Master's thesis

NTNU
Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Computer Science

Håvard Anda Estensen

Towards Better User Privacy with Decentralization

Master's thesis in Computer Science
Supervisor: Svein Erik Bratsberg
June 2019

NTNU
Norwegian University of
Science and Technology

Håvard Anda Estensen

Towards Better User Privacy with Decentralization

Master's thesis in Computer Science
Supervisor: Svein Erik Bratsberg
June 2019

Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Computer Science

Summary

User privacy has been put into the limelight after numerous scandals where passwords and other personal user data have been leaked. Europe's General Data Privacy Regulation, which took effect in 2018, addresses some of these issues but makes it not so straightforward to use a blockchain to store personal data. The right to be forgotten dictates that a user can request a company to delete all their personal data, while the blockchain is an immutable data structure that does not support deletion. This thesis explores how companies can move storage to the edge, away from centralized databases, while still being able to aggregate data and use machine learning without compromising privacy. The main contribution is a caching mechanism for linkable ring signatures that enables orders of magnitude faster authentication while providing the same privacy guarantees as standard linkable ring signatures.

Sammendrag

Brukeres personvern har blitt satt i rampelyset etter mange skandaler der passord og andre personlige brukerdata har blitt lekket. Europas nye personvernlov (GDPR), som trådte i kraft i 2018, tar opp noen av disse problemene, men gjør det ikke så enkelt å bruke en blokkkjede til å lagre personopplysninger. Retten til å bli glemt dikterer at en bruker kan be om at et selskap sletter alle personopplysninger de har om dem, mens blokkjeder er en uforanderlig datastruktur som ikke støtter sletting. Denne oppgaven undersøker hvordan bedrifter kan flytte lagring til kanten, vekk fra sentraliserte databaser, samtidig som de fortsatt kan samle data og bruke maskinlæring uten å ødelegge personvern. Hovedbidraget er en hurtigminne-mekanisme for linkbare ringsignaturer som muliggjør en størrelsesorden raskere autentisering samtidig som de gir samme personvern-garantier som linkbare ringsignaturer.

Preface

This project thesis is written for the Department of Computer Science (IDI) at Norwegian University of Science and Technology (NTNU). The research is conducted by Håvard Anda Estensen, under the supervision of Professor Svein Erik Bratsberg.

I want to thank Svein Erik Bratsberg for feedback on my ideas and writing, and for being flexible with scheduling meetings. Thank you for motivating me!

Table of Contents

List of Tables

List of Figures

Abbreviations

AI	=	Artificial Intelligence
AWS	=	Amazon Web Services
BFT	=	Byzantine Fault Tolerance
BTC	=	Bitcoin
DAG	=	Directed Acyclic Graph
DAO	=	Autonomous Organization
DH	=	Diffie-Hellmann
DHT	=	Distributed Hash Table
DoS	=	Denial-of-Service
DP	=	Differential Privacy
dPoS	=	Delegated PoS
DNS	=	Domain Name System
DSN	=	Decentralized Storage Networks
ECC	=	Elliptic Curve Cryptography
ETH	=	Ether
EU	=	European Union
EVM	=	Ethereum Virtial Machine
FBA	=	Federated Byzantine Agreement
GDPR	=	General Data Protection Regulation
HE	=	Homomorphic Encryption
IPFS	=	InterPlanetary File System
LSAG	=	Linkable Spontaneously Anonymous Group
ML	=	Machine Learning
P2P	=	Peer-to-peer
PII	=	Personally Identifiable Information
PoRep	=	Proof of replication
PoW	=	Proof-of-Work
PoS	=	Proof-of-Stake
RSA	=	Rivest Shamir Adleman
RTT	=	Round-Trip Time
SEP	=	Secure Enclave Processor
SGX	=	Software Guard Extensions
SNIP	=	Secret-shared Non-Interactive Proof
UTXO	=	Unspent Transaction Output
VM	=	Virtual Machine
ZK	=	Zero-Knowledge
ZK-SNARK	=	ZK Succinct Non-Interactive ARgument of Knowledge
ZK-STARK	=	ZK Scalable Transparent ARgument of Knowledge

Chapter 1

Introduction

1.1 Purpose

The modern web leaves no place for anonymity. Authentication is usually done with an email-password pair or social media login, which ties the user to a real-world person. Data about these users is stored in huge centralized databases, where leaks can affect millions of people at a time. The motive of this research is to investigate how we can improve user privacy with decentralization and distributed systems.

Blockchains pave the way to make the world decentralized but meet oppression from lawmakers and governors that want to maintain power. The General Data Protection Regulation (GDPR) affects how a blockchain can store personally identifiable information (PII). A blockchain creates a way to guarantee that the information is not tampered with while it is stored. However, storing personal data as plaintext on an immutable data structure is a direct violation of GDPR's "right to be forgotten", so privacy-enhancing techniques have to be applied to make the PII private before adding it to a blockchain.

GDPR imposes responsibilities on data controllers and data processors. It assumes that a single entity controls compute and storage, not a Decentralized Autonomous Organization (DAO) where no one can be held personally accountable for mistakes. Centralized storage is a recipe for misuse and data leaks. By decentralizing storage, we can have cheaper, faster, and more secure file storage. GDPR dictates where information can be stored. Blockchains are built for a global world that knows no national borders.

Presently, the most important use case for blockchain is digital money. However, storing value like cryptocurrencies on a blockchain is different than storing information. Cryptocurrency transactions can be compressed with off-chain solutions like zk-SNARKS to prove that a transaction happened, while for information storage, you sometimes need all the information from the transaction. With value storage, the cryptography can be upgraded so attackers cannot take control over your funds. However, encrypted data is only as secure as the encryption schemed first used to encrypt it and since anyone can create a copy of the encrypted data, it is meaningless to re-encrypt it with a stronger scheme later. This means that all publicly stored information will be exposed when quantum computers are made available. We have to make a trade-off between decentralization, privacy, and scalability.

People think cryptocurrencies are anonymous, but most of them are not. This thesis attempts to raise awareness of what information you expose about yourself interacting with cryptocurrencies by mapping the information you expose when buying cryptocurrencies. Exposing your social security number, passport number, or credit card number should not be necessary. You should only be required to prove your ID without providing it. At the same time, blockchains with strong privacy-guarantees like anonymous cryptocurrencies face stiff opposition from governments that demands transparency.

There are many public blockchains under active development, and most of them store data that can be considered personal. Because of its nature, most of these projects are global, so personal data is not only stored in countries with adequate privacy laws, as GDPR requires. To become genuinely global, blockchain projects have to adhere to the strictest privacy laws. The complexity of these systems makes them hard to reason about.

The research in this thesis is done through the eyes of a computer scientist, not a lawyer, and contains a technical interpretation of GDPR, not a legal one.

1.2 Research Questions

RQ1: How can we increase user privacy?

There has been much research lately about the security and privacy for public blockchains. Since there is a strong financial incentive to exploit these systems, they must adhere to even stricter requirements than private ones (although that is security by obscurity). The goal is to learn from these distributed systems and find novel ways to apply cryptographic techniques.

RQ2: Can we use a blockchain to store personally identifiable information?

Can we transform data so it can be stored on a blockchain? When is data sufficiently anonymized?

1.3 Thesis Structure

The thesis is structured into four parts. The State of Internet Privacy chapter describes what privacy is, defines personally identifiable information, explains how machine learning makes data more useful, and discusses use-cases for blockchains. The Technical Solutions to Obtain Privacy chapter introduces technical blockchain concepts and privacy-enhancing techniques needed to explain more complex topics later. The Experiment chapter investigates and analyses how storage can be moved towards the edge, and users are given control of their data. The Related Work and Conclusion chapter evaluates the related work and compares the experiments with the research goals.

Chapter 2

The State of Internet Privacy

This section explains what privacy is, how GDPR affects data collection and processing for European citizens, and how GDPR makes it hard to use blockchains and machine learning.

2.1 Privacy

“Privacy is the power to selectively reveal oneself to the world” – Eric Hughes in a Cypherpunk’s Manifesto[1]

2.1.1 Definition

Privacy is a human right[2]. It does not mean secrecy, a secret is something you do not want anyone to know, but it gives you the ability to control what can be accessed by others. This applies to governments and police, although some governments seem to care less about it.

If we want privacy, we must disclose as little as possible about ourselves when we transact with other parties. The other party should only know what is directly necessary for the transaction. Showing your drivers license to a bouncer not only reveal your age, but also your name and social security number and when you got the permission to drive. Now, this information is exposed. To achieve privacy, we need an anonymous transaction system where we can selectively reveal the information the other party requires. This requires cryptography, which will be described in Chapter 3.

In the digital age, we often reveal ourselves. When we buy something from a store and pay with a credit card, the purchase can be used to direct advertisement against us in the future. Before we handed cash to the cashier and remained anonymous, however, cash is going away.

2.1.2 I have nothing to hide

Some people might say that they do not care if their privacy is violated because they do not think they have anything to hide. This is a false premise. You do not need to justify why you need a human right. You cover your windows with curtains in your home not because you are doing anything immoral or illegal, but just because you do not want people to watch you. Even if you have nothing useful to say, does that mean free speech not import? Moreover, if you do not need your rights, that does not imply that others do not need them.

If payment information is stored on a computer or phone, a hacker can steal your money. If the door to your computer is open, a hacker can encrypt all your data and demand ransom[3, 4]. If you have written something critical about someone that can have consequences. By letting advertisers learn about you from your data, they will eventually learn to know you better than yourself and use that information to manipulate you.

2.1.3 Platforms Dictates the Rules of the Game

Data is now arguably the most valuable commodity in the world. Four out of the five most valuable companies in the world, Alphabet, Amazon, Facebook, and Microsoft[5] all play significant roles in our lives. They provide free services in exchange for user data and monetize with the advertisement. These platforms have been hugely successful in generating revenue at the expense of user privacy. They have sold data to third parties without being transparent about it[6, 7, 8]. However, it is hard to imagine a life without search, file storage, navigation, messaging, and online shopping. Power has become concentrated at these gatekeepers of the truth. They dictate the rules and companies using their platform has to obey them. Who can publish an app on the app store? What content should be censored? How much is an artist paid when you listen to their song? Platforms dictate the rules and can change them at any time[9].

