A decentralized and secure network inside browsers allowing any application to work on top of it protecting the privacy and anonymity of the users, allowing new disruptive services with browsers acting as autonomous nodes replacing servers

CONVERGENCE

1. Excellence

1.1 Sound concepts and objectives

1.1.1 Background and rationale

The problem today is that **we must invent one network per need** if we want to evade big data centralization and protect privacy/anonymity: to browse, to chat, to email, to exchange files, to do social networking or cooperative work, to do crypto currency, to protect the users from their connected objects, to handle peer identities.

Different systems are trying to mix those different networks or propose their own solution, whether centralized or not, or a mix of both, whether open source or not, without being fully convincing so far, and each solution needs a specific installation/software on a specific platform/device with the associated risks (like software loading and integrity) and complication for the users, in addition some solutions are completely unusable by normal people.

We propose to define a **decentralized and secure architecture including a digital identity management system using the most widely spread unified/open/standard system (not platform/device dependent): browsers, that would allow any solution to work on top of it, without requiring specific installation or skills to use it**.

The study will define also how those applications, namely chat, email, file sharing, social/cooperative networking and crypto currency, can work on top of the proposed architecture as well as studying how it can protect the privacy/anonymity of the users of connected objects.

Browsers can run javascript applications as standalone applications and act as autonomous nodes to relay data but have still some paradoxal limitations.

<patent>

Then we will show how additional applications can be run inside browsers with these new features: anonymous browsing, uProxy [1] and Tor nodes [2] inside browsers, Figures 1 and 2 below do summarize the concepts.

The installation process of Convergence and related applications will be as simple as browsing a site that will deliver the code without requiring any effort from the users, the applications will then execute inside browsers with the possibility to continue to run in background when they are closed.

However, we will study an alternative where the application can deploy inside the network without the use of web sites and since P2P networks need a sufficient number of users to work we will study how to motivate the peers to participate and sustain the network.

Those new principles raise the question of an alternative model for the web applications that cannot necessarily be tied to a domain with the introduction of an entityID instead, as well as a network and public peerID management system.

1.1.2 Key technological aspects

The anonymizer networks are thousands of nodes, the bitcoin network is millions of peers, the bittorrent network is hundreds of millions of peers, **browsers are billions**.

So browsers are a good candidate to build a huge decentralized network, they allow to easily use applications without any installation on any device.

In addition browsers have all required features: **browsers can discuss between each other** (WebRTC [3]), **browsers can discuss with networks** (WebSockets [4], XHR [5]), **browsers can store data and manipulate files** (indexedDB [6], File API [7]), **browsers can stream** (built in solution and hopefully future Streams API [8]), **browsers can handle work in background** (Workers [9]), **browsers can handle crypto** (WebCrypto [10]), **browsers can proxy** (SOCKS [11]), **browsers can handle modules as independent applications** (Web Components [43])

One past drawback of using a browser for applications was that it was required to let it open to run them. This is now solved with the Service Workers [12] that allow **to run a browser instance and associated applications in background**.

<patent>

Figure 1

Regarding the network architecture, it will be inspired from the **Peer Shared Mesh technology** (Peersm) [14] (which is already inspired from other projects as stated in next section) **where peers (browsers) are connected to each other via the Tor protocol using WebRTC** as shown on figure 2:

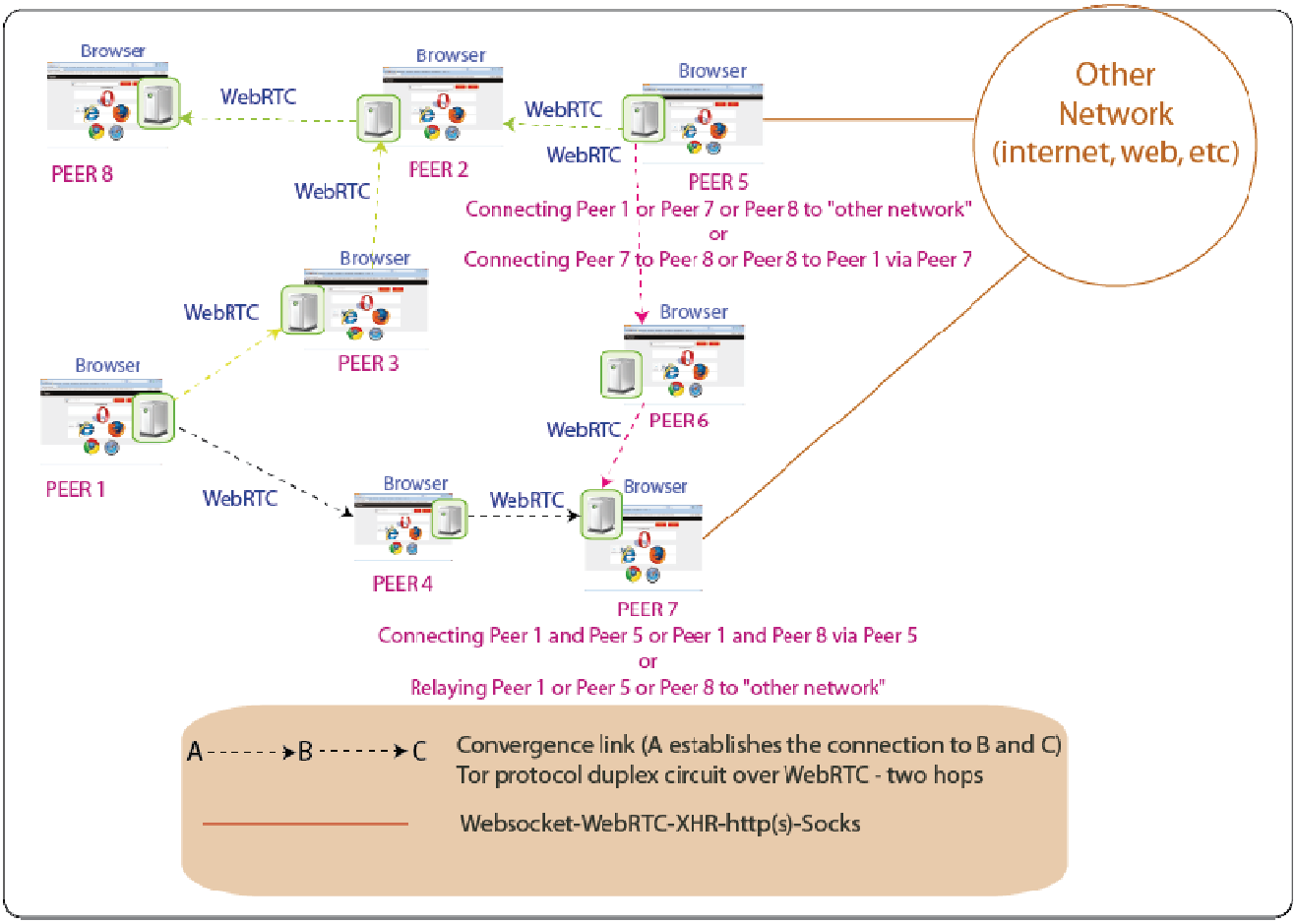

Figure 2

Peer Shared Mesh (Peersm) is a technology from NAIS applied to anonymous P2P file sharing, while we don't intend to clone the project we will **reuse the Peer Shared Mesh concepts including the peer discovery/introduction system (WebRTC Distributed Hash Table [15] - DHT), extend it adding a long term peerID management system and the specific concepts of peers connected to each other via two hops implementing the Tor protocol, and the specific principles of a first direct level for peers discovery/registration before invoking the DHT**.

