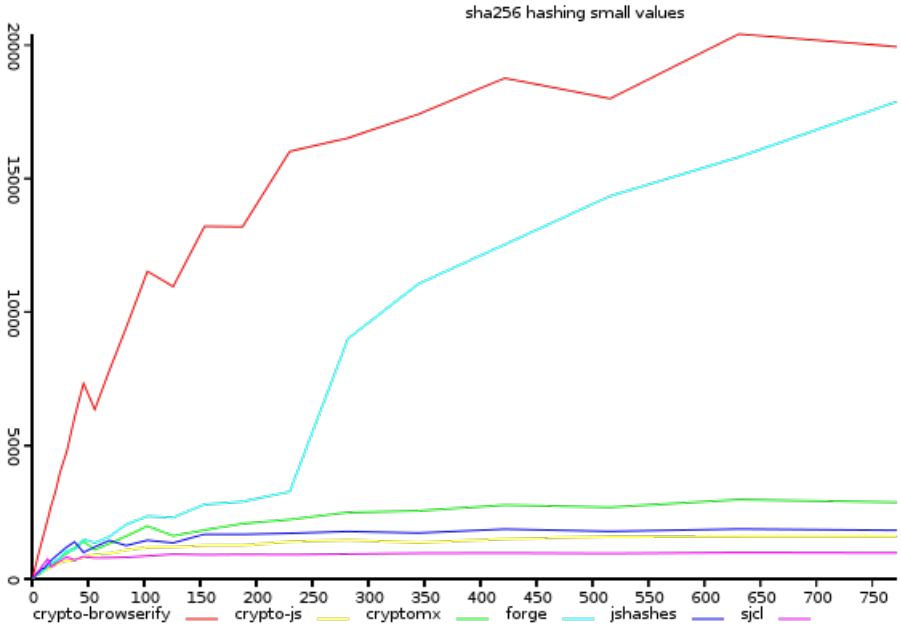


Figure 18: *y-axis shows size/time, higher is better*

Figure 19: *y-axis shows size/time, higher is better*



8.4 JS Cryptography Library Tables

The following tables have been made based on **Dominic Tarr**'s[82] benchmarks.

Table outcome must be interpreted in the following order.

- Awful
- Bad
- Ugly
- Meh
- Neutral
- Okay
- Nice
- Good
- Amazing

Table 3: Hashing 0-10MB Files /milliseconds based on figures 10, 11, 12, 13

Libraries	Sha1 (size)	Sha1 (hash)	Sha256 (size)	Sha256 (hash)
sjcl	Awful	Awful	Awful	Awful
crypto-js	Bad	Ugly	Meh	Bad
forge	Okay	Okay	**Amazing**	**Amazing**
crypto-browserify	Nice	Nice	Good	Good
crypto-mx	Neutral	Neutral	Okay	Ugly
git-sha1	Good	Good	Neutral	Neutral
jshashes	Meh	Meh	Ugly	Bad
rusha	**Amazing**	**Amazing**	Neutral	Neutral

Table 4: Hashing Small Files /milliseconds based on figures 18, 19

Libraries	Sha1 (size)	Sha256 (size)
sjcl	Bad	Bad
crypto-js	Ugly	Ugly
forge	Nice	Good
crypto-browserify	Good	**Amazing**
crypto-mx	Neutral	Okay
git-sha1	Okay	Neutral
jshashes	Meh	Meh
rusha	**Amazing**	null