university of groningen / faculty of mathematics and natural sciences / computing science

University of Groningen

Decentralized Link Sharing, Towards a Framework for Decentralized Applications

Master Thesis

Laurence de Jong

August 17, 2015

Supervisors:
prof. dr. ir. M. (Marco) Aiello
dr. A. (Alexander) Lazovik

Research performed at Media2B, under supervision of:
dr. B. (Berco) Beute
dr. H. (Henk) Doornbos

Acknowledgments

First, I would like to express my sincere gratitude to Berco and Henk, for introducing me to the topic and for their daily guidance and inspiration. I would like to thank my supervisors, prof. Marco Aiello and dr. Alexander Lazovik, for allowing me to perform research on this subject. I thank Leon and Nykle for their feedback and discussions. I would like to thank my girlfriend for listening to my endless rambling about technology. Lastly, I would like to thank all others who have supported me.

Laurence de Jong
Groningen
August 17, 2015

Abstract

The current internet is ruled by centralized entities. These central entities are trusted with users' data in good faith. However, services shut down, data is not always protected, and governments perform mass surveillance (ab)using these easy targets. Another issue is that when a service becomes more popular, more resources are required. However, the users of a service often have unused resources, like disk space and bandwidth, available. In an attempt to solve these issues, decentralization is mentioned as a possible solution. To discover what the architecture of a decentralized solution should be, a common use case on the internet is selected: link sharing. Reddit is the best known link sharing website. Reddit suffers from its popularity and controls a huge amount of data, data which is created by its users. A novel decentralized link sharing system is designed in this research, with the goal of making a small step towards finding a general architecture for decentralized systems.

Research starts with describing projects and protocols which use decentralization as a means to achieve scalability, preserve privacy or to let users control their own data. In the analyzed projects, the three common technologies are BitTorrent, distributed hash tables (DHT) and blockchains. Blockchains are used as a store for data which can be trusted. Unique names and public keys are examples of data which should be stored in blockchains. Distributed hash tables are used as a key value store which is distributed across nodes and used for data with lower requirements. BitTorrent technology is used to create active replication of data across nodes. Together, these technologies can be used to create fully decentralized applications.

A novel decentralized link sharing system is designed in this research, employing the previously mentioned technologies. A blockchain is used for the registration of users and the storage of public keys. With the help of BitTorrent, links are actively replicated across nodes, resulting in instant updates of links in topics the user is interested in. Links are stored in a distributed hash table as well, which makes sure data is available, even where there are no seeders for a torrent.

The qualitative evaluation of the novel decentralized link sharing application shows that security, protection of privacy and scalability are characteristics which do not come for free. The fact that applications are distributed over many systems and platforms, without developers having control over them, means that they are significantly harder to secure. In fact, users carry more responsibilities themselves, whereas users currently trust a company to take care of the responsibilities. Research will need to be carried out to make communication secure and encrypted by default.

To conclude, the blockchain is important in the development of decentralized systems. Together with distributed hash tables and BitTorrent, they form the foundation for decentralized applications right now and potentially for many to come.

Contents

List of Figures

Chapter 1

Introduction

> Start where you are. Use what you have. Do what you can.
>
> Arthur Ashe

The internet plays a significant role in many lives of people around the world. Organizing your life online is the new normal. Reading the news, watching videos, planning your work and chatting with family and friends. All are activities which are common for most nowadays, and they will become normal for a larger group of people in the near future. The internet fulfils our human nature to share, communicate and collaborate. Look around, and imagine that most of the twenty year-olds and younger do not know a world without internet. They have embraced it as if it has always been there, and many of them do not know what to do without it.

1.1 The power of central control

The network we call the internet is often described as a worldwide system of interconnected computers and networks, enabling billions of devices to connect to each other. That description hides the *information silos* as they are on the internet today.

In the beginning the internet was a network of computers all communicating with each other. Take for example Usenet, it is a distributed worldwide discussion platform established in 1981. All that was needed was and initial connection to one of the other computers in the network in order to send and receive messages. Usenet was famous for having no central control.

Over the years the internet became more and more popular, and the decentralized network became more and more centralized, both on the services layer and the infrastructure layer. A small number of online operators has a large concentration of power when it comes to services. Both the topology and the ownership of internet infrastructure are centralized as well. Huge backbones with national and international cables are owned and maintained by a small number of companies. These centralization forces are mainly because of two big advantages, it is easier to manage centralized services and it is easier to make money off them. Google can deliver a better email service than the system administrators at your work, or than your Internet Service Provider (ISP). And of course they have a better spam filter, they can look into everybodys

email. In the end it comes down to convenience. It is easier to upgrade one system instead of many, and it is easier to monitor one system instead of many.

A regular visit to a website involves a client and a server, where the client sends a request to the server, and the server responds with the requested website. Most of the infrastructure used in handling a request is not owned by the client nor the server. Large glass fiber networks are owned by large consortia, while the company giving you access to these networks is your ISP. What if one of these companies decided to put a tap on their cables to monitor the traffic that is going over their cables? Or worse, what if they are altering the websites you visit to obtain access to your accounts?[1]

Big centralized services are an easy target for surveillance, censorship and hacking. With the (ongoing) revelations of Edward Snowden[2] it became clear that global mass surveillance is big and happening. Government surveillance programs have contracts with the biggest companies[3] in the internet world, which allows them to extract data and monitor citizens on a truly massive scale. It is not a matter of monitoring a single suspect, it is a matter of systematic registration of all activities of everybody, regardless of their intentions. These centralized services are the perfect place to tune in on in order to obtain all the information you want. They can register who are communicating, for how long, at what times, where you are, which devices are used, and so on. Every website you visit, every personal chat with your loved ones, every thought you share on Facebook, every song you listen to on Spotify and every movie you watch on Netflix. Sometimes they know it because they force the company to give the information, and sometimes they know because they tapped all the traffic from and to these central hubs.

1.2 ”Your data is safe with us”

Uploading work-related documents to Dropbox, holiday pictures to iCloud and videos to YouTube. You are giving your data to a company, and trust them to care of it. They have to take care of it because otherwise you will go to another company (right?). Nonetheless, over the last couple of years there have been numerous reports of hacks at large companies. Some of the largest data breaches can be seen in figure 1.1. Often huge amounts of data and databases are copied, and later offered for some amount of money, or just publicly posted for anyone to verify whether they have been affected by the hack.[4]

We have to trust them with storing our data, storing it safely and being open about what happens with the data. It can take days, weeks or months before companies reveal they have been hacked and your unencrypted passwords have been leaked. During this time your account details may have been compromised. Even when they promise to protect your data, they might still sell it. The recently bankrupted RadioShack is aiming to sell the data of millions of customers.[6]

[1] ”Facebook’s login system is being hijacked by China’s Great Firewall” - The Verge `http://theverge.com/e/8272158`

[2] The NSA files - The Guardian `http://www.theguardian.com/us-news/the-nsa-files/all`

[3] ”NSA PRISM program taps in to user data of Apple, Google and others” - The Guardian `http://gu.com/p/3gd58`

[4] Verify whether account details have been leaked at `https://haveibeenpwned.com/`

[6] ”Bankrupt RadioShack aims to sell 67m customer names and addresses” - The Guardian `http://gu.com/p/492ht/sbl`

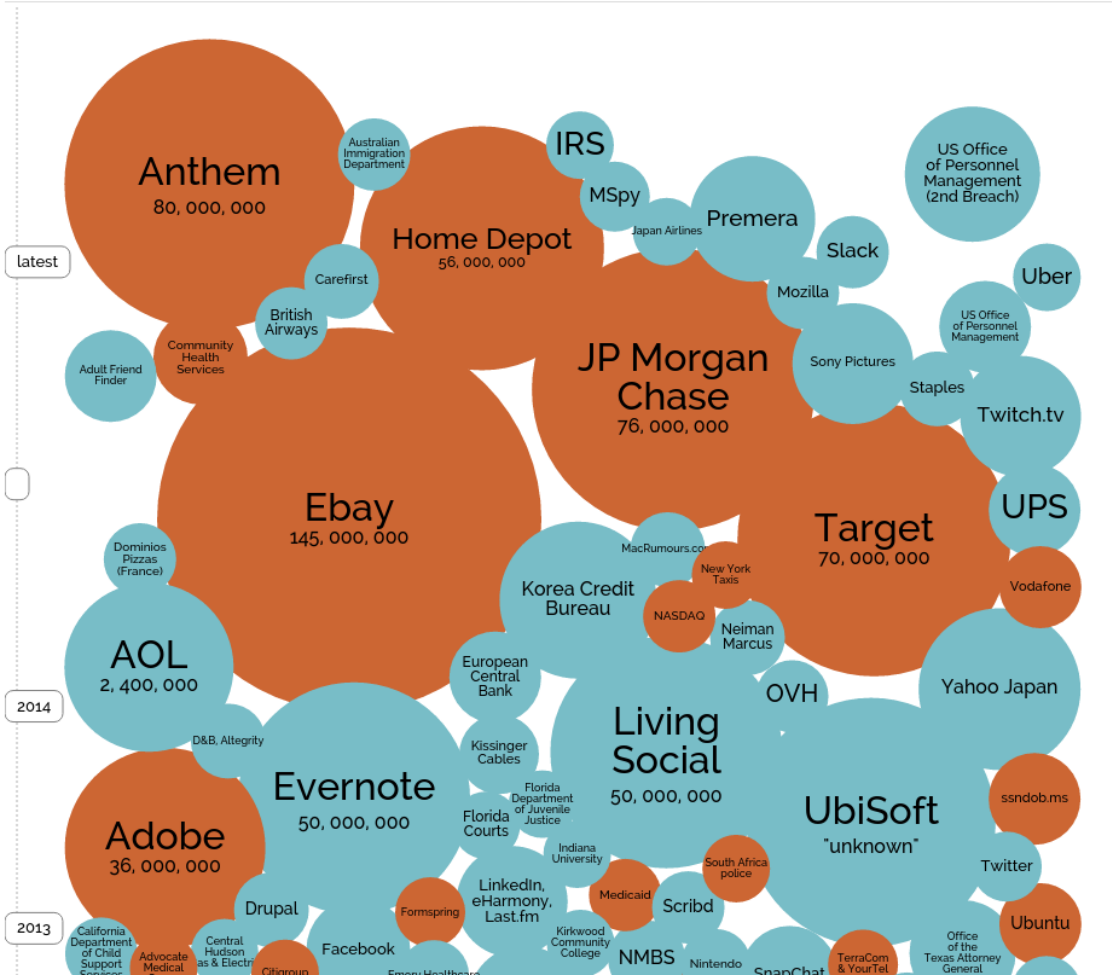

Figure 1.1: World's Biggest Data Breaches of more than 30,000 records. (Updated 13 June 2015)[5]

Let us assume your data is stored safely and will not be sold. Are you certain your that you can retrieve your data when you need it? After all, you do not have the data, it is stored at a server owned by a company. What if they decide to bail out, to end a service, or are forced to close? Examples are Google Reader, Google Wave, the Dutch social network Hyves and Megaupload. Your data is at their servers, and if they shut down you can only *hope* you get the opportunity to retrieve your data. In the best cases you can transfer your data to another service, like with the shut down of Google Code.[7]

The central services on the internet today are closed platforms, or otherwise called *walled gardens*. Companies, such as Google, Apple and Facebook, build complete software ecosystems intended to serve all your needs. They control the data, and the access to the data. Not only for users, but developers as well. Restrictions on which content can be accessed, which ap-

[7] "Bidding farewell to Google Code" `http://google-opensource.blogspot.nl/2015/03/farewell-to-google-code.html`

plications can be installed, and measures which make it hard to switch to other companies, are common place. As a result, these isolated stacks often lead to introduction of expensive licenses, incompatibilities, vendor lock in, and overall full control over data of their users.

1.3 Censorship

In many countries the right to speak your mind, and read the information you want, is not as common as one might expect. Censorship is happening and not just in suppressive governments. There are several technical ways censorship can be performed. Websites can be removed from DNS servers, or packets of data can be filtered based on forbidden words. During the Arab Spring in 2010, Egypt, Syria and Libya shut down all internet traffic to prevent people from organizing events targeted at the government.[8] The power of centralization is used to perform censorship and to suppress people. It is clear from the 2014 edition of the annual Internet Monitor report [1], that online censorship is a problem which people are aware of and try to overcome. The report stated that nearly seventy percent of all Iranian young people reported to use proxy servers to circumvent online censorship.[9] The software used to connect with these servers has been declared illegal by the government.

1.4 about:reddit

Ten years ago the website reddit.com was created by two roommates from the University of Virginia. In 2014 the website served 71 billion page views[10] and it is one of the most popular websites in the world.[11] Reddit is like a bulletin board and their sections are called *subreddits*. There are over eight thousand active subreddits on a wide variety of topics. From pictures of cute animals[12] to presidential question and answer sessions[13], from technology discussions to adult content. Reddit is known to be relatively open, and has only a few rules[14], most of them are there either to comply with the law or to make sure the website continues functioning normally.

Each of the subreddits has *moderators* which establish the rules for their community, just as real-life communities. The content is user generated and consists of links to external websites, they are called *posts* or *submissions*. Another form of content are so-called self-posts. These are posts consisting of just text and are often used for announcements, or for initiating discussions. Users can discuss the submissions in the form of *comments*. The submissions and comments can be *upvoted* and *downvoted*, which results in community curated lists of the best posts per subreddit.

[8] https://en.wikipedia.org/wiki/Internet_censorship_in_the_Arab_Spring

[9] http://www.dailysabah.com/mideast/2014/09/08/70-percent-of-young-iranians-are-online-illegally

[10] Reddit in 2014 report http://www.redditblog.com/2014/12/reddit-in-2014.html

[11] Reddit on Alexa http://www.alexa.com/siteinfo/reddit.com

[12] "aww" subreddit https://www.reddit.com/r/aww/

[13] IAmA session with Barack Obama https://www.reddit.com/r/IAmA/comments/z1c9z/i_am_barack_obama_president_of_the_united_states/

[14] Reddit's rules. https://www.reddit.com/rules/

Figure 1.2: Reddit's logo

Reddit is known for both its positive and its negative effects.[15] There are many cute and funny pictures, constructive discussions about social issues, and educational posts. On the other hand, people argue that Reddit is racist[16], does not intervene and act in cases of bullying, humiliation, *doxing*[17] and threats. Moreover, there are discussions about the subreddits inciting harm or violence againt groups and individuals, subreddits containing adult material, and subreddits to accomodate gun trades.[18] The company behind Reddit has often found itself in a situation in which it did not want to intervene but actually had to.[19] Some websites even declared Reddit as "failed".[20][21]

1.4.1 Hug of death

With the success of Reddit comes a powerful side effect. Having an enormous user base is like controlling an enormous fire hose of page views. When a link is posted, it can become popular in a couple of hours. The link will be followed by thousands of users. Smaller websites, such as a local newspaper or a personal blog, are not designed for these flash crowds. As a result, the servers will be overloaded and services will be unavailable for hours on end. The effect already had the name of *Slashdot effect*, due to the same effect from the website Slashdot.[22] For Reddit it has earned the name "Hug of death".

It is not just the websites Reddit is linking to that are affected. Reddit itself is often coping which large amounts of traffic, resulting in the feared 503 error message as illustrated in figure 1.3. Posts, comments and votes are often queued because the high load.[23]

1.4.2 Technology

Reddit is open source and built using Python, with Pylons as the web framework.[24] The database behind Reddit is Apache Cassandra. An Application Programming Interface (API)

[15] Statistics about the content of Reddit. http://minimaxir.com/2014/12/reddit-statistics/

[16] Subreddit moderators asking Reddit to take action against racism. https://www.reddit.com/r/blackladies/comments/2ejg1b/we_have_a_racist_user_problem_and_reddit_wont/

[17] The researching and disclosure of personal information. https://en.wikipedia.org/wiki/Doxing

[18] Controversial subreddits. https://en.wikipedia.org/wiki/Controversial_Reddit_communities

[19] Reddit declaring not to be responsible. http://www.redditblog.com/2014/09/every-man-is-responsible-for-his-own.html

[20] "Reddit is a failed state" - The Verge http://theverge.com/e/5885404

[21] "Is Reddit broken beyond repair?" - The Kernel http://kernelmag.dailydot.com/issue-sections/staff-editorials/10749/reddit-broken-beyond-repair/

[22] https://en.wikipedia.org/wiki/Slashdot_effect

[23] Reddit's status page showing number of comments to be tallied http://www.redditstatus.com/

[24] https://github.com/reddit/reddit

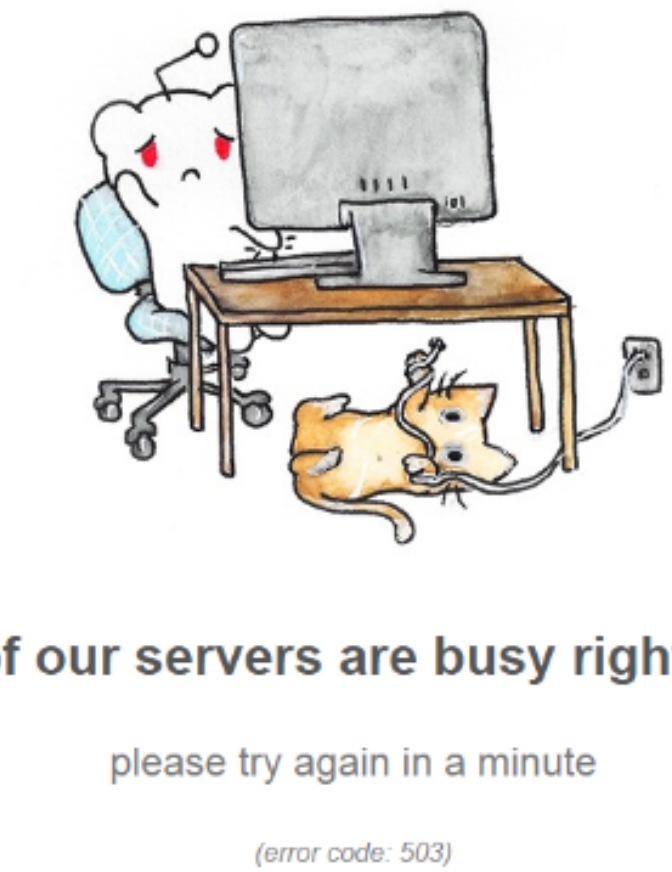

Figure 1.3: Reddit's servers are busy, resulting in a 503 error.

is used to allow developers to create their own applications for mobile platforms. Numerous apps have been developed for iOS, Android and Windows.

1.4.3 Revenue model

Although Reddit used to be owned by the company Advance Publications, it is now an independent company with its own management board. There are a couple of main revenue streams providing streams of income. Advertisements, Reddit gold, merchandise and donations are the main income sources. Reddit gold can be bought for a small fee which allows users to, among other things, disable ads, access special features and subreddits, and allows users to "gild" a post or comment.[25]

[25] `https://www.reddit.com/gold/about`

1.4.4 Evaluation

Reddit and the websites it is linking to suffer from flash crowds and suffer from server overload. Reddit is struggling with its position of having an open platform and has often been accused of censorship. Reddit is an independent, commercial company which is hosting one of the most popular discussion platforms on the internet. In the past, technology companies have been bought or shut down resulting in the loss of enormous amounts of data and the loss of an invaluable platform. Controlling Reddit means controlling large amounts of data. Should a platform like Reddit be in private hands?

1.4.5 Current affairs

During the writing of this thesis, a number events have taken place on Reddit. A Reddit employee, Victoria, has been fired. Victoria was responsible for the organization of the previously mentioned AMA sessions. Reddit did not communicate to the community about firing Victoria in time. The scheduled AMA sessions with celebrities could not continue and moderators had a hard time contacting the celebrities, as Reddit was their main medium for communication. In the months before this event, users of Reddit complained about management decisions which had a negative impact on Reddit and its communities. The firing of Victoria was the last straw for the communities. Moderators set their subreddits to 'private' or 'dark' as a form of demonstration against the recent events. As a result, dozens of subreddits could not be reached, and users were unable to reach the content in these subreddits[26]. In the days after this 'subreddit drama', the CEO of Reddit resigned and one of the founders Reddit stepped in as new CEO[27], announcing that clear content policies will take place. A couple of days later the chief engineer at Reddit resigned, stating that she had lost confidence in the direction of the company[28]. Again a couple of days later, the new content policies are discussed by the Reddit team and its communities[29]. The main point many subreddits fear is that "offensive" content and subreddits will be banned. They argue that anything can be offensive for other people. Since the banning of Victoria, many redditors have been searching for alternatives. A subreddit for creating a reddit alternative and a subreddit for a decentralized reddit have been created. A small Reddit alternative called Voat could not handle the large group of redditors checking out their website and has been offline for long periods. There are a couple of lessons to be learned from these series of events. Reddit and the moderators have huge control over their communities. There is huge demand for a Reddit in which users can do what they want, without fearing to be censored or banned. Voat, just as Reddit, is not able to handle sudden increases in popularity.

1.5 Research questions

Centralized solutions are easy targets for attacks, and they can be abused for censorship and mass-surveillance. In these solutions, users are often bound to lose because they have no control

[26] https://www.reddit.com/r/SubredditDrama/comments/3bwgjf/riama_set_to_private_over_mod_firing/
[27] https://www.reddit.com/r/announcements/comments/3cucye/an_old_team_at_reddit/
[28] http://arstechnica.com/business/2015/07/reddit-loses-another-prominent-female-employee-as-chief-engineer-quits/
[29] https://www.reddit.com/r/announcements/comments/3djjxw/lets_talk_content_ama/

over their data and have to trust the creators to handle their data with care. When a company quits, users have to trust the company not to sell their data, hope to retrieve their data, and hope there is an alternative to import the data into. Websites often struggle to maintain a big enough server fleet to handle peak performances required for flash crowds.

Previous research in Computer Science has shown (section 3 that, decentralized applications offer solutions to the before mentioned problems. However, it is unclear how a decentralized link sharing application should be constructed. The state of the art, when it comes to decentralized applications, is discussed in chapters 2 and 3. There is no known research for creating a decentralized link sharing application. However, many other decentralized applications have been built which show the potential of decentralized applications. Relevant decentralized applications include Twister (a microblogging application), IPFS (a file sharing application), and Tox (a chat application).

In this thesis, I embark on a mission to research how a decentralized link sharing system should be constructed, with privacy, scalability, and user control over data as the main driving forces. These driving forces are evaluated according to a qualitative approach. Such a link sharing sharing system has not been researched before, and can function as a blueprint or framework for future decentralized applications.

In order to find out how a decentralized link sharing should be constructed, the following research questions will be answered in this thesis:

1. Which technologies are suitable for creating a decentralized link sharing application?
2. How should these technologies be used?
3. How do these technologies score when it comes to privacy, scalability and user control over data?

1.6 Methodology

The methodology applied for the qualitative research in this thesis consists of two phases. The first part is the research of related work and previous technologies. Research in the area of decentralized applications has had a major development with the introduction of blockchain technology in 2009. Blockchain technology is the basis for the peer-to-peer payment network called Bitcoin (more on blockchain in section 2.1.2). Despite being mentioned in scientific papers, the paper explaining the workings of the blockchain has not been published by any of the major established scientific publishers. Instead, the paper has been published under a pseudonym on the website of Bitcoin itself. It is not uncommon for scientifically significant papers in the area of decentralization to use other methods of publishing instead of the established publishers. The reasons for this trend are out of scope for thesis, but the open source and peer-to-peer nature of decentralized networks appears to attracts researcher whom prefer openness in research. Thereby, making it harder to perform research in the form of a systematic literature review. Although papers are not always published by publishers, references to these papers can be found in other papers which are published by publishers. Google Scholar *does* include these paper. However, Google Scholar is *not* an established major publisher and is known to include grey literature. Consequently, papers which are of low value to the scientific community might be included in the results as well.