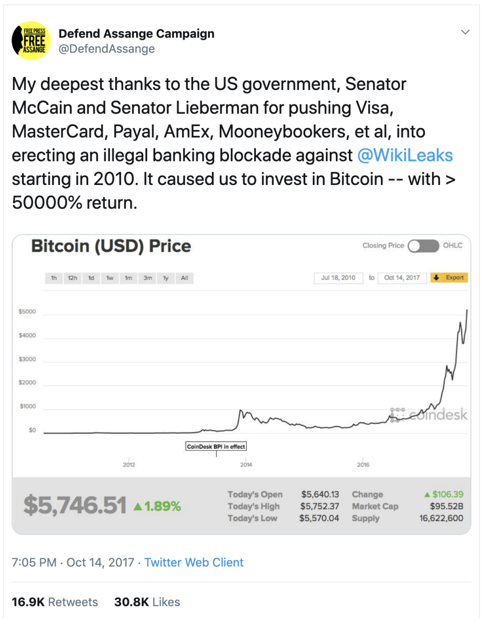
When building a platform for a global audience, local cultural views are often overlooked. The iconic photo of a naked 9-year old girl fleeing from napalm bombs was infamously censored from Facebook as it was flagged as nudity, but Facebook was later forced to restore it[10]. Should a single entity be allowed to decide what we are allowed to share on a global scale? To decide what can be considered free speech, and what is not? Fake news should be censored, but it is difficult to define. If President Trump says something untrue, should it be considered fake news? Most politicians never tell the whole truth. No news site is free of mistakes and biases, but some make an honest effort to find out the truth. One person might consider something fake news, while another consider it free speech. Tech giants are told to do more about fake news by the European Commission[11].

2.1.4 Censorship

There is both business and political risk to let a central entity manage data. It becomes easier for countries to suppress free speech, like China's blocking of Wikipedia[12], Egypt blocking Tor[13] and Venezuela blocking cryptocurrency exchanges[14]. Surveillance can catch criminals, but who will overthrow a corrupt regime if the regime is totalitarian?

In 2011 the US cut WikiLeaks off from their funding for being whistleblowers[16]. As 97% of the global payment market was blocked, it became harder to donate, and the blockade was without democratic oversight and transparency[17]. Funny enough, this lead WikiLeaks to accepting cryptocurrencies as donations and making millions shown in Figure 2.1. Still, whistleblowing has consequences, and you should be able to remain anonymous. It is not sufficient to get legal protection if you risk someone slipping some plutonium in your morning coffee.

Governments want more control, and it seems like we are heading towards an Orwellian world. France has suggested a ban on privacy-oriented cryptocurrencies[18]. The UK wants to ban encryption[19]. Australia forces tech companies to add a backdoor in the software, so they can access data even when it is end-to-end encrypted[20].



Defend Assange Campaign
@DefendAssange

My deepest thanks to the US government, Senator McCain and Senator Lieberman for pushing Visa, MasterCard, Payal, AmEx, Mooneybookers, et al, into erecting an illegal banking blockade against @WikiLeaks starting in 2010. It caused us to invest in Bitcoin -- with > 50000% return.

7:05 PM · Oct 14, 2017 · Twitter Web Client

16.9K Retweets 30.8K Likes

Figure 2.1: Consequence of the banking blockade of WikiLeaks[15]

2.2 GDPR

2.2.1 Definition

The General Data Protection Regulation[21] (GDPR) is a legal framework that provides guidelines to companies for how personal information can be collected and processed for individuals that live in the European Union (EU). This includes companies not based in the EU. GDPR was put into effect the 25th of May, 2018 and promise more transparency, accountability, and fines up to 4% of the annual turnover.

2.2.2 Personal Identifiable Information

Personal Identifiable Information (PII) is personal data that can directly or indirectly identify a person (data subject). Examples are names, date of birth, email address, IP address, or location. PII is used to create better and more personalized services but is also sold to third parties to create targeted advertisements. GDPR does not apply to anonymized data, but it is hard to tell if data is truly anonymous. It is difficult to know the consequence of indirectly linked data when gathering data. Although not specified in the GDPR, it is later stated in WP29 that data that is indirectly given through observations of your activities, like access logs, also is considered as PII[22].

GDPR encourages pseudonymization of data. Pseudonymisation is the process of removing identifiers, so data no longer is directly identifying. This reduces the risk of data processing while retaining the usefulness of the data. GDPR is more relaxed about pseudonymized data, so it allows it to be used for other purposed than the original collection purpose. It also gives leeway to process PII for scientific, historical, and statistical purposes.

2.2.3 Data controllers

Data controllers are entities that dictate how and why PII is going to be used by the organization. A data controller can process the collected data or work with a third party. For PII to be processed, the data subject has to give explicit consent, and it is the responsibility of the data controller to collect the consent. To ensure a transparent process, the data controllers must create a privacy policy that outlines:

- What data is collected
- How the data is stored
- How the data is used
- Whom the data is shared with
- When and how data is deleted

Sometimes data is stored on servers in different countries. Some countries, like Argentina, Switzerland, and New Zealand, have adequate privacy laws, and data can be transferred freely[23]. Others do not, and here, data cannot be transferred without the data subject explicitly giving consent after having been provided with the associated risks. The US has ”partial” adequacy since they do not have a general data protection law; however, companies that participate in the Privacy Shield Framework[24] are allowed to transfer data into the US.

Data controllers must be able to prove that they are compliant with the GDPR principles. This is referred to like the accountability principle.

2.2.4 Data processors

Data processors process data on behalf of the data controller. If the controller and the processor are not the same, they have to have a contract that dictates for example, what happens with the data if the contract is terminated. In general, data processors have less responsibility than data controllers.

2.2.5 The Right to Data Portability

A data subject has the right to request PII about them in a structured format so they can without hindrance transmit those data to another controller. It is not the data controllers obligation to transmit the data in a compatible format to the new data controller, but the data has to be a commonly used machine-readable format.

2.2.6 The Right to be Forgotten

A data subject has the right to ask the data controller to delete PII concerning them. However, the data subject does not have an absolute right to erasure. If the data is anonymized, it no longer counts as PII.

This also applies to search engines. Two UK businessmen demanded Google delete their criminal conviction from its search engine results[25]. Google first refused, but the London High Court ruled in favor of one of the criminals because he expressed genuine remorse. The other man's request was rejected because he had not accepted his guilt and was likely to repeat his offense.

2.2.7 The Right to Explanation

The right to explanation has become a controversial subject in GDPR[26]. GDPR states that data subjects should have ”meaningful information about the logic involved” in automated decision-making processes (article 15h), but does not further specify what this entails. Data subjects should also be able to opt out of automated processing and have the right to a human intervention, where they can express their point of view and challenge the decision (recital 71).

2.3 Data Mining

Data mining is the process of turning raw data into useful information. Learning about data patterns can help businesses develop more effective strategies, decrease costs, and increase revenue. By continuously analyzing data, a company can automate decision-making without waiting for human judgment. Insurance companies can detect fraud, stores can forecast demand, and farmers can learn how much water is optimal for a tomato plant. The strategy for many companies has been to hoard data, expecting it will be useful in the future. It is challenging to handle the volume, variety, veracity, and velocity of the data. Data mining quickly becomes slow and expensive.

2.3.1 Artificial Intelligence

Artificial Intelligence (AI) is a field of computer science where machines learn from data to make intelligent decisions. AI represents a breakthrough for almost all applications as it allows for unprecedented precision and recall in automation. At a high level, one of the big goals is to create general intelligence[27]. To be able to perform any intellectual task like a human.

Machine learning (ML) is a subset of AI, where statistical methods and algorithms are used to perform a task without explicit instructions, like spam filtering, but also life-saving ones like a cancer diagnosis. Machine learning trains a model to predict some output for a given input. The algorithms run on the data and capture patterns from the data and transfer the learning to the model. ML is classified depending on how models are trained and can be broadly classified into supervised learning, unsupervised learning, and reinforcement learning.

Supervised learning requires a lot of training data with labels. For spam filtering emails would need to be labeled as ”spam” or ”not spam,” and after seeing many samples, the algorithms will eventually figure out what characteristics make an email likely to be spam. To verify that the model is a precise prediction, some of the training data is put aside and only used for verification. If the model is too similar to the training data, it is considered overfitted, and will only perform well on data that is precisely the same as the training data. However, this is no good, and we want to generalize the model enough so it can handle similar, but unknown input. Showing one picture of a dog is not sufficient for it to learn what a dog is.

Unsupervised learning can learn from data without labels and use clustering to evaluate how different samples are. Data like pictures can sometimes be very rich and can hardly be described with a single label (e.g., dog). Unsupervised learning attempts to learn everything that can be learned about the data without a particular task in mind; to learn for the sake of learning.

Reinforcement learning allows an agent to learn by trial an error by interacting with the environment instead of inspecting data. Feedback is given by using rewards and punishments as signals for positive and negative behavior. The agent seeks to maximize the reward and minimize its penalty. The advantage of reinforcement learning is that you are not dependent on data or manual feature engineering.

2.3.2 The Cost of ML

It is hard to get an overview of what companies use your PII for. Personal data is creating better and more personalized services, but it can also be used for evil. Cambridge Analytica harvested in 2014 PII from 50M Facebook users and used it to target them with personalized political advertisements before the 2016 US election[28].

It is clear that tracking your opinions and political stance can give you trouble when you apply for a new job. However, trivial things are also essential to keep private. Tracking toilet breaks tracks your health. Tracking if you smoke an insurance company can decide you need to pay more for insurance. Tracking your mouse movement might spot Parkinson's before you do, and you might be denied insurance.

Training with biased data also leads to biased decision-making. Face recognition systems fed fewer photos of dark-skinned people is worse at recognizing dark-skinned faces[29]. Amazon's internal recruitment tool discriminated female candidates because their historical hiring decisions favored men over women[30].

2.3.3 Explainable AI

Biased AI is hard to fix because machine learning is not transparent. Machine learning today is a black box of magic, and it is hard to identify bias before later, and fixing it retroactively is not trivial. This problematic in combination with GDPR's "right to explanation." A company must be able to give you a better explanation for why they denied your insurance than "the algorithms say so."

2.4 Blockchain

Blockchains are a type of distributed ledgers technology that offers a way to keep records of transactions. Cryptography enables blockchains to run without the control of a central authority because everyone participating can audit the history.

Blockchains have similar motivations as GDPR in that power should not be centralized at a few large actors. However, GDPR still assumes that power is still centralized but at smaller actors. Blockchains challenges this centralized world and aims to replace centralized entities all together with decentralized ones. The challenge is, how can a decentralized world be regulated?