As shown on Figure 2 each peer is connected to several other peers via two hops who are acting as Rendez-vous points to relay the messages between the different peers, similar somewhere to the Tor hidden services mechanisms (who sometimes are used to connect peers which is not what they have been designed for and cannot be efficient with the Tor network, because far too small).

We will define the concepts of the WebRTC DHT which to summarize allow the peers to discover and introduce each other without the help of any server.

We will study the use of a decentralized system for secure peerID, entityID (and applications code) securization/management and related data storage controlabe by the users like combined use of blockchains and DHT.

The whole system will constitute **an encrypted serverless anonymous P2P network making extremely difficult to trace/identify the users and what they are doing/exchanging**.

We will define the protocol messages to be used by the applications to work on top of this architecture and address all the security/privacy aspects.

1.1.3 State of the art, proof of concept and innovation

The architecture definition **leverages the state of the art regarding different existing architectures** (mainly The Tor network [2], I2P [16], freenet [17], bittorrent [18] and cjdns [19]) and the below mentioned projects for the application part, **defines new innovative concepts and adapt it to browsers**.

The Tor network has been designed for browsing only, is centralized and far too small to envision any P2P application on top of it, I2P and freenet have been designed to share content (although they now propose new services but not convincingly), with a level of complexity not accessible to normal users, the bittorrent network is used to share content only too, is quite efficient but not designed at all to protect privacy (although some means exist to protect as shown in our study [20] but still anonymity cannot be insured), cjdns proposes the same type of architecture than Convergence but is more designed to prevent DDOS attacks and does not protect privacy/anonymity.

Some combines the different networks mentioned above to propose different applications they have not been designed for to support, or to attempt to protect, leading always to the contrary of the intent, ie security and privacy leaks (like for example running bittorrent over Tor or I2P).

The proof of concept with browsers has been demonstrated by the Peersm project, which implements all the crypto of the Tor protocol inside browsers (node-Tor [21]) and other projects like WebTorrent [22], which is a bittorrent network inside browsers using WebRTC (and a WebRTC DHT not fully specified) or Flashproxy [23] using browsers to relay the Tor traffic between Tor nodes and users via WebSockets (now replaced by WebRTC SnowFlake [24] to avoid the NAT traversal issue of Flashproxy).

Some other projects like Peer5 [25] or Maelstrom [26] are implementing decentralized web connections, like P2P XHR for Peer5 retrieving the data from different peers and P2P browsing from Maelstrom retrieving web sites data from peers.

The Tribler project [27] proposes a bittorrent network using the Tor protocol, it can only be used for file sharing and put at risk the peers that are exiting in the clear toward the bittorrent network, from our standpoint it follows too much the Tor network rules for things that do not necessarily apply to a P2P network (Guards concept, 3 hops, etc), at contrario Convergence is using the Tor protocol adapted to a P2P environment optimized for browsers with a new protocol (ie not the bittorrent protocol)

The FreeDOM project [28] is an example of distributed content inside browsers (distributed wiki) but for general content only, not related for example to peers' personal data, social networking and cooperative work.

Diaspora project [29] tried to manage private data, but in a centralized way that could not be sustainable.

So none of those projects envision to implement the level of decentralization and protection that offers Convergence, the possibility to aggregate different applications, the specific peers connection mechanisms, the specific two levels peer introduction/discovery system and a WebRTC DHT clearly specified.

<patent>

WebRTC inherently induces many security and privacy issues like peer introduction securization [24], STUN servers and possible address IP leak [25], NAT traversal, that we will address.

WebRTC and WebSockets use a specific protocol so might be blockable, we will study too the obfuscation means to prevent this, this is briefly addressed by the uProxy project [1] but so far the uProxy team did not deem necessary to write specifications, protocol fingerprinting is addressed by other projects like Snowflake ([24] and [51]). Tor pluggable transports [52] for obfuscation will be analyzed too.

The ensemble is complicate and challenging to put together, all security/privacy issues will be carefully studied and measured, using the known vectors of attacks for the Tor protocol/network and P2P networks.

In addition a special attention will be paid to recent attacks like the logjam attack ([30]) and suspicion of deanonymization ([31])

We will not list here all the chat, messaging applications but we will evaluate the security of the applications over the Convergence platform using the indicators described in [32] and [33] as shown in figure 3.

Figure 3

For the browsing application we will compare with the Tor Browser features, it is likely that allowing anonymous browsing directly from any browser (ie without using a localhost server like the Tor Browser) will not eliminate the need for the specific features developed inside the Tor Browser.

For the file sharing application we will reuse the concepts of Peersm project and clarify some TODOs (mainly streaming and data validation)

Social networking will analyze how Convergence can handle private data based on the peerID management system and how to share it, as well as how to handle and share cooperative work. The data can be hosted by one peer or be decentralized among several ones.

Regarding crypto currency we will analyze the benefit to run a crypto currency network over Convergence knowing that even without including privacy features the bitcoin network is already difficult to trace.

Specifics of browsers and javascript (especially the code loading) will be studied, of course this has nothing to do with the sometimes wrong common belief that browsers and javascript are insecure.

We will show how to implement uProxy and Tor nodes inside browsers with Convergence and how Convergence or the underlying technology can protect the privacy and anonymity of the users of connected objects.

Finally we will study how to deploy automatically applications inside the Convergence network and how crypto currency micro payments to the peers can boost the system.

So to summarize we do not intend to reinvent the wheel but will reuse the best of what exists, define and introduce new concepts for which the state of the art does not exist and innovate going beyond the state of the art for what exists.

1.1.4 Objectives

- **Objective 1: design the Convergence architecture, peer introduction and discovery system (including WebRTC DHT), peerID management system**

  Means to achievement: redefine the Peersm mechanisms and complete them, define a completely new protocol that can be used by the Convergence applications inside browsers

  Measurable outcome: challenge the architecture resistance to known attacks (network and browsers as defined above) and determine its usuability by the Convergence applications (including <patent>) and users (NAT traversal for example). The architecture will necessarily involve the use of self-signed certificates which are problematic inside browsers, or by design the use of a secure Convergence protocol over a non secure one, similar to the http/https mixed content issues inside a web page, which is problematic too. The proposed architecture will explain how to solve this.

- **Objective 2: <patent>**

- **Objective 3: define how the chat and messaging applications can work**

  Means to achievement: define the chat and messaging application using the Convergence protocol and ID management system, as well as additional security mechanisms required.

  Measurable outcome: assess the usability and test the resistance to known attacks with the methodology explained in the previous paragraph (EFF and secushare method, as shown in Figure 3).

- **Objective 4: define how the file sharing application can work**

  Means to achievement: complete the Peersm specifications, as an anonymous bittorrent like network, especially for live streaming and specific data/pieces validation.