A literature research has been performed while bearing in mind the previously mentioned obstacles. Each paper has been evaluated according to the study quality assessment criteria as defined in "Guidelines for performing Systematic Literature Reviews in Software Engineering" [2]. Studies have been included or excluded based on three quality concepts as described in table 1.4.

Term	Synonyms	Definition
Bias	Systematic error	A tendency to produce results that depart systematically from the true results. Unbiased results are internally valid.
Internal validity	Validity	The extent to which the design and conduct of the study are likely to prevent systematic error. Internal validity is a prerequisite for external validity.
External validity	Generalisability, Applicability	The extent to which the effects observed in the study are applicable outside of the study.

Figure 1.4: Quality concepts for including and excluding studies

The second part of the research consisted of analyzing and evaluating projects in the area of decentralized applications. As a starting point I have taking the list of projects in decentralized applications as documented by Jagerman et al. in their paper titled "The fifteen year struggle of decentralizing privacy-enhancing technology". They state that the largest repository of decentralization attempts can be found at redecentralize.org. The list of projects in their repository[30] is extensive and includes dead projects, current projects and projects which are in development. Not all projects from this repository have been analysed, and not all projects discussed in this paper are on the list. The projects discussed in this paper are a combination of projects from the list and from projects mentioned in scientific papers. They have been assessed on their quality and their contribution in terms of their potential towards the goals defined in this paper.

1.7 Scope

The scope of this research is limited to a literature research in the area of decentralization. In the research the most prominent and technologically interesting projects will be described. However, it is not the goal to give an elaborated list of *all* projects in the field of research, nor is it the goal to create an exhaustive comparison of all technologies.

There will be *no* proof of concept or implementation for the decentralized system within the time frame of the research. The number of topics touched by this projects is large. Understanding and explaining why and how technologies can be used in a decentralized application are, in my opinion, of greater value than rushing towards a proof-of-concept without scientific foundation. The thesis includes examples of how to implement a decentralized link sharing. These examples can be used directly in a proof of concept.

[30] `https://github.com/redecentralize/alternative-internet`

1.8 Contribution

With this thesis I contribute to the scientific community by offering a theoretical solution for a decentralized link sharing system. The system is designed with the goals of preserving privacy of the end-users, using the end-users resources to create a system which scales with the number of users, and to let end-users retain control over their data. I have investigated the technologies used in decentralized applications and selected the technologies best suited and currently available to attain the stated goals. I evaluated the proposal, thereby uncovering weaknesses in the proposal. The final contributions in this paper are directions for future research aimed at creating a framework for decentralized applications.

1.9 Organization

The rest of the document is built up in the following way: the next chapter describes the current, past and near future projects in the area of decentralization. Chapter 3 discusses related scientific articles and researches. The chapter includes recommendations done in previous research. Chapter 4 starts with describing the requirements for a link sharing system. The chapter follows up with discussing how a decentralized link sharing system would work, based on analyzed technologies. There is a qualitative evaluation on the goals of privacy, security and user control over data in chapter 5. Based on the performed research there is a discussion including lessons learned in chapter 6. Finally there are conclusions in chapter 7, followed by recommendations for future work in chapter 8s.

Chapter 2

Related Technologies

> If I have seen further, it is by
> standing on the shoulders of giants.
>
> Isaac Newton

Abstract

Over the years many applications and protocols have been designed and implemented in the field of decentralization. They have different goals, different origins and various levels of decentralization. The common technologies among decentralized applications are BitTorrent, distributed hash tables and blockchain technology. Blockchains are used for creating a shared state among nodes, for example to store usernames. Distributed hash tables are used as a decentralized key value store and BitTorrent is used for actively replicating data between nodes.

There are many projects aiming to provide privacy or scalability by using decentralization. Projects ranging from file sharing applications to network architectures, and social networks to domain name systems. For almost every goal there is a project trying to create a decentralized alternative. In this chapter the knowledge from these projects in the field of decentralization is employed to discover how a decentralized link sharing system can be built. To understand the projects, technologies which are commonly used are explained first.

2.1 Fundamental technologies

Decentralized systems often make use of already existing technologies. These technologies are often fundamental to understand why and how an application works the way it does. This section introduces these concepts which will be used throughout this chapter and the remaining part of this document.

2.1.1 Distributed Hash Table

A distributed hash table (DHT) is a structure which provides a dictionary-like interface for storing key-value pairs (items). It is comparable to a map in C++, a hashmap in Java and a dictionary in Python. Although there is one big difference, the responsibility for the keys is distributed across a network of nodes. And in order to facilitate an even spread of the responsibilities, the key is hashed (more about hashing in section 2.1.2.

Any node participating in the network can retrieve a value based on the associated key. The key is hashed and used to look up the associated value as in figure 2.1. A node is responsible for a subset of all the possible keys thereby distributing the *keyspace* over the participating nodes[3].

```
dht = DHT()   # assume DHT interface

key = "key_to_store_value"
value = "This one goes to 11."

# Store the value at the hash of the key
dht.put(hash(key), value)

# Retrieve the value from the hash of the key
stored_value = dht.get(hash(key))
assert(stored_value == value)
```

Figure 2.1: Storing and retrieving data from a DHT

In order to find the node responsible for a certain key a routing system is required. There are different routing solutions available but the routing from Kademlia is explained here [4]. When a node joins the DHT network it is responsible for the keys closest[1] to its id. To find the node responsible for a key it looks at all the other nodes it is connected with. The request for a key is sent to the node which is closest to the node responsible for the key. This node will either respond with the value belonging to the related key, or it will respond with the address information of the node which it thinks is closest to the requested key. These steps are repeated until the node responsible for the key is found, and has responded with the value.

A node does not know about all nodes in the network, it only knows about a subset of the nodes in the network. It stores the nodes in its *finger table*. The finger table gets updated according to some rules. The finger table should not get too big, nor should it be so small that it risks losing its connection with the network. Nodes which are active more recently, are more useful to store in the finger table. It makes more sense to store connections with nodes across the keyspace, than nodes which are all close (XOR distance) to each other.

Problems

Although DHTs are used at large scale in practice, for example in BitTorrent, they do have security[5] and privacy vulnerabilities[6]. An example of an attack on a DHT is a so-called Sybil attack. When a Sybil attack takes place, an adversary inserts a large amount of nodes into the network. The nodes will impair the lookup of keys by either not providing information to other nodes, or by forwarding requests to other malicious nodes[7].

Another problem with DHTs is that they often make it possible to link IP addresses with the files they are sharing. To overcome this problem onion routing can be used. Tor only supports TCP connections by default, and DHTs are often built using UDP due to their asynchronous

[1] Closeness in Kademlia is the XOR (binary operator $\oplus$) distance.

nature[8]. Therefore, to use onion routing, either TCP needs to be used in a DHT, or onion routing needs to be extended to support UDP.

2.1.2 Blockchain

The blockchain is a distributed database developed by Satoshi Nakamoto. The blockchain is the fundamental technology behind the digital peer-to-peer currency Bitcoin[9]. The blockchain is used as a world wide ledger, recording all transactions that happen with Bitcoin. When a transaction is made, it is sent to all nodes in the network after which it will be validated and incorporated into a block. When a new block is generated, it points to the previous block resulting in a chain of blocks, hence the name blockchain. Bitcoin is not owned by a company, nor is there a central entity issuing the currency. Everybody can join the network, and download a copy of the blockchain from other nodes[10]. Following is a description of a number of the concepts used in blockchain technology. For a complete description of Bitcoin and related concepts, I suggest the paper by Florian Tschorsch and Bjrn Scheuermann called "Bitcoin and Beyond: A Technical Survey on Decentralized Digital Currencies" [11].

Transactions

The blockchain does not record the amounts of Bitcoin available at addresses (comparable with an account number), it only records the transactions. After processing all transactions, one can say what amount of Bitcoin is available at an address.

```
from Crypto.PublicKey import RSA
from Crypto import Random
# generate a public and private key pair
random_generator = Random.new().read
private_key = RSA.generate(bits=4096, random_generator)
public_key = private_key.publickey()
# distribute the public key to the other party
...
# other party creates a message to sign
challenge = 'This one goes to 11.'
...
# create the signature for the challenge with the private key
signature = private_key.sign(challenge, '')
...
# other party verifies that the signature for the
# message is correct
assert(public_key.verify(challenge, signature))
```

Figure 2.2: Prove ownership of the private key with a challenge

A transaction describes the transfer of an amount of coins from one address to another. In order to transfer money, one has to prove one is owner of the address from where the amount

comes. To prove ownership of an address, the public and private key pair associated with the address are needed. Figure 2.2 shows an example of how one can prove to be in possession of the private key by providing a message, a signature and the public key. A Bitcoin address can be derived from the public key, but not the other way around.

Proof-of-Work

The transactions are collected by nodes who will try to create a block from the transactions. A block is generated by solving a cryptographic puzzle. The puzzle consists of finding a hash which meets some requirements.

A cryptographic hashing algorithm has the following properties:

Deterministic The hashing process is deterministic, this means that taking the hash of a value always results in the same hash. Demonstrated in figure 2.3.

One-way A hash cannot be practically reverted to the input. Any amount of data can be put in a hash function with no predictable hash as result. The resulting hash reveals no information about the input. Additionally, the hash of a value cannot be predicted faster than by computing the hash of the value. This means that the fastest way to find the hash of a value is by taking the hash of a value. Therefore, brute-forcing is resource intensive and unrealistic.

Continuity A single bit change in the value will result in a completely different hash[2]. Demonstrated in figure 2.3.

Uniformity The hash results should be uniformly distributed over all the possible outcomes.

```
>>> from hashlib import sha256
>>> value1 = 'This one goes to 11.'
>>> # Deterministic hashing
>>> sha256(value1) == sha256(value1)
True
>>> value2 = 'This one goes to 11'  # mind the missing dot
>>> # Continuity
>>> sha256(value1).hexdigest()
'eb9579995c6dbe53363c8b53b7b37e2472e69b88d0d0dd382c77b94f1c31780a'
>>> sha256(value2).hexdigest()
'ccffbc7be118aaaa10debf3c1e5d92d18f90b4c1fde8c9a0e714f542ad1a283e'
```

Figure 2.3: Properties of a hashing algorithm

The cryptographic puzzle in Bitcoin consists of finding a hash which has a leading number of zeroes. The leading number of zeroes are determined by the difficulty. The only way to find a hash which conforms to this difficulty, is by creating hashes of different inputs. A hash

[2]Two inputs resulting in the same hash are called a *hash collision*. There are no known hash collisions for the SHA-2 (including SHA-256 which is used by Bitcoin) and SHA-3 families. https://en.wikipedia.org/wiki/SHA-2

is generated from the list of transactions. When the list of transactions has been hashed the resulting hash will always be the same. Nonetheless, a hash with a number of leading zeroes needs to be found. Therefore there is a counter, called a *nonce*, which must be increased after every try in order to create a different input, which will result in a different hash.

```
>>> from hashlib import sha256
>>>
>>> nonce = -1
>>> result = ''
>>> value = 'This one goes to {0}.'
>>>
>>> while not result.startswith('0' * 3):
>>>     nonce += 1
>>>     result = sha256(value.format(nonce)).hexdigest()
>>>
>>> print(result)
'000f70983339fdaac2af4bd9729e67d0448ac2f312fb72d73f59e21379416d3c'
>>> print(value.format(nonce))
'This one goes to 4136.'
```

Figure 2.4: Finding a hash with a number of leading zeroes.

Figure 2.4 shows that finding a hash with a leading number of zeroes took 4136 iterations. It took a large quantity of resources to create the hash compared to the resources required to verify the hash is correct, which can be done in one step. The generation of the hash is called a *Proof-of-Work* (PoW). The difficulty for the Proof-of-Work is calibrated after every 2016 blocks, based on the amount of calculation power of the network. The number of hashes which can be performed every second is called the *hashrate*. The difficulty is adjusted so that, based on statistical chance of finding a correct hash, finding a correct hash takes about ten minutes (called block time).

Mining

The computers solving the cryptographic puzzle, and the people operating the computers are both referred to as miners. There are two reasons for a miner to generate a block. The first reason is that when the transactions are collected, a special transaction is inserted which creates a number of coins. These newly created coins will be transferred to an address the miner has access to, thereby earning Bitcoins for creating valid blocks. The number of created coins, or *reward*, is currently 25 coins, this reward halves every 210.000 blocks (roughly four years).

The reward is not the only income for the miner, the other source of income is transaction fees. With the diminishing rewards, the transaction fees should become the primary incentive for miners to continue mining. Generally, a miner only mines when the rewards for mining outweigh the costs for resources such as electricity.

Chaining the blocks

Each block contains transactions and, among other things, a block header. The block header consists of six parts:

Field	Purpose
Version	The version of the Bitcoin miner software
hashPrevBlock	A 256-bit hash of the previous block header
hashMerkleRoot	A 256-bit hash of all the transactions in a the block
Time	The current timestamp in Epoch time
Bits	The target hash needs to be below this difficulty
Nonce	The counter which is incremented for finding the right hash

Table 2.1: The Bitcoin block header.

As can be seen in table 2.1 the block header contains a reference to the previous block. The hash of the block header of the previous block is included in the current block. This means that, if a changed were to be made in the previous block, the hash of the block would be different and, consequently, the current block has to be changed as well. Moreover, since the hash required a leading number of zeroes, a new valid hash has to be generated, which has the right difficulty. The more blocks there are after a block (confirmations), the more blocks require to be updated or recalculated in order to get the right hashes. During this period, the rest of the network continues with adding blocks to the longest chain. This implies that, in order to change a block, more computing power is needed than the rest of the network combined. The same goes for the transactions, a hash of the transactions is in the block header. Therefore, if a change is made to a transaction all the blocks from the block with the change, all the way to the newest block, require to be updated in order to get it approved by the rest of the network.

The hashrate of the Bitcoin network is currently more than the top 500 supercomputers in the world combined[12]. Thereby making a change in the blockchain would require vast amounts of resources, making it infeasible to change the history of the blockchain. Never before in the history of mankind has there existed an unforgeable record of events which is open to the public to read from and append to.

Pseudonymity

Bitcoin is often called anonymous, but this is only partially true. Indeed there is no registration of names or account numbers. Although, when Bitcoins are bought for Euros, for example at an exchange, the exchange is required to ask for identity information[3]. From then on the identity can be linked to a Bitcoin address. And by following the transactions, one can deduce which addresses belong to whom. Therefore, it might be better to call Bitcoin pseudo-anonymous or *pseudonymous*. Some Bitcoin users contact other users whom are already registered at an exchange, so they can buy Bitcoin from them directly without registering their identity. This is one of the ways to stay anonymous when using Bitcoin. Although, as soon as a transaction links

[3]In the USA this is due to the Know Your Customer (KYC) and Anti-Money Laundering (AML) regulations. In the Netherlands there are the equivalent WWFT and WTT laws

a person to a Bitcoin address, the chain of transactions can be followed to link more addresses to a person.

Altcoins

All the previously discussed features of the blockchain can be tweaked and changed. As a result, such a blockchain might not be compatible with the Bitcoin blockchain. These other blockchains are called altcoins. Some blockchains have lower difficulties resulting in shorter block times, others use different hashing algorithms, and again others provide features for anonymization.

Problems

The blockchain Bitcoin is using, commenced at the beginning of 2009, and has been running for over six years. During these six years, the blockchain has been growing to a size of over 35GB[4]. The most rapid expansion being the most recent due to increasing popularity. If the blockchain continues growing at the current pace, it will become infeasible for normal users to run the Bitcoin software. There are suggestions for overcoming this problem. A technology called SPV (Simple Payment Verification) clients is already used. SPV clients only store the block headers and only request blocks which contain the transactions in which they are interested.

Although Bitcoin is running for six years, it is unclear what the future will bring. Will the transaction fees be enough to compensate for the diminishing rewards? Will the network be capable to grow to the size of international payment processors? Bitcoin can currently process a theoretical maximum of 7 transactions per second. This limit is due to the maximum size of a block (1MB) and the minimum size of a transaction[5]. For comparison, the VISA network processes about 2.000 transactions per second on average, and can handle up to 56.000 transactions per second.[6] However, Bitcoin and VISA are incommensurable due to two reasons. First, Bitcoin can expands its transaction rate without increasing the costs of running the network. Second, where VISA only processes transactions, Bitcoin is a decentralized network which can be used as the foundation for a complete financial infrastructure.[7]

2.2 Storage and File sharing

File sharing and storage can be divided in two groups, there are the centralized and the decentralized solutions. Examples of centralized solutions are Dropbox, Google Drive, Microsoft OneDrive and iCloud. They provide a web interface, an API, and desktop and mobile applications. They are used as cloud storage solutions which are accessible from everywhere around the world. Most of these services provide sharing options like sharing files or folders with other users, or making a publicly available link to share files with everybody. In all of these cases, either the website needs to be trusted with the data, or manual actions to encrypt the data have to

[4] Blockchain size - `https://blockchain.info/charts/blocks-size?timespan=all`

[5] A minimal transaction, containing one from-address and two destination addresses, is 220 bytes. $\frac{1MB/220bytes}{10*60second} \approx 7.5transactions/second$

[6] VISA handling 56k transactions per second.`http://usa.visa.com/about-visa/our-business/visa-transaction.jsp`

[7] `https://medium.com/@FavorableCarry/relax-about-bitcoin-electricity-use-1ca2446ae5ab`

be taken. Replicating files over multiple cloud storage providers might appear a good solution to overcome the problems of centralization. A downside to that solution is that multiple locations need to be synchronized. Several solutions to the privacy and availability problems have been proposed, a couple of them will be discussed here.

2.2.1 BitTorrent

The most widely used protocol for peer-to-peer file sharing. A torrent file is downloaded which includes a list of trackers. The tracker will maintain records of which node has which blocks of data, but the transfer of the block happens directly from one peer to the other[13]. This makes trackers a vulnerable single point of failure. Therefore DHT based trackers have been introduced. Only one initial connection to a node is required to join the DHT after which all data blocks can be found.

BitTorrent technology is used by Blizzard for distributing patches of games[8], by Linux distributions for the distribution of their software and by the Internet Archive for distributing their archives[9]. By using torrents the load of a server can be vastly reduced because all users of the torrent help distributing the load.

File sharing via BitTorrent has shown that a quarter of a million people can share a file[10] and that trackers are facilitating file sharing for millions of files and tens of millions of users[11].

In order to make nodes cooperate in the BitTorrent network, most clients implement a form of a tit-for-tat strategy[14]. A node will only exchange data with other nodes which willing to exchange data. With this strategy, a node which does not reserve any bandwidth for uploading, will get its bandwidth choked, resulting in lower or zero download speed. Therefore cooperation is better than gaming the system, which should work as an incentive to enable uploading.

Problems

As described earlier, a BitTorrent tracker is a single point of failure. When a torrent is only registered at one tracker and the tracker is attacked or shut down, the torrent cannot be downloaded anymore. There are two common solutions to this. The first solution is registering torrents at multiple trackers. The second solution is using a DHT, which is used as a decentralized tracker.

Although the distribution of files via BitTorrent is decentralized, the distribution of torrent files often is not. They are hosted on web servers which allow for easy access to and searching indexes for torrent files. These centralized torrent websites are often the point of attack for governments and copyright holders for enabling the distribution of illegal content. They are either raided and taken down or their domain names are revoked. As a response, The Pirate Bay proposed to create a decentralized browser where the indexes, which are normally hosted on web servers, are distributed using torrents as well, thereby eliminating all centralized components[12].

[8] How is it that Blizzard can distribute such large files to the public? http://us.blizzard.com/en-us/company/about/legal-faq.html

[9] https://archive.org/details/bittorrent

[10] https://torrentfreak.com/game-thrones-season-finale-sets-piracy-record-140616/

[11] https://torrentfreak.com/top-torrent-trackers-now-handle-up-to-56-million-peers-each-150531/

[12] http://www.techspot.com/news/55238-the-pirate-bay-to-introduce-decentralization-in-attempts-to-beat-censorship-for-good.html

Incorporated in BitTorrent is the before-mentioned tit-for-tat incentive system. However, when the transfer of files is finished, there is no reason to seed the torrent other than helping others to receive the files. Accordingly, less popular content has less availability and is harder to come by the older the torrent is. Private torrent websites and trackers overcome this problem by requiring an upload/download ratio, or by requiring a minimum amount of seed time. Suggestions have been made to incorporate micropayments into BitTorrent clients, thereby rewarding seeding and creating incentive for seeding a torrent. However, peers in BitTorrent networks are already monitored for copyright infringement. In some countries ISPs will shut down internet connections and copyright holders sue seeders for thousands of Euros. Adding micropayments, so earning money over copyrighted material, might not be the wisest option.

BitTorrent is anonymous in the sense that no account with personal data or email is required. It is not anonymous because any node can ask the DHT or tracker on which IP a block can be found. The exposure of IP addresses, especially in combination with knowing which content is available on that address, can be dangerous. Sharing of copyrighted material can lead to takedown notices and being cut off from the internet by ISPs.

2.2.2 Tribler

A client for BitTorrent created by the TU Delft which focusses specifically on the problems with BitTorrent (see 2.2.1). Where in a regular BitTorrent configuration, the torrent files are searched for in a centralized index website like The Pirate Bay, with Tribler the searching of files happens peer-to-peer as well. A node will search other nodes for files that they are sharing. Due to this decentralization it becomes impossible to take down[15].

Tribler hides the IP addresses of their clients by using onion routing. The onion routing network implemented by Tribler is different from Tor. Where Tor only allows TCP connections, Triblers onion routing is made for UDP connections.

Problems

One of the major problems with Tribler is the network speed. Where filetransfer in BitTorrent is limited by the up and download speeds of their users, Tribler is limited due to onion routing and encryption. Tribler reaches a maximum speed of 5 MByte/s with encryption disabled and 0.5 MByte/s with encryption enabled.

The goal of Tribler is to provide an anonymous direct video streaming service based on BitTorrent technology. A full-HD video with a length of 90 minutes takes up about 8GB of space. In order to stream such a video, the download speed should exceed 1.5 MByte/s[13]. The calculation of this number does not take into account the overhead caused by encrypting the data and onion routing[14].

Another self-recognized problem with the onion routing in Tribler is the possibility of Sybil attacks. This is a common problem in decentralized applictions. Any attacker able to insert a large amount of nodes into a network can form a threat to the security of the network. For

[13] $\frac{8000MByte}{90minutes*60seconds} = 1.5MByte/s$

[14] https://github.com/Tribler/tribler/wiki/Anonymous-Downloading-and-Streaming-specifications

Tribler this is a known problem, and they estimate that a developer can solve this issue in about a year of full-time work[15].

Users of Tribler have reported to receive take-down notices from copyright holders and some claim to have been raided by law enforcement agencies[16]. Tribler does not only connect with nodes within the Tribler network. It connects to nodes and trackers outside of the network as well. The option to stay within the Tribler network is off by default, which makes users vulnerable. Tribler is well-aware[17] of the problem and aims at solving the issue as soon as possible.

2.2.3 Storj.io

The Storj project takes a new approach in file storage. Files are cut into pieces, encrypted and stored over several nodes in the network. Only the owner will be able to open the files because the owner has the private keys. To make sure users participate in the network and not just throw away the files, there is an incentive for users for storing data called proof-of-storage. When a user can proof the availability and integrity of the data, the owner will reward the user for storing the files. This means that it will cost money to store data in the network. However, making disk space available in the network, money can be earned as well. Thereby making online storage free, as long as a user contributes as much to the network as he uses[16].