2.4.1 Decentralization

Paul Baran's diagram shown in Figure 2.2 from "On Distributed Communications Networks"[31] is often used to explain the difference between decentralization and distributed systems. Older literature uses different definitions for decentralized and distributed that we use today and provide a lot of confusion[32, 33, 34]. With our modern definition, decentralized and distributed switch places.

Decentralized systems have no central point of control. Distributed systems use messages to coordinate actions across components to achieve a common goal. Decentralization can

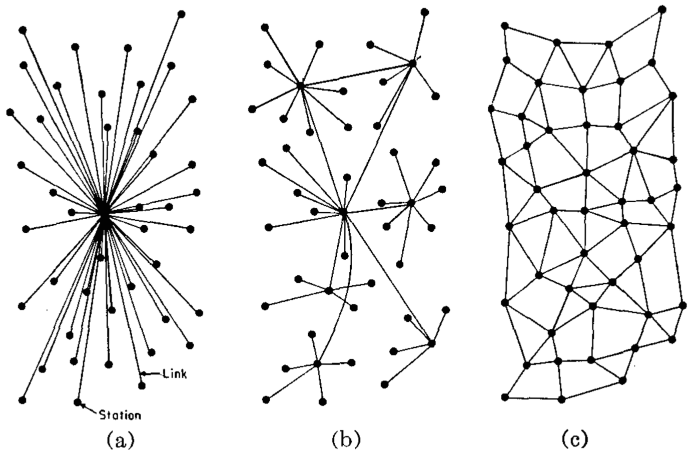


Figure 2.2: Degree of Decentralization: (a) Centralized. (b) Decentralized. (c) Distributed networks[31]

be further divided into three categories.

- Architectural decentralization
- Political decentralization
- Logical decentralization

Architectural decentralization is how physically decentralized the system is. How many computers can fail before the system breaks down? Political decentralization is how decentralized governance is. How many individuals control the system? Logical decentralization is how monolithic the system is. If the system is cut in half, will the two halves be able to operate independently?

When centralized companies grow, they make their services more valuable for users. After reaching the top of their growth curve, they start optimizing for profit and stops adding value for users. For users to get more value, we need there to be another company for competition. Without competition, we end up in a monopolistic state. A decentralized non-profit organization never end up in a state where they optimize for profit; they optimize to have users keep using the project. Projects that have a token have a built-in incentive for maintainers and developers to improve the project.

Decentralized systems also protect the system from failing, either by accident or by malicious intent. They are fault tolerant, so they are less likely to fail when one component fails. Attacks on the systems are more expensive to carry out as they lack sensitive central points to be attacked at a lower cost than the economic gain. It is harder for participants to collude by doing coordination we do not like.

2.4.2 Cryptocurrencies

Cryptocurrencies are digital currencies, secured with cryptography and a blockchain. Distributed consensus replaces a central authority and creates trust where there before could be none without a trusted third party. Not trusting a third party makes cryptocurrencies less vulnerable to censorship.

Digital payments are not private. When you purchase something with your credit card, your bank will learn what you bought, where you bought it and how much you paid for it. Cryptocurrencies can also be used in a traceable way if, for example, addresses are reused[35]. This puts the burden of ensuring privacy over on the user.

2.4.3 Third Parties are Expensive and Slow

An important use case for cryptocurrencies is remittance. Remittance is money sent to support family members in another country. If your bank is not connected to the other bank, you have to use a money transfer service, which is expensive and slow. The World Bank estimates that $689 billion was sent as a remittance in 2018 and that the global average cost for remittance in Q1 2019 to be 6.94%[36]. Sub-Saharan Africa is the most expensive region to send money to, with a total average cost of 9.25%. These are some of the poorest people in the world and the people who can least afford it. Cryptocurrencies offer a way to send money globally with low fees instantaneously. The World Food Programme tested Ethereum to send aid money to more than 100,000 refugees and found that they save more than $40,000 per month in bank transfer costs[37].

2.4.4 Open the world for more cooperation

Today's centralized systems store vast amounts of data in silos. The interoperability of data between these platforms is minimal. The vendor lock-in does not allow for sharing data like pictures between platforms. Why is it not even possible to use one chat program to communicate with everyone?

The free, open-source social network Mastodon[1] try to address data siloing by encouraging people to host server instances and allowing for communication between the instances, making data portable between services. However, hosting instances requires technical know-how and is arguably only accessible to the technocratic elite. Mastodon also ends up not being as decentralized as its vision as most users only join the top 5 instances[38].

Payment is not a solved problem. There are many currencies and payment systems. They speak different languages, and they have a hard time understanding each other. Moving money is slow and expensive. International wire transfers take multiple days to clear and

[1]https://joinmastodon.org/

move through multiple intermediaries that each take a cut. Streaming payments cannot work with today's infrastructure as the fees are too high. Networks like Ripple[2] and Stellar[3] can handle this by integrating with other payment systems so you can send money cheaply between countries.

[2] https://ripple.com/xrp/
[3] https://www.stellar.org/

Chapter 3

Technical Solutions to Obtain Privacy

This section explores the challenges of preserving privacy when aggregating data. We start by explaining the disadvantages of using the cloud and argue that users should be able to decide where their data is stored. We then define cryptographic primitives and blockchain for building decentralized, private, and secure services. Lastly, we discuss how we can use these techniques to decouple storage from applications.

3.1 Local-first Software

3.1.1 Slow Software

When opening an app, it sometimes takes seconds to load data before it is displayed. A spinner icon is displayed instead, and the app feels slow. The multiple round-trips the app has to do to fetch data is not all of the perceived slowness. The operating system, the garbage collector in the runtime and latency from keyboards all add up total response time. If it takes more than 100ms from user input to the response, the response will not feel immediate[39]. Overall response time also impacts the bottom line. AWS found that a 100ms increase in response time decreased their revenue with 1%[40]. When you store data in the cloud, it becomes tough to create a fast app, and this provides motivation to store user data locally and synchronize data in the background. It is often useful to access data from the service, even if it might be slightly out of date.

	C	O	V	I	Si	T	Se	Sy	SP
C	0(1)	22(2)	65(13)	136(5)	189(12)	113(5)	142(12)	159(2)	185(11)
O	22(2)	1(1)	88(14(	125(2)	166(13)	101(11)	131(13)	178(3)	182(11)
V	65(13)	88(14)	1(16)	73(13)	220(22)	156(16)	179(29)	219(13)	121(16)
I	136(5)	125(2)	73(13)	0(0)	180(18)	211(10)	233(14)	301(5)	185(12)
Si	189(11)	166(12)	220(22)	180(17)	1(9)	68(8)	97(13)	169(8)	329(21)
T	133(5)	101(11)	156(18)	211(10)	68(9)	0(3)	32(9)	104(2)	263(15)
Se	142(9)	131(13)	179(20)	233(13)	97(13)	32(10)	1(9)	133(8)	290(16)
Sy	159(2)	178(3)	219(12)	301(5)	169(10)	104(2)	133(8)	1(0)	338(11)
SP	185(13)	182(12)	121(17)	185(13)	329(23)	263(16)	290(18)	338(14)	1(11)

Table 3.1: Average measured RTT between AWS data centers in milliseconds with standard deviation in parentheses

To commit data to a global data store, it is not sufficient to send the data to the closest data center[41]. Data needs to be replicated to multiple data centers, and this will often require multiple round-trips to other data centers. Table 3.1 shows the average measured Round-Trip Time (RTT) between AWS data centers in California (C), Oregon (O), Virginia (V), Ireland (I), Singapore (Si), Tokyo (T), Seoul (Se), Sydney (Sy), and São Paulo (SP). To achieve consensus between the three data centers that are closest together (C, O, V) we are already at more than 100ms.

3.1.2 The Cloud in a Support Role

When we store data in the cloud, we do not have ownership of the data, even though we created it. If the cloud is unavailable, we often cannot use the software. If the service decides to shut down, there might not be an easy way for us to export the data and run the service locally. Stringify[42], Google Reader[43], and Mailbox[44] are just a couple of examples.

For apps with cloud storage, the clients at best cache data and must send all modifications to the cloud, or they did not happen. Reversing this order, we make the client the primary and the cloud secondary for backup and synchronization assistance. Using the cloud as a backup has two significant advantages. First, only storing data locally is not redundant enough for data like family photos. It is also not sufficient to back up to multiple devices if they are all in the same house. What if the house burns down? Second, the primary device might not always be online or might only be connected to a mobile network. The cloud makes synchronization possible and limits mobile data usage.

3.2 Cryptographic Primitives

To create a private and secure system, we need two basic primitives. We need to be able to hide data from others, and we need to be able to prove that the data is ours. This is done with encryption and digital signatures. Both of these rely on cryptographic hash functions.

3.2.1 Hash functions

A hash function maps data of arbitrary size to data of a fixed size. Hash functions are deterministic - meaning an input must always give the same hash value. Because the input data might be larger than the output data, there will be inputs that map to the same output (pigeonhole principle), but collisions should be rare. A good hash function is uniform in the sense that every output hash value should be generated with about the same probability.

3.2.2 Cryptography

Cryptography is the study of techniques to prevent a third party or the public from reading our messages. The un-encrypted message, called plaintext, is converted into ciphertext with an encryption algorithm to allow for secure communication.

3.2.3 Cryptographic hash functions

A cryptographic hash function is a deterministic function H that maps a message M to a small fixed-size output $H(M)$:

$$H(M_1) = H(M_2), M_1 \neq M_2 \tag{3.1}$$

Not only should collisions be rare, but it should not be practical to generate these. Sometimes cryptographic hash functions are broken, like SHA-1 was broken in 2017, but remain widely used[45].

3.2.4 Symmetric-key Encryption

Symmetric-key encryption uses the same key for encryption and decryption. This requires that the parties that are communicating have to exchange a secret before they start communicating. Exposing the symmetric key affects both confidentiality and authentication. An unauthorized person can not only decrypt messages but also encrypt new messages and send them back to the parties that were originally communicating.

$$C = E_k(M) \tag{3.2}$$

$$M = E_k^{-1}(C) \tag{3.3}$$

We define E_k to be our encryption algorithm that uses the key k. We use E_k to encrypt a plaintext M to a ciphertext C. We observe that the decryption algorithm is the inverse of E_k and that it uses the same key.