  Measurable outcome: evaluate the privacy/anonymity features, usability and evaluate the multi-peer download performances/rates inside browsers

- **Objective 5: define how the social networking application can work**

  Means to achievement: define the social networking application bases using the Convergence protocol and ID management system, define the private and public data management, protection, portability..

  Measurable outcome: assess the usability, sharing of public data, protection of private data and control for the users over their data and their integrity.

- **Objective 6: define how the cooperative application can work**

  Means to achievement: define an application allowing to share a document modified within a group and distributed among different peers, using the Convergence protocol and ID management system. Probably some concepts of Objective 5 will be reused.

  Measurable outcome: assess the usability and the performance of secure document retrieval, update and distributed storage

- **Objective 7: define how the crypto currency application can work**

  Means to achievement: define how to anonymize a bitcoin like network using Convergence and how to link payments to the peerIDs.

  Measurable outcome: assess the usability and security/traçability compared to the bitcoin network alone

- **Objective 8: define how the browsing application can work**

  Means to achievement: define using <patent> how anonymous browsing can be achieved directly inside browsers, study if direct exit (toward web sites) can be performed by the browsers relaying the traffic or if the browsers must exit through the Tor network.

  Measurable outcome: assess the usability and security/fingerprinting, compare with the Tor Browser features, determine the most secure solution for users and browsers relaying the traffic.

- **Objective 9: define how the uProxy application can work**

  Means to achievement: define a uProxy application directly inside browsers using <patent>

  Measurable outcome: assess the usability and security aspects, especially study the authorization procedure for the peer that is exiting the data and/or if the traffic can exit directly or must be relayed through the Tor network, study the impact of the same origin policy.

- **Objective 10: define how a Tor node can work inside browser**

  Means to achievement: define how a Tor relay can use a browser to relay the data inside the Convergence network and/or exit to the Tor network back or directly to a web site.

Measurable outcome: study if the traffic can exit directly or must be relayed through the Tor network, study the impact of the same origin policy. A limiting factor inside browsers might be the upload bandwidth of the peers, which can really reduce the performances of a usual Tor circuit, study if the Convergence architecture can solve this issue involving different peers to relay the data received from a Tor relay and exit it with the equivalent bandwidth and define the related modifications that must be performed at the Tor protocol and routing levels.

- **Objective 11: Internet of Things, Smart Cities, show how the Convergence concepts can protect the connected objects**

  Means to achievement: some connected objects are using browsers interfaces and can therefore join the Convergence network to insure privacy and anonymity for the information they send about their users, some others can aggregate their data using for example the FIWARE platform, which supports WebRTC and could be used to anonymously communicate between the users and their connected objects.

  Measurable outcome: assess how connected objects can become peers inside the Convergence network or can become "aggregated peers" and/or could reuse the underlying technology to insure privacy and anonymity

- **Objective 12: Applications deployment**

  Means to achievement: secure the standard code loading from websites and define a way so the applications can deploy directly inside the Convergence network without the need of web sites, using the peers with an opt-in mechanism.

  Measurable outcome: assess the security of the mechanisms and required changes inside browsers.

- **Objective 13: Crypto currency micro payment**

  Means to achievement: define a way to sustain/scale the network so the peers get in return micro payments for their participation.

  Measurable outcome: assess the effectiveness of the system.

1.2 Relation to the work programme .

- *Today the net is a place unlocking rapid innovation.*

Browsers are evolving quickly on all devices (especially smartphones) with the raise of html5 standards which brings new features allowing the browsers to communicate between each other and with network entities, using WebSockets, WebRTC and XHR, which the Convergence project is using **allowing to aggregate applications on the same secure network available from browsers without requiring any installation from any device (including smartphones) and working as native applications**. In addition the Convergence project is going beyond html5 (who somewhere starts showing its age) proposing new concepts inside browsers like <patent> and Web Components [43].

- *This potential is too often left un-exploited, i.e. Europe does not do enough to turn RTD & I outcomes into business success.*

  RTD&I projects **must often create one solution per need**, especially to protect privacy and anonymity, knowing that the said solution must be declined and maintained for all different devices and platforms it operates on, which induces a lot of development work not always sustainable by the entities that perform it.
  In addition it's difficult for European entities to distribute widely a project in an internet dominated by major US companies (like Google, Microsoft, Mozilla and Apple).
  The Convergence network intends to solve this problem with an open secure architecture potentially composed of billions of peers on a common platform (browsers) making it easy to create and use applications on top of it.
  It's difficult for European entities to weigh in standards, like the W3C or WHATWG, since they are dominated again by the major US companies, who specify the Web according to their needs, leading to important design mistakes (see [35]) since they don't anticipate some uses like those proposed by the Convergence project which will open browsers to multiple possibilities, the Convergence project will propose some modifications to the specifications and new ones for adoption by the standards body.

- *In particular open platforms offer opportunities for the development of new services and applications. FIWARE, for example, is an open platform that demonstrates the capacity to become a preferred service platform, but its potential is currently underused.*

  The Convergence network proposes a distributed alternative to centralized platforms, like FIWARE, to build secure applications protecting privacy and anonymity.
  But a distributed network/application still might require cloud services depending on its goals, like Bridges for the WebRTC DHT, anonymization of aggregated connected objects information, duplication/storage of data or simply hosting the application that will work on Convergence (ie a host to retrieve the code), which can be performed using the FIWARE platform, as well as using FIWARE as an aggregator in order to bridge communications with the Convergence network (to protect IOT for example as we have seen before).

- *Current centralised platforms for big and social data management consolidate the dominance of existing incumbent actors, stifling innovation and allowing less and less control over the data by citizens. Distributed architectures and decentralised platforms have a huge potential to enable the creation of viable alternatives to current dominant models.*

  The hegemony of major internet companies for data management and storage (personal or not) and their financial/technical capacity to manage huge platform and big data **makes it difficult for a newcomer to compete with and for citizens to have control over their data**.
  The situation is so critical that citizens have started to give up with their privacy and anonymity, worse, since the trend appear not likely to revert, especially with the Internet of Things, some started to be interested by projects offering to consciously monetize part of their private data [36].
  The bittorrent network has shown the strength and capacities of a decentralized network, but without any privacy/anonymity related concerns and limited to file sharing, **the Convergence platform proposes to solve this problem using the huge potential of billions of peers to securely exchange, store, manage, relay, control data via a complete open decentralized network whose size and capacities can easily compete with major structures and that can support any type of application**.

Some platforms like Ethereum [64] proposes decentralized architecture and control over the data stored, **per analogy Convergence is for applications what Ethereum is for smart contracts.**

- *More generally, key players and ecosystems, startups and SMEs often do not have sufficiently innovative technology in their hands to innovate on the net. Outcomes of Future Internet RTD & I need to be transferred faster into real life.*

  For the reasons stated previously **the major internet companies do not make it easy for smaller entities to innovate**, since they dominate the market, the technical aspects and the standards. Smaller entities can develop innovative projects and concepts that are difficult to distribute, can be obsolete when they ship due to the time it took to develop them or due to a major internet company effort to suddenly develop it faster.
  Browsers are probably the best platform to innovate quickly and ship innovative applications, since they are widely deployed, the applications are easy to install on any device just browsing a web site and can work standalone as native applications, **Convergence uses this potential open to everybody to deploy quickly innovative applications with a focus on privacy/anonymity and control over the data**.