Problems

The project is in closed beta so it remains to be seen how the project will operate on a large scale in a worldwide network. One of the main questions people are asking is, when they participate, and allow their disk space to be used, whether people can use it for storing illegal content. Storj addresses this question by introducing something called graylists[18]. A graylist will be a list of hashes of content of which users would opt-out to be hosting. Such an opt-out list would be maintained by users and other users can choose the lists they wish to use. However, a hash based opt-out system could be easily gamed by changing a single bit in the original file, leading to a different hash as demonstrated in figure 2.3.

2.3 Network

The internet is a large network where packets of data are sent from one machine to the next. The Internet Protocol (IP) suite, or the protocols used for communication on the internet, specifies how messages should be routed, packetized, transmitted and addressed. What the IP suite is lacking, is encryption of the data transferred using these protocols. David P. Reed, "the designer of UDP", argued that end-to-end encryption should be part of the TCP layer, although this has never been implemented[19]. However, the UDP and TCP protocols are built into most hardware

[15] https://github.com/Tribler/tribler/wiki/Anonymous-Downloading-and-Streaming-specifications#known-weaknesses

[16] https://www.reddit.com/r/torrents/comments/30jlz0/raided_after_using_tribler/

[17] http://forum.tribler.org/viewtopic.php?t=7143&f=2

[18] http://storj.io/faq.html#faq-5-2

[19] David P. Reed about UDP http://www.reed.com/blog-dpr/?page_id=6

at the moment, making it practically impossible to update TCP to include encryption or make other changes. Other projects, such as Tor and I2P, try to add encryption and other measures to preserve privacy on top of the communication protocols.

2.3.1 Tor (The Onion Router)

Probably the most media covered piece of software of the last couple of years. Tor is free software allowing to anonymously browse the internet. Each request will be directed through a set of relay nodes. Each node peels off a layer of encryption of the request until the last nodes (exit node) performs the actual request (see figure 2.5). Within the Tor network there are custom DNS entries for the top level domain .onion. These .onion addresses are only reachable from within the Tor network and are not indexed by search engines like Google[17]. The websites in the Tor network are sometimes called the darknet or deep web.

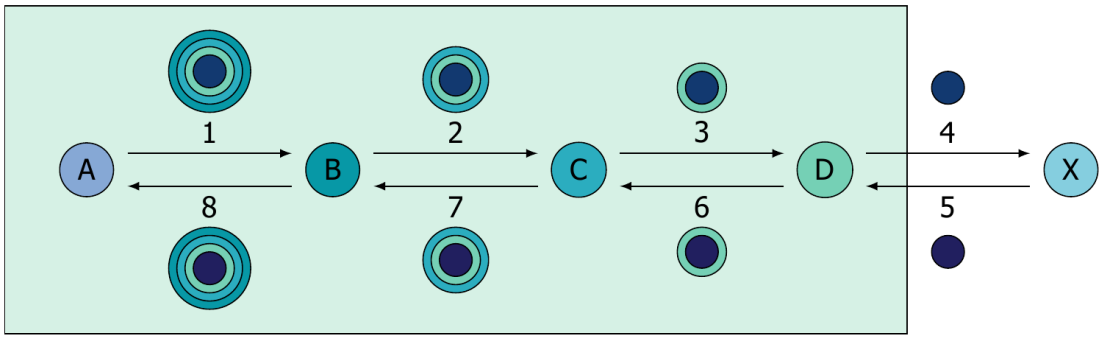

Figure 2.5: Onion routing: peeling layers

Although the operator of an exit node might able to eavesdrop, there is no known method to perform mass-surveillance on the complete Tor network. Or as the NSA puts it in one of their presentations: "Tor Stinks"[20].

In order for a node to establish a connection over the Tor network, relays have to be selected. These are the nodes peeling off layers of encryption as illustrated in figure 2.5. The list of relay nodes is provided by a group of so-called authorities or directory servers. These servers retain a list of all known relays and their public keys[21]. An overview of how directory servers and nodes in the network cooperate can be seen in figure 2.6

Problems

The before mentioned authorities might be assumed to be the Achilles heel of the Tor network. However, these authorities are not providing the lists of relays directly. The lists are served by caching servers. It would require most or all authorities to be compromised to be able to perform an attack to trace IP addresses of clients.

[20] http://www.theguardian.com/world/interactive/2013/oct/04/tor-stinks-nsa-presentation-document

[21] https://www.torproject.org/docs/faq.html.en#KeyManagement

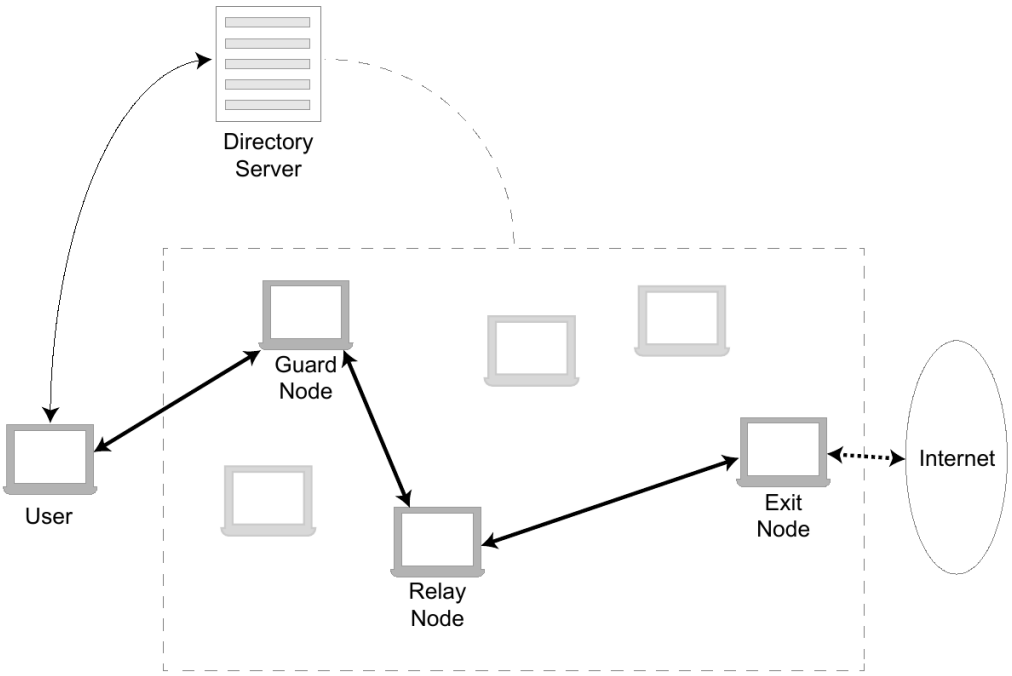

Figure 2.6: Onion routing using a Directory Server

The list of relay and exit nodes is accessible by everybody. This has led to some websites using the addresses used in Tor as a blacklist. Wikipedia for example disables editing of their pages when accessed from an address known to be used in the Tor network.

However, the use of Tor is not inherently safe. For example, browsers plugins and pdf readers might make a connection outside of the Tor network, which may result in revealing identifying information. Since exit nodes peel off the last layer of encryption, any request that is not encrypted be read and manipulated by the exit node. So in case login data or payment data is sent unencrypted via the Tor network, it might be intercepted. A Swedish researcher did steal data and published login data. Later he was arrested by the Swedish police[22]. Due to the anonymity features of Tor and the so called hidden services, Tor tends to attract users with ill intents. Which has led to an increase of illegal activity on the network.

Running an exit node in the Tor network is not without danger. Since the exit node is performing the actual request to the destination, for the destination it appears as if the exit node performed the request. Several owners of exit nodes have been raided or arrested by the police due to suspicious activity coming from their address.

Traffic over the Tor network is not and will not be fast. The connections over the network are routed around the world running on the computers of volunteers[23].

[22] http://archive.wired.com/politics/security/news/2007/09/embassy_hacks?currentPage=all
[23] https://www.torproject.org/docs/faq.html.en#WhySlow

2.3.2 I2P

An anonymizing network layer using a variant of the onion routing technology. While at first Tor and I2P might strike as the same, there are actually quite some differences[24]. Where Tor is using some centralized servers which allows them to efficiently address Sybil attacks, I2P aims at being fully decentralized and self organizing. Where Tor uses directory servers for announcing relay nodes, I2P uses a DHT based system called netDB[25]. Although DHTs suffer from all kinds of attacks, netDB tries to overcome the Sybil attack by making a small adjustment to the way they choose their keys.

Problems

A general problem with onion routing is the fact that, due to the encryption of all traffic and because network is maintained by volunteers, it is quite slow[18]. The people behind I2P have constructed a large list with possible attacks and their defenses against them.[26] Most, but not all, of these attacks are prevented already.

2.3.3 Cjdns/Hyperboria

Hyperboria is the name of a group of people connecting their computers and sharing their resources. The protocol used to connect the computers is called cjdns. The network uses IPv6 and adds end-to-end encryption for all communication. The routing protocol is build on a Kademlia-like DHT. To join the network a connection needs to be made to a friend who already is part of the network. By creating mesh networks and connecting them together they hope to cover more and more cities. Joining a mesh network would be free. An Internet Gateway Provider (IGP) somewhere in the network would provide access to the "rest" of the internet for a small fee. This way, use of a normal ISPs will be eliminated. It would be relatively easy to switch IGP in case it is too expensive, or when it is too slow. Content inside the network would be free to access, and by linking up cities an expanding network can be established[19].

Problems

The routing table in cjdns uses a DHT. Therefore, the vulnerabilities applicable to DHTs are applicable here as well. Other than that, cjdns uses a friend-to-friend system where, in order to join the network, somebody in the network needs to grant access. Cjdns has not gone mainstream yet and it is unclear whether the technology will be able to handle being the internet replacement it intents to be[20].

2.3.4 Open Libernet

The project Open Libernet proposes a new strategy for incentivising people to join, extend and maintain mesh networks. Users will be rewarded for each MB of data they successfully transfer. The credit can be used to make use of the network themselves. The credit can be donated

[24] https://geti2p.net/en/comparison/tor
[25] https://geti2p.net/en/docs/how/network-database
[26] https://geti2p.net/en/docs/how/threat-model

or sold. Therefore, it is beneficial to make resources, such as bandwidth and disk space, available to the network. It can even create incentive for upgrading hardware. Instead of paying a monthly fee to an ISP for arbitrary bandwidth and data limits, services will be on a pay-what-you-use basis. Hardware which is low quality, and cables which are unused will naturally go extinct because they do not result in income. The project appears to be mainly theoretical although their goals are clear [21].

2.4 Domain Name System (DNS)

Computers on the internet are reachable via IP addresses. These IP addresses are not easy to remember and might be changing from time to time. The DNS system translates domain names (like en.wikipedia.org) to IP addresses (for example 91.198.174.192) which is called a lookup. A look up is initiated at a computer and can go through the router to an ISP. In case the ISP cannot resolve the domain name, the request might be forwarded to another DNS server which can do the look up. Every server used for the lookup of a domain name and eavesdropper on the network, can register the IP address and the domain name of the lookup[27]. Therefore, the current DNS system is considered to be not privacy conserving.

2.4.1 KadNode

Rather than relying on a central authority, the KadNode project puts the DNS functionality in a DHT. All users can register domain names on the .p2p top level domain for free. Multiple IP addresses can be registered for the the same domain as well, meaning a domain name is not unique. To make sure the domain name resolves to the right IP address, a public-private key system is used. A user can register a domain name with a private key, and distribute the public key to end users. In case multiple IP addresses are registered on the same domain, only the IP address which can be verified with the public key would be used[22].

Problems

Resolving a domain name correctly in this system becomes a problem of attaining the right public key. In this solution globally unique names are not achieved. The layers in most applications which build on top of a DNS system assume names to be globally unique.

2.4.2 Namecoin

An alternative to (not just) DNS is Namecoin. It is a fully decentralized open source information registration and transfer system. Namecoin allows for registration of for example domain names, usernames and identification (Onename) and uses a blockchain. At the moment Namecoin allows for registration of unique domain names ending with .bit. A node in the Namecoin network can look in their local database and translate the .bit address to an IP address. This way no lookup has to be done on other servers, all that is needed is a Namecoin node. A transaction can change the IP address for a domain name or transfer the domain name to another user [23].

[27] DNSSEC can be used to make sure the request and response are not altered along the route. However, it does not provide encryption.

There are multiple specifications for providing name and identity information. Projects can introduce their own specification as well, in case the currently available ones are not suitable[28].

One of the most often mentioned problems in Namecoin is the ability for so called name-squatting. Although it does cost money to register a name in Namecoin, occupying a large number of (interesting) names is possible. However, Namecoin has a solution for that problem. In the current name registration system, name-squatters can be ordered to, or forced to give up the domain name. In Namecoin, a domain name can only be transferred with the private key that was used to register the domain name. This leads to a situation in which names can be unusable forever. This is opposite of the current system where there is always a company with the ability to correct unjust use. In Namecoin, the problem of squatting is solved by making the registration of a domain name only valid for a certain amount of time. After this period, the name requires a renewal to stay valid. At the moment renewal is free, but as soon as there is a fee for renewal it can become expensive and less interesting the squat domain names[29].

Problems

Namecoin requires the running of a full node with a complete copy of the blockchain. Such a system will not be feasible for every situation, for example routers and phones might not be suited for a full blockchain node.

2.4.3 okTurtles

Where the Namecoin project required a complete copy of the blockchain locally, the okTurtles project aims at being more user friendly. They are creating a browser-plugin which will query a server which is a gateway to the Namecoin blockchain. A browser-plugin is the only thing needed to make DNS secure and private. They plan on integrating with the identity features of Namecoin (and Onename) as well. When an input-field on a website is detected, for example on a Twitter or Facebook page, okTurtles will try to discover whether or not this person has an identity on Namecoin. If so, then the messages can be sent encrypted to this person, allowing for easy, private and secure communication via untrusted channels[24].

Problems

By using a gateway instead of integrating a Namecoin node, the gateways become interesting targets for attacks. It is a clear trade-off between easy integration and decentralization.

2.5 Identity

Websites and applications often require an account. Access to these accounts is usually obtained with a username or an email address and a password, often called credentials. Credentials often differ across services. On one hand this is due to different requirements for the credentials, and on the other due it being much safer having different credentials. Different credentials means that, in case they are stolen, one set of credentials does not give access to all services at

[28] https://wiki.namecoin.org/index.php?title=Category:NEP
[29] https://wiki.namecoin.org/index.php?title=Squatting

once. Remembering these credentials is practically impossible. Credentials are written down, stored by the browser or stored in a password management systems. Other initiatives propose a system in which one account is used to log in everywhere. It is common practice to use Facebook's, Twitter's or Google's accounts for other services. However, this gives these companies power over not just on their own services, but over the services that integrate with them as well. Several solutions have been proposed to remove these centralized identity services by giving decentralized alternatives.

2.5.1 OpenID

The OpenID Foundation has developed an open standard and decentralized protocol which allows users to authenticate on websites which support the protocol. A user can register an account on any of the OpenID providers and use this account to log in[25].

Problems

There are many problems with OpenID, ranging from security problems which allow for phishing to usability problems.[30] The most interesting problem for this research, is the trust problem. An OpenID provider has to be trusted. Everyone can become an OpenID provider and issue valid identities. There is no way a website accepting an identity from OpenID can verify the validity of the identity. Another big problem is the fact that an OpenID provider can track on which websites the identity is used.

2.5.2 Keybase

On Keybase, an account consists of a public key and profiles from other websites which are linked. A message needs to be signed with a private key and posted on the public profile. The websites which are supported by Keybase include Twitter, Github and Reddit[31]. Changes in an account will be registered as transactions. The root of a Merkle Tree of these transactions will be stored in the Bitcoin blockchain, so every user can verify the integrity of the data stored by Keybase [26].

Problems

The accounts created at Keybase are not stored in the blockchain. Just a hash is stored in the blockchain. It is not possible to retrieve account information from the blockchain, it is merely possible to verify that the data received from Keybase is valid by verifying it with the hash from the blockchain. This makes the Keybase server a central point of failure.

2.5.3 Onename

The idea behind Onename is like Keybase, except that Onename stores all data in a blockchain. In this case not the Bitcoin blockchain, but the Namecoin blockchain. Running a node in the Namecoin network means that account data is available on that node. The accounts stored

[30] `http://www.untrusted.ca/cache/openid.html`
[31] `https://keybase.io/docs/server_security`

in the blockchain can be used as accounts for any service. The result is a fully decentralized approach to an online identity [27].

Problems

One of the often stated problems with Onename is name squatting, the practice of taking as many popular usernames as possible. However, as discussed in section 2.4.2, this problem is already solved in the Namecoin blockchain underlying Onename.

Most applications using Onename use a centralized gateway to access the contents of the blockchain. This is because the use of the Namecoin blockchain is not a widely accepted. And instead of forcing users to download the blockchain, a gateway is used. The use of a gateway is more convenient, but it goes at the cost of decentralization.

2.6 Communication

Communication between applications requires an established protocol. What should a message look like? How are messages addressed? How do messages reach a destination? The protocol used in WhatsApp determines that a message should always be sent to their servers. From there on the message will be sent to the destination. Most communication protocols works this way: send the message to a central server, then relay it to the destination. Decentralized communication protocols remove the need of a central server, thereby making it harder to eavesdrop.

2.6.1 XMPP

Originally called Jabber, XMPP is an open standard managed by the Internet Engineering Task Force (IETF). It is a decentralized messaging protocol and has been used by, among others, Google Talk. Facebook Chat and Microsoft Message Service both support XMPP, although their implementation features only an XMPP interface, and not a full server[28].

Problems

Although XMPP is decentralized, it does use a client-server model for communication. Clients connect via a central server and all communication goes through the server as well. Clients do not connect directly with each other. XMPP is decentralized because it allows anyone to set up a server. However, anyone running a server can see the metadata. So, whoever runs an XMPP server can see who are communicating and when. However, the conversation itself is forward encrypted with TLS, indecipherable even if someone obtains the keys after the fact. XMPP has deniable authentication, meaning that authenticity is guaranteed during while communicating. However, the authenticity cannot be proven to a third party after the fact.

2.6.2 Off-the-Record Messaging (OTR)

The OTR protocol can be used over existing messaging protocols like XMPP. It will make sure no eavesdropper can perform a man-in-the-middle attack by using the Diffy-Hellman key ex-

change algorithm. The algorithm is used in many privacy-first chat applications and has deniable authentication [29].

2.6.3 Telehash

Used by IBM in their ADEPT proof-of-concept for the future of Internet of Things (more about IoT in section 3.6). Telehash is the brainchild from the creator of XMPP, Jeremie Miller. The goal is to make applications talk privately, in real-time and completely distributed. The routing of messages is inspired by Kademlia. The third version of Telehash is currently under development and matures while being integrated in projects[32]. Telehash strives for end-to-end encryption, for all communication, all the time, without revealing contents, identities or metadata [30].

2.6.4 WebRTC

Developed by the World Wide Web Consortium (W3C), WebRTC is a protocol for *browsers* to perform real-time peer-to-peer communication. It is developed for performing audio, video and data communication without the need of plugins or third party software. The browsers Google Chrome, Mozilla Firefox and Opera support the protocol. Examples of projects demonstrating the powerful potential of WebRTC are peer.to, talky.io and sharefest.me[31].

Problems

WebRTC is not integrated in the browser Internet Explorer and Safari by default. There are plugins available although this clashes with the philosophy of allowing real-time communication without external plugins.

2.6.5 Tox

A messaging protocol as well as the name of the application. Tox is an open source, end-to-end encrypted messaging application for multiple platform including mobile platforms. It supports texts, images, voice chats and group chats. It uses a DHT as routing mechanism to find where users can be reached. Connections to relay messages are establishes from peer to peer directly. In order to prevent tracking of users across multiple IP addresses, onion routing has been implemented[33]. Tox implements an onion routing algorithm similar to Off-the-Record Messaging (OTR). The DHT is not used to store messages, it is only used for routing, thereby eliminating the need for a central server [32].

2.6.6 Bitmessage

The goal of Bitmessage is to give a secure, encrypted alternative to email. A Bitmessage user is reachable via a 36-character random string, which is their public key as well. When a message is sent it will be broadcasts to all peers, just as with the flooding of transactions in Bitcoin. The

[32] http://quartzjer.tumblr.com/post/109339320204/lockchain-fun

[33] https://github.com/irungentoo/toxcore/blob/522f90fee138087db660dccc08413c53f388f604/docs/Prevent_Tracking.txt

only way to know whether a message has certain destination, is by trying to decrypt it with the private key. This requires that every node in the network processes every message. Messages will be stored by nodes for two days. When a nodes joins the network, all messages from the past two days are retrieved to find out whether a message is for the node. When a node receives a message, it can send an acknowledgement. An acknowledgement is treated like a normal message, and broadcast to all peers. To prevent spam, it is required for each message to have a proof-of-work, which will take about four minutes on an average computer[33].

The flooding of messages to all nodes appears to be a non-scalable solution. When the number of nodes becomes big, each node will be processing messages all the time. To prevent this from happening, a mechanism has been designed which puts nodes in ”streams”. Each stream is responsible for a subset of all messages. When a message needs to be delivered to another stream, a connection is made to nodes in the other stream.

Problems

Because every node needs to process all messages, and because a proof-of-work needs to be calculated, a full client on a phone might not be ideal for battery and bandwidth usage. Gateways have been developed to facilitate mobile usage and usage at places where no application can be installed. However, this will introduce security and privacy problems in the same fashion as with centralized applications.

2.7 Decentralized web browsing

Websites are generally a centralized environment. A server hosts a website and a client requests the website from a server. The more clients a server processes, the larger the resources on the server have to be. But what if clients could act as server as well? They already have the website, so they might serve it to other clients as well. Several projects already created the technology for the distribution of websites in a peer-to-peer manner.

2.7.1 BitTorrent Maelstrom

The creators of BitTorrent only recently released their browser called Maelstrom. It is based on chromium and currently it is closed source. A torrent file and magnet link are created from a website and will include the selected files. When another user opens the magnet link in the Maelstrom browser the files will be transferred and the website will be displayed. There is support for downloading with the help of trackers, but trackless (DHT) downloading is supported as well [34].

Problems

There is no support for user generated content nor the possibility to verify whether the website is created by a certain person. Moreover, there is no means to update websites other than to create a new torrent.

2.7.2 IPFS

It is a new peer-to-peer hypermedia protocol as they call it. With a DHT as a basis, a client does a look up of a hash which will return the location of the website. The website changes are propagated as changes or diffs with the help of git instead of distributing full new websites [35]. IPFS has a built-in incentive system. In order to participate in the network, files need to be seeded. IPFS can run in the background while performing all the hard work. There is a web interface which allows users to interact with the application.

Nodes which are to enter the network have to generate an identifier. The generation of an identifier is like solving a cryptographic puzzle such as a Proof-of-Work. This puzzle makes it expensive to generate large numbers of identifiers which should prevent Sybil attacks.

The messages sent by IPFS can be transported over any transport protocol in a reliable manner. The integrity and authenticity of the messages can be verified using hash checksums and public keys.

IPFS uses a block exchange called BitSwap which is based on BitTorrent. In BitTorrent nodes announce which blocks they want and which blocks they have. However, these blocks are limited to the torrent a node is interacting with. With BitSwap the blocks can be acquired regardless of the files they belong to. The BitSwap protocol uses a credit-like system which makes sure peers cannot only leech but will have to seed as well to participate in the system. The exchange protocol can be replaced with different strategies which will have varying effects on the network.