3.2.5 Diffie-Hellman Key Exchange

A Diffie-Hellman (DH) Key Exchange is a way to exchange cryptographic keys over a public channel. The original protocol uses a multiplicative group of integers modulo p, where p is prime and a generator g that is a primitive root modulo p.

p and g are defined in public. Alice chooses a secret a and sends $A = g^a \mod p$ to Bob. Bob chooses a secret b and sends $B = g^b \mod p$ to Alice. Alice computes $B^a \mod p$. Bob computes $A^b \mod p$.

$$(g^a \mod p)^b \mod p = (g^b \mod p)^a \mod p \tag{3.4}$$

Alice and Bob will arrive at the same value, but an attacker Eve will not because of the discrete logarithm problem.

3.2.6 Public-key Cryptography

Modern cryptography uses the idea that you can make the key used to encrypt your data public, while the key used to decrypt data is kept private. Because the keys are different, we call the encryption schemes asymmetric, and because we can expose one of the keys to the public, we call the systems public key cryptography systems. The most commonly used of these are RSA[46], which is named by the inventors of the algorithm: Ron Rivest, Adi Shamir, and Leonard Adleman. There is a big advantage of not having to do a secret key exchange before starting communication, but it comes at the cost of about 9x key length[47].

Algorithms like RSA are based on Trapdoor one-way permutations. Essentially, this means two algorithms where one is easy to compute, and one is hard.

To let Bob send her encrypted data Alice starts by creating a secret key $S_k = d_k$ and derive the corresponding P_k public key from it $P_k = (n_k, e_k)$, which specifies the trapdoor one-way permutation f_k of $\mathbb{Z}_{n_k}$:

$$f_k(x) = x^{e_k} \pmod{n_k} \tag{3.5}$$

We assume that only Alice knows how the inverse permutation f_k^{-1} can be computed efficiently. This is the original Diffie-Hellman model.

$$C = M^{e_k} \pmod{n_k} \tag{3.6}$$

Bob produces a ciphertext C by encrypting a message M with Alices' public key (n, e).

$$M = C^{d_k} \pmod{n_k} \tag{3.7}$$

Alice can easily decrypt C with her secret key d_k, but an attacker Eve would, in the worst case, have to factor n_k to compute the private key from the public key, which is very computationally hard. However, more efficient factoring algorithms like the General Number Field Sieve and the Quadratic Sieve have been moderately successful than the naïve approach of guessing pairs of known primes. This leads to longer keys and is unsustainable for mobile and low-powered devices with limited computational power. We need a better trapdoor function.

3.2.7 Elliptic Curve Cryptography

Elliptic Curve Cryptography[48] (ECC) is gaining traction over RSA and DH because it offers smaller key sizes and more efficient implementations while providing the same level of security. The advantage of EC over RSA is the key size. 256 bit ECC provides the security of a 3072 bit RSA/DH and computations are done approximately 10 times faster[47]. There have also been no compromising attacks on ECC since when it introduced in 1985.

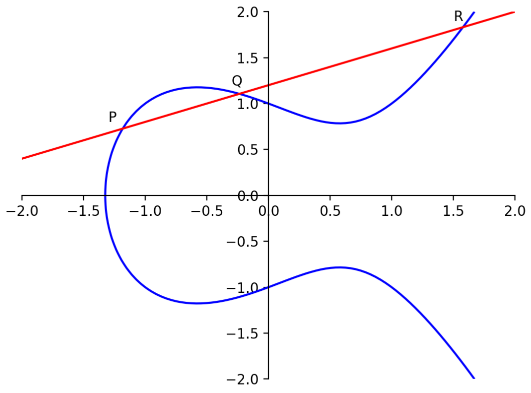


Figure 3.1: Point addition on Elliptic Curves (P + Q = R)

An Elliptic Curve (EC) is a curve defined by an equation on the form $y^2 = x^3 + ax + b$, with no cusps or self-intersections. The curve is symmetric over the x-axis, and a line through two points on the curve will intersect the curve exactly one more place. To move around on the EC points are added together to produce a third point, as shown in Figure 3.1. The addition is done n times to end up at some point on the curve. The Elliptic curve discrete logarithm problem is that to compute the number of addition it takes to get to that point we actually have to do additions until we find a matching point. To illustrate this, one can imagine a game of billiards. A ball starts at some point and is bounced around until it ends up at some other point on the curve. Someone observing would easily be able to count the number of times the ball bounced the wall. But someone not observing would have to redo the billiard game from start to finish.

3.2.8 Digital Signatures

To have trust in a system, we need authentication, integrity, and non-repudiation. Authentication verifies that a message comes from a certain user. Integrity verifies that the content has not been altered. This is to avoid a man-in-the-middle attack and random corruption. Non-repudiation is proof that someone has signed a document. If someone signs a document, they cannot later deny that they have signed it.

Whole messages are not signed, the hash of the message is signed instead for efficiency, compatibility, and integrity. Because of this, digital signatures rely on a cryptographically secure hash function.

3.2.9 Secret Sharing

Secret sharing is methods to distribute a secret to a group of participants. Each participant is allocated s, a share of the secret. The secret can only be reconstructed when sufficient numbers of shares are combined.

Trivial secret sharing

$$s \oplus s_1 \oplus s_2 \oplus \cdots \oplus s_{n-1} = s_n \tag{3.8}$$

Create $n - 1$ random numbers (a_1 to s_{n-1}) and XOR them with the secret. The random numbers and the result of the operations are the shares.

$$s = s_1 \oplus s_2 \oplus \cdots \oplus s_n \tag{3.9}$$

The weakness in this scheme is that it requires all shares to recover the secret. If one share is corrupted or lost, the secret is gone. Shamir's scheme provides a way to recover the secret with t out of n shares[49]. However, this weakens privacy or requires more participants.

Secret sharing works well for storage but exposes the secret to someone when we want to do useful operations on the secret. This someone is often a service provider that we do not want to trust.

3.2.10 Homomorphic Encryption

Homomorphic Encryption[50, 51] (HE) allows computations on ciphertext where the result when decrypted would match the result if the operations had been done on the plaintext. This allows privacy to be preserved when users share sensitive data.

An example of HE is a service provider that shares a public key with some users. The users encrypt their data and share their encrypted data with the service provider. The service provider does computations, like adding the encrypted user data and decrypts the result. The result is then shared with everyone.

Using HE does not mean that information cannot leak. If the service provider is dishonest, they can decrypt the user's data straight away. If an attacker controls the network, they can control $n - 1$ inputs and differentiate the result to extract a user's information. HE is malleable, so an attacker can transform the ciphertext even though they do not know what it decrypts to.

HE is computationally expensive, so anonymizing data and storing that data as plaintext might be more suited for data aggregation.

3.2.11 Blinding

Blinding is a technique where an agent can provide a service without knowing the real input or the real output. This can prevent side-channel attacks like if an attacker tries to recover information by measuring computation time or power consumption.

An example of blinding is that Alice has an input , x and Bob has a function f. Alice wants Bob to compute $y = f(x)$ without revealing y or x. Alice cannot compute y herself because she might not have enough resources, or she might not know f. By encoding x with some other input $E(x)$ that is a bijection of the input space of f, Alice blinds the message. Alice then gives the encoded message to Bob and Bob computes $f(E(x))$ for her. Alice can decode the message by applying D and obtaining $y = D(f(E(x)))$

3.2.12 Zero-knowledge proofs

Usually, we expose more information about ourselves than necessary when we share personal information. To get into a nightclub, we often show the bouncer our drivers license, which contains our birthday, which our age can be derived from. There is also a picture so the bouncer can be sure that the ID belongs to us. But not only that, the license contains more information like our social security number, which can be memorized and misused. The bouncer has no business knowing this about us, and we would like a system which allows us to reveal information about ourselves selectively.

Exposing too much information is also a problem for web applications. For example, if a client wishes to log into a web server, it often has to prove that it knows the password. Most often the password is transmitted in cleartext over a secure channel. The server then hashes the transmitted password and compares it with the stored password hash. The server, therefore, knows the cleartext password, and we are reliant on the fact that the server is not compromised.

Having one complex password is not sufficient because they are frequently leaked[52]. It would be safer if servers only stored salted hashes, but many do not follow these best practices. Re-using password exposes users to many risks. Because we cannot trust other parties, we want to have a system where we can prove that we know the password without revealing the password itself.

Zero-Knowledge (ZK) proofs allows us to prove possession of certain information without revealing the information itself. The first efficient examples of ZK proofs were mentioned in 1985 by the cryptographers Shafi Goldwasser, Silvio Micali, and Charles Rackoff[53]. The MIT researchers proposed an interactive and probabilistic proof system where no information is leaked from the proof, except that it is correct (sound).

To illustrate this, imagine that you are colorblind and a friend has to pens which she claims are different colors, but otherwise identical. But you cannot tell the difference. Your friend wants to prove that the pens are different colors without revealing their color. She gives the pens to you and asks you to hold one in each hand and put them behind your back. You are then allowed to shuffle them, or not. Then presenting the pens to your friend, she has to guess if you shuffled. She has a 50% chance of guessing correctly. But doing this multiple times the probability of her cheating will become very small. For an input of n-bit, the probability of cheating can be generalized to $\frac{1}{2^n}$. Because this proof applies to graph coloring, it can be generalized to all NP-C problems.

However, the problem with interactive constructions like this is the latency it would introduce in a vast network where nodes might be thousands of kilometers from each other. It would also not be efficient to send many messages for each proof. For cryptocurrencies, interaction is also a problem. However, zero-knowledge is still a very desirable feature. You want to be able to prove that a transaction took place without revealing the amount sent or the parties involved.

3.2.13 ZK-SNARKS

Zero-Knowledge Succinct Non-interactive ARgument of Knowledge (ZK-SNARKS) is a non-interactive ZK system[54]. To be practical, the proof can be verified within a few milliseconds and have a short length not to use too much storage space. The cryptocurrency Zcash was the first widespread use of ZK-SNARKs[55]. The big drawback with ZK-SNARKS is that they require a trusted setup to generate randomness. To avoid compromising the security Zcash organized a multi-party ceremony where each party generates a shard of a public-private key pair. You only need the public key, so the private key needs to be destroyed to avoid actors being able to print money.