1.3 Concept and methodology, quality of the coordination and support measures

We will describe in this section the global concepts of the Convergence architecture (which are subject to evolve during the study) and main technical aspects to study and solve, as well as the criteria to measure the outcome.

1.3.1 Global concepts

The General architecture is summarized in Figure 2.

Each peer generates a public/private key and a corresponding self-signed certificate (nodeID certificate), its fingerprint (or nodeID) is the fingerprint of its public key which is its onion key too (ie the one used for the Tor protocol between peers), those keys are generated at each connexion and therefore change.

Each peer generates also long term key using the same principles, except that those keys don't change in time and the fingerprint of the public key might represent the peerID, ie the long term ID of the peer, "might" because in case of compromised keys the peerID will have to change.

The peers are connected to each other via two hops using the Tor protocol (see Figure 2), so via encrypted links, the last one in the path can't know who is the first one and the first one can't know it is the first one. The last one is performing the relay function which consists in relaying the data from one connection to another, it can't know who originated the data, what it is and where it is going.

In addition the "relayer" can't know if he is serving data directly or via another relayer, in other words it can't know how many hops are separating him from the requester (see Figure 2 again and data exchange between Peer 1 and Peer 8 for example).

Per design the" relayer" sees the data decrypted and re-encrypt it to pass it forward, depending on its objectives, the Convergence application will add mechanisms such as key negociation between the requester and the recipient to encrypt data so the relayers don't see them in the clear.

The peers are communicating their peerID and what peerID they know to the “relayers” they are connected to, and the “relayers” (as peers) do the same. They send the [peerID, public Key] information signed by the private key to make sure that the advertiser really owns this peerID and then store the peerID information associated with the nodeID to reach this peerID.

In addition the peers are implementing a Kademlia Distributed Hash Table (DHT) using their nodeIDs.

So, each peer implements a routing table containing the nodeIDs of known peers and the peerIDs they know which per analogy with the Bittorrent DHT represents the infohash.

Each time a “relayer” has a new peerID it registers itself in the DHT, so the data stored in the DHT do not relate to the peers that have the corresponding peerID but to those that know other peers that know how to contact this peerID.

When receiving a request the “relayer” connects to the requested peer if it knows it, if not it sends the request to another connection/peer it knows has this peerID information (without knowing who it is), if the peer does not have any longer the peerID information it can reject the request or send it to another connection/peer that might have it, on rejection the “relayer” can send the request to another connection/peer, when a given request has been forwarded too many time it is discarded and the originator restarts the request with different “relayers”. If the “relayer” do not know anything about the requested peerID, they consult the DHT and extend the path to the relevant “relayer”.

For example on Figure 2 Peer 1 knows that Peer 7 knows how to contact Peer 8 via Peer 5, Peer 1 will send a request to Peer 7 who will send it to Peer 5 who will send it to Peer 8, at the end Peer 1 and Peer 8 will be connected through Peer 7 and Peer 5.

So the peer discovery system has two layers and the content discovery system will work the same way.

A difficulty of WebRTC is that a peer cannot connect directly to another one, for security reasons, so to connect two nodeIDs a third party is needed to perform the peer introduction (basically for the requested peer to send back to the requesting peer an authorization to connect to him on [this IP] and [this port number]).

Figure 4 describes the standard connection process between two peers for WebRTC.

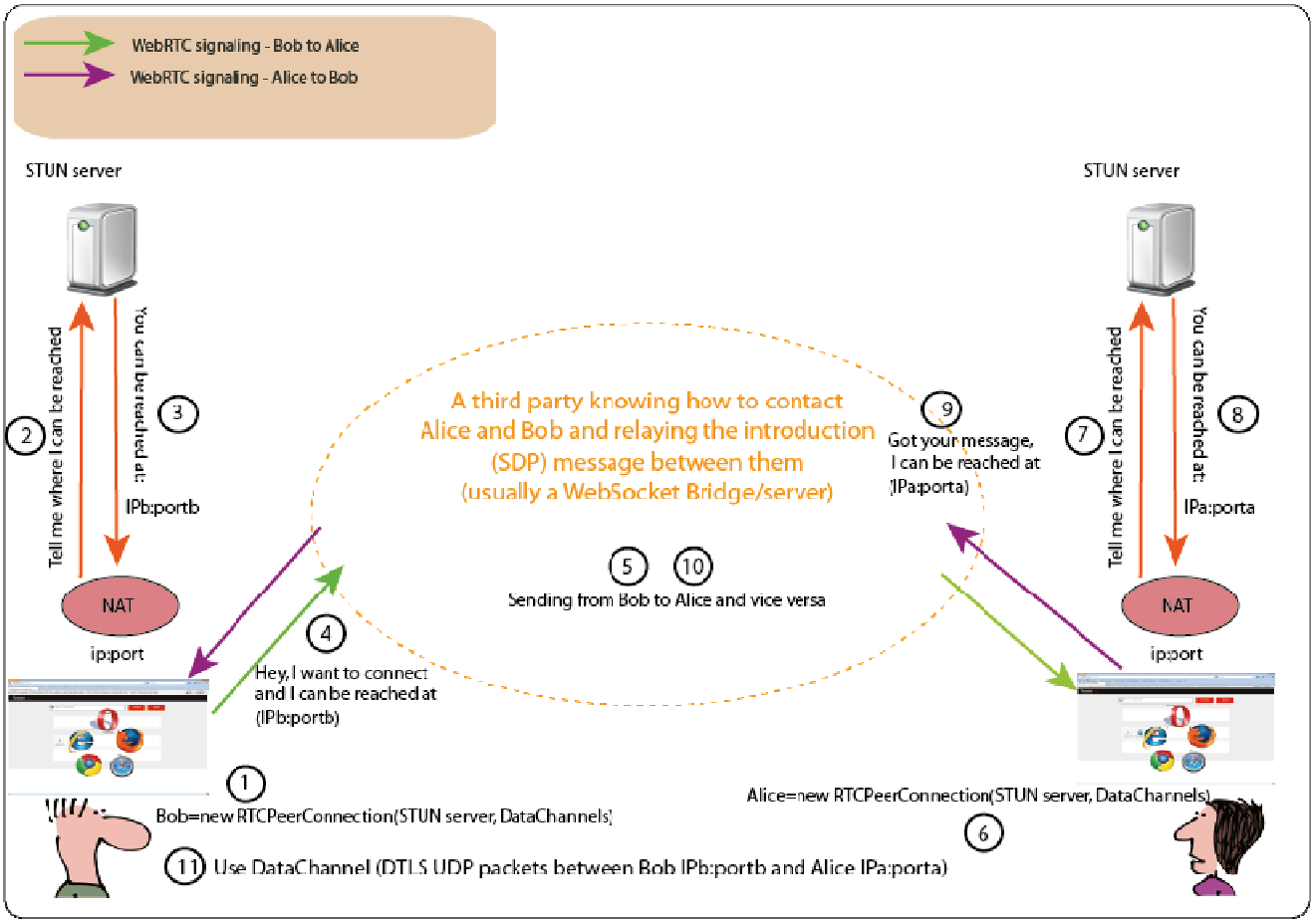

Figure 4 – Peer introduction, standard case

So, to the above mechanism we must add the WebRTC DHT concepts as shown on Figure 4 boostraped by WebSocket Bridges (the study will address in more details some concerns like bridges blocking, number of bridges, bridges DHT and probability of peers connected to them) where basically the final intent is to allow peers to introduce themselves without using bridges (ie A is connected to B, B is connected to C, B performs the introduction for A to connect to C):