The DHT of IPFS stores information in two ways. When a piece of information is small ($\leqslant 1$ kb) it is stored directly in the DHT, otherwise the DHT will store the location of the nodes where the files can be found.

2.7.3 ZeroNet

Decentralized websites using Bitcoin crypto and the BitTorrent network [36]. As with IPFS, websites can be visited by doing a lookup of a hash. However, with ZeroNet the hash is looked up in a torrent tracker instead of a DHT. The torrent tracker will return a list of peers currently seeding the website. When connecting to the peers the website can be requested based on the hash. This means that it is quite easy to obtain a list of IP addresses of users seeding a certain website. ZeroNet supports the use of Tor which can hide IP addresses of users.

For each website, a file is generated which contains the names and signatures of all the files of the website. This way, the integrity for each file and the complete website can be verified using the public key of a website. Once a website is received, it will be seeded for other users to retrieve it. However, there is no incentive mechanism incorporated in the project. Which means there is no reason continue seeding a website.

ZeroNet has support for dynamic content, meaning there is way for users of website to, for example, post messages. The owner of a website creates a separate file for each user and makes the user owner of that specific file. These files are only changeable by their owner, the user. When a user posts a message, the message will be added to the specific file of the user on the local machine. When the message is added, the file will be synchronized to all other users seeding the website.

Remembering hashes to visit a website is not a human-friendly way of browsing. Therefore

ZeroNet has recently incorporated Namecoin. There is not a Namecoin client included in ZeroNet, instead a lookup of a url is done on an API which will return the hash which identifies the website.

Problems

Although aiming at a fully-decentralized solution, some parts can be more decentralized but at the cost of ease of use[34]. ZeroNet is using a tracker instead of a DHT, the decentralized alternative. Moreover, the integration with Namecoin is currently with the use of a gateway instead of a full client. If Namecoin releases a lightweight client, ZeroNet will incorporate it in the project. Using Namecoin as an identity service is considered as well.

The method for handling dynamic content is not a scalable solution. Imagine a forum with thousands of users. Every post, comment or vote a user makes results in updates to the website for all users who are seeding the website. A better scalable solution would distribute dynamic content more finely grained.

2.8 Full stack solutions

The before-mentioned solutions are either protocols or solutions incorporating a couple of features. On the other hand, there are projects setting up a full stack of technologies in order to facilitate a decentralized internet.

2.8.1 Freenet

With the goal of providing freedom of speech on the internet, Ioan Clarke designed a peer-to-peer platform which is resistant to censorship of communication [37]. The platform, for which the development initiated around 2000, harbors many applications like websites, forums, email, chat and blogging.

The platform is a file sharing network which allows files to be stored on and retrieved from the network. When a file is inserted into the network the file is split in parts and encrypted. These parts are spread over multiple nodes to make sure when a single node goes offline the file is still available. There is no way to remove a file from the network. Each node caches data, and when new data is coming in it will ”forget” the data which is the has not been requested for the longest time.

The nodes in the network know only a subset of the nodes of the network. When a message is routed from the source to the destination there will be a number of nodes in between routing the messages. None of these nodes in between know about the source or destination of the message, they only know where to forward the message to. The technology is similar to onion routing.

To create incentive for users to share bandwidth, the number of nodes connected to a node is proportional to the bandwidth of the node. Because of the higher degree of connectivity,

[34] https://www.reddit.com/r/zeronet/comments/376w40/new_feature_trusted_authorization_providers/crk69q0

a larger group of nodes can be reached in a shorter timespan. Therefore, a high-degree node experiences a better performance.

With 60,000 nodes discovered over a period of 8 weeks, Freenet is one of the most successful decentralized censorship resistant communication systems. However, due to the chosen routing algorithm and the combination of Opennet and Darknet (friend-to-friend) the performance of the network has a large margin for improvement.

Problems

The Freenet network suffers from unsatisfactory performance due to the current routing algorithms. It can take relatively long for websites to load for the first time due to inefficient routing [38].

Users of Freenet do not have control over the data that is stored on their machine when they become part of the network. Although it is very hard or even impossible to detect which data a node is storing, it might not be pleasant for everybody to not know what their computer is serving to the network. Especially since Freenet is often associated with condemnable or illegal content[35].

Using Freenet in open-mode (Opennet) provides only limited anonymity and is relatively easy to block. Therefore Freenet is actively encouraging users to use the friend-to-friend mode. However, friend-to-friend mode requires convincing friends to join the network and this is not that easy with experimental projects such as Freenet.

2.8.2 GNUnet

The GNUnet application is currently mainly used for secure file sharing. The framework can communicate over various transport networks and is fully decentralized. It uses a Kademlia-like DHT for its name system, overlay network and file sharing [39]. The application has its own application interface and it not accessible with a browser. GNUnet features a name system called the GNUnet Name System (GNS). With GNS each user manages his own mappings and is, consequently, not reliant on a central authority. Users can use other users' mapping as well, by using a DHT to store and look up mappings. However, a major drawback with this solution is that names are not unique in the system.

2.8.3 Ethereum

The yet-to-be-released project for decentralizing the web. The aim is to develop a decentralized platform which can be used by Dapps (decentralized applications). The project consists of a blockchain, a communication protocol called Whisper, decentralized storage called Swarm[36] and a browser, capable of interacting with the other parts, called Mist. The blockchain, which will be used for their coin called Ether, will support *smart contracts* which can be written in different programming languages [40].

[35] `https://freenetproject.org/faq.html#offensive`
[36] `https://github.com/ethereum/cpp-ethereum/wiki/Swarm`

Smart contracts

A transaction recorded on the blockchain contains a small piece of code. This piece of code transfers a certain amount of value from one address to another. However, there are more possibilities for pieces of code in a transaction. Bitcoin for example supports m-out-of-n payments. Which means that at least M out of N people have to sign a transaction for the transaction to be valid. These scripts are like small contracts describing what should happen when the premises of the contract have been fulfilled. More advanced contracts are called *smart contracts* [41]. Examples of smart contracts could include a lottery where, after 100 payments, all payments will be directed to one of the participants. Or a door which can be unlocked with a phone after payment to the smart contract of the door.

Whisper

Whisper is not a DHT and it is not just peer-to-peer connections. It will be used for communication in the form of small messages between Dapps. The aim is to make the communication resistant to traffic analysis attacks. Nodes can listen and advertise for messages which match a certain topic. The size of the part which has to match can be configured per message which is equal to deciding how much is revealed about the topic one is looking for. There will be a direct tradeoff between latency and the disclosure of information[37].

Mist

The Mist browser will incorporate all parts of Ethereum and display the Dapps. However, Dapps can be used in combination with normal browsers as well. A javascript library is available which is an interface to Ethereum and all the parts like Whisper and Swarm to be able to use the Dapps in other browsers.

Problems

The Bitcoin network does not support smart contracts the way that Ethereum does. Therefore, Ethereum cannot use the Bitcoin blockchain and as a result, the project will create their own blockchain. However, initiating a new blockchain comes with many challenges. For one, the security of the blockchain is partially determined by the amount of computation power in the network. Therefore, a new blockchain should gain traction quickly to surpass the initial *bootstrapping* phase as soon as possible.

2.8.4 MaidSafe

For almost ten years is MaidSafe under development. It has the goal of completely decentralizing the internet [42]. Decentralized storage, decentralized communication and a coin called Safecoin (not blockchain based) [43] are all part of the project. There is no planned release date although there is roadmap towards a public beta. The project will use an altered form of Kademlia [44]. Storage in the network will be secured by a system of vaults [45]. These vaults make sure data is replicated a number of times across the network in order maintain the availability

[37] `https://github.com/ethereum/wiki/wiki/Whisper`

of files on the network. The vault system can be compared to the Storj project, except the vault system will not use a blockchain.

2.9 Social networks

The majority of decentralized online social networks (DOSN) are theoretical or do not have a significant user base. The major player in the field of DOSNs is Diaspora. However, Twister is catching up fast in the field of microblogging. Two other alternatives, Aether and Synereo, are both worth mentioning and will be described in the following sections.

2.9.1 Diaspora*

A privacy-aware distributed open source social network with comparable features to Facebook. An important requirement for the project was to be able to import posts from other social networks to lower the barrier of entry. Users select a pod to host their data but are able to interact with all users in the network. With about 70.000 active users spread out over 100 pods in the last 6 months[38] it can be said that the number of users has significantly shrunk [46].

Problems

It is not required to install Diaspora to join the network, one can go to one of the hundred so-called pods to sign up and login. This results in the top 4 pods providing services to 2/3 of all the Diaspora users which directly clashes with the goals of the project to address privacy issues related to centralized social networks.

Any user can host a pod, which can lead to problematic situations. Where in the case of a fully centralized social network a company needs to be trusted with the data, in de decentralized case either an individual has to be trusted. Another option is to host a pod, but this is not for the faint-hearted.

2.9.2 Twister

As a reaction to the centralized forces of Twitter developer Miguel Freitas created a fully decentralized microblogging platform which he called Twister. The network has the same features as Twitter with followers, hashtags and a 140 character limit for posts but it does not allow cross-posting due to the terms-of-service of Twitter [47]. Twister can run on Windows, Linux and Mac and uses a normal web browser as the interface.

The application contains a blockchain client for a Twister-only blockchain which is used for the registration of users. Posts, the avatar and profile information of users are stored in a Kademlia-based DHT. However, because a DHT cannot push messages, this would result in Twister clients polling all followers to determine whether or not a user has a new post. This would result in a non-scalable situation where clients send many requests to update their posts. Therefore when a user follows another user, the client joins a torrent swarm[39] which results in live updates when a user posts a new message. Posts and private messages are signed and can

[38] `http://podupti.me/`

[39] A swarm is a network of nodes sharing the same file. This includes all seeders and leechers.

be verified with the public key registered in the blockchain. A user can stay anonymous in the sense that no personal information is required for signing up. The registration in the blockchain makes sure that other users are able to verify that a message comes from a certain user [48].

Problems

While peers are connected with each other and exposure of IP addresses may be a problem, users are advised to use Twister in combination with Tor. However, Twister uses a DHT which uses UDP and UDP is not supported for Tor. On the other hand, the peer-to-peer connections from the blockchain are over TCP and can be routed over the Tor network. To make Twister compatible with Tor, there is an option to route all DHT requests over the blockchain peer-to-peer network where nodes which do support UDP are used as gateways[40]. Although this does make Twister compatible with Tor, the network would stop functioning if all nodes were to operate over Tor because there would be no gateways available any more. A second solution would be to create a custom onion routing protocol which can be used over UDP (as in Tribler and Tox).

2.9.3 Aether

An alternative to Reddit is the decentralized, open-source and anonymous Aether [49]. It is a standalone application which connects with other nodes and synchronizes topics, posts and messages. The application features a well designed and good looking interface.

Problems

There is no spam-prevention, uniqueness in usernames and no encryption or privacy protection. Moreover, Aether's scalability is limited. All messages are synchronized to all clients, without the possibility to limit the use of resources[41].

2.9.4 Synereo

Our attention is worth something according to Synereo. They plan to create a decentralized social network which will be centred around an *attention economy*. Posts will be propagated through the network based on the fuel of the post. The fuel of the post depends on the popularity of the user and the amount of likes, comments and shares of a post and maybe even the amount of time looked at a post. The value in Synereo is expressed in AMPS, the cryptocurrency designed for Synereo [50]. The hoster of the application can earn money, thereby creating incentive for users to set up the application on their own machine. Synereo will be accessible with normal web browsers which will make it easy to use with a phone when Synereo is configured as a gateway[42]. The project has recently done a crowd funding campaign, and the team is currently working on the first working product to be tested in a closed environment.

[40] http://twister.net.co/?p=363
[41] http://getaether.net/how.html
[42] https://www.reddit.com/r/Synereo/comments/339u2f/ama_with_synereo_founders_ask_us_anything_live_now/

2.10 Other notable applications

From the previously described projects it may have become clear that it is not already straightforward to categorize a project. IPFS, for example, is categorized as a project for decentralized web browsing, although it might be classified as a file sharing application as well. These last couple of projects could not be easily fit into any of the categories, but do deserve to be mentioned.

2.10.1 GitTorrent

Although the name might suggest otherwise, distributed version control systems (DVCS) such as Git are not that decentralized in practice. A large portion of the repositories is concentrated at github.com which makes them vulnerable for attacks like the recent DDoS attack from China.[43] A clone of a repository on a machine does not mean the repository is available for other users because most users do not run a git server on their machine[44].

The only truly decentralized version control system appeared at the end of May 2015. It is called GitTorrent and as the name might have given away, it uses BitTorrent. Repositories are identified by hashes which can be looked up via BitTorrent in order to request the files from the repository. An extension of Git has been made in order to facilitate the functionality over BitTorrent, as it is not supported by default. The notoriously hard to remember hashes are stored in a blockchain which makes it possible to translate user and repository names to hashes which allows for the retrieval of the repository.

2.10.2 OpenBazaar

There are a number of decentralized marketplaces being developed, one of them is OpenBazaar [51]. It uses BitMessage (see section 2.6.6) for communication, Bitcoin for pseudonymous payments and a DHT for establishing a network and storing data. OpenBazaar does not require the installation of a local Bitcoin client, it uses a hosted Bitcoin client. The application is accessible via a browser after installation. OpenBazaar takes out the middlemen in both setting up shops, and payments as can be seen in figure 2.7.

Problems

OpenBazaar does not support onion routing yet because not all connections are made with TCP[45]. However, a solution involving custom onion routing over UDP can be implemented as well.

[43] http://arstechnica.com/security/2015/04/meet-great-cannon-the-man-in-the-middle-weapon-china-used-on-github/

[44] http://blog.printf.net/articles/2015/05/29/announcing-gittorrent-a-decentralized-github/

[45] https://github.com/OpenBazaar/OpenBazaar/issues/866

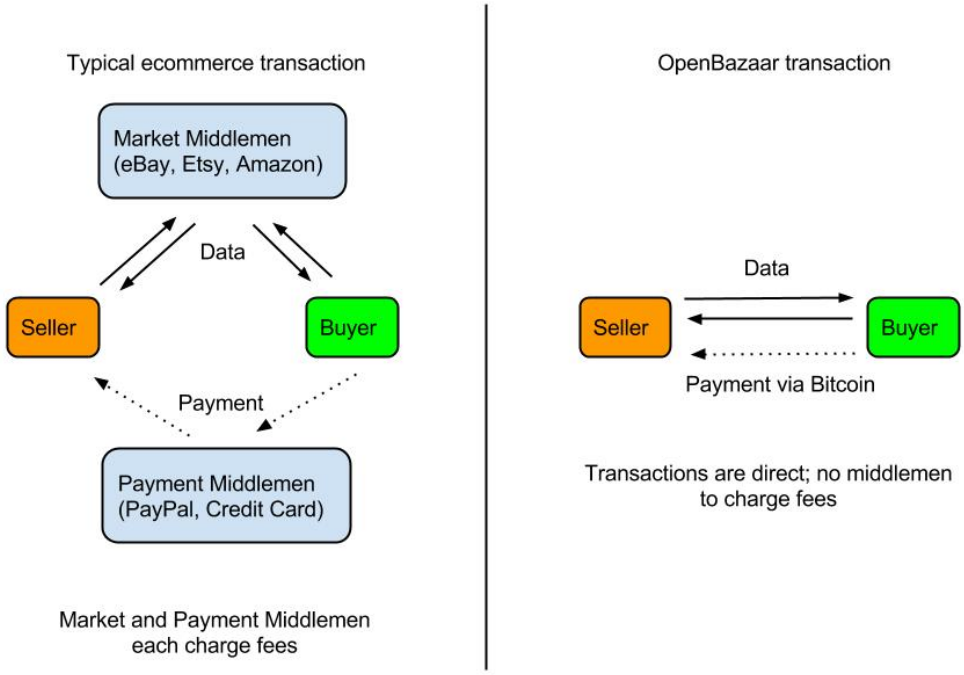

Figure 2.7: OpenBazaar: taking out the middlemen

2.10.3 2gather

The company Eris Industries created a video sharing decentralized application based on their blockchain and IPFS[46] (section 2.7.2). IPFS is used to share the videos via the content-addressable decentralized file sharing mechanism. An AngularJS application, which works in any browser, communicates with a back-end which provides the interaction with the distributed file store and the smart-contract enable blockchain[47]. The blockchain, which is a fork of Ethereum, is under complete control of Eris Industries which allows them to reset the blockchain if they wish to do so.

2.10.4 WebTorrent

Downloading files via torrents happens with clients like uTorrent, qBittorrent, Transmission or Vuze. These applications require installation of software. And according to the developer of WebTorrent, they do not look user friendly with all their statistics, windows and configuration panels. He came up with a solution to create a full torrent client in javascript which can

[46] https://eng.erisindustries.com/tutorials/2015/04/07/2gather/
[47] https://github.com/eris-ltd/2gather

run in a web browser. The idea is to not just create a torrent client, but create a YouTube-like video streaming service, which is fully decentralized and will receive the video using torrents in the browser. While browsers cannot create TCP and UDP connections directly (only via supported protocols such as HTTP and HTTPS), all connections will be performed over WebRTC. Browsers can currently only connect to other browsers with WebRTC. There are plans to create a hybrid torrent client which could support normal and WebRTC connections. If other torrent clients would support WebRTC connections, the border between browser and non-browser clients could fade away. Then, the power of torrents can be used in browsers for distributing any content from peer to peer.

2.11 Overview

Table 2.8 is an overview of the discussed technologies.

Project	Blockchain	DHT	State	Note
Storage and File sharing				
BitTorrent		✓	Operational for 14+ years	Decentralized file sharing. Can also be used without DHT.
Tribler		✓	Operational for 6+ years	Fully decentralized BitTorrent file sharing including search.
Storj.io	✓	✓	Beta	BitTorrent based decentralized file storage. Has incentive for sharing resources.
Network				
Tor			Operational for 12+ years	Anonymity network. Uses centralized directory servers as routing mechanism.
I2P		✓	Operational for 12+ years	Anonymity network. Uses a DHT to replace directory servers.
Cjdns/Hyperboria		✓	Operational	Anonymity network. Uses a DHT as routing mechanism. Requires a friend in the network to join.
Open Libernet	✓		Conceptual	Meshnetworking using cryptocurrency for micropayments.
Domain Name Systems				
KadNode		✓	Operational	Domain names are not unique and verified using public keys.
Namecoin	✓		Operational for 4+ years	Uses a blockchain to register unique names.
okTurtles	✓		In development	Uses blockchain based name systems to use unique names.
Identity				
OpenID			Operational for 8+ years	Federated identity system that relies on co-operating websites.
Keybase	(✓)		Operational for more than 1 year	Allows for correctness verification of identity information using a blockchain.
Onename	✓		Operational	Username and identity protocol stored as a new namespace in the Namecoin blockchain.
Communication				
XMPP			Exists for 15+ years	A communication protocol with architecture similar to email.
OTR			Exists for 10+ years	Encryption for instant messaging.
Telehash		(✓)	V2 exists for 2 years	End-to-end encrypted mesh networking protocol. Supports routing using a DHT.
WebRTC			Exists for 4+ years	Allows for browsers-to-browser communication. Not supported by all browsers.
Tox		✓	Operational for 2+ years	Instant messaging using a DHT as routing mechanism.
BitMessage			Operational for 2+ years	E-mail-like encrypted peer-to-peer communication.
Decentralized web browsing				
Maelstrom		(✓)	Operational <1 year	Chromium fork supporting websites as torrents. Can use a DHT as routing mechanism.
IPFS		✓	Operational <1 year	BitTorrent-like file sharing using a DHT as routing mechanism.
ZeroNet	(✓)	(✓)	Operational <1 year	BitTorrent based websites, supports Namecoin for DNS and Identities. Uses trackers but DHT support is planned.

Project	Blockchain	DHT	State	Note
Full stack solutions				
Freenet			Operational for 15+ years	Platform for censorship-resistant communication. Nodes will "forget" data when it is not accessed. There is no incentive to store data.
GNUnet		✓	Operational for 13+ years	Decentralized peer-to-peer networking mainly used for file-sharing.
Ethereum	✓		In development	Smart-contract enabled blockchain. Project includes decentralized file-sharing, traffic-analysis resistant communication and a web browser.
MaidSafe		✓	In development	Decentralized internet platform including storage and payment system. Uses a DHT for routing.
Social networks				
Diaspora			Operational for 4+ years	Federated social network of connected servers.
Twister	✓	✓	Operational for >1 year	Microblogging application. Uses a DHT for routing, uses BitTorrent for data-sharing. Registers usernames in blockchain.
Aether			Operational for >1 year	Link sharing application. Synchronizes all content, usernames are not unique. No spam prevention.
Synereo	✓		In development	Social network operating as neural network. Creates an economy for attention.
Other notable applications				
GitTorrent	✓	✓	Operational for <1 year.	Version control system sharing repositories using BitTorrent. Uses a DHT for routing and a blockchain for the registration of usernames and repository names.
OpenBazaar	✓	✓	In development	Decentralized marketplace. Uses a DHT as routing mechanism and Bitcoin as payment network.
2gather	✓	✓	Demo application	Decentralized video sharing application. Uses a blockchain for the registration of video channels. Uses BitTorrent with a DHT as routing for distributing videos.
WebTorrent		✓	Operational for <1 year.	Javascript library enabling BitTorrent technology to be used in browsers. Uses a DHT as routing mechanism. Uses WebRTC for all communication.

Figure 2.8: Technologies overview

Chapter 3

Related Work

> If we knew what it was we were doing, it would not be called research, would it?
>
> Albert Einstein

Creating a link sharing system touches upon many subjects in Computing Science. Adding the decentralization aspect only expands the number of subjects. This chapter dives into the research performed on some of the subjects. Privacy, blockchain technologies and social networks are among the discussed topics.

3.1 Privacy

A large body of scientific work in peer-to-peer systems is focussed on preservation of privacy for end-users. The main reason this happens in peer-to-peer systems, it because these technologies are used as a way to circumvent central power and to prevent mass-surveillance. Therefore, the pressure on privacy is high in these peer-to-peer systems, resulting in a large body of work. Centralized services often benefit from the data their users generate and have no interest in these privacy preserving technologies. Where end-to-end encrypted communication used to be one of the highest goals, the aim is now even higher by going for untraceable endpoints and creating traffic-analysis resistant communication [1].

The annual report of the Internet Monitor project highlights the most compelling event and trends in the digital world. The 2014 edition[1], which focusses on, among other things, the tension between protecting privacy and using big data for social good, offers a great piece of insight in current problems on the internet.

The report states that, due to the revelations of Snowden, there is a revived interest in promoting decentralized and peer-to-peer approaches to network infrastructure. The approaches are used as a measure to support private communication, and be more resilient against censorship, surveillance and network outages. However, due to the decentralized nature of these networks, maintenance and security are performed by the communities using these networks. Security and privacy might actually be harder to "get right" for these communities then they would be for an Internet Service Provider.

An essay in the report is about the relation between data and privacy. Users are often unaware of the practices of their service providers. Even if users are aware, switching to privacy preserving alternatives is often an impractical solution. Users would no longer be able to use most popular services. However, users are often not in a position to evaluate which services

would be better privacy-wise. Another challenge described in the report is the fact that most privacy preserving applications are only used in small circles of activists, technologists and journalists. Often due to use of cryptographic jargon which is indecipherable for normal users.