3.2.14 ZK-STARKS

The trusted setup issues with ZK-SNARKs was addressed by Eli Ben-Sasson et al. in 2018 when they introduced a transparent ZK system: Zero-Knowledge Scalable Transparent ARgument of Knowledge (ZK-STARKS)[56]. They correctly identified the lack of scalability, transparency, and post-quantum security as severe issues with the current ZK systems. Scalable means that the verification of proofs scales exponentially faster than the data size. For a blockchain, a full node can thus generate proof in quasi-linear time ($n \log n$) and convince other nodes what the current state of the ledger is without requiring them to store the entire blockchain or re-compute the current state. Being transparent means that no trusted setup is required; the randomness used by the verifier is public.

No other universal realized ZK system than ZK-STARKS scales, provide transparency, and is post-quantum secure. However, there is, however, also a significant drawback with ZK-STARKS. Current ZK-STARKS proofs are about 1000x longer than ZK-SNARKS, so it is likely that research will be on making the proofs shorter or aggregate and compress them. Scalability is a big debate in the cryptocurrency space as the most successful projects only manage to do a tiny amount of transactions compared to VISA, as shown later in the scalability section.

3.2.15 Identity-linked ZK-SNARKS

ZK-SNARKS does not provide a concept of identity; they merely prove that someone knows something. If you are presenting a proof that you are old enough to buy alcohol, you must also prove that you generated the proof. This can be done with key derivation on the secret key. Provide a secret key and compute the associated public key to prove knowledge of the secret key.

Proofs are likewise very sensitive. If Alice gives Bob a proof that she held more than 1,000 BTC, she might not want the whole world to know, despite that the exact amount is hidden. This can be avoided by creating a proof that guarantees either:

1. The prover knows the secret and the sender's secret key.
2. The prover knows the receiver's secret key.

Using XOR on the previous statements Alice can create a proof because she knows her secret key as well as the secret. Bob knows that Alice does not know his secret key, so Alice must have created the proof. However, Dave cannot tell if it was Alice or Bob that created the proof, so Alice has plausible denyability[57].

3.2.16 Bulletproofs

An alternative to SNARKS and STARKS are Bulletproofs[58]. They are short, non-interactive, and requires no trusted setup. However, a Bulletproof is more time-consuming to verify than a SNARK proof. Bulletproofs support proof aggregation, so proving that Bitcoin transactions happened instead of storing the transactions themselves would shrink the UTXO from 160 GB to 17GB.

3.2.17 Group Signatures

Group signatures allow any member to sign on behalf of a group without revealing their identity[59]. This can be useful when we want to make sure someone in an organization has signed a document, but we want to provide plausible deniability for the signer. Group signatures are useful when members want to cooperate and requires a trusted manager to set up the system. This manager can reveal the signer if there are disputes and group signature are, therefore, a centralized solution.

3.2.18 Ring Signatures

A ring signature is a group signature but without a group manager[60]. A user can spontaneously create a ring signature with a set of public keys that may or may not belong to anyone else. Other members may not be aware that they are participating in the ring. Ring signatures provide a way for whistleblowers to remain anonymous while making them a trustworthy source of information. A person in the government could leak a secret to a journalist without revealing himself, and the journalist can be convinced it came from someone in the government. Ring signatures are useful when members of a group do not want to cooperate. The signature scheme can be simplified to a disjunctive statement where the signer shows that he has knowledge of one of the secret keys of a set of users: I know $S_1 \vee S_2 \vee \cdots \vee S_n$, where S_i is a secret key.

Signing

A ring signature σ is created from a set of public keys $P_1, P_2, \cdots, P_r$ for r ring members, the signers secret key S_s and a message m. The signature σ includes the public keys used to generate it. Because of this, the size of ring signatures grows linearly with the numbers of public keys. We define a combining function $C_{k,v}(y_1, y_2, \cdots, y_r)$ which is initialized with a key k, value v and random values $y_1, y_2, \cdots, y_r$ that act as ”fake” secret keys. The combining function uses the symmetric encryption function E_k generated with the message m to encrypt the output of XOR operations like this shown in equation 3.10.

$$C_{k,v}(y_1, y_2, \cdots, y_r) = E_k(Y_r \oplus E_k(y_{r-1} \oplus E_k(\cdots \oplus E_k(y1 \oplus v) \cdots))) \tag{3.10}$$

Solving the ring equation $C_{k,v}(y_1, y_2, \cdots, y_r) = v$ we calculate a unique value for y_s that satisfies the equation. The signer obtain x_s by inverting g_s on y_s:

$$x_s = g_s^{-1}(y_s)$$

Only the signer has the knowledge to invert his trapdoor, and it should be infeasible to solve the equation without a trapdoor. Figure 3.2 illustrates the combining function and Figure 3.3 depict why ring signatures are called ring signatures.

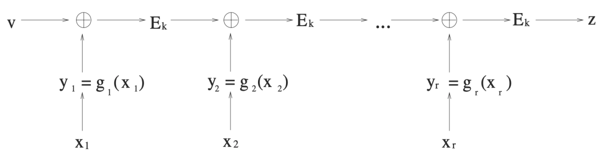


Figure 3.2: The Original RSA Ring Signature Scheme Combining Function[60]

Verifying

To verify a ring signature the verifier computes the encryption key k and $y_i = g_i(x_i)$ for all members of the ring. If the fundamental ring equation is satisfied with these values, the ring signature is valid.

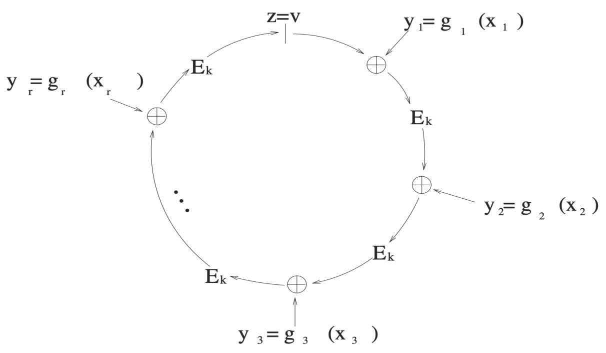


Figure 3.3: The Original RSA Ring Signature Scheme[60]

3.2.19 Linkable Ring Signatures

Linkable ring signatures provide anonymity but also allows verifiers to determine if the same ring member[61] has issued two signatures. This is useful in a voting scheme where voters should remain anonymous but are not be allowed to vote twice in the same election. It is also useful if we require multiple independent sources to trust a government leak.

We define linkability as signatures that were generated with the same secret key. The linkability is dependent on the list L of public keys being fixed. If a user signs any two messages with the same ring, the verifier can tell that the signer was the same (but the signer's identity is not revealed).

LSAG Signature Scheme

Linkable Spontaneously Anonymous Group (LSAG) is a linkable signature that satisfies three properties: $Linkability$ implies that we can link two signatures by the same signer; $Spontaneity$ implies that we have no group secret and thus require no shared setup; $Anonymity$ implies that we have signer indistinguishability.

We define $G = \langle g \rangle$ as a group of prime order q, so the underlying discrete logarithm problem is intractable. Let $L = \{ y_1, \cdots, y_n \}$ is a list of n public keys. H_1 and H_2 are independent hash functions where $H_1 \colon \{0,1\}^* \longrightarrow \mathbb{Z}_q$ and $H_2 \colon \{0,1\}^* \longrightarrow G$. Each user i has a secret key x_i such that $y_i = g^{x_i}$.

Signing

1. To get linkability the signer creates a tag $\tilde{y}$ by computing $h = H_2(L)$ and $\tilde{y} = h^{x_\pi}$. This ties the signature to a user's secret key
2. Generate a random seed value u and compute $c_{\pi+1} = H_1(L, \tilde{y}, m, g^u, h^u)$
3. For $i = 1, \cdots n$ generate a "fake" secret key s_i and compute
   $c_{i+1} = H_1(L, \tilde{y}, m, g^{s_i} y_i^{c_i}, h^{s_i} \tilde{y}^{c_i})$
4. Compute secret $s_\pi = u - x_\pi c_\pi \mod q$

The signature then becomes $\sigma_L(m) = (c_1, s_1, \cdots, s_n, \tilde{y})$

Verifying

1. Compute $h = H_2(L)$
2. For $i = 1, \cdots n$ compute the following
   $z'_i = g^{s_i} y_i^{c_i}$,
   $z''_i = h^{s_i} \tilde{y}^{c_i}$,
   $c_{i+1} = H_1(L, \tilde{y}, m, z'_n, z''_n)$
3. Check that $c_1 \stackrel{?}{=} H_1(L, \tilde{y}, m, z'_n, z''_n)$. If yes, accept the signature

Linkability

Given two signatures $\sigma'_L(m') = (c'_1, s'_1, \cdots, s'_n, \tilde{y}')$ and $\sigma''_L(m'') = (c''_1, s''_1, \cdots, s''_n, \tilde{y}'')$ check that both signatures are valid and that $\tilde{y}' = \tilde{y}''$. The signatures are created if the congruence holds.

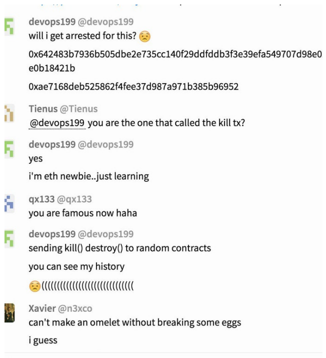
Linkable ring signature provides less anonymity than standard ring signatures. If an investigator subpoenas a user's private key he can use that key to generate a new ring signature and test the linkability with the signature of the leaked document to check the signer is behind the leak.

3.2.20 Formal verification

Extensive code reviews and testing is not enough to discover all faults in a system. The complexity in modern distributed systems increases the probability of human error in the design and code. Formal verification is a method to prove that a program is correct for all inputs. The correctness of the algorithm is proven with math, For example, that a state machine cannot reach a specific state.

Formal verification can be a valuable tool for concurrent and distributed programs. We can, for example, use it to prove safety properties: The system cannot lose committed data, and liveness properties: If the system receives a request, it must eventually respond.

Using TLA+, a formal specification language that is based on discrete mathematics, AWS found three subtle bugs in DynamoDB where the shortest error trace was 35 high-level steps[62]. They also found bugs in S3, EBS, and the Internal distributed lock manager.


devops199 @devops199
will i get arrested for this? 😟
0x642483b7936b505dbe2e735cc140f29ddfddb3f3e39efa549707d98e0e0b18421b
0xae7168deb525862f4fee37d987a971b385b96952

Tienus @Tienus
@devops199 you are the one that called the kill tx?

devops199 @devops199
yes
i'm eth newbie..just learning

qx133 @qx133
you are famous now haha

devops199 @devops199
sending kill() destroy() to random contracts
you can see my history
😟((((((((((((((((((((((((((((((

Xavier @n3xco
can't make an omelet without breaking some eggs
i guess

Figure 3.4: The newbie developer that killed the Parity library contract

Finding bugs is crucial for cryptocurrencies, where millions of dollars are at stake. Everyone is incentivized to maximize their economic gains by exploiting bugs. One of the most famous examples of a bug in a cryptocurrency is the Parity suicidal contract bug. Parity is an Ethereum client and also a multi-signature wallet. A developer new to Ethereum killed[63] a library contract which the main Parity contract relies on upon and locked \$169m in Ether[1].