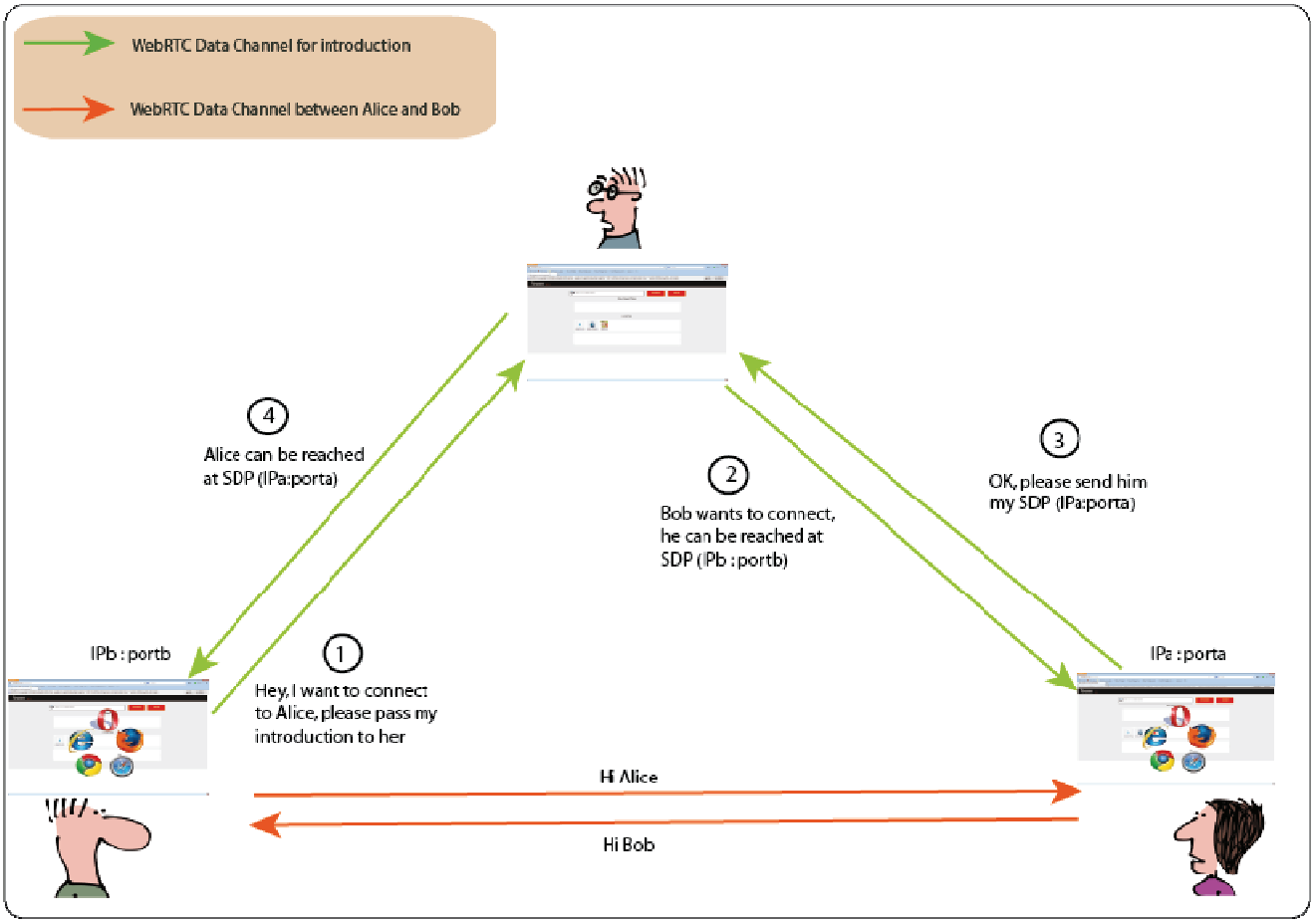

Figure 5 – WebRTC Peer introduction

A connects to a bridge, the bridge registers A in the bridges DHT.

A sends to the bridge it is connected to a FIND_NODE request [IDA,modulusA], the bridge registers A, queries the other bridges to find the closest node to A and replies with a FOUND_NODE request [IDB,modulusB,IP:port] of several peers close to A where [IP:port] corresponds to the bridge that can perform the peers introduction, if missing the queried bridge can perform it

A can send different requests to differents bridges to discover some peers.

A connects to B using the related bridge for peers introduction (INTRODUCE [IDB,SDPA offer], INTRODUCED [IDA,SDPB answer])

A sends a FIND_NODE to B, B replies with the closest nodes to A connected to it [IDN,modulusN,IP:port] where IP:port is the bridge where N is connected to in B routing tables.

If B is connected to nobody (rare and unlikely case since all the nodes are connected to others), B does the same than A to discover the peers and connect to them, then passes the result to A.

A create circuits (Tor protocol) with the nodes connected to B that will act as the “relays”.

A adds the peers in its routing table [IDN,modulusN,IP:port(N bridge),[IDx]] where [IDx] is a list of peers that can perform the introduction to N without requesting the bridges (A and B at least here).

B does the same [IDA,modulusA,IP:port(B bridge),IDB]

Each peer connected to A adds A in its routing table and does the same as above.

Peers are “relayers” and “relayers” are peers but the two functions should not be mixed, even if it can be confusing since the same code and port are used for both functions.

Each peer maintains connections with five “relayers”.

We now have described how internally inside the Convergence network the peer introduction and discovery system based on their peerID would work.

But we need on top of this a mechanism protecting the peerID and correlating it to human readable information, giving the details the peers want to share (name, alias, email address, etc) in a secure way, it will be studied how this can be achieved, probably using a blockchain (like Namecoin [37] or Onename [38] who is using a DHT too to minimize the information stored in the blockchain) and/or Web of Trust concepts [53] and/or a mix of both like keybase.io [56], referencing all the peers information and peerIDs, with a focus on peer verification forbidding name/ID squatting and spoofing.

This, as a whole, solves the problematic known as “Zooko’s triangle” [54] where security, user-chosen/human-meaningful names, and decentralization were three properties conjectured to be impossible to solve inside a single system.

1.3.2 Technical challenges, issues and measurability

The Convergence architecture and applications are **mixing many different interdisciplinary topics** that must be carefully studied.

- *Global architecture*

    - Crypto inside browsers

    Projects such as Peersm have demonstrated the possibility to implement crypto in javascript inside browsers with good performances (2 Mbps streaming with the Tor protocol and 4 hops, therefore 4 encryption/decryption + 4 hashes + 1 TLS decryption for each 500 B block).

    Some parts will remain in javascript since for example the WebCrypto API does not handle certificates, and unfortunately the WebCrypto API does not handle progressive operations too (encryption, decryption, hash) and even if it could, this would not be compatible with the Tor protocol (see ISSUE 22 [39]).