3.2 Blockchain and Cryptocurrencies

The technology behind Bitcoin is called the blockchain (a technical description is at section 2.1.2). Although the blockchain has first been described and implemented for Bitcoin, the ideas used already existed. The Proof-of-Work which Bitcoin requires for mining originate from the Hashcash algorithm [52]. Hashcash is an algorithm that requires an amount of work to be performed, just as with Bitcoin. The goal for Hashcash was to incorporate such a proof-of-work in emails and to prevent spam with it. If somebody sends an email, the proof-of-work would demonstrate that the sender performed work to send the email, thereby the email would probably not be spam. Sending one email would not be a problem. However, sending email in large amounts, as spammers do, requires a substantial amount of resources, thereby making it less attractive to send spam. Proof-of-Work, together with cryptography, hashing and theories of money[53] resulted in the system Bitcoin has become.

The amount of research in blockchain technology and its applications has been rapidly increasing over the last couple of years.[1] The blockchain is an interesting technology from many perspectives. When talking about Bitcoin and the blockchain, most people only think about digital cash. However, after examining the books about Bitcoin and the blockchain,it becomes clear there are many more interesting perspectives.

The blockchain is a breakthrough in the field of computer science. It is the first digital form of cash which solves the double-spend problem without the need of a central authority. The trusted central authority, like a bank or Paypal, would verify that each portion of money can be spent only once. The blockchain removes the need of a central authority by two means. First, the ledger which records all transactions, is public and is verified to be correct by each party in the network. Second, the proof-of-work makes is impractical and expensive to change any part in the history of the blockchain [54]. When a user makes a transaction, trust in the receiving party nor any intermediary is required, only the Bitcoin software has to be trusted [10]. Because there are no intermediaries able to change or control the network, any person can make a transaction to any other person. Thereby creating a system which allows for international transactions which do not require permissions from banks, can be performed at any time of the day, and for a small fee. It is hard, or even impossible, to find a comparable process in current economic infrastructures [55].

Blockchain technology is used to perform timestamping of documents, creating an undeniable proof of the existence of a document at a certain time [56]. It used to register assets, to create prediction markets and to create identity and name systems [12]. It has even been suggested to provide the basis for a largely digital government. The blockchain allows for the removal of middlemen from society in the largest form. The government acts as a middlemen for registering our properties, distributing funds, and coordinating society. It is argued by Wright et al. that the blockchain might ultimately result in what they call "the rise of Lex Cryptographia" [57].

[1] http://suitpossum.blogspot.co.uk/2014/12/academic-bitcoin-research.html

Zyskind at al. performed research in the area of personal data storage. They intended to protect the privacy of users and the security of their data. They conclude that personal and sensitive data should not be trusted to third-parties because they are susceptible to attacks and misuse. Their solution to the problem of trusted third parties consists of a blockchain registering ownership over data. The blockchain register which entities can have access to data. Finally, they state that blockchains can be harnessed to solve trusted computing problems in society.

3.3 Name/identity systems

The idea of unique identities originated from before the internet [58]. A public and private key pair could be used to create private communication, no third party was needed. It allows for ideal communication where knowing ones name and address is enough for sending private messages of which the signatures can be verified. However, these systems are not common practice on the internet due to their usability issues [59].

Public key infrastructures (PKI) allow for the binding of user identities to public keys. PKI's were commonly deployed as central servers containing the bindings[2]. However, soon after the spread of PKIs, it became clear there was a major drawback[60]. The storage of public keys happened at so called Trusted Third Parties (TTP). If two people wanted to communicate in a secure way, they had to use a third party to look up each other public keys. These TTPs were recognized as problem a number of reasons. They were, among many other problems, single points of failure, were dependent on Certificate Authorities which were single points of failure as well, and they allowed the owner of a PKI to register which user is requesting which key.

Fromknecht et al. performed a research at MIT to create a PKI with identity retention in a decentralized solution. They identified the Namecoin blockchain (section 2.4.2) as a potential solution. Based on the Namecoin blockchain, they create Certcoin, which is tailored to their requirements. They conclude that blockchain based identity providers can offer better guarantees than either Certificate Authorities or current centralized PKIs [61].

Wachs et al. performed research to the feasibility of a of decentralized name system with the goal of being censorship resistant [62]. They explored the theoretical design fore a name system and started by discussing Zooko's Triangle.

Zooko's Triangle consists of three requirements for name system of which, he conjectured, only two can be attained. The requirements are that in a name system, the names should be **memorable**, the system should be **secure** and the names should be **globally unique**. Zookos conjecture states that it is impossible to achieve a system that achieves memorable, secure and globally unique names *at the same time*[62].

A system like Namecoin might appear to contradict Zooko's Triangle. However, Wachs et al. argue that the Namecoin system's security depends on the computing power of the network. An adversary with more computation power than the network would be able to create an alternative timeline with different domain names. However, since the release of their paper at the start of 2014, the hashrate of Namecoin has dramatically increased due to *merged mining*.[3] The Namecoin hashrate is about a third of that of Bitcoin, thereby making it practically impossible to gather a majority in the Namecoin computation network.

[2]The University of Groningen used to have a PKI located at `http://certs.rug.nl`

[3]`https://bitinfocharts.com/comparison/hashrate-nmc.html`

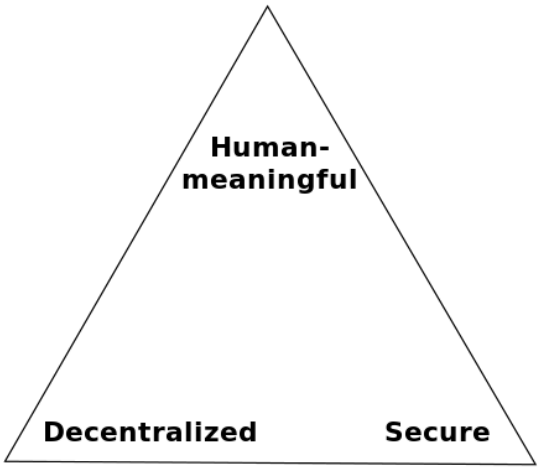

Figure 3.1: Zooko's Triangle

Frederic Jacobs[63] and Garman et al. [64] have selected blockchain-based technologies for creating decentralized name systems. Both recognize that DNS, Certificate Authorities and login providers such as Google and Facebook, act as centralized forces in name and identity systems. Our reliance on these companies raises concerns about privacy. All widely adopted solutions have a provider guaranteeing claims on unique names. Their solution to the privacy issue include using a blockchain to replace the trusted third parties. Both researches conclude that blockchains can replace trusted third parties. Although the scalability of blockchain-based systems remain an issue for the long-term future.

3.4 Distributed Hash Tables

A large number of distributed hash tables (DHTs) have been proposed in the past fifteen years. They are used as distributed key-value stores (a technical explanation can be found in section 2.1.1). One of the largest DHTs is the Mainline DHT implementation used in BitTorrent clients. Its size was measured over a period from 2011 to 2013, and ranged from 15 to 27 million nodes on a daily basis. Thereby making it the largest known overlay network [65].

The distribution of the keyspace, the routing between nodes and solutions to joining and leaving nodes, are all implementation specific. There are many implementations of DHTs among which Kademlia[4], Chord[66], CAN[67] and Pastry[68] are the most well known ones. Many variations exist on these DHTs. In fact, KAD, Mainline DHT and Vuze DHT are all variations of Kademlia. However, they are incompatible with each other.

The most important characteristics of a successful DHT implementation can be summarized in three points. First, the network has no central coordination or control, the network is autonomous and decentralized. Second, the system should be able to scale to many millions of nodes (as is demonstrated by Mainline DHT). And third, a successful DHT should be able to handle churn well. Churn is the process of nodes joining and leaving the network. When there is a large amount of churn, the topology of the network will change constantly, and the distri-

bution of the keyspace changes as well.

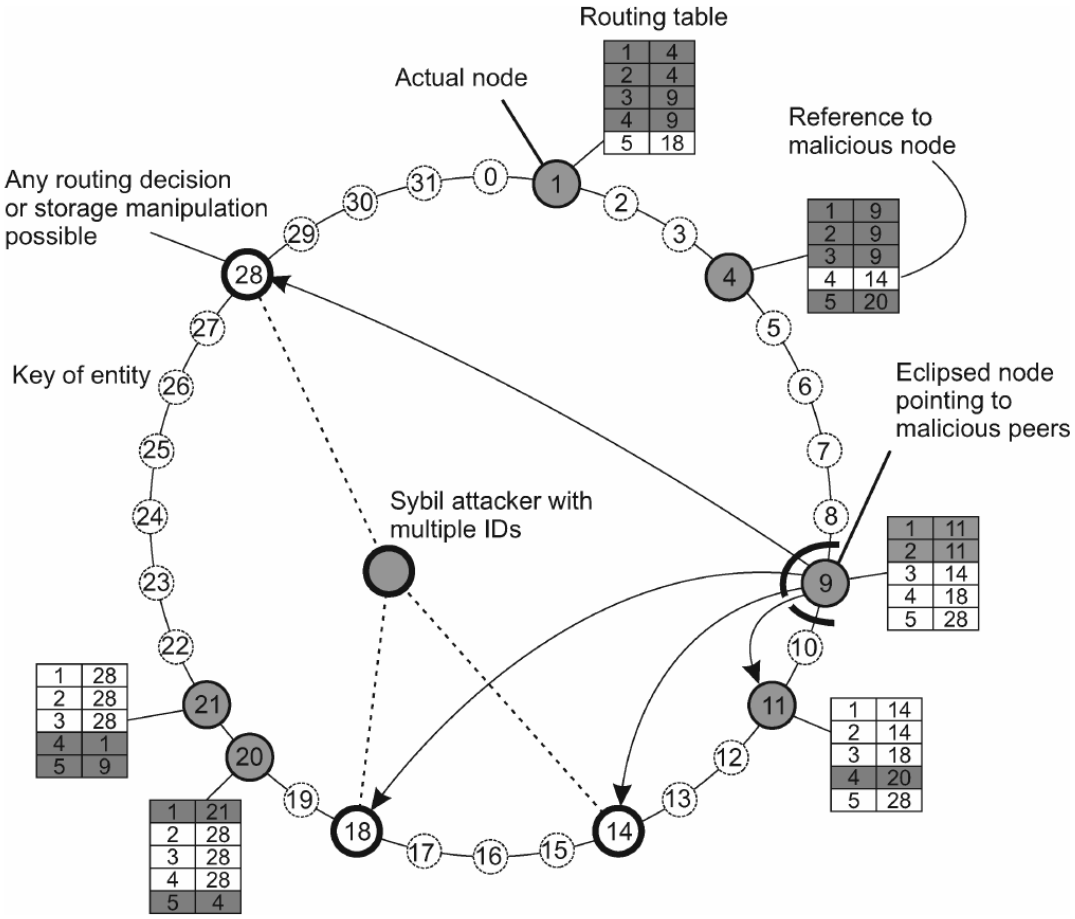

Figure 3.2: DHT Sybil attack

Sybil attacks are still considered an open problem in DHTs. During a Sybil attack, an attackers inserts a large number of nodes into a DHT. By doing so, the attacker can abuse the system by hiding information or sending wrong information through the network. An example of a Sybil attack is visualized in figure 3.2. There are proposals like Speedy[69], Whanau[70], Persea[71] and Octopus[72] which attempt to be resistant against Sybil attacks. Most of them use social graphs to select nodes which can be trusted.

3.5 Decentralized social networks

Social networking is an important part of users' online lives. Websites like Facebook, Weibo, Twitter and Ello have millions of users on a daily basis. There are two main problems with these social networks. First, they form centralized stores of information. The information is hard to extract and cannot easily be used on other sites. Second, the sites allow for only limited

control over personal information. Several researchers have suggested to use decentralization as a means to solve these two problems. Decentralization gives the opportunity to provide control over private data and over the users' privacy [73].

In 2009, the research team of Cutillo et al. propsed to solve the same centralization issues with a decentralized approach as well. They propsed to create a system which is secure, provides privacy to its users and lets users have control over their personal data. Their proposal consisted of three parts: *matryoshkas*, a peer to peer substrate (e.g. DHT), and a trusted identification service. The matryoshkas are the logical equivalent to what is now known as onion routing (e.g. Tor (The Onion Router)). The trusted identification service provides the users with a unique pseudonym, a unique identifier and certificates for signing data. There proposal creates the identification service as a centralized service, because no decentralized service existed at the moment. They argue that their solution is resistant against impersonation attacks, is end-to-end encrypted, preserves privacy and allows for verifying the integrity of data [74]. Later, the project got named Safebook, and formed the basis for many attempt at decentralized social networking [75].

Many decentralized social network have been proposed. Although most have been merely academic, Diaspora appears to be to only one with a reasonable user basis [76][77]. The success of a social network is largely dependent on three factors: 1) Which social network is used by family and friends? 2) What is the quality of the content on the network? 3) How is the user experience?[78]

3.6 Internet of Things: a hard push for decentralization

The European Parliament recognizes the Internet of Things (in the form of Smart Homes) as a potentially life changing technology. Their greatest concern with the technology is the protection of data and the privacy of smart home users. They state an uncertainty when it comes to smart devices and liability. Who is responsible when a smart fridge orders food? Who would be responsible in case problems arose?[79]

Over the past couple of years the terms Internet of Things (IoT) has grown in popularity. Stories of refrigerators ordering milk, and self-owning self-driving cars which pay for their own fuel with the money they earned with their services, may sound far-fetched. However, they might appear relatively soon according to IBM. IBM initiated research to discover which technologies can be used in the IoT. They expect that the current 1 billion devices will have grown to over 25 billion by 2020. According to their research, there are five main forces why today's internet will not be able to accommodate the 25 billions devices[80]. These forces are:

Middleman
: The high cost of connectivity because of centralized clouds, server farms and the service costs of middleman.

Broken trust
: The broken trust and lack of privacy which have been revealed by Snowden. An IoT which is built on trusting central companies controlling personal devices is hard to imagine and maybe even harder to engineer. Closed source approaches are obsolete and must be replaced with security by transparency.

Companies are not future proof
A company creating smart devices is quick to enter the market and sell products. However, IoT devices will be anything from doorknobs to doorbells and they are expected to last for years if not decades. In case they contain a critical security bug, the costs for updating might be incalculable. That is, if the original company still exists.

Lack of functional value
A microwave connected to the internet does not increase its functional value per se. The core functionality or user experience should be increased, which is something most successful IoT devices have done without subscription or app.

Broken business models
There is a lack of business models which will work in the IoT future. Advertisements, analytic,s and overly optimistic revenue models have been marked as improbable.

Following the argumentation from IBM they state that most of the interactions on the web are in fact transactions. And following Moores law there will be vast amounts of unused bandwidth, storage and processing power which can be used in order to process the transactions which will be made by the billions of IoT devices which will be connected.

Therefore, the recommendations from IBM to overcome the before mentioned problems include decentralized computing in the form of peer-to-peer systems. However, decentralization is not the only factor for success, trustless computing is another big factor. An environment without central party that needs to be trusted, where the other peers do not have to be trusted and in which there is no single point of failure [80]. Their technical recommendations can be summarized in three points:

- Trustless peer-to-peer messaging
- Secure distributed data sharing
- A robust and scalable form of device coordination

3.7 Recommendations

After decades of work in the field of decentralization, many lessons learned have been learned. This section describes the recommendations as they have been collected from previous research.

Narayanan et al. [76] made recommendations specifically for developers creating decentralized systems. The list of seven recommendations can be summarized to:

1. Create economic incentives for entities participating in the system.
2. Analyze your target audience. Which features do they want? Do they value encrypted communication over usability?
3. Achieve privacy not only by technology, but by socio-legal approaches as well.
4. Privacy should not be the only advantage.
5. Interoperate with other systems by writing glue code and using open standards.

6. Start with a minimum viable product.

7. Work with regulators in order to push for interventions such as transparency and opt-out.

Narayanan et al. described the challenges for decentralized applications. In a centralized application where all data is stored in a single place, for example a database, asking a question about all data is rather common. In a decentralized application there is no unified view of all available data. Asking for example how many users are part of a DHT can become a challenge to answer. Detection of fraud or spam becomes harder as there is no single location where data is going through where it can be filtered. Since decentralized applications make connections between many nodes, over many networks and even spreading continents, network connections may be unreliable more often.

Geambasu et al. [81] created a system in which data will self-destruct after a certain amount of time in order to increase data privacy. They created a Firefox plugin called FireVanish which used the system. In the anecdotal section they concluded that a practical solution with a minimalistic interface is relatively intuitive. They stated: "We theorize that solutions that require new infrastructures have a greater barrier to adoption than solutions that can 'parasitically' leverage existing infrastructures."

Sharma et al. [82] published a paper about an architecture for decentralized online social networks (DOSN). They conclude that using the resources from end users, as is for example the case with a DHT, does not mean that scalability or robustness are characteristics of the system by definition. Moreover, privacy and security do not come naturally as well, in fact they are more vulnerable due to the lack of any accountable authority.

Paul et al. [77] performed a survey on decentralized online social networks. They made conclusions about the user experience as a recommendation for DOSNs. Installing software on machines may require administrative rights which may make it a barrier for adoption. Moreover, installations can result in interoperability issues due to different operating systems. Therefore, they argue that DOSNs: "...should be running at every web-connected device without installation obstacles...".

For privacy fighters, early adopters and some developers topics as privacy, security and decentralization may be important. However, Paul et al, and the privacy-first chat application Hemlis, came to the same conclusion about the other users of the system. Paul et al. describe: "Finally, not everyone is equally passionate about privacy issues, and in the absence of a critical mass, even if security related technical challenges to realize a decentralized online social network are overcome, the exercise may be futile."[77] And on the final blog post of Hemlis the creators concluded: "Each new attempt have made us understand that our goal of creating a mass market messenger just based on the fact that it is private, secure and beautiful, is not nearly enough."[4]

[4] https://hemlismessenger.wordpress.com/2015/04/22/sometimes-you-understand/

Chapter 4

Towards decentralized link sharing

> The original idea of the web was that it should be a collaborative space where you can communicate through sharing information.
>
> Tim Berners-Lee

Abstract

A link sharing system allows users to create an account, post links and comment on the links. Users can vote on links and comments, which results in a user-curated lists of content. A decentralized link sharing system can create user accounts by using a blockchain to store a combination of usernames and public keys. A distributed hash table can be used to store all content posted by users. To reduce the number of requests in the distributed hash table and to allow for lower loading times, BitTorrent can be used to actively synchronize content.

Creating a decentralized link sharing application requires knowledge about the requirements for link sharing. Based on these requirements, and the technologies discussed in the previous chapter, a decentralized link sharing application will be proposed. After that, an evaluation of the proposed system can be done, based on the goals of preserving privacy, having control over data and increased scalability.

4.1 Requirements

The requirements for a decentralized link sharing system are spread out over multiple facets which all have an effect on the research questions (defined in section 1.5). These facets are discussed individually in the following sections. They include functionality, usability and decentralization requirements.

4.1.1 Posts, comments and votes

The main functionality of link sharing can be described as an application where posts (links or text) can be shared, after which users can comment on them. Users can vote on posts and comments, and have the ability to sort them, for example by most upvotes, or by the time they were added. The voting results in a user moderated list of posts.

The posts are categorized in sections[1]. Each section can set its own rules when it comes to the content of the posts and comments. A section can have multiple moderators which enforce the rules and can ban users in case the rules have been broken.

4.1.2 Registration

The registration of users should be non-restrictive, meaning an email address is optional and so is the verification of an email address. This makes it easy to create an account and allows for the creation of one-off or throwaway accounts[2]. In cases where controversial or personal information is exchanged but where the poster wants to stay anonymous these one-off accounts can be used. Alex Leavitt performed research about the use of throwaway accounts. He recommends the following: Given the beneficial nature of these [throwaway] accounts, especially for those individuals who tend to feel less anonymous in particular situations, designers should consider alternatives to permit these identity performances within their own systems.[83] Being able to associate a username with an IP address might reveal enough information to identify a person.

Accounts can be used to explicitly link the account of the user with a person in real-life. On Reddit, this is used for ask my anything (AMA) sessions, where the user has to provide proof that he or she is the actual owner of the account. Figure 4.1 illustrates Edward Snowden's photo used as proof of giving and AMA session[3].

4.1.3 Usability

Smartphones become more and more powerful, and are more ubiquitous in everyday life. A link sharing system will have to work on mobile devices as well as laptops with varying specifications. Phones as well as laptops find themselves in changing environments as they move from one location to the next. They often find themselves behind a router or NAT over which they have no control, for example with public wifi hotspots or at the office. Moreover, people are less device-bound, and often assume they can log in anywhere and have their data available. Last but not least, using the service should be mom-proof. Meaning it does not require a Computer Science degree in order to set it up and to use it.

4.1.4 Decentralization

In the previous chapters decentralization has been set as the way to go for a link sharing system. There are a couple of requirements which are inherent to the use of decentralization technologies.

First of all, it has to be open source. Users, or at least Computer Scientists, should be able to verify what the application is doing. Stating that an application using a decentralized architecture and performs end-to-end encryption is no proof of actually implementing those features.

Second, it has to be as operating system agnostic as possible. A decentralized application will run on many devices and operating systems. They all have different components and features which might make it harder to be supported everywhere.

[1] On Reddit, sections are called subreddits

[2] On Reddit, one-off accounts are called throwaway accounts

[3] `https://www.reddit.com/r/IAmA/comments/2wwdep/we_are_edward_snowden_laura_poitras_and_glenn`

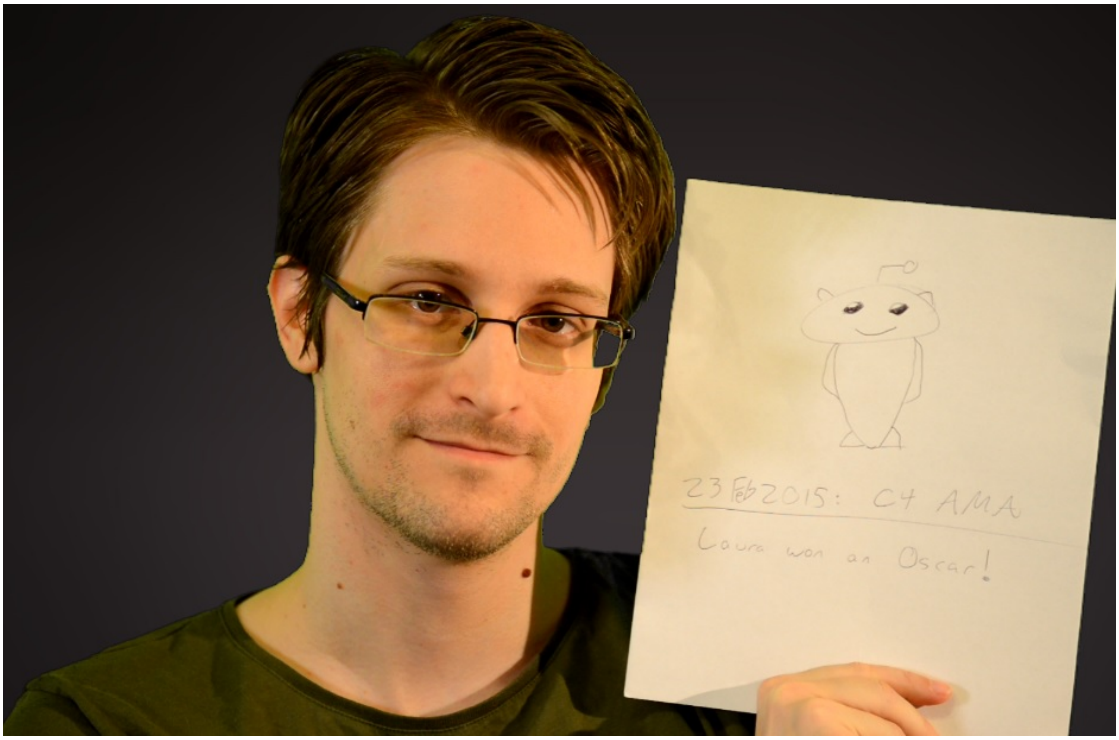

Figure 4.1: Edward Snowden giving proof for his AMA.