[1] https://github.com/paritytech/parity-ethereum/issues/6995

3.3 Privacy-Preserving Data Collection

"Data cannot be fully anonymized and remain useful" – Dwork/Roth[64]

To make a collection of data useful, we need to perform operations on it. We are not interested in individual data records, but the aggregates. This section will discuss various techniques for preserving individual privacy when aggregating data.

3.3.1 Anonymization or Pseudonymization?

In general, data is more useful, the richer it is[64]. However, the more rich data is, the more difficult it is to achieve privacy with anonymization. Removing obvious personally identifiable information is not sufficient. Sometimes a particular combination of fields or attributes, like a ZIP code, gender, and date of birth can identify an individual with remarkable accuracy. In 2000 Sweeny showed that 87% of Americans could be identified with only those three pieces of information[65]. The attack is performed by linking the records with non-anonymised records in another data set. By linking anonymized medical records and voting records, both publicly available) they were able to expose the governor of Massachusetts's medical records.

Membership or not membership in a data set also leaks information. If you live in a sparsely populated area and you know your neighbor has been at the emergency room you can probably guess their diagnosis by matching the time.

Even revealing seemingly trivial facts is not ok. If Dave buys bread consistently every day until suddenly not buying bread anymore, an analyst could conclude that he got Type 2 diabetes. The conclusion might be correct, but either way, Dave's privacy is harmed. This poses a difficult problem: When is personal data sufficiently anonymized? It is challenging to anonymize personal data because you do not know what it takes to de-anonymize it.

3.3.2 Federated learning

Typically, machine learning requires that training data is centralized and models are trained in the cloud. Federated learning is a new approach that enables training on sensitive data[66]. Multiple hospitals might want to cooperate with training a model, but the data cannot leave the hospital, even in an encrypted form. Also, competitors in business might choose to train a model together without exposing their secrets to each other.

Another motivation for doing federated learning is to distribute heavy computations to devices on the edge. This will reduce the costs of training models and enable training on rich data, like video, when the connection bandwidth is low. For a self-driving car, it would be next to impossible (or costly) to send all the sensor data to the cloud for processing.

While aggregating the models from the clients in the federated learning system, we must treat the information as confidential. The ML model can also leak information[67]. The next sections explore techniques of secure aggregation.

3.3.3 Differential Privacy

The traditional way to extract useful information from data is to collect raw usage information in a centralized database and then run queries on it. This data is vulnerable to nosy programmers, hackers, and subpoenas from governments. While it is possible to use homomorphic encryption when doing operations on encrypted data, this is still not efficient enough for large data sets.

Differential Privacy (DP) is a technique where a user's identity is obscured by adding mathematical noise to a sample of the usage pattern[64]. When multiple users share the same pattern, a general pattern starts to emerge, but with some noise. Raw data is not collected, and this has several advantages. First, it requires less storage for the service provider. Second, it provides plausible deniability for the user about their answers. A medical company might want to do a study where people report if they smoke or not. The analysis might help us understand how smoking affects our health. If an insurance company learns that you smoke they might increase your premium. We need a way to analyze without compromising the user's privacy by not revealing if the user was part of the data set. On the other side, predictions are useless if they are too imprecise. Add too much noise in a model used to diagnose disease and you will kill patients[68]. Differential privacy is a trade-off between usefulness and privacy.

An example of DP is a research method used in structured survey interviews called randomized response[69]. A user reports an answer to a potentially embarrassing question, like whether he had sex with a prostitute in the last year. Before he answers, he flips a coin without revealing the result to the interviewer. If the coin comes up head, he answers truthfully. Else he answers yes. We can assume that half of the people that honestly would have said no had to say yes because of the law of large numbers for sizeable randomized sampling. Therefore, if 40% answered no and 60% answered yes, the true fraction that had sex with a prostitute would be around 20%. This leaks information about people who respond ”no”, so we can generalize the response by also providing plausible deniability for ”no”:

1. Flip a coin
2. If head, answer truthfully
3. If tails, flip coin again
4. If head, answer ”yes”
5. If tails, answer ”no”

Of course, there would be no actual coin flipping involved, but it would be built into the client. However, coins are great for illustrating examples. In this second example, we get privacy for any outcome.

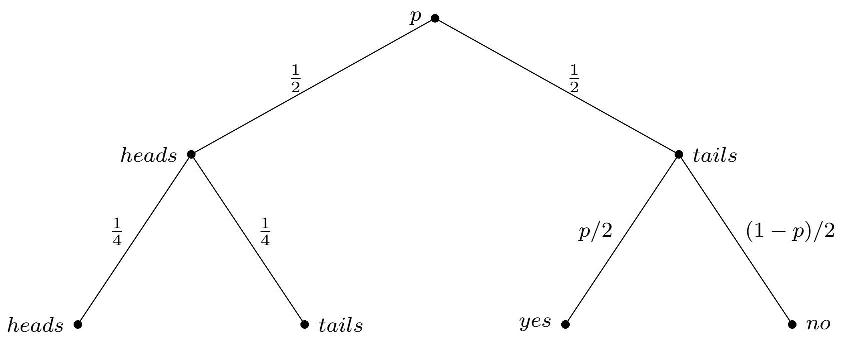


Figure 3.5: Probability Tree for Randomized Response

With plausibility deniability for both ”yes” and ”no” we estimate the real fraction of participants having the property to be $2(Y - 0.25)$, where Y is the ”yes” answers collected.

Another method is output randomization. A database stores the information, and a researcher queries the database for information. The database ads noise to the true answer before sending a response. The database has to make sure each answer does not leak too much information about the data and that the answers with noise are close to the original answers by adding noise from a Laplace or a Gaussian distribution[70, 71]. However, both randomized response techniques leak data if the participants are questioned multiple times, as we only add a small amount of noise to make the collected data useful.

RAPPOR

RAPPOR is an implementation of the randomized response technique used by Google for collecting usage statistics in Google Chrome[69, 72]. Instead of only ”answering” one question at a time, answers are encoded as strings and added as vectors in a Bloom filter. This allows for collecting queries repeatedly without the loss of privacy. Still, RAPPOR adds noise, which makes it unsuitable for some applications. DP remains hard to get right, and a sign of that is that there are papers written only to point out errors in existing papers about DP[73].

3.3.4 Prio

Instead of adding noise to the data Prio[74] takes another approach. Data is divided into shares on the clients and sent to multiple servers for aggregation. As long as at least one server is honest (and servers can be managed by different entities), the servers learn nearly nothing about the information. This is used by Mozilla Firefox to learn how many users that use ”private browsing” without revealing who these are[75].

Prio Scheme

1. Clients split its private data into s shares, one for each server, and sends the data on a secure channel to the servers

2. Servers aggregate the shares

3. Servers publish their accumulators, and a total sum of the accumulators yields the correct sum

Robustness and Correctness

Choosing a privacy advocating company to administer one of the servers make it next to impossible for the other servers to learn any personal information. The downside of this is the robustness of the protocol. If one server fails to do is a job, the aggregation will not work. We could make the protocol with s servers handle k faulty servers but only protect the privacy of $s - k - 1$ malicious servers. Placing a server in a country with strict privacy laws and no extradition would probably keep the data safe.

To protect against faulty or malicious clients, Prio uses a ZK proof called Secret-shared Non-Interactive Proofs (SNIP) to prove that data is inside a valid range. Prio provides privacy with only a 5.7x slowdown compared to naïve data collection, which is a lot faster than alternative systems.

By having different entities manage the Prio servers, we can ensure privacy as long as all of the parties avoid colluding. E.g., one server can be managed by app developers, and one can be managed by the mobile application platform (e.g., App Store or Google Play).

One thing that Prio does not solve is participation in the aggregation. To avoid spam it is suggested to use API keys for authentication, but this leaks information about participation and can over a long time leak data the same way as DP if the same data is collected multiple times. This problem will be discussed further in the Experiment section.

3.4 Blockchain Building Blocks

3.4.1 Definition

A blockchain is a database where the participants do not need to trust a central party to maintain the state of the database. The blockchain is structured as a linked list, where the elements are called blocks, and the pointers chain the blocks together. All the nodes on the network store the blockchain (also called a distributed ledger) and can validate the current state and the new state when blocks are added. Blocks can never be removed. Each block contains the cryptographic hash of the previous block, a timestamp and transaction data. By design, a blockchain is protected against modification of blocks and can store data in a permanent and verifiable way.

3.4.2 Cryptocurrencies

Cryptocurrencies are digital assets that use cryptography to secure financial transactions and to print more money. Unlike the central banking system control is decentralized and typically managed with a blockchain, though some projects are more centralized than others.

3.4.3 Bitcoin

Bitcoin[76] (BTC) is the blockchain-based payment system with the highest market cap and regarded as the first decentralized cryptocurrency. The decentralization came as a result of Satoshi Nakamoto solving the double-spending and centralized money creation problem in 2008. Double-spending refers to an individual being able to spend the same money twice. Because network communication in the network is not instantaneous different parts of the network might receive different next blocks. This network partition results in a fork and creates different states in the network. Because of this, it is not enough to trust a transaction from a block on the ledger. It must be hardened first. The conflict is solved when a new block is mined for one of the forks. Bitcoin chooses the longest chain, and the shorter one will be discarded. It is therefore recommended to wait for 6 more blocks to be added after your block (confirmations) before you consider something added. The finality is still only probabilistic since you could theoretically have a 51% attack.

3.4.4 Ethereum

Ethereum (ETH) is a decentralized platform that runs smart contracts. The Ethereum Virtual Machine (EVM) is a Turing-complete virtual machine that runs on the Ethereum blockchain, which can execute scripts on a decentralized network of nodes. The script requires gas to be deployed and run. This is a mechanism to mitigate spam and allocate resources on the network. Smart contracts are programs stored on a blockchain. They can run (in theory) without any possibility of downtime, censorship, fraud, or third-party interference. In Ethereum smart contracts are written in a higher-level language and compiled down to EVM bytecode before being deployed.