    Indeed the Tor protocol hash to validate blocks is specific and the encryption of blocks progressive, since performances of WebCrypto will be far better than javascript it must be studied if the Tor protocol needs to be modified, for example packet size to allow progressive encryption and modification of hash mechanism.

But keeping in mind that for some applications it must stay compatible with the original Tor protocol.

Another typical issue with crypto inside browsers is the pseudo random number generator, it has to be studied which one can be used, this can be the one of javascript, generally criticized, the WebCrypto one, or a hand-made based for example on the entropy generated from the Tor protocol and connections to the peers that usually generate unpredictable effects.

- Number of hops

Since WebRTC with the peer introduction mechanism makes it difficult to create connections, it appears not easy to build the standard three hops for each connection.

In addition, Convergence does not foresee to use the CREATE_FAST cells, therefore the first node in a circuit can't know it is the first one, it's probably enough but must be studied, especially it has to be noted that two hops is not necessarily the number of hops since the circuits are likely to be extended.

- The Guards concept

This concept of the Tor network insures to keep the same "well established"/long-running first node during a circuit lifetime which reduces the probability of connecting to someone that controls the first and the last node of a circuit and could therefore correlate traffic.

We consider that this does not apply to Convergence, since browsers are volatile, subject to change and given the potential size of the Convergence network, compared to the small size of the Tor network, it's unlikely that the same peer controls the first and last node of a circuit.

- Keys management

If WebCrypto is to be used the keys are stored in an opaque format and are not extractable, if not the onion keys stay in memory and change at each session, but not the long term ID keys, the study will determine what's the best method to be used to protect the keys.

- Blocking browsers - obfuscation

Browsers are difficult to block since users are usually behind a NAT and their IP address change, nevertheless WebSockets and WebRTC protocols have specific fingerprints and could be blocked, there are no signs today that this will happen (for example the Peersm project easily passes the Great China Firewall) but some obfuscation methods must be studied.

- Invading/spying the DHT (Sybils 1)

The initial peers returned by the bridge could be compromised, therefore they could send only compromised peers.

But the onion ID does change for each session then if the peers are continuously returning peers that do not seem close enough to your ID, you could detect that they are compromised.

We have made a study [20] to see how the bittorrent spies do behave and position themselves in the DHT, but unlike bittorrent it is not possible for the peers that would like to compromise the DHT to choose their ID since it is the fingerprint of their public key.

The DHT represents the public table of all the peers, it's unlikely that it's entirely compromised.

If the users don't trust the bridges they can choose the peers "manually" or use the peers introduction feature of the WebRTC DHT.

- Security compared to Tor

For the general architecture if we compare to all the attacks/threats studied in the context of the Tor network, most of them cannot apply to Convergence. Because most of the attacks are linked to the inherent dangerousness of browsing the web and related implications.

If we take the correlation attacks where several peers in the path are colluding to deanonymize someone, as explained previously this is unlikely.

- Adversaries tactics and tracking

As we have seen the guard concept is not necessary for two hops and it’s difficult for the peers that cannot choose their fingerprints to constantly propose themselves as a close peer.

We believe it matters that a node knows it is the first one (like the Tor network nodes), because it can deanonymize those that are connected to it, that’s why Convergence will not use the CREATE_FAST cells, the nodes cannot know in what position they are (unlike the Tor network the first node cannot check among a finite known list of relays if the previous one is a relay or an user).

In addition, two hops is not necessarily the path that will follow the messages, the peers discovery system (which is not the DHT only) makes that a queried peer can extend the path to others it is connected too and ultimately extend to new connections, limited to 5 more hops, it makes it very difficult to trace.

- Spoofing/polluting the network (Sybils 2)

Spoofing/polluting the network is not easy since those that are announcing things are doing it for others, the announced peer IDs are protected by the key mechanism but it has to be studied what could be the impact of possible fake announcements. Indeed some peers could announce some fake peer IDs that matches the key mechanism but the motivation of doing such is not obvious.

More important is, as explained previously, that they cannot position themselves as they like inside the network and possibly attract some peers (as demonstrated in study [20]) in order to mislead them, so for example they can never reach the requested peer ID. Doing such would require enormous resources and therefore seems unlikely.

In addition the attacker would need to invade both levels of the discovery system (the direct one and the DHT) which seems impossible.

- Peers identification and freeriding

As we have seen peers are “relays” and “relays” are peers, each peer must contribute to the network and cannot freeride, for example not performing the relay function.

And the peers are relaying the messages from someone they don’t know for someone else they don’t know, not knowing what the content is about if encrypted end to end by the Convergence application or if the peer is not performing the “relay” for this message.

The list of peers is not centralized like the Tor network, so not easy to get as a whole, but some authorities could walk the DHT and collect the peers.

For some authorities it can make a difference what function operates a peer, for example any form of relay, it has to be studied if in some cases some peers are allowed not to perform all the functions without encouraging freeriding, which could stop the network.

- NAT traversal

Convergence has no other choice than using what WebRTC offers, so using STUN servers for NAT traversal (but TURN relay servers are not considered), we don’t think that it matters that STUN servers know about the peers (and therefore could be able to trace a connection) but this has to be studied, as well as other possible means.

- Code loading and centralization

The code of the Convergence network and related applications is loaded from a web site, typically the Convergence web site and the applications web sites.

But this is not centralized since the code can be duplicated on other web sites in case the above ones are blocked, or can be retrieved from different sources such as social networks and the cloud.

Securing the code loading of a web application is an unsolvable problem if we don’t involve a third party that can validate the code, we will study the different possibilities knowing that it must stay easy to use and must use https.

And we will study another alternative, which is to allow applications to automatically deploy inside the Convergence network, so newcomers/new apps can quickly extend and be used, the mechanism will need the peers to opt-in for the automatic installation of a given application (ie it will be prompted “would you like to install application X?), maybe a blockchain mechanism could be used to secure the applications and their code, a plausible solution could be to use a similar mechanism than Ethereum smart contracts to deliver/deploy the code.

And to motivate the peers we will study if a bitcoin micro payment system can be used according to the participation of the peers for a given application.

- WebRTC mode and block size

WebRTC empiric uses regarding packet loss possibilities advises a size of 1024B < payload of IP, UDP, DTLS, and SCTP protocols ~1150 B - unreliable mode

So, since the Tor protocol is fragmenting by blocks of 498B it seems logical to keep this size (we could change this but the system must be compatible with the Tor network again) but this

introduces some other difficulties that must be studied (see data validation below for the file sharing application)

- WebRTC security issue

See [40], WebRTC is using self-signed certificates to avoid fingerprinting, the SDP (peer introduction) does include the fingerprint of the certificate, this is not enough to guarantee that there is not a MITM peer in the middle. Therefore the standards foresee to add another mechanism where the fingerprint of the DTLS certificate will be signed by a third party that knows you, typically a social network where you have an account.

This is of course far from protecting your anonymity and privacy and cannot be used in Convergence context, so we will study how to use the Tor protocol CERTS cells mechanism to secure this aspect.

- Browsers issues

The code loading of the Convergence networks and applications must use https. The WebRTC DHT is establishing WebSocket connections with the Bridges, that are likely to use self-signed certificates for the TLS connections of the Tor protocol.