With decentralization technologies come their associated attacks. There is of course no technology which is not hackable. The only possibilities are either making it expensive, or making it time-consuming to perform attacks. These often go hand-in-hand.

4.1.5 Social requirements

A case could be made for implementing requirements regarding the social consequences of the application. Tools are often used and abused by criminals or for activities not intended. Therefore a decentralized link sharing application can be abused for money laundering, child pornography and organization of criminal activities. However, no technical solution will solve a social problem directly. And there is no reason to believe that implementing technical solutions against these problems will solve the social issues causing them. Furthermore, uses for a system might be very different morally but are indistinguishable from a technical point of view[4].

4.2 Proposal

Based on the previously set requirements and technologies a proposal has been made to create a decentralized link sharing system. The decentralized link sharing application will use two peer-to-peer networks. The first one is a blockchain peer-to-peer network, to other one is a

[4] http://rys.io/en/94

bittorrent network. The bittorrent network will be used in combination with a DHT instead of a tracker. The DHT will take the responsibility of routing messages to destination nodes. An overview of how these networks are used in the decentralized link sharing application can be seen in figure 4.2.

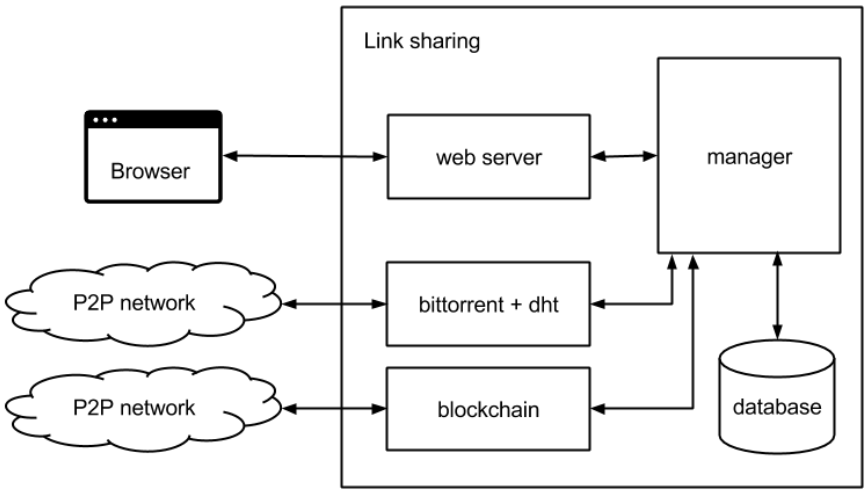

Figure 4.2: Link sharing overview

Users can interact with the application using a browser. The browser will be pointed to an address on the local machine (for example: https://127.0.0.1:10897). The web server will be listening for requests from the browser, and forward the requests to the manager. The manager processes all requests. When the requested data is present in the database it will respond with the requested data. If the data is not available it might request the data from either the bittorrent or the blockchain peer-to-peer network.

4.2.1 User Registration

One of the most striking differences between centralized and decentralized applications is the way user accounts are handled. On a normal website a user creates an account, so he is allowed to post messages or perform actions. The account is usually accessed by using credentials like a username and password. In a decentralized application, where are these credentials stored? Where are the credentials verified? How can other nodes verify an account is logged in?

As an example the Twister application is examined. There is no central authority storing identities and verifying credentials. All interactions happens with signed messages. When a user creates an account, a public and private key pair are generated. When a message is signed with the private key it can be verified using the public key. The only thing left is to distribute the public key to other users. As has been identified in sections 2.5 and 3.3, a blockchain based identity system can be a reliable store for identity and public key mappings. Twister uses an architecture in line with these recommendations and uses the blockchain to store public keys and usernames. Then, in order to verify that a message is from a certain account, the message is signed with the private key and verified with the public key from the blockchain belonging to the user who sent the message. In other words, *there is no need to login*.

A blockchain can be used for the registration of usernames and public keys. Following the recommendations in section 3.7 it is best to "parasitically" make use of an already existing infrastructure. The Namecoin blockchain appears to be the most mature blockchain designed to be used as name provider. However, the future of the Namecoin blockchain is uncertain, and therefore the adoption of multiple name systems should be considered. An architecture in which identities can be provided by multiple name systems will be advised. The Namecoin blockchain can either be used for any of the already existing name formats, for example Onename and NameID[5], or a new name format can be proposed. Following the recommendation of using what already exists, using the Onename name system appears to be the most appropriate.

The Namecoin blockchain provides all of the features required for secure user registration. It guarantees uniqueness of usernames, it provides security by requiring a Proof-of-Work for the blocks in the blockchain, and it creates incentive to join the network because the coins associated with Namecoin can be traded for other currencies. Finally, the blockchain provides a dictionary for mapping usernames to public keys.

As an example of how a public key would be retrieved from the blockchain, the already existing Namecoin project is used. Assume a message has been received containing a username and a signature. In order to verify the signature, the public key bound to the username has to be retrieved from the blockchain. The most recent public key for the username is the one that is needed to verify the signature. The most recent public key for the username can be anywhere in the blockchain. This means that all the blocks need to be traversed, starting with the most recent block, in order to find the most recent public key for a username. Since the blockchain might become multiple gigabytes (it is 2.6 GB at the moment), it can take a long time before the public key is found. Imagine having to do this for every message that is received. The Namecoin blockchain does not only store usernames and public keys, it is used for domain names as well but it is used for other data as well. To make looking up the public key faster, a database can be build storing just the username and public key combinations. The database will be updated for every block received via the network. Figure 4.3 demonstrates how every received message can be verified using the public key from the database.

4.2.2 Links and comments

A large portion of decentralized applications consists of static data. Data which does not change at all (video and audio files, once created they are distributed as-is) or changes relatively little

[5] `https://nameid.org/`

```
from Crypto.PublicKey import RSA

# assume a received message with the following structure
message = {
        "signature": "some_signature",
        "data": {
                "username": "some_user",
                # Contains data, for example a posted link
        }
}

# retrieve the public key from the database
public_key_string = database.query(message["data"]["username"])

# import the key
public_key = RSA.importKey(public_key_string)

# assert the data is signed
assert(public_key.verify(message["signature"], message["data"]))

# Take action based on verification
```

Figure 4.3: Verify all received message with the stored public keys

(blog posts, wiki). This data can be identified by the hash of the data. Since the data does not change (often) the hash can usually be used to request the data it is associated with.

However many websites are dynamic in the content they are portraying. Dynamic content is anything which will be gathered from possibly multiple locations or networks and assembled into something which is for example time or person dependent. Dynamic content makes decentralized websites hard. Imagine a decentralized social network with millions of users. To generate a page with latest updates from friends, would require the updates from friends. A node can ask for the data when the user is asking for it. This might take some time, considering there are many nodes involved in the storage and retrieval of the data. The other extreme would be to send all data to all nodes, all the time. This would allow for instant displaying of the latest updates. However, this would result in a higher bandwidth usage. Furthermore, a large quantity of data is received in which the user is not interested. A decentralized application will need to make a trade-off between rendering content instantly (possibly faster than a normal website), and by not consuming too much of the available resources (bandwidth, storage). This is where a decentralized application has to make a well balanced decision.

Links will be posted in sections (subreddits) which will be BitTorrent swarm, comparable to the way Twister creates a BitTorrent swarm for a user. When the swarm is joined, the data added to the swarm will be retrieved automatically. This means that when a user posts a link to a swarm the other users will receive the link. Data can be added to a torrent using the

functionality for adding pieces to an existing torrent[6].

In order to join the swarm a node needs to connect to a node already in the swarm. The addresses of nodes which are already in the swarm will be available in a DHT. The name of the section can be hashed and the hash will be the key in the DHT where to find the nodes in the swarm for the section. The key in the DHT will act as a tracker for the torrent. The key can be created as can be seen in figure 4.4

```
section_name = "Bitcoin"
key = sha256(section_name + "tracker")
```

Figure 4.4: The hash in a DHT for discovering the nodes in the swarm of a section

Nodes seeding the torrent for a section will announce that they are seeding. When an announce message is received it will be *appended* to the list of nodes seeding the torrent. When the node is appended, a timestamp is included. After a expiration time, the node will be remove from the list of nodes that are seeding the torrent.

There are two types of keys in the DHT. Some keys can contain a *single* value which can only be changed by one user. An example would be the comment of a user. The other type of keys can contain *multiple* values. These are keys like the tracker key. The keys which can contain multiple values are append-only. When a node announces to be seeding a torrent, the information will be appended to the value.

The same procedure can be done for receiving the comments for a link. The link itself will be unchangeable and unique. The link will be hashed, which creates a key for the DHT where the comments for the link will be shared in a torrent swarm. The key can be created as demonstrated in figure 4.5. When a key for a tracker is created, it can be used to join the torrent for the section or link.

```
link = "https://www.bitcoin.org/"
section = "Bitcoin"
key = sha256(link + "tracker" + section)
```

Figure 4.5: The hash in a DHT for discovering the nodes in the swarm of a link

A link might be posted to multiple sections. To track to which sections a link has been posted, another key is used. When a section is added to the list of sections for a link, the tracker key for the link in that section will be evaluated. If there are seeders for the section, the section can be added. This is demonstrated in figure 4.6. The keys for tracking the sections in which links are posted have multiple values and are append only.

4.2.3 Votes

The storage of votes can be done in a DHT. When a link is submitted, users can vote on the link. They can either give an upvote or a downvote. A user will cast a vote by sending a signed

[6] http://www.rasterbar.com/products/libtorrent/manual.html#add-piece

```
# assume dht interface
dht = DHT()

# The data is part of a signed message
link = "https://www.bitcoin.org/"
section = "Bitcoin"

nodes = dht.get(link + "tracker" + section])

if len(nodes) > 0:
        key = sha256(link + "sections")

        # Append section to the list of sections
        dht.put(key, section)
```

Figure 4.6: The hash in a DHT for discovering the nodes in the swarm of a link

message to the vote address associated with the link. The key for the votes address can be generate as can be seen in figure 4.7

```
link = "https://www.bitcoin.org/"
section = "Bitcoin"
key = sha256(link + "votes" + section)
```

Figure 4.7: The hash in a DHT for the votes on a link

The message with the vote will contain the username, whether it is an upvote or a downvote and time at which the vote was made. The whole message will be signed with the private key of the user and stored at the vote address of the link. The message can be verified to be sent by a certain user and within a certain time frame.

4.2.4 Spam

There is no central control on the usage of accounts. Therefore actions like spamming, might be harder to control than in a centralized solution, where all information flows can be controlled. When a user posts a message, the message might be distributed to other nodes interested in the message. However, if here is no limit on the amount of messages which can be posted, a node can fill the network with data.

Two good examples of spam prevention are BitMessage and Twister. BitMessage requires a message to have a Proof-of-Work. This means that if a node were to send spam, a Proof-of-Work needs to be generated for every message. This will either take time or resources making it unattractive or even infeasible to send spam. Twister uses another method to prevent spam in the network. Each message includes the number of the message for that user and the number of the block which was the most recent when the message was posted. When the number of

messages posted during the period of one block is too large, nodes will drop the messages. The messages will not be stored nor propagated through the network. If a node would try to send another message without increasing the number, the resulting key would be the same. The node can verify that the value for the key is not empty and therefore determine fraud.

To make sure a user in the decentralized link sharing system is not posting too many links, the spam prevention method of Twister can be used. When a user posts a link the link will be stored at an address of the user as well. The key where the message can be stored can be seen in figure 4.8.

```
1 key = sha256(username + "links" + k)
2 # k is the number of the link for this user
```

Figure 4.8: Storing a link for a user.

The message sending the link will contain the hash of the most recent block in the blockchain. The node which is going to store the link will first check the previous link of the user and check if the message is posted during the same block. This repeats until a certain number is reached, for example 5. When there are 5 link posted in the time frame of one block the message will be discarded and not stored. The same method can be applied to comments and votes.

There is another form of spam, or unwanted messages, which on Reddit is controlled by moderators. Users will for example post links which are outside of the topic of the subreddit, or they will violate any of the other rules of the subreddit. There are two proposals for implementing a form of moderation. The first is the way Reddit allows for moderation: the user creating a subreddit becomes the moderator of the subreddit. A moderator can assign new moderators and set the rules for the subreddit. In case the users of the subreddit are not happy with the moderators, they can create a new subreddit and moderate it according to their needs.

The second proposal is to allow all users to propose themselves as moderator of a subreddit. Other users can then subscribe to the moderatorm or multiple moderators. These moderators will maintain a list of usernames of users which are banned, and of comments which should be removed. When a node receives a link or comment on one of these lists, the node will drop the message and not distribute it to other users. This way there is still an option for a moderated section, but there is freedom in choosing who should be the moderator.

Chapter 5

Evaluation

> Arguing that you don't care about the right to privacy because you have nothing to hide is no different than saying you don't care about free speech because you have nothing to say.
>
> Edward Snowden

Abstract

The proposed decentralized link sharing application has been evaluated based on its ability to protects a users' privacy, its ability to scale and to which extent a user can keep control over data. The privacy of users is hard to protect in both centralized and decentralized applications. A decentralized application can protect privacy slightly better when implemented correctly. However, there are many aspect which can be improved. The control over data is not implicit by using a decentralized application. However, it can remove the reliance on trusted third parties. The scalability of a decentralized application is superior over a centralized application. The more users join a network, the more resources are available. The costs of scaling a decentralized application are replaced by users adding their resources to the peer-to-peer networks of the application.

With the proposed decentralized link sharing system as described in the previous chapter, it is time to evaluate the system based on the research questions and the goals set in section 1.5.

5.1 Privacy

Looking for privacy issues in a public link sharing system might appear out of place, considering the fact that it is public. However, there is still the privacy of users which needs to be taken into account. In this section there is a comparison of the privacy in a centralized link sharing system, for example Reddit, and the proposed decentralized link sharing system. First an analysis of adversaries is presented.

5.1.1 Adversaries

There are many reasons for adversaries to try to find out who is visiting which website. For example, visiting websites of LGBT communities, or browsing health information can contain

sensitive information which one would not like to be exploited by other people. Who would be interested in finding out about the websites a person a visiting and why?

ISPs Providers of access to internet, whether it be consumer ISPs or public wifi hotspots, are well positioned to monitor behaviour of users. They can use data mining to learn about, among other things, the websites a person is visiting. The data is valuable for advertising, either directly or by selling the data to other interested parties [84].

Corporate setting An employer usually has full access to all traffic in the local traffic, regardless of whether the device is personal or for corporate use. An employee can be monitored to register whether or not the network is used for personal activities. Investigating the contents of the traffic could reveal much more about an employee, and has been abused many times in the past [85].

Surveillance Governments can perform surveillance for many reasons. There are many goals such as preventing terrorism and protecting national security. Government surveillance might also be used to monitor people with opposing political views or actively prevent demonstrations [86].

Censorship Taking it one step further than surveillance, censorship can be performed to make the requested content unavailable. In China the so called Great Firewall is used to block IP addresses, filter content and perform man-in-the-middle attacks[1]. Other countries, such as Turkey, block Twitter regularly[2].

5.1.2 Current situation: reddit.com

When a user visits Reddit, the following happens: a user types in a URL, the url is resolved to an address using DNS, a request is sent to the servers of Reddit, one of the servers records which page is requested, the page is sent back over the internet, and finally the website is displayed in the browser of the user. An overview of this process is in figure 5.1.

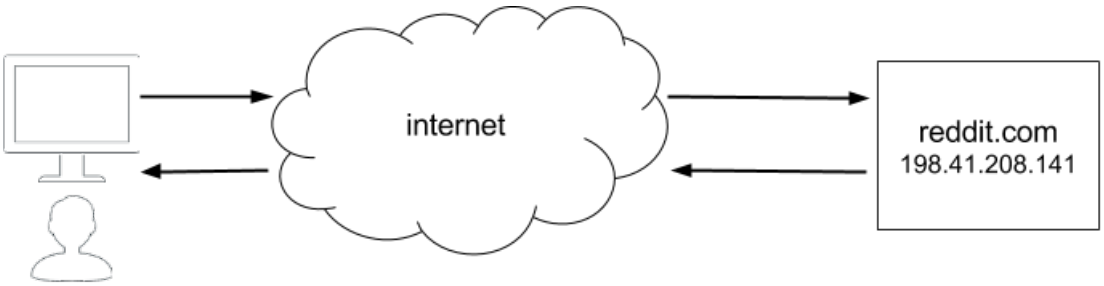

Figure 5.1: Browse Reddit

In this process there are numerous ways the privacy of a user can be harmed. The resolving of reddit.com to an IP address happens using DNS. The DNS requests shows which IP address

[1]https://www.theverge.com/2015/4/28/8508117/facebook-connect-great-firewall-great-cannon-censorship
[2]http://www.bloomberg.com/news/articles/2015-07-22/turkey-blocks-access-to-twitter-on-bombing-footage-hurriyet

it trying to resolve which domain name. The DNS message itself is not encrypted, allowing for sniffing of DNS queries[3]. Any party between the user and the DNS resolver, and the DNS resolver itself, can inspect the messages. When the domain name has been resolved, a request to the servers of Reddit will be sent. This request was using HTTP by default, but for reddit it became HTTP**S** by default from the 29th of June 2015[4]. With HTTP, the full request is readable and changeable by any party between the user and the reddit.com server. There is no verification in the HTTP to make sure the request has not been changed, making it vulnerable to man-in-the-middle attacks. With HTTPS browsing the web is more secure. HTTP requests are sent over an SSL tunnel, which encrypt all the data going over it. Although, this is a big improvement over HTTP, it is still possible to see from where to which destination a request is going. Therefore, it is possible to determine a user is visiting reddit.com. Even the contents of a request are not safe when traffic analysis is performed[87]. The certificates used in HTTPS connections are issued by Certificate Authorities (CAs). When a certificate is created by a CA, it is accepted by the browser as it is accepts certificates from over a 1000 CAs. There are many ways to compromise a CA, for example by breaking into one, just as has happened with the Dutch CA DigiNotar[5]. Every browser accepting certificates from DigiNotar needed to be updated to solve the problem.

5.1.3 Decentralized application

In the decentralized proposal, the application is not connecting to one server, instead it is part of peer-to-peer networks. The messages do not go to one single destination, they are spread out over the nodes participating in the network. This means that any node serving the requested content, could monitor the use of that content. However, it would only know a small part of all communication. One method to prevent that a node discovers the origin of a request is by using onion routing such as Tor (The Onion Router).

If an attacker wanted to link a user to an IP address, the attacker could track the propagation of messages through the network. If the attacker would be the first receiver of a message, the origin of the message could be revealed. The use of Tor could prevent the revealing of the origin of a message.

5.1.4 Comparison

In this section the centralized and decentralized link sharing applications are compared based on situations. The first situation, as shown in table 5.2, examines if the use of a link sharing application can be detected. The use of a link sharing application might be prohibited in certain countries. Detecting the use of an application can make it easy to either shut down all communication for this application, or link a user to the use of an application.

In the second situation, a user is reading certain sections. On Reddit these are called subreddits. Who can find out if a user is visiting a certain section? Table 5.3 shows the results of

[3]DNSSEC, the DNS version with security extensions, does not encrypt requests, it makes sure the integrity of the data is protected, and the origin of the data can be verified

[4]https://np.reddit.com/r/redditdev/comments/39zje0/reddit_will_soon_only_be_available_over_https/

[5]https://www.f-secure.com/weblog/archives/00002228.html

Adversary	Centralized	Decentralized
ISP	Easily, the destination of messages is always the same.	Yes. With effort, traffic analysis can be performed.
Corporate	Yes. The destination of messages is always the same.	Yes. With effort.
Government	Yes. By demanding information from the ISP, the server of the website, or by monitoring the hardware coming from the ISP or going to the website.	Yes. With effort, by demanding information form the ISP. Joining the network will reveal a lot of information as well, although no central registry of, for example, all IP addresses is kept.
Application	Yes. The website itself can find out whether an IP, and in case the user is logged in, which user is visiting.	-
Other users	Only when username and person are linked.	Only when username and person are linked.

Figure 5.2: Can the use of link sharing be detected?

the analysis. The same line of reasoning as in the previous situation is followed here. Visiting certain sections of a link sharing website or application might associate a user to certain topics.

Adversary	Centralized	Decentralized
ISP	Yes. In case of HTTPS, with traffic analysis.	Yes. With traffic analysis.
Corporate	Yes. In case of HTTPS, with traffic analysis.	Yes. With traffic analysis.
Government	Yes. By colluding with ISP or website.	Yes. By colluding with ISP or by joining with many nodes.
Application	Yes. The website itself can find out everything.	-
Other users	No.	When username and person are linked, by following the user.

Figure 5.3: Finding out which sections are visited

In the third situation the user is actively participating in the website or application. Is it possible to link users to actual persons, and thereby register their participating in certain sections or topics? The results of the analysis are in table 5.4.

The fourth situation analyses the possibility to alter content as it has been sent from the location it is stored, to the receiver. Does the user have a way to verify that the content received is actually created by the stated creator? Table 5.5 contains the results of the evaluation.

In the last situation, the goal is not to alter the content, but to completely censor the content. It should become impossible for users to retrieve certain content. The results of the analysis are

Adversary	Centralized	Decentralized
ISP	Yes. In case of HTTPS, with traffic analysis.	Yes. With traffic analysis.
Corporate	Yes. In case of HTTPS, with traffic analysis.	Yes. With traffic analysis.
Government	Yes. By colluding with ISP or website.	Yes. By colluding with ISP or by joining the networks with many nodes.
Application	Yes. The website itself can find out everything.	-
Other users	When username and person are linked, by following the users posts, comments.	When username and person are linked, by following the users posts, comments.

Figure 5.4: Finding out if a user participates in a certain section.

Adversary	Centralized	Decentralized
ISP	Yes. When using HTTPS, after CA breach.	No.
Corporate	Yes. When using HTTPS, after CA breach.	No.
Government	Yes. When using HTTPS, after CA breach.	No.
Application	Yes.	-
Other users	No.	After obtaining other users private key.

Figure 5.5: Is is possible to alter content before it is received by the user?

in table 5.6.

Adversary	Centralized	Decentralized
ISP	Yes.	Yes.
Corporate	Yes.	Yes.
Government	Yes.	Yes, by working together with the ISP.
Application	Yes.	-
Other users	Yes, in case they are moderator for a subreddit.	After colluding with all other nodes responsible for the same key in the DHT, yes.

Figure 5.6: Is is possible to censor content?

Looking at the situation described above, it is clear that ISPs and governments can harm the privacy of users of both centralized and decentralized link sharing applications. Although, in

a decentralized application it is slightly harder due to the large range of nodes cooperating in a system. Therefore, the privacy of a user in a decentralized link sharing application cannot be guaranteed, yet monitoring all users becomes expensive.

5.2 Control over data

To evaluate the proposed system on the control over data by its users, a comparison is made with Reddit based on a number of scenarios. The control over data is defined in two problems. First, the information users create is stored on websites which do often not allow for re-usability in other websites. The websites are information silos. Second, since users insert information in a website, they loose control over their data. Users do not control how their personal data is disseminated [73]. Websites have complete control over the data of their users, and are a trusted third party when it comes to storing and transferring information.