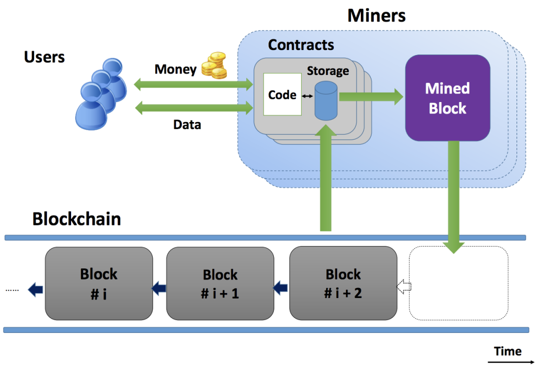


Figure 3.6: How a smart contract interacts with a blockchain[77]

3.4.5 Making Blockchains Secure

Blockchains rely on cryptography for integrity, authentication, and non-repudiation. Digital signatures ensure that a document has not been altered while it was transferred. Authentication makes sure the owner of a document can be verified based on a digital signature. Non-repudiation means that the sender of a signed document cannot deny signing it. Both Bitcoin and Ethereum use the Elliptic Curve Digital Signature Algorithm (ECDSA) for signatures as the key size is a lot smaller for the same level of security.

3.4.6 Merkle Tree

Merkle Tree is a data structure that allows efficient and secure validation of large data structures. With a tree of hashes, two lists of data can be compared with only the root hash. This makes validation of blocks in blockchains much faster.

3.4.7 Smart Contracts

Smart contracts[79] are contracts written with code that facilitate interaction without a trusted third party. A blockchain is used to store the contracts, so they are immutable and

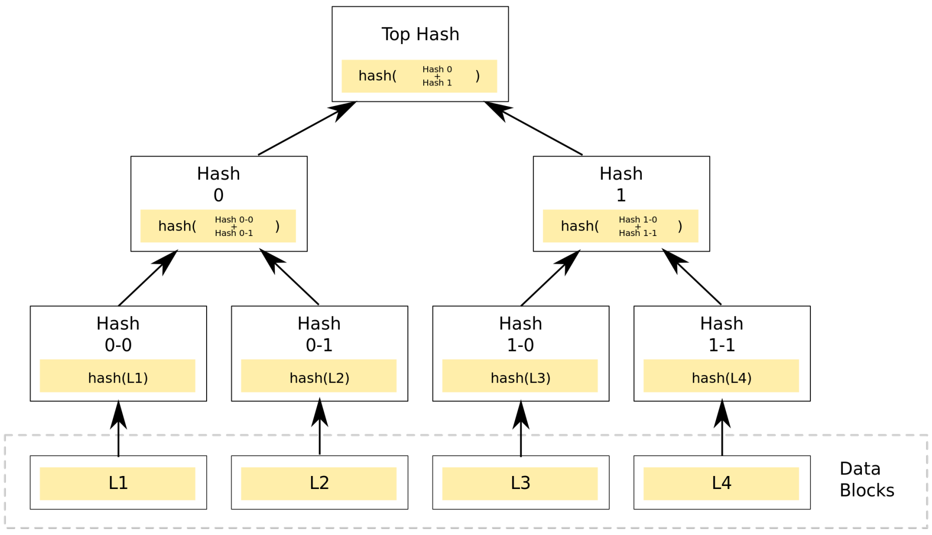


Figure 3.7: Merkle Trees - a tree of hashes[78]

decentralized. Since vulnerabilities cannot be removed once a contract is deployed, the contracts should be formally analyzed to avoid exploits[80].

Smart contracts let us create a decentralized version of Kickstarter[2] that does not charge an 8% fee[81]. A company or entrepreneur can create a smart contract that only gives them access to the fund if a specific funding goal is reached. If this does not happen, everyone that put money into the contract are automatically refunded.

3.4.8 Desirable Privacy Properties For Transactional Data

We do not want anyone to know when we spend money. *Unlinkability* is when you can't tell who sent money. If someone sends money twice, you cannot even tell that it is the same person sending money. The same applies to storing data. If an outsider can observe that you are storing the same bytes again, they can learn something about that data.

Another desired property is not to be able to trace funds. If someone knows you have money, they should not be able to tell when you spend it. This is called *untracability*. This also makes money fungible.

[2]https://www.kickstarter.com/

3.4.9 Consensus

To avoid double-spending, the network has to agree on the state. It is unpractical to wait an hour for 6 confirmations, and there exist other means to get consensus. The consensus is harder when we have a public network where we do not control all actors. When money is involved actors have an incentive to try to game the system, and we have to protect the network not only against network forks but malicious behavior.

FLP impossibility theorem

The FLP impossibility theorem[82] is a formal proof that consensus is not always reachable in limited time for an asynchronous environment where one process may fail by crashing. The system cannot distinguish between processes being slow or crash failures. The result of this is that consensus protocols must choose between $safety$ and $liveness$. Safety means that nothing bad will ever happen. Liveness means that we will have progress.

Proof of work

Proof of work[76] (PoW) is the consensus mechanism used in Bitcoin and is a mechanism to prevent spam on the network. The network rewards the first participant that solve a hard cryptographic puzzle. The first node to solve the puzzle is allowed to create a new block. Since each block is dependent on the input from the previous block, miners cannot precompute blocks. On average, it takes 10 min for someone to solve the puzzle. If hashing power is added to the network that average will go down, so the network adjusts itself by making the puzzle harder.

When a miner has found a valid solution for the PoW puzzle, they broadcast their solution to the rest of the network alongside transactions for the new block. Miners use electricity and get tokens as rewards. Miners also get mining fees for mining blocks and transaction fees. If a higher transaction fee is added to a transaction, it will be processed faster (at least miners are economically incentivized to do so)

Current fee mechanism for Bitcoin is to reward miners for mining blocks. This will not be sustainable in the long run; the number of coins that will be created is fixed — the transition from relying on block rewards into higher fees on transactions. Having fees for transactions is not clear-cut. There is only room for a certain amount of transactions per block, and the people that are willing to pay the most to will get their transaction in that block. It is difficult for a user to know how much they should pay because the network might congest in a non-predicable way. They probably end up paying more than needed. A better fee market[83] would be only to pay what the lowest bidder fee that was accepted into the block. This avoids that some parties accidentally pay millions for a single transaction[84] and represents the actual demand instead of first-price auctions.

PoW is dependent on miners to secure the network with useless puzzles. This makes the network require a lot of energy to be secured and transactions expensive. Yet, lowering transaction fees and mining rewards would result in the tragedy of the commons. Smaller rewards will result in fewer miners. Fewer miners and the network might be susceptible to a 51% attack.

Proof of Stake

Proof of Stake[85] (PoS) is another consensus mechanism where miners are people with wealth, not computing power. People that stake their wealth and are chosen at random to be validators and are typically punished if they try to cheat. There is no solving of cryptographic puzzles, so PoS is much more energy efficient.

To carry out a 51% attack on PoW, you have to temporarily obtain 51% of the hash power of the network. To carry out a 51% attack on PoS, an attacker would need to obtain 51% of the tokens. This is a lot more expensive.

dPoS

Delegated Proof of Stake (dPoS) is a PoS system where token holders voter for delegates to run the system. Each token holders voting power is determined by how many tokens they hold. The delegates with the most votes are elected to run the system. If they misbehave, voters can vote for someone else. Because the delegates that run the protocol are rewarded, there are backups to take their place. This makes for an efficient and effective protocol and mirrors democracy in many ways. However, just as in a real-world election, a large number of users might not care about voting, and it is easier to organize cartels and perform attacks since the system is more centralized.

Byzantine Fault Tolerance

Byzantine Fault Tolerance (BFT) solves the Byzantine Generals' Problem in a synchronized environment. By signing and passing messages, no expensive mining is required, and this results in low fees. The catch is that the network needs 2/3 or more honest and reliable nodes. BFT also has closed membership, which means that there has to be a list of recommended validators. Usually, the company behind the protocols manages that list, which makes BFT centralized. Finality is however guaranteed, and the system will not fork as PoW does.

Federated Byzantine Agreement

Federated Byzantine Agreement (FBA) refines BFT and makes membership open and control decentralized. No central entity creates a list of who can participate. Each node creates a quorum slice of other parties they trust. For example, a non-profit can trust that a transaction happened if Mozilla says it happened and Bank of America or JPMorgan Chase says it happened. FBA favors safety over liveness and will never fork, but can become unavailable, if, for example, the network splits. The drawback of FBA is that more setup is required by the node operator.

Consensus Protocol	Membership	Safety	Liveness	Scalable
PoW	Open	No	Yes	No
PoS	Open	No	Yes	Somewhat
BFT	Closed	Yes	No	Yes
FBA	Open	Yes	No	Yes

Table 3.2: Comparison of Consensus Algorithms

3.4.10 Decentralized Storage

Decentralized Storage Networks (DSNs) store your files on strangers computers and paying them with cryptocurrency tokens. The InterPlanetary File System[3] (IPFS) an example of one of these systems, but many more exist[4][5][6]. Data is content-addressed instead of location-addressed. Instead of telling the browser to get the data stored at example.com's IP address, you ask for the content that has the hash value of the data.

$$Qmc5gCcjYypU7y28oCALwfSvxCBskLuPKWpK4qpterKC7z \tag{3.11}$$

The hash in equation 3.11 is the hash for 'Hello World!'. The first byte indicates the hash function that was used to produce the hash (default now is SHA-256). The second byte indicates the length of the hash, and the rest is the output of the hash function. Like Bitcoin, IPFS uses Base58 to make the hash more user-friendly, by avoiding characters that can be mistaken for each other in certain fonts (like capital O and zero). The advantage of specifying the hash function in the hash is that it is easy to later upgrade to a better hash function while making sure the system is backward compatible. This is called Multihash[7].

To store who has what content IPFS relies on a Distributed Hash Table (DHT). No single node holds information about what other nodes store, but knows where to ask to get that

[3] https://ipfs.io/
[4] https://storj.io
[5] https://sia.tech/
[6] http://swarm-gateways.net/bzz:/theswarm.eth/
[7] https://multiformats.io/multihash/

information. This is similar to how DNS works. Files are duplicated, so they are resilient, but not in a wasteful way.