Therefore TLS WebSockets (wss) cannot be used since browsers do not accept self-signed certificates.

And non TLS WebSockets (ws) cannot be used neither since browsers do not allow to establish non TLS connections inside a https page.

This issue is being discussed on the W3C lists [41] since we see no reasons why ws could not be used inside a https page with a protocol more secure than TLS (ie the Tor protocol).

One alternative is possible, which is to use http and not https to load the code but this is insecure, this problem will be studied.

The same issue occurs with Service Workers, only https is allowed for service workers (therefore they cannot run non TLS WebSockets) which we are planning to use for some applications so they can continue to run in background after the user closed his browser.

- Blockchain/WOT for peerIDs and data

A blockchain will probably be part of the system used to store and protect peers data, it can be combined with other technics such as Web of Trust or keybase-like who generally present the inconvenient to be centralized, this subject is actively discussed as shown by [61] who presents different alternatives not blockchain-based for now that can be related to Letsencrypt [62] too which we will consider since it can be combined with blockchains as well.

Blockchains are presenting different difficulties experienced by projects such as Namecoin and Onename, please see [42], among those the always increasing size of the blockchain, what proof should be used (work, stake, burn, allocation), what network should host it and how the data can be securely stored with a P2P model while not overloading the blockchain (for example using a DHT allowing to retrieve data from the references stored in the blockchain) and forbidding ID squatting/spoofing, some projects like “The internet of coins” [63] are

starting to envision some solutions to solve some of these issues, like a P2P meta language allowing different blockchains/systems to interact between each other without the need for the peers to retrieve the complete blockchains.

- *<patent>*

  <patent>

- *Applications*

  - *Chat and messaging applications*

    Define an architecture that respects the requirements of Figure 3.

  - *File sharing*

    The file sharing application will be defined on the Peersm model, where basically the content is referenced by infohashes that follow the same rules than peer IDs for content discovery (ie add to the WebRTC DHT the infohashes registration)

    The streaming mechanisms will be clarified.

    There is one identified domain that the state of the art seems not to be able to solve: data validation.

    Unlike bittorrent, there is no metadata file to describe pieces and check their integrity.

    A file or a stream has been brought initially to the network by one seeder that used his private key or generated one to sign each piece, the corresponding public key is sent with the answer to the first chunk query: [size, type, public key]

    The signing process is based on asymmetric crypography, it does include a timestamp for live streaming.

    This is the theory but given the size of the blocks (498 B) it appears not realistic to perform a signature operation for each block, 4 Bytes are allocated to the signature, they must insure with a good probability that a piece has good chances to be correct. Unfortunately this “light” signature mechanism seems not to exist and usual methods like Bloom filters seem to reach quickly their limits (multi GB files for example), we will study how this can possibly be solved.

  - *Social networking*

    The specific difficulty of this application is to define where the data are hosted, how they can be shared depending on the rights given by the owners and how they can be managed, moved if required or transferred in case of a compromised peer ID or platform change.

  - *Cooperative application*

    The same difficulty applies to this application regarding how the data are hosted and shared, as well as how insure their integrity/update in a cooperative environment.

- *Crypto currency*

  Crypto currency networks are already difficult to trace, it will be studied if there is an interest to run a crypto currency network on Convergence, and what could be the future of blockchains whose size increase rapidly and are already generating a phenomenon of recentralization of the crypto currency networks.

- *Browsing application*

  <patent>

  It will be studied if the "relays" that exit toward the web can reasonably exit in the clear or must exit through the Tor network, if the "relays" are really necessary and if the peer cannot exit itself.

- *uProxy*

  <patent>… another difficulty is to determine how the final peers authorizes others to exit (peers that want to evade censorship are supposed to know someone that has installed uProxy and authorize him to exit)

- *Tor nodes*

  As highlighted previously the difficulty here is to manage the upload bandwidth limitation of browsers, so a peer connected to a Tor node via WebSockets does not degrade completely the bandwidth of the circuit established.

  Most likely the network Tor node will have to send data to several Convergence Tor nodes (peers) which induces a mechanism of data sequencing.

- *IOT*

  The difficulty is that users can usually not connect to their connected objects (at home for example), and connected objects have different architectures, some might be able to become peers inside the Convergence network but maybe the best solution would be to define how aggregated traffic of connected objects can be anonymized to their users.

2. Impact

2.1 Expected impacts

2.1.1 Expected impact

- *To demonstrate how a distributed architecture can enable new data services and disruptive (e.g. commons-based) economic models, and become a viable decentralised alternative to the current dominant data management platforms which are gathering big data at global scale in a centralised manner;*

  How this is addressed:

The Convergence architecture is a fully decentralized serverless architecture, is autonomous and can't be controlled.

The decentralized, distributed computing and open source aspects of the ensemble will allow new services to be easily built without requiring substantial resources given the strength provided by such network and without depending on other structures, especially for startups and SMEs, allowing innovation and new economic models.

Convergence shows new concepts, both for the base of its architecture and modifications inside browsers (<patent>) **allowing to define disruptive services inside browsers not needing any longer expensive servers or infrastructures, easy to scale and easy to sustain, making it easy to define the associated economic models (**for example based on crypto currency payments to the application providers by the users based on their digital peerID).

The crypto currency application on Convergence network intends to anticipate the ineluctable recentralization of the bitcoin network since it will become difficult for normal peers to store/update the entire blockchain, or at least it will become difficult for newcomers to retrieve the entire blockchain given its size, therefore the big entities that will store the blockchain might become like banks, breaking the intent of the whole system.

The potential size of the Convergence networks gives it the ability both to store big data but to compute them too in a distributed way.

How this is measured:

The Convergence code and applications can be installed from any website in the world, from an iframe or from new methods that we will define involving the Web Components and Service Workers. They can work in background and continue to run even if the browser is closed.

Therefore anybody visiting such web sites can participate to the network during the navigation or install Convergence and its applications.

In addition it will be studied how an application can be installed without the need of websites and with a peer opt-in mechanism **so a newcomer can spread a new application quickly inside the Convergence network**.

With billions of browsers in the world this gives to Convergence the potential to be the biggest network in the world, far bigger than any possible centralized system and main data management platforms, or other P2P networks.

To motivate the peers to scale and sustain the network we will study the possibility of crypto currency micro payments, such as the one mentioned above but not for the application providers but for those that are contributing to the network.

- *To demonstrate that citizens' generated data can be made available as part of a common distributed and decentralised architecture, open to all, so to allow new entrants to aggregate data on demand, bringing unanticipated features and innovative services;*

How this is addressed:

The social networking, file sharing and cooperative work applications show how citizens can bring data to the network, **whether private or public and how they can manage it**, simply from their browser, depending of the type of data it can be stored inside their browser and be accessible by others, or distributed inside the decentralized network, themselves participating to store the data of others and share it.

The list of services in this proposal are not exhaustive, the peerID blockchain can be used for quite a lot of new services involving for example payment, health, identification data.