5.2.1 Current situation: Reddit

Users of Reddit are dependent on the availability of their data through Reddit. There are a number of reasons why Reddit go offline. Reddit could have decided to quit running the website. The website has a hard time making Reddit profitable, relying mostly on investments. If Reddit were to decide to stop operating the service, all data would unavailable. It is completely in the hands of Reddit to decide what will happen with the data. Will they allow users to download their data? Will they make all data available as a torrent?

A second solution to making Reddit profitable, would be to introduce a payment scheme. This can be anything from a monthly subscription fee with free access to a limited number of posts, to only allowing paying members to comment. In this situation, the users would dependent on the decisions of Reddit. As described in section 1.4.5, not just Reddit itself has control over the data of users. To some extent the moderators have control as well. They can decide who can join a subreddit, they can make it private, and they can decide and control which users are allowed to post in the subreddit. Moderators cannot just control, they can do it without being open about which users and posts have been removed and why they have been removed.

Next to the internal threats, there are external threats. Distributed denial-of-service (DDoS) attacks happen thousands of times each day [88]. Reasons for performing DDoS attacks vary from extortion to disrupting competition, and from hacktivism to unintentional attacks. As described in section 1.4.1, Reddit can create an increase in traffic to servers, often resulting in downtime. However, when a DDoS attack happens on Reddit, their servers might not be able to withstand the amount of data sent their way. As a result, users might not be able to access their data, and in the end users are dependent on the actions of Reddit to overcome the attack.

When the services of Reddit are unavailable, users cannot use their data unless preventative measure have been taken. A user might post a link on multiple websites to reduce the chance of losing data or not having the data available. However, if discussions about a link are spread across multiple websites, the data might end up spread out as well. Developers can built tools using the Reddit API[6]. The API is mostly used for mobile apps but is also used for desktop

[6] https://github.com/reddit/reddit/wiki/API

applications. It can also be used as a so called scraper or crawler, allowing users to retrieve their and other users' data in JSON format. The specification for the data is Reddit specific, but due to the structure of JSON the data can be reused in other applications. Users of Reddit have noticed their dependency on Reddit. A researcher has put effort into extracting all publicly available data from Reddit. The compressed dataset of posts and comments has a size of 250GB.[7]

The last problem to discuss about the control over data is the Streisand effect [89]. When access to data is restricted, for example due to users removing data or due to censorship, the attention and attraction to the data are increased. The resulting effect is paradoxical, leading to an increase in demand for the data. Then, if the data is still available somewhere, the spread of the data can happen in a rapid pace. Central hubs such as Reddit can be used to both, make the data unavailable as soon as possible, as well as be used to spread the data as fast as possible. It is a battle between users and moderators which often leads to the defeat of the party with the least manpower, the moderators. Not just Reddit itself, other services such as Google or the internet archive make it difficult for data to be entirely removed from the internet. However, there is a conundrum on the internet. On one hand there is the "right to be forgotten", which resulted from people being confronted with their actions from the past, due to their names reappearing in search engine listings [90]. On the other hand, there is the issue of "link rot", or the problem of links and references to website which do not exist any more [91].

5.2.2 Decentralized application

The decentralized proposal makes use of peer-to-peer networking protocols. BitTorrent, one of the protocols used in the proposal, allows for the replication of data across many nodes. Each of the nodes which replicated the data has to, either remove the data, or go offline, at the same time in order to make data unavailable. This means users will have a large chance that their data will be available when it has been replicated to a number of nodes. However, this means that users are dependent on other users for the availability of their data.

There is no company controlling the application. There might be a company developing the application, but this does not automatically result in full control over the network. Since the created software will be open source, it would allow for users to modify the software to their own needs. Thereby eliminating the option for a central entity to steer a project in the other direction than their users had in mind. Consequently, there is no central party in control, making it hard to remove content which is unwanted or illegal.

A decentralized link sharing application, as proposed in the previous chapter, does not imply that users will always have access to their data, nor does it imply that data is reusable in other networks or applications. When a user is unable to reach the network, only the data on the local machine would be available. Therefore, the application needs to store the data posted by user locally as well. This is at the expense of disk storage and less mobility. If a users stores all posted links on a desktop computer, action is required to make the data available on other personal devices.

There are many attacks possible at peer-to-peer networks. A Sybil attack, as described in section 3.4, can disturb the operation of a peer-to-peer network, with possible loss of data as result. BitTorrent technology has been used to perform DDoS attacks in the past by directing

[7] `https://archive.org/details/2015_reddit_comments_corpus`

traffic at a victim [92]. The users in a BitTorrent swarm are often in large numbers and spread over continents, resulting in a perfect network for attacking websites. However, taking down the BitTorrent swarm itself appears to be more difficult. Trackers do form a central point of attack, but the use of multiple trackers has largely mitigated that risk. Trackers and torrent indexes are the central points in the BitTorrent network which are attacked and shut down. When using a DHT, there is no central server to attack. The BitTorrent client Tribler uses a DHT and has not seen any complete network downtime during the six years it has been running.

A blockchain will be used to register users in the proposed decentralized link sharing application. Letting data disappear from a blockchain is practically impossible. Each node in the blockchain network will have an unforgeable exact copy of the data. If a user wanted to have its username removed from the blockchain, it would be impossible.

5.2.3 Comparison

Figure 5.7 is an overview comparing Reddit to the decentralized alternative.

Situation	**Centralized**	**Decentralized**
Service quits	Possible loss of all data. Full dependency on company.	There is no entity to quit.
Forced to quit	Possible loss of all data. Full dependency on company.	There is no entity to be forced quit.
Down due to attack	Possible loss of all data. Full dependency on company.	Possible loss of all data. No company to depend on.
Limiting access	Full dependency on company.	There is no entity to limit access.
Reusability of data	Not inherent, requires standardization. Accounts can be reused but require trust in company.	Not inherent, requires standardization. Accounts can be reused but require trust in blockchain technology.
Removal of data	Trust company to remove data. Other entities might copy data.	Trust other nodes to remove data. Other entities might copy data. Data in blockchain is impossible to remove.

Figure 5.7: Comparison of control over data in a centralized and a decentralized application.

5.3 Scalability

To evaluate the scalability of a centralized and a decentralized link sharing application, a comparison will be made between the proposal and Reddit. The scalability of both will be measured based on a couple of factors. What is the effect of an increase in number of users? What is the effect of a daily cycle of users using the services for some time and then leaving it again? How does the application scale when time progresses? What is the effect for a user when he uses the service?

5.3.1 Current situation: Reddit

Reddit exists for ten years at the moment, and it keeps all posts available. However, when a post on Reddit is 6 months old, it is archived. When a post is archived, comments and votes cannot be added or changed any more. The reason posts are archived is to reduce the load on the database servers. When the data for a post cannot be changed any more, it can be treated as static data, which has many benefits such as the ability to cache it.

Reddit experiences a double peek of users throughout the day. Most users are American with Europeans following on second place. Users are most active on Reddit in the evening, resulting in a peek from American users. A couple of hours later, after the Americans have gone to bed, the Europeans become active and a second spike of activity can be seen on the graphs. The number of pageviews can vary from 100k to 300k throughout the day. In order to handle the large fluctuations in pageviews, Reddit moved their website to Amazon Web Services in 2009. The services from Amazon, in combination with an architecture focused on scaling, allow Reddit to adapt to changes in the number of pageviews. The costs of running Reddit correlate with the popularity of the website. However, it turns out to be difficult to generate a profit from the content on Reddit.

As mentioned before, a user created an archive of all publicly available posts and comments. The archive has a size of 250GB. The longer the website exists, the more users there are, the greater the required storage capacity.

When a user on Reddit subscribes to more subreddits, there will be no increase in use of resources for the user. When the user browses more, the use of bandwidth correlates with the use of the website. However, browsing a website does not require a huge bandwidth.

5.3.2 Decentralized application

The proposed decentralized application uses two peer-to-peer networks. They are a blockchain, and a BitTorrent network with a DHT used as routing mechanism. The largest operating DHT is the BitTorrent Mainline DHT, a variant of Kademlia. The DHT started with the introduction of it in BitTorrent clients in 2005 and has been running since then. The network was measured in 2013 and shows that the number of nodes in the network ranges from 10 to 27 million. The number of nodes has a daily churn of about 10 million nodes, meaning the network doubles and halved in size in a daily cycle. The operating costs of the network are spread out over all users, where every users pays by providing its resources to the network. The more users join the network, the more resources are available. However, there is a major difference between the use of BitTorrent as a file sharing protocol and BitTorrent as a protocol for sharing data in the proposed decentralized link sharing application. The DHT in BitTorrent removes an entry after it has not been updated for 8 hours. If links and comments in the link sharing application would be treated the same way, it could result in losing links and comments when no seeders are available. Therefore, the links and comments are not just shared using BitTorrent, they are stored in the DHT itself as well. These records should not be dropped after 8 hours. This means that the size of the DHT will only grow, and not reduce in size.

The blockchain is used for the registration of user accounts. Namecoin's blockchain is currently 2.6 GB. The blockchain can only grow in size as it is an append-only structure. A way to reduce the size of this blockchain is by using an SPV client. An SPV client would decrease the

required storage capacity but would increase the bandwidth usage. Regardless, currently there is no SPV client available for Namecoin. The size of the blockchain might appear to be rather large when it is just used for the registration of users. However, the accounts are not just for the decentralized link sharing system. The Namecoin blockchain can be used as an alternative for a general authentication mechanism which can benefit many websites and applications.

When a user of a decentralized link sharing application subscribes to more sections, more torrents will be joined. Each torrent will increase the bandwidth usage dependent on the size and popularity of the section.

5.3.3 Comparison

Figure 5.8 is an overview comparing Reddit to the decentralized alternative when it comes to scalability.

Situation	**Centralized**	**Decentralized**
Increase in number of users	Requires bigger servers with more resources. Becomes more expensive when it becomes more popular	More resources are available when more users join the network.
Daily cycle of users	Requires dynamic scaling architecture.	DHT and blockchain are built for scaling.
Time progresses	Requires more resources from company.	Requires more resources from users.
Increased usage	Marginal increase in bandwidth usage.	Resource usage correlates with service usage.

Figure 5.8: Comparison of scalability between a centralized and a decentralized application.

Chapter 6

Discussion

> Art is never finished, only abandoned.
>
> Leonardo da Vinci

While researching decentralized applications and designing a decentralized link sharing application, many lessons have been learned. A decentralized application uses many technologies and has many factors to take into account. Most of these technologies are not common knowledge for Computer Scientists. Combining peer-to-peer networks while keeping in mind that it has to be secure, easy-to-use, respects users' privacy and should be scalable is a complex task. Although decisions have been made about which technologies to use and why, many discussions preceded these decisions. A number of these discussions are laid out in this chapter. They include lessons, which have been learned from the research and the discussions, that might serve as a starting point for future researchers and developers.

6.1 To browser or not to browser

The proposed link sharing application uses TCP and UDP connections. This means a separate application has to run in the background in order to communicate with other nodes in the network. There are four categories of applications when separated on their use of browsers. They are:

Standalone applications
: They do not use a browser and are run as a separate applications. Examples are BitMessage, GNUnet, Aether and Tribler. These applications built their own interface.

Browser as interface
: Projects like ZeroNet, IPFS, Freenet and Twister run an application in the background and use the browser as a means of interaction. The applications often consist of a built-in webserver which creates a website which can be visited with a normal browser. The benefits are that general web development technologies can be used even though they are decentralized applications.

Web applications
: A good example of a decentralized application which is a web application only is WebTorrent. The application is written in JavaScript and runs in the browser. A visit to a normal

url downloads the application and starts it in the webbrowser. Content, like for example videos[1], is downloaded from other users instead of a server. The communication with other nodes happens in the browser as well using WebRTC (section 2.6.4).

Own browser
The last category of applications are complete browsers. The Mist browser by Ethereum and the Maelstrom browser by BitTorrent fall in this category.

There are many reason to choose which category a decentralized application should be in. The Mist and Maelstrom browser try to create a bridge between decentralized applications and the "old internet". Their logic for communicating with peer-to-peer networks is built into their browsers which allows for decentralized applications. The browser acts as an interface for both decentralized applications and normal websites. This results in a smooth transition between normal and decentralized applications, without having the feeling of switching contexts. The web applications take it a step further by moving all their logic to a website. Since browsers cannot make TCP and UDP connections to other applications, WebRTC is used. This means that only other browsers supporting WebRTC can be used for these application. Furthermore, when a web application is closed there is no more participation in the network. Users are only part of the network when they leave the tab for the website open. Another concern with web applications is that cryptography in Javascript is considered unsafe[93][2], making it not recommended to do public key encryption in the browser.

Which kind of application a decentralized application should be is hard to answer. The recommendations (section 3.7) stated that using existing infrastructure and allowing all browser-enabled devices to join a network is preferred. However, using browsers comes at the cost of less freedom in using peer-to-peer networks.

The lesson learned is that the infrastructure for web applications is focussed on "regular" websites. Using a decentralized application or website is not part of that. All described categories are used in practice, but which category of applications will be successful for decentralized applications remains to be seen.

6.2 Communication

From the research performed for this thesis, it became clear that most applications use either UDP or TCP connections. This is no coincidence, and for a good reason, most hardware only supports these two kinds of packets. Any application performing end-to-end encryption has to perform the encryption in the higher layers, commonly in the application layer. It happens too often that packets from applications are sent unencrypted over the network. I argue that encryption of data in packets should be default. Transport layer security (TLS) is already widely applied in transport of data. A project like Telehash performs end-to-end encryption, all the time. Research in Telehash or equivalent projects should lead to a new standard for communication between machines. If Telehash would become the new way for creating sockets, then network hardware could be optimized by implementing dedicated encryption hardware.

[1] http://fastcast.nz/

[2] https://www.nccgroup.trust/us/about-us/newsroom-and-events/blog/2011/august/javascript-cryptography-considered-harmful/

However, encryption alone is not enough. By performing traffic analysis it can become clear who is communicating with who. There are several methods for making it harder to perform traffic analysis. Whisper, the communication protocol by Ethereum, is one of the projects trying to create traffic-analysis resistant communication. This project is not developed enough to be used in large scale. I am not sure whether it is feasible to make all communication traffic-analysis resistant. I do think there are many use cases where untraceable communication is important. Therefore, research and development of these technologies is important.

Current technologies for hiding IP addresses are already widespread. However, the speeds of these networks are disappointingly slow. Private communication should be as easy and fast as normal communication. Optimizing speeds and functionality in onion routing software can deliver benefits right now. The lesson to be learned here, is that the internet infrastructure is not built to keep communication private. Although there are many solutions, none of them are part of standard communication at the moment.

6.3 Website infrastructure

An interesting observation in applications like ZeroNet and Maelstrom is that a large part of tools, which are standard in website infrastructure, appear to be missing. Once the application is downloaded and runs, visiting a decentralized website is like visiting a normal website. A large difference is that developers of websites do not have to configure load balancers and web proxies. There is no need to create infrastructure to maintain sessions alive over multiple servers in case one of the servers goes down. There is no need to configure database replication across data centres, nor the necessity to monitor servers or collect logs from them. Although this might sound like heaven for some developers, these benefits come at a huge cost. The developer has no control over the application, nor the environment it runs in. This means applications will run in unpredictable environments, which can lead to problems. Users may forget to update applications, resulting in compatibility issues between versions or unpatched security issues. Web development for decentralized applications are completely different from regular web development.

6.4 Scalability

For the other technology, the DHT, the scalability remains to be seen. In the proposal there is no option to remove links or comments which means there is an append only DHT. The DHT would continue to grow, using more and more resources from the participating nodes. A suggestion to reduce the size of the DHT would by means of archiving links. After a certain amount of time, lets say 6 months, a read-only archive will be created from a link and all of its comments and votes. The user who posted the link can seed the archive for other users to read. Archives could be stored using a decentralized storage application like Storj or specialized archive seeding services which will be rewarded for seeding the archives.

One other aspect noted in the problem description (section 1.4.1) is the so-called ”Hug of death”. A link becomes popular which means a larger number of users is visiting the website. The server hosting the website might not be capable of handling these amounts of load. This problem is not solved by creating a decentralized link sharing system. However, if the websites

would be hosted in a decentralized network like ZeroNet or IPFS, scalability similar to these systems can be reached.

6.5 Incentive

Taking part in a decentralized system is most often two way traffic. BitTorrent makes sure users are not only leeching but seeding with the tit-for-tat system as well. Bitcoin creates incentive for verifying blocks and transactions by rewarding miners with Bitcoins and transaction bounties. Storj rewards the nodes storing data by using micro transactions, and users can earn credit by renting out their hard disk. Twister users mine the blocks for the blockchain to make sure their accounts and public keys are safe, and as reward they are allowed to send a limited number of messages which will be seen by all users. Such incentives, and especially economic incentives, have been recommended (section 3.7) for decentralized systems. An economic incentive system can have multiple benefits. It can be used to combat spam, or prevent vote rigging. It can create incentive to put resources into a network as well. However, creating an incentive system is a complex task. There are many requirements and parameters to take into account. The lesson to be learned is that incentive models have helped decentralized applications, but they are hard to invent and it is even harder to get the balance right.

Chapter 7

Conclusion

> It's more fun to arrive a conclusion than to justify it.
>
> Malcolm Forbes

This research describes a novel architecture for a decentralized link sharing application. The research started by analyzing projects and researches in the area of decentralized applications. Based on this research, the proposal for a decentralized link sharing application has been made. The first two goals described in this thesis are to find out which technologies can be used and how they can be used. The conclusion is that BitTorrent technology can be used to share data between users. BitTorrent usually involves the use of a tracker. A tracker is a centralized part in BitTorrent which tracks the availability of blocks of data across nodes. However, the tracker is a central point of failure. A distributed hash tables can be used as a routing mechanism which can replace the functionality of a tracker. Together, BitTorrent and a DHT can be used as a fully decentralized technology for sharing data, including links. In a link sharing website, links are posted by users which can be identified by a username. The search for a decentralized identity system has concluded with the use of a blockchain. Blockchain technology, the foundation for the cryptocurrency Bitcoin, is a relatively new technology with a large potential. It is used as the common truth for transactions in Bitcoin. Other projects and research have suggested to use the blockchain as a replaced for the currently centralized public key infrastructures. In a decentralized link sharing application, a blockchain can be used to store combinations of usernames and public keys. With the use of a blockchain as a decentralized public key infrastructure, the architecture of a decentralized link sharing application is complete.

The final goal for this thesis is to evaluate the proposed architecture on three aspects. The architecture is compared to traditional client-server based web browsing on protection of the privacy of users, control over data by users and the scalability of the network. The conclusion is that protection of privacy is hard achieve in both, the centralized and decentralized situations. However, the decentralized proposal has a slight advantage because it is harder, or more expensive, to monitor users due to the spread of sources and destinations. The control over data is not implicit by using a decentralized architecture. However, the required trust in a third party is reduced if not omitted. There is no company to shut down, get shut down or to perform censorship. The scalability of the decentralized proposal is largely influenced by the used technologies. All technologies, BitTorrent, DHT, and blockchain, have proven to be highly scalable with minimal to no downtime. Both BitTorrent and DHTs have been successful for over 10 years. Blockchain technology is relatively new with its existence of of 6 years, and has some

unknowns ahead, such as the increased size of the blockchain.

The decentralization of applications does not come for free. There is a trade-off between the pro-active distribution and replication of data, and the time to wait for content. Pro-active replication has the benefits of fast access to data which is already replicated, but at the cost of higher bandwidth and disk space usage. When no pro-active replication is performed, less bandwidth and disk space are used. However, it comes at the cost of a larger delay between the request and retrieval of data. Especially because a decentralized application runs on a network operated on and by volunteers.

The proposed solution is not a perfect solution. It requires that the application should be installed on a machine, which can be an obstacle for users and in some situations. The communication of data has many aspects which can be improved. Although the communication is encrypted, it is not resistant to traffic analysis at all. And, above all, the infrastructure used to facilitate the communication is often owned by a small number of large companies. This makes surveillance and traffic-analysis a hard problem to solve. On another note, due to the increased bandwidth and disk space usage, the application is not suitable for phones. Other than that, there is no economic incentive in the application. This means that there is no reason for users to set up the application in order to provider their resources to the network. The solution described in this document can serve as a starting point for further exploration for a decentralized link sharing application. It can form the basis for research towards a generalized framework for decentralized applications.

Chapter 8

Future Work

> A society grows great when old men plant trees whose shade they know they shall never sit in.
>
> Greek Proverb

The technologies behind decentralized applications are just in their infancy. The many possible use-cases for the blockchain are already uncountable. The blockchain is often compared to the beginnings of the internet and the TCP/IP stack. TCP/IP was less beautiful and with fewer features than its competitors. It was too slow and would never be capable of transferring audio or video. However, from a simple protocol moving packets from one location the next, emerged the internet as we know it right now. And we are all watching videos over it on our smartphones. And here it is, blockchain technology. The same stories go around. The blockchain will never be able to do this or that. It is just a protocol for transactions from one address to another. What will emerge when the blockchain will be as common as TCP/IP?

Progress can only be made by investing time and energy. The future for decentralized applications depends on the current researchers and developers. During my research I have seen many possibilities, and many problems. This chapter focusses on the topics which, in my opinion, are important to invest in to create the successful future which I, and with me many others, endorse decentralized applications.

8.1 Layers

One of the favourite activities of computer scientists is to create abstraction layers. By creating layers, the underlying implementation details can be hidden, which results in a separation of concerns. Looking at the projects described in the previous chapter, there are common patterns which have been identified as the layers in decentralized applications. The initial description and analysis defined in this section can be used as a starting point for creating a generalized model for decentralized applications.

Transport

The transport layer manages the connections between peers and makes sure the messages arrive correctly. Protocols which operate on this level are transport protocols like UDP and TCP. UDP-

based protocols such as uTP[1], QUIC[2] and swift[3] are operating on this level as well.

Protocols like UDP and TCP do not have any form of encryption for their packets. Although there are checksums to verify the integrity of the packets and sequence numbers to track which packets have been arrived or not, there is no encryption of the actual data sent nor verification that the message actually comes from the original source. As a result, other protocols have been developed on top of them, for example SSL and TLS. However, these protocols operate, according to the internet standard[4], on the application layer. This leaves the decision whether to encrypt data or not to the developer.

Various protocols have been developed to operate on the transport layer and which perform encryption of the data. For example Tcpcrypt[94] which has been revived by IETF after the revelations of Snowden. Telehash (section 2.6.3) is another project creating transport layer protocols which are fully end-to-end encrypted, all the time. Telehash not only aims to protect content and identity, they also want to protect metadata.

Following the recommendations in section 3.7, it should be default and required to use encryption. The transport layer is a fundamental layer, if encryption becomes the de facto standard in this layer, it will become harder, or at least more expensive, to perform mass surveillance on the data going over the internet.

Routing

The routing layer is responsible for knowing where to route messages and where to find peers. When visiting a normal website, there is no specific routing layer. A DNS request will be resolved resulting in an IP address. A HTTP request for a certain website will be sent to the server of the website and a response will return the website. The routers and switches in between the machine requesting the website and the machine delivering the website know where to route a message. However, they do not know where content is. It is not possible to ask a router, please find the data corresponding to this key. There is no way to get an address for content, there is only a way to route a message to an address.