This network is more resilient than a traditional one because the data could be stored in multiple nodes, so if one node goes down, others can provide the data. Re-hashing the content after downloading it makes sure you got the file you were asking for. No one can change the data when it is being transmitted through the network without you knowing.

Data cannot be removed but will be forgotten by the swarm if peers do not have an incentive to store it. This problem can be solved by incentivizing nodes to store and keep content available by offering the Filecoin token. To get paid, node operators must prove that they are storing the data, not just claiming they are. This $proofofreplication$ can be done with Verifiable Delay Functions[86].

3.5 Scaling Blockchains

This section describes the scalability of current blockchains and the work that is being done to increase scalability.

The first public comment on the Bitcoin whitepaper was how can this scale?[87]. When Cryptokitties[8] became popular, Ethereum experienced a sixfold increase in pending transactions, and almost it almost brought the network to a grinding halt[88]. Scalability has arguably become the most important debate[89, 90] in the cryptocurrency space, as Bitcoin consumes more electricity than 159 countries at peak[91].

To scale blockchains, we can build on decades of research of distributed systems. We start by defining a naïve blockchain. All nodes store and process the same information. Adding nodes will not make the network able to store or process more transactions, but will make the network more collusion-resistant. One can argue that at a point, adding nodes for this system is a waste of electricity. To add more capacity to the network, we scale vertically by using more powerful nodes. We can process faster and store more information, but these increased hardware requirements restrict who can operate nodes and make the network less secure and more vulnerable to collusion. Mining is centralized to only powerful node operators running specialized hardware.

The scalability trilemma[92] dictates that a blockchain can at most have two out of the three properties:

- Decentralization
- Scalability
- Security

[8] `https://www.cryptokitties.co/`

We observe in Table 3.3 the different priorities of payment systems and that none achieve all three properties[93, 89, 94, 95, 96].

System	Degree of Decentralization	Peak TPS	Finality time	Private
Bitcoin	Distributed	7	1 hour	Maybe
Ethereum	Distributed	15	2 min	Maybe
EOS	Decentralized	250	Not guaranteed	Maybe
IOTA	Centralized	1,500	1 min	Maybe
Ripple	Decentralized	1,500	4 sec	Maybe
Stellar	Distributed	4,000	3-5 sec	Maybe
Visa	Centralized	56,000	Days	No

Table 3.3: Classifying payment systems with regards to the scalability trilemma

Bitcoin and Ethereum are secure and decentralized, but not speedy. In the other end, Visa is very speedy but is centralized and prone to censorship. In between, we have private systems like Monero and Zcash and systems that are more decentralized than distributed, like Ripple and EOS. They have many nodes, but few validator nodes, so they are less censorship resistant. However, still possible to host a node and keep a local copy of the blockchain. However, there is no mining reward and therefore, little incentive for regular users. However, more prominent entities could be incentives to do so.

3.5.1 Layer 1

Layer 1 scaling refers to scaling the core blockchain protocol to achieve scalability. This can be done with sharding, better consensus algorithms, use more powerful nodes, or use an alternative data structure to store blocks.

Sharding

Sharding is horizontal scaling by partitioning data into shards of data. If the data is too big to be stored on one node, split the data and store the parts on different nodes. This might seem straightforward. Split the key space from A-M and N-Z and store it on two different nodes. However, what if the keys in the first key space are more common than in the second one? A hotspot like this might create more load on one node, which also might change over time. The system must be able to rebalance itself, preferably in real-time. Sharding for blockchains is even more complex as the hashing power used to secure the network is split over shards. Less computing power is needed for an adversary to do a 51% attack.

Other ”Blockchains”

There exist multiple data structures that compete with blockchains. IOTA’s tangle[97] is a DAG where a new transaction must do work to approve two earlier transactions. No miners are required as users do work to add their transactions. This works both as an incentive and an anti-spam mechanism. Removing miners enables zero fees and high scalability. However, for now, the tangle requires a centralized coordinator.

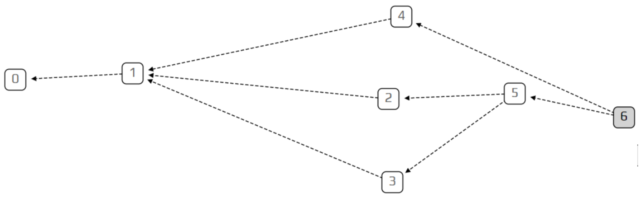


Figure 3.8: IOTA’s tangle - a DAG[98]. The grey node is unconfirmed as no other transaction links to it.

Nano’s block lattice[99] has users store their account blockchain off-chain. A transaction requires two transactions: one for the sender and one for the receiver. As with IOTA Nano have zero fees, and the unpruned ledger containing 14 million transactions is only 7.5GB. Zero fees might sound like a good thing, but if no one is incentivized to secure the network, an attacker might be able to carry out an attack.

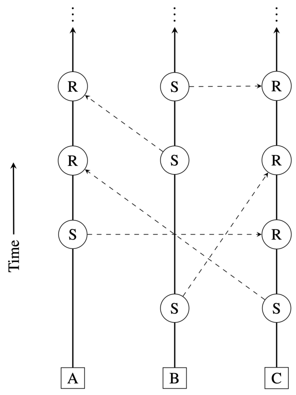


Figure 3.9: Nano’s block lattice[99]. S represent a send transaction and R represent a receive transaction

3.5.2 Layer 2

Layer 2 scaling refers to building a more performant layer which can fall back on the base layer for safety in case a black swan scenario[9] emerges. These systems are often off-chain and require the base layer to lean towards safety and decentralization.

The simple solution for scaling is to avoid putting transactions on the blockchain because they are expensive. Interaction with layer 1 happens only if there is a conflict or for checkpointing. This enables micropayments as transactions are fast and cheap.

Lightning network

Lightning Network is a layer 2 protocol built for Bitcoin. The lightweight software users to run nodes on commodity hardware since no mining is involved. There exists several implementations[10][11][12][13] and this provides decentralization. If there is found a bug in one of the clients, users can switch to another.

Plasma

A current limitation of smart contracts on Ethereum is that they are expensive to call. After all, the computation is duplicated over thousands of nodes. Computationally expensive transactions are therefore much more expensive to run on Ethereum than on a cloud VM. Plasma organizes blockchains in a hierarchical tree - to spawn child chains, which also can spawn child chains and do expensive operations on the child chains. Commitments will periodically be relayed to the root blockchain. A child chain can charge lower transaction fees because the operation does not have to be replicated over all the nodes in the root blockchain. Plasma guarantees that everyone can withdraw their funds to the main chain at any time, and if anyone tries to cheat, they are penalized because they have to stake a deposit to participate. This makes it safe even if a single entity controls the entire child chain.

3.5.3 Verification of Computational Integrity for Confidential Data Sets

When a new Bitcoin node is added to the network, it should verify the blockchain. To verify the integrity of the blockchain, we compare the hash of the last block with everyone

[9] An event that is extremely difficult to predict, yet have catastrophic ramifications
[10] https://github.com/mit-dci/lit
[11] https://github.com/ElementsProject/lightning
[12] https://github.com/ACINQ/eclair
[13] https://github.com/lightningnetwork/lnd

else. If it is the same, we know that all the transactions before are the same, unless a 51% attack has happened. Verifying smart contracts is more tricky, as the verification takes as much time as executing the smart contract.

A party P reports that doing a computation C on a data set D will result in particular output. If a data set D is public any party can verify $C(D)$ by re-execute C and compare the output with what was reported by P and ensure that it is $C(D)$. This naïve solution does not scale as we have to read the whole data set D linearly. If the data set D contains confidential data, a third party cannot use this naïve solution as it would violate privacy. To solve this in the real world, we often rely on a trusted third party, like an accountant, to verify the computation on behalf of everyone else. Putting trust in a single third party opens up the possibility for collusion. ZK proofs can replace these third parties and create scalable and transparent proofs that we can verify efficiently. The proofs are shorter than D and are verified faster than a naïve examination.

3.6 Privacy in a Cloud-Native World

3.6.1 A Naive Way to Build a Secure Storage

Databases are an important part of the modern computing experience. They are a central part where users can store and access the same data. An advantage over only storing the data on a user device is the redundancy you get. Replication over multiple drives and geographical zones make sure you do not lose your family photos if disaster strikes. Databases also provide ”unlimited” scalability. The pay-as-you-go model ensures you can buy more capacity instead of buying a new device when the storage capacity is exceeded.

Because databases often contain sensitive data, there is (or at least should be) a strong incentive to encrypt the data. The goal is to make sure you can access your data, and others can not — even the database provider which have physical access to the server. A naive solution to this problem would be to encrypt the data on the client and upload the encrypted data to the database while storing the encryption key on the client. However, this renders the modern database to a simple key-value store, unable to do range queries. Data would no longer be sorted because that would leak information. If records were sorted, an attacker could insert other records and leak information about specific records.

However, being able to range queries is one of the most useful features of a relational database and not something we want to forego. Instead of offloading the work on the database, the client has to download all the data rows and query themselves. This might not be possible because of the sheer size it takes to store the data on the client, and the time it takes to download it all makes it unusable from a usability perspective. While this is not great for database records, it works well for file storage, as we will see later.

Deterministic encryption creates the same ciphertext for the same plaintext. If queries are done on surnames, the attacker can guess the name by observing repeated queries since some names are more common. Adding a unique salt to the encryption scheme would avoid this but come at the cost of duplication.

Order-preserving encryption leaks the order of the records and in combination with the data distribution is it possible to reconstruct the majority of records in some databases[100].

In a world where data breaches are the norm[101] maybe any type of encryption is better than storing records in plaintext? Alternatively, maybe this gives the developers corpulent confidence in the security of the system? Maybe this problem has no good solution[102]? They do not have network access to our system, so it is ok that the internal communication in our system is unencrypted. Too many rely on security by obscurity.

3.6.2 Central Storage and Trust

Can we trust the cloud provider? If data is not encrypted, they have physical access to the hardware and can extract the entire data set. Even encrypting the data is not enough. By observing RAM access patterns, one can perform statistical attacks to learn what information is accessed. Centralizing storage creates a valuable target to hack. Economics dictates that it is worth it to attack if the reward is more significant than the resources used to attack and the risk. Storing data multiple places divides the resources used to attack and makes the reward for a successful attack smaller.

3.6.3 Trusted Hardware

Sometimes encrypting data is not sufficient to ensure privacy. If an adversary can observe access patterns to encrypted storage or memory, they can still learn sensitive information. Oblivious