Disruptive non existing services are presented in the proposal to contribute to this effort, like the general concepts of peers relaying the data for others and unanticipated services where the peers are acting as autonomous nodes using the <patent> features to bring new features on top of the existing ones (relaying data for other networks or <patent>), allowing to extend the distribution and decentralization of data, or like the IOT application allowing to aggregate the users data and to make it available anonymously.

How this is measured:

Each application will assess how the data are distributed stored and managed

- *To develop an architecture and open standards allowing European citizens to retain full control over their digital identities, and to move their personal profiles between different platforms, for distributed or centralised (data portability);*

  How this is addressed:

  The Convergence network and applications are based on the Web standards, the code is open source. Some applications could decide not to open source their code but in any case what is doing the application can be verified since a javascript application cannot hide its code.

  The Convergence network includes an identity management system based on cryptographic keys that are owned by the citizens and stored inside their browser, the keys are not extractable but can be transferred, the users have the total control of their identities and data, they can use them for chat, messaging and the social networking application defines how they create personal profiles, share them and the associated data, decide what must stay locally, what can be distributed, where the data is stored (ie peers, servers, blockchain, mix of all) and **how they can remove distributed data or reaggregate data to centralize it or redistribute it on another platform**

  How this is measured:

  Assess how the users keep control on their identity/data and portability of it.

- *To create a level playing field for the development of new collaborative applications and services based on emerging participatory innovation models that are intrinsically respectful of privacy and ethics.*

  How this is addressed:

The Convergence network defines a level playing field (comparable as already mentioned to Ethereum) where anybody can create new innovative applications, test them in an already deployed network and distribute them, this is rendered possible by the inherent collaboration of all the peers and their participation to scale and sustain the network allowing the fast and easy deployment of applications including collaborative applications.

While this might not be mandatory for any application, **the core concepts of the Convergence architecture are to protect systematically privacy and anonymity**, protecting by default the users, therefore being respectful of the ethics and bringing new innovative models like built-in anonymous browsing.

How this is measured:

Assess how privacy and ethics are respected and how the playing field makes it easy to deploy applications, for example based on the automatic application deployment mentioned above.

2.1.2 Strengthening Innovation Capacity, Competitiveness and Growth

The Convergence network offers a complete solution to deploy widely and quickly new applications without the need of any other resources than the applications themselves, enhancing innovation, competitiveness and growth of the providers that can now compete with the centralized expensive infrastructures of the main actors.

2.1.3 Societal and environmental impact

The design of the Convergence network provides privacy and anonymity, in addition it is extremely difficult to block or to censor, allowing access to anybody, even in censored countries, to its applications.

Browsers are replacing servers and big data centers **allowing consequent energy savings**.

2.1.4 Contribution to standards

The Convergence concepts will help the standards to evolve, and **the <patent> mechanisms are expected to be standardized, as well as the underlying problematic that Convergence concepts bring**, the main one being that the model of an application inside browsers tied to a domain and secured by certificates linked to this domain appears obsolete for future applications where entities cannot be tied to a domain (typically a WebRTC peer), showing the necessity to move to an entityID management system.

References

[1] https://www.uproxy.org/
[2] https://www.torproject.org/
[3] https://www.w3.org/TR/webrtc/
[4] https://www.w3.org/TR/2011/WD-websockets-20110929/
[5] https://www.w3.org/TR/XMLHttpRequest2/
[6] https://www.w3.org/TR/IndexedDB/
[7] https://www.w3.org/TR/FileAPI/

[8] https://streams.spec.whatwg.org/
[9] https://www.w3.org/TR/workers/
[10] https://www.w3.org/TR/WebCryptoAPI/
[11] https://en.wikipedia.org/wiki/SOCKS
[12] https://www.w3.org/TR/service-workers/
[13] https://en.wikipedia.org/wiki/Proxy_auto-config
[14] https://github.com/Ayms/node-Tor#anonymous-serverless-p2p-inside-browsers---peersm-specs
[15] https://en.wikipedia.org/wiki/Kademlia
[16] https://geti2p.net/en/
[17] https://freenetproject.org/
[18] http://www.bittorrent.com
[19] https://en.wikipedia.org/wiki/Cjdns
[20] https://github.com/Ayms/torrent-live
[21] https://github.com/Ayms/node-Tor
[22] https://webtorrent.io/
[23] https://crypto.stanford.edu/flashproxy/
[24] https://trac.torproject.org/projects/tor/wiki/doc/Snowflake
[25] https://www.peer5.com/
[26] http://project-maelstrom.bittorrent.com/
[27] http://www.tribler.org/
[28] http://conferences.sigcomm.org/hotnets/2012/papers/hotnets12-final141.pdf
[29] https://diasporafoundation.org/
[30] https://en.wikipedia.org/wiki/Logjam_%28computer_security%29
[31] https://www.deepdotweb.com/2015/11/27/police-log-ips-making-arrest-by-planting-own-nodes-in-freenet/
[32] https://www.eff.org/secure-messaging-scorecard
[33] http://secushare.org/comparison
[34] https://en.wikipedia.org/wiki/Same-origin_policy
[35] http://lists.w3.org/Archives/Public/public-webapps/2015OctDec/0187.html
[36] http://rekla.im
[37] https://namecoin.info/
[38] https://onename.com/
[39] https://www.w3.org/2012/webcrypto/track/issues/22
[40] https://torrentfreak.com/huge-security-flaw-leaks-vpn-users-real-ip-addresses-150130/
[41] http://lists.w3.org/Archives/Public/public-webapps/2015OctDec/0200.html
[42] http://blog.onename.com/namecoin-to-bitcoin/
[43] https://github.com/w3c/webcomponents
[44] https://mailman.stanford.edu/mailman/listinfo/liberationtech
[45] https://lists.torproject.org/cgi-bin/mailman/listinfo/tor-talk
[46] http://lists.zooko.com/mailman/listinfo/p2p-hackers
[47] https://www.cypherpunks.to/list/
[48] http://lists.w3.org/Archives/Public/public-webapps/
[49] https://mail.mozilla.org/listinfo/es-discuss
[50] https://lists.w3.org/Archives/Public/public-script-coord/
[51] https://trac.torproject.org/projects/tor/wiki/doc/Snowflake/Fingerprinting
[52] https://trac.torproject.org/projects/tor/wiki/doc/PluggableTransports
[53] https://en.wikipedia.org/wiki/Web_of_trust
[54] https://en.wikipedia.org/wiki/Zooko's_triangle

[55] https://freedom-to-tinker.com/blog/mcarlsten/an-empirical-study-of-namecoin-and-lessons-for-decentralized-namespace-design/
[56] https://keybase.io/
[57] https://mailman.stanford.edu/pipermail/liberationtech/2015-November/015680.html and https://mailman.stanford.edu/pipermail/liberationtech/2015-November/015695.html
[58] http://lists.w3.org/Archives/Public/public-webapps/2015OctDec/0200.html
[59] https://lists.torproject.org/pipermail/tor-talk/2016-February/040441.html
[60] https://lists.torproject.org/pipermail/tor-talk/2016-February/040451.html
[61] https://github.com/saint/w2sp-2015/blob/master/SP_SPSI-2015-09-0170.R1_Syverson.pdf
[62] https://letsencrypt.org/
[63] http://internetofcoins.org/home
[64] https://ethereum.org/
[65] http://internetofcoins.org/blog