One way to create a content-addressable network is to use a DHT (section 2.1.1. A node in the network can request the data which corresponds to a certain key. However, a node does not know about all other nodes in the network. It knows about a limited amount of nodes called neighbours and it knows for which part of the keyspace these nodes are responsible. The links between a node and its neighbours form an overlay network. When a node does not know a node responsible for the key it is interested in it sends the message to the node which it thinks is closest to the key[5].

BitTorrent can use a DHT for routing as well. In a Kademlia DHT there is a notion of buckets. These buckets have a certain size k. This is the number of nodes which are all responsible for the same part of the keyspace. If a popular piece of data would be stored at the nodes responsible for the key, these nodes might use a large amount of resources because they will have to provide the data. In BitTorrent, the DHT does not store the data in the DHT, but is stores the location of the data in the DHT. The location data is a list of address and port combinations where the data

[1] http://www.bittorrent.org/beps/bep_0029.html
[2] https://www.chromium.org/quic
[3] http://libswift.org/
[4] https://tools.ietf.org/html/rfc1122
[5] In Kademlia closest means the XOR distance, not the physical distance

can be found. When a piece of data is downloaded the location of the node which downloaded it will be added to the DHT. Thereby creating more and more locations where the data can be downloaded resulting in a more scalable solution. BitTorrent has two layers of routing, first the DHT has to find the node responsible for the key, and second the value corresponding to the keys contains the addresses where the actual data can be found. Therefore, BitTorrent uses a DHT for routing to determine where data can be downloaded.

Another kind of routing which is present in for example Bitcoin and BitMessage is the so-called flooding. There are pieces of data (blocks, transactions, messages) which do not have one destination. All nodes in the network are interested in all data in this case. The transactions in Bitcoin have to be present at miners to be incorporated in blocks, and once a block is created as many nodes as possible have to receive the block as fast as possible. With BitMessage a node can only find out whether a message is for it by trying to decrypt it. It has to receive all messages to find the messages which belong to it.

Both kinds of routing (DHT and flooding) are not present in normal web browsing. Web browsers are merely endpoints receiving data, a browser is not relaying its data to other nodes interested in the website. But what about WebRTC? WebRTC does peer-to-peer connections between browsers. However, the only way to find other browsers is by using a central signaling server where both browsers connect to first.

Application

Using the previous two layers, the transport and routing of data are possible. On top of these two layers, an application can be build using its own set of rules. The application layer is comparable to the application layer (7) in the OSI-model and generally defines the algorithm an application follows. It defines how the data is structured which is exchanged between nodes and how it should be interpreted. However, in the OSI-model the session (5) and presentation (6) layers are between the transport (4) and the application (7) layer. The boundaries between these layers (5, 6, 7) are often debated and it is unclear which layer has which responsibility. In the decentralized model these layers are combined to one layer called the application layer which specifies the technology specific algorithm, data structures and sessions.

Composition

The composition layer is the layer which uses the underlying application protocols. When multiple peer-to-peer networks are included by using multiple transport, routing and application stacks the composition layer can interface with them. Data can be send and retrieved to and from these networks and can be visualized for the user. The application layer exposes an application interface which will be used in the composition layer. An example in current technology stacks would be an application which requests data from a database, a task scheduling system and a web API. The composition layer uses the interface exposed by the applications or networks it interacts with.

Examples

Although this is only an initial attempt at identifying the layers, it is still interesting to see how decentralized applications would fit such a model. Looking at Twister, the layers in figure

8.1 can been identified. In this case it is clear that Twister is using two different peer-to-peer networks to operate. A BitTorrent network is used for directly sharing posts with other users, and to store posts directly in the DHT as well. The blockchain network is used for storing user accounts and public keys, which are used to validate messages distributed via the BitTorrent network.

Transport	Direct sharing	UDP/TCP*	Storage	UDP/TCP*	Accounts	TCP
Routing		DHT		DHT		flooding
Application		BitTorrent		BitTorrent**		Blockchain***
Composition	Twister	Storing messages in a DHT, direct sharing of messages via BitTorrent, registration of users in the blockchain and validation of messages with public keys available in the blockchain.				

* DHT requests may be tunneled over TCP, using the peer-to-peer connections from the blockchain. However, these peer will be used as a proxy and still require other nodes with UDP connections.
** The BitTorrent library is used for access to the DHT directly
*** The blockchain is a (nameless) fork of Bitcoin.

Figure 8.1: Twister in layers

The GitTorrent project is also using two kinds of peer-to-peer networks to operate as can be seen in figure 8.2. Repositories are made available via BitTorrent. The repository itself is not stored in the DHT, just the addresses where the repository can be found, in the same manner as file sharing via BitTorrent happens. To find a repository, the right key in the DHT has to be found. This key can be found in a blockchain, which registers the usernames and repository names.

Transport	Direct sharing	UDP	Names	TCP
Routing		DHT		flooding
Application		BitTorrent		Blockchain
Composition	GitTorrent	Sharing repositories via BitTorrent. Registering user and repository names in the blockchain.		

Figure 8.2: GitTorrent in layers

Both Twister and GitTorrent appear to fit the layers perfectly. What about the proposed decentralized link sharing system? From figure 8.3 it is clear that the link sharing system will use three peer-to-peer networks.

Research into a model for decentralized applications would be a big step towards the realization of a framework for creating decentralized applications. A framework in which interaction with multiple peer-to-peer networks would be as easy as using a database. Multiple blockchains and peer-to-peer networks already exist, although integrating with them is not for the faint hearted.

Transport	Direct sharing	UDP	Storage	UDP	Accounts	TCP
Routing		DHT		DHT		flooding
Application		BitTorrent		BitTorrent**		Blockchain
Composition	Link sharing	Registration of user accounts in the Namecoin blockchain using the Onename name format. Links and comments will be stored in a DHT as well as directly shared using a BitTorrent swarm.				

Figure 8.3: Link sharing in layers

8.2 Internet of Things

Although the internet of things might still sound futuristic for some, it is heading our way rapidly. Companies like 21 inc[6] are developing chips to be embedded in all kinds of devices. Even the smartphones of today could benefit enormously from these developments, access to decentralized infrastructures could be reachable for everybody. A major drawback of using decentralized applications is the increased use of resources like power and bandwidth. The Android application for the messenger Tox only uses wifi default due to the increased data usage. Just being part of the DHT in Tox results in a data usage of over a GB in month. There are two solutions which, in my opinion, should be combined. The first solution is a more efficient distribution of data. Should an application actively push data or be more pull based? An application like Synereo might be turn out to be a good solution: using the network as a neural network. The second solution is better hardware. Increased battery capacities and wireless charging, combined with higher download speeds and raised limits on data, will give a better opportunity for decentralized applications on smartphones.

8.3 Identity

At the moment usernames and passwords are the most used way to identify a person on the internet. However, public-key cryptography is advancing. Facebook recently gave the option for using PGP encryption in email notifications.[7]. Bitcoin uses public-key encryption all over the place. There is a consensus that public-key encryption can be used to create a more private and secure internet. However, the private key part, used to sign and decrypt messages, is still a bit rough when it comes to user-friendliness. They are often bound to devices, hard to transfer and hard to store safely. These private keys are often used as identification. Future research should focus on the general problem of identification. Solutions currently used, like credentials, private keys, id-cards or biometrics, all try to solve the same problem. How can a person be identified? And how can this method be made unforgeable? Combinations of something physical or unique, with something to remember, like a password, appear to be the most common solutions. However, the level of standardization for identification lacks behinds.

[6] `https://21.co/`

[7] `http://www.theguardian.com/technology/2015/jun/01/facebook-introduces-pgp-encryption-for-sensitive-emails`

It makes it hard to do it right, and even harder for users to know how to recognize what is right.

Bibliography

[1] Urs Gasser, Jonathan Zittrain, Robert Faris, and Rebekah Heacock Jones. Internet monitor 2014: Reflections on the digital world: Platforms, policy, privacy, and public discourse. *Berkman Center Research Publication*, (2014-17), 2014.

[2] Staffs Keele. Guidelines for performing systematic literature reviews in software engineering. In *Technical report, Ver. 2.3 EBSE Technical Report. EBSE*. 2007.

[3] Hao Zhang, Yonggang Wen, Haiyong Xie, and Nenghai Yu. Distributed hash table: Theory, platforms and applications. 2013.

[4] Petar Maymounkov and David Mazieres. Kademlia: A peer-to-peer information system based on the xor metric. In *Peer-to-Peer Systems*, pages 53–65. Springer, 2002.

[5] Guido Urdaneta, Guillaume Pierre, and Maarten Van Steen. A survey of dht security techniques. *ACM Computing Surveys (CSUR)*, 43(2):8, 2011.

[6] Mebratu Kassahun. Secure routing in structured p2p overlay: Simulating secure routing on chord dht. 2015.

[7] Miguel Castro, Peter Druschel, Ayalvadi Ganesh, Antony Rowstron, and Dan S Wallach. Secure routing for structured peer-to-peer overlay networks. *ACM SIGOPS Operating Systems Review*, 36(SI):299–314, 2002.

[8] Stevens Le Blond, Pere Manils, Chaabane Abdelberi, Mohamed Ali Dali Kaafar, Claude Castelluccia, Arnaud Legout, and Walid Dabbous. One bad apple spoils the bunch: exploiting p2p applications to trace and profile tor users. *arXiv preprint arXiv:1103.1518*, 2011.

[9] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system. *Consulted*, 1(2012):28, 2008.

[10] Melanie Swan. *Blockchain: Blueprint for a New Economy*. " O'Reilly Media, Inc.", 2015.

[11] Florian Tschorsch and Björn Scheuermann. Bitcoin and beyond: A technical survey on decentralized digital currencies.

[12] Tim Swanson. The anatomy of a money-like informational commodity, a study of bitcoin, 2014.

[13] Bram Cohen. The bittorrent protocol specification, 2008.

[14] Bram Cohen. Incentives build robustness in bittorrent. In *Workshop on Economics of Peer-to-Peer systems*, volume 6, pages 68–72, 2003.

[15] Johan A Pouwelse, Pawel Garbacki, Jun Wang, Arno Bakker, Jie Yang, Alexandru Iosup, Dick HJ Epema, Marcel Reinders, Maarten R Van Steen, and Henk J Sips. Tribler: a social-based peer-to-peer system. *Concurrency and Computation: Practice and Experience*, 20(2):127–138, 2008.

[16] Tome Boshevski, Josh Brandoff, and Vitalik Buterin. Shawn wilkinson (shawn@ storj. io). 2014.

[17] Roger Dingledine, Nick Mathewson, and Paul Syverson. Tor: The second-generation onion router. Technical report, DTIC Document, 2004.

[18] Michael Herrmann and Christian Grothoff. Privacy-implications of performance-based peer selection by onion-routers: a real-world case study using i2p. In *Privacy Enhancing Technologies*, pages 155–174. Springer, 2011.

[19] Caleb James DeLisle. *cjdns Whitepaper*, 2015 (accessed June 25, 2015). `https://github.com/cjdelisle/cjdns/blob/master/doc/Whitepaper.md`.

[20] Jack VanDrunen. Security of the cjdns address space. 2014.

[21] Jhon Smith. *Open Libernet*, 2014 (accessed May 12, 2015). `http://openlibernet.org/paper/open-libernet.pdf`.

[22] Moritz Warning. *KadNode*, 2015 (accessed June 25, 2015). `https://github.com/mwarning/kadnode`.

[23] Vincent Durham. *Namecoin*, 2015 (accessed May 22, 2015). `https://namecoin.info/`.

[24] Greg Slepak. Dnschain+ okturtles.

[25] David Recordon and Drummond Reed. Openid 2.0: a platform for user-centric identity management. In *Proceedings of the second ACM workshop on Digital identity management*, pages 11–16. ACM, 2006.

[26] Chris Coyne Max Krohn. *Keybase*, 2015 (accessed May 13, 2015). `https://keybase.io/docs`.

[27] Muneeb Ali Ryan Shea. *Onename*, 2015 (accessed April 4, 2015). `https://onename.com/`.

[28] Peter Saint-Andre. Extensible messaging and presence protocol (xmpp): Instant messaging and presence. 2011.

[29] Nikita Borisov, Ian Goldberg, and Eric Brewer. Off-the-record communication, or, why not to use pgp. In *Proceedings of the 2004 ACM workshop on Privacy in the electronic society*, pages 77–84. ACM, 2004.

[30] Jeremie Miller. *Telehash*, 2015 (accessed April 28, 2015). http://telehash.org/v3/spec/v3.0.0-stable.pdf.

[31] Eric Rescorla. Webrtc security architecture. 2014.

[32] Irungentoo. *Tox*, 2015 (accessed March 26, 2015). https://tox.im/tox.pdf.

[33] Jonathan Warren. Bitmessage: A peer-to-peer message authentication and delivery system. *white paper (27 November 2012), https://bitmessage. org/bitmessage. pdf*, 2012.

[34] Eric Klinker. *Project Maelstrom*, 2015 (accessed April 12, 2015). http://blog.bittorrent.com/2015/04/10/project-maelstrom-enters-beta/.

[35] Juan Benet. Ipfs-content addressed, versioned, p2p file system. *arXiv preprint arXiv:1407.3561*, 2014.

[36] Tamas Kocsis. *ZeroNet*, 2015 (accessed June 18, 2015). http://zeronet.io/.

[37] Ian Clarke, Oskar Sandberg, Brandon Wiley, and Theodore W Hong. Freenet: A distributed anonymous information storage and retrieval system. In *Designing Privacy Enhancing Technologies*, pages 46–66. Springer, 2001.

[38] Stefanie Roos, Benjamin Schiller, Stefan Hacker, and Thorsten Strufe. Measuring freenet in the wild: Censorship-resilience under observation. In *Privacy Enhancing Technologies*, pages 263–282. Springer, 2014.

[39] Krista Bennett, Christian Grothoff, Tzvetan Horozov, Ioana Patrascu, and Tiberiu Stef. Gnunet-a truly anonymous networking infrastructure. In *In: Proc. Privacy Enhancing Technologies Workshop (PET*. Citeseer, 2002.

[40] DR GAVIN WOOD. Ethereum: A secure decentralised generalised transaction ledger, 2014.

[41] Nick Szabo. Formalizing and securing relationships on public networks. *First Monday*, 2(9), 1997.

[42] Nick Lambert and Benjamin Bollen. The safe network: a new, decentralised internet. 2014.

[43] Qi Ma, Nick Lambert, and David Irvine. Safecoin: The decentralised network token. 2015.

[44] David Irvine. Maidsafe distributed hash table. 2010.

[45] Greig Paul, Fraser Hutchison, and James Irvine. Security of the maidsafe vault network. In *Wireless World Research Forum Meeting 32 (WWRF32)*, 2014.

[46] Ames Bielenberg, Lara Helm, Anthony Gentilucci, Dan Stefanescu, and Honggang Zhang. The growth of diaspora-a decentralized online social network in the wild. In *Computer Communications Workshops (INFOCOM WKSHPS), 2012 IEEE Conference on*, pages 13–18. IEEE, 2012.

[47] Robert W Gehl. Building a better twitter: A study of the twitter alternatives gnu social, quitter, rstat. us, and twister. *Forthcoming in Fibreculture*, 2015.

[48] Miguel Freitas. twister-a p2p microblogging platform. *arXiv preprint arXiv:1312.7152*, 2013.

[49] Burak Nehbit. *Aether*, 2014 (accessed June 15, 2015). `http://getaether.net/`.

[50] Dor Konforty, Yuval Adam, Daniel Estrada, and Lucius Gregory Meredith. Synereo: The decentralized and distributed social network. 2015.

[51] Brian Hoffman. *Open Bazaar*, 2015 (accessed July 1, 2015). `https://openbazaar.org/`.

[52] Adam Back. Hashcash, 1997.

[53] Nick Szabo. Shelling out–the origins of money, copyright by n. *Szabo, available online at http://szabo. best. vwh. net/shell. html*, 2005.

[54] David S Evans. Economic aspects of bitcoin and other decentralized public-ledger currency platforms. *University of Chicago Coase-Sandor Institute for Law & Economics Research Paper*, (685), 2014.

[55] Andreas M Antonopoulos. *Mastering Bitcoin: unlocking digital cryptocurrencies*. " O'Reilly Media, Inc.", 2014.

[56] Bela Gipp, Norman Meuschke, and André Gernandt. Decentralized trusted timestamping using the crypto currency bitcoin. *arXiv preprint arXiv:1502.04015*, 2015.

[57] Aaron Wright and Primavera De Filippi. Decentralized blockchain technology and the rise of lex cryptographia. *Available at SSRN 2580664*, 2015.

[58] Adi Shamir. Identity-based cryptosystems and signature schemes. In *Advances in cryptology*, pages 47–53. Springer, 1985.

[59] Hanno Böck. A look at the pgp ecosystem through the key server data. 2015.

[60] Ian Grigg. Pki considered harmful, 2008.

[61] Conner Fromknecht, Dragos Velicanu, and Sophia Yakoubov. A decentralized public key infrastructure with identity retention. Technical report, Cryptology ePrint Archive, Report 2014/803, 2014. http://eprint. iacr. org, 2014.

[62] Matthias Wachs, Martin Schanzenbach, and Christian Grothoff. On the feasibility of a censorship resistant decentralized name system. In *Foundations and Practice of Security*, pages 19–30. Springer, 2014.

[63] Frederic Jacobs. Providing better confidentiality and authentication on the internet using namecoin and minimalt. *arXiv preprint arXiv:1407.6453*, 2014.

[64] Christina Garman, Matthew Green, and Ian Miers. Decentralized anonymous credentials. *IACR Cryptology ePrint Archive*, 2013:622, 2013.

[65] Liang Wang and Jussi Kangasharju. Measuring large-scale distributed systems: case of bittorrent mainline dht. In *Peer-to-Peer Computing (P2P), 2013 IEEE Thirteenth International Conference on*, pages 1–10. IEEE, 2013.

[66] Ion Stoica, Robert Morris, David Karger, M Frans Kaashoek, and Hari Balakrishnan. Chord: A scalable peer-to-peer lookup service for internet applications. *ACM SIGCOMM Computer Communication Review*, 31(4):149–160, 2001.

[67] Sylvia Ratnasamy, Paul Francis, Mark Handley, Richard Karp, and Scott Shenker. A scalable content-addressable network. *ACM SIGCOMM Computer Communication Review*, 31(4):161–172, 2001.

[68] Antony Rowstron and Peter Druschel. Pastry: Scalable, decentralized object location, and routing for large-scale peer-to-peer systems. In *Middleware 2001*, pages 329–350. Springer, 2001.

[69] Quanquan Liu, Qian Long, Pratiksha Thaker, and Wenting Zheng. Speedy: a sybil-resistant dht implementation. 2014.

[70] Mahdi N Al-Ameen and Matthew Wright. Persea: a sybil-resistant social dht. In *Proceedings of the third ACM conference on Data and application security and privacy*, pages 169–172. ACM, 2013.

[71] Chris Lesniewski-Lass and M Frans Kaashoek. Whanau: A sybil-proof distributed hash table. NSDI, 2010.

[72] Benjamin Fabian and Tobias Feldhaus. Privacy-preserving data infrastructure for smart home appliances based on the octopus dht. *Computers in Industry*, 65(8):1147–1160, 2014.

[73] Ching-man Au Yeung, Ilaria Liccardi, Kanghao Lu, Oshani Seneviratne, and Tim Berners-Lee. Decentralization: The future of online social networking. *W3C Workshop on the Future of Social Networking Position Papers*, 2:2–7, 2009.

[74] Leucio Antonio Cutillo, Reik Molva, and Thorsten Strufe. Privacy preserving social networking through decentralization. In *Wireless On-Demand Network Systems and Services, 2009. WONS 2009. Sixth International Conference on*, pages 145–152. IEEE, 2009.

[75] Leucio Antonio Cutillo, Refik Molva, and Melek Önen. Safebook: A distributed privacy preserving online social network. In *World of Wireless, Mobile and Multimedia Networks (WoWMoM), 2011 IEEE International Symposium on a*, pages 1–3. IEEE, 2011.

[76] Arvind Narayanan, Vincent Toubiana, Solon Barocas, Helen Nissenbaum, and Dan Boneh. A critical look at decentralized personal data architectures. *arXiv preprint arXiv:1202.4503*, 2012.

[77] Thomas Paul, Antonino Famulari, and Thorsten Strufe. A survey on decentralized online social networks. *Computer Networks*, 75:437–452, 2014.

[78] Simon Thiel, Mohamed Bourimi, Rafael Giménez, Simon Scerri, Andreas Schuller, Massimo Valla, Sophie Wrobel, Cristina Frà, and Fabian Hermann. A requirements-driven approach towards decentralized social networks. In *Future Information Technology, Application, and Service*, pages 709–718. Springer, 2012.

[79] Laura Panades-Estruch Lieve Van Woensel, Geoff Archer. Ten technologies which could change our lives: Potential impacts and policy implications. *European Parliament, Directorate-General for Parliamentary Research Services*, 2015.

[80] P Brody and V Pureswaran. Device democracy: Saving the future of the internet of things. *IBM Global Business Services Executive Report*, 2014.

[81] Roxana Geambasu, Tadayoshi Kohno, Amit A Levy, and Henry M Levy. Vanish: Increasing data privacy with self-destructing data. In *USENIX Security Symposium*, pages 299–316, 2009.

[82] Rajesh Sharma and Anwitaman Datta. Supernova: Super-peers based architecture for decentralized online social networks. In *Communication Systems and Networks (COMSNETS), 2012 Fourth International Conference on*, pages 1–10. IEEE, 2012.

[83] Alex Leavitt. This is a throwaway account: Temporary technical identities and perceptions of anonymity in a massive online community. In *Proceedings of the 18th ACM Conference on Computer Supported Cooperative Work & Social Computing*, pages 317–327. ACM, 2015.

[84] Paul Ohm. The rise and fall of invasive isp surveillance. *University of Illinois Law Review*, pages 08–22, 2009.

[85] Corey A Ciocchetti. The eavesdropping employer: A twenty-first century framework for employee monitoring. *American Business Law Journal*, 48(2):285–369, 2011.

[86] Susan Landau. Making sense from snowden: What's significant in the nsa surveillance revelations. *IEEE Security & Privacy*, (4):54–63, 2013.

[87] Brad Miller, Ling Huang, Anthony D Joseph, and J Doug Tygar. I know why you went to the clinic: Risks and realization of https traffic analysis. In *Privacy Enhancing Technologies*, pages 143–163. Springer, 2014.

[88] D Anstee, A Cockburn, and G Sockrider. Worldwide infrastructure security report. Technical report, Technical report, Burlington, MA, USA, 2014.

[89] Sue Curry Jansen and Brian Martin. The streisand effect and censorship backfire. 2015.

[90] Jeffrey Rosen. The right to be forgotten. *Stanford law review online*, 64:88, 2012.

[91] Jonathan Zittrain, Kendra Albert, and Lawrence Lessig. Perma: Scoping and addressing the problem of link and reference rot in legal citations. *Legal Information Management*, 14(02):88–99, 2014.

[92] Karim El Defrawy, Minas Gjoka, and Athina Markopoulou. Bottorrent: misusing bittorrent to launch ddos attacks. In *Proceedings of the 3rd USENIX workshop on Steps to reducing unwanted traffic on the internet*, pages 1–6. USENIX Association, 2007.

[93] Nick Hofstede and Nick Van den Bleeken. Using the w3c webcrypto api for document signing. In *WASH*, pages 10–16, 2013.

[94] David Mazieres, Dan Boneh, Quinn Slack, Mike Hamburg, Andrea Bittau, and Mark Handley. Cryptographic protection of tcp streams (tcpcrypt). 2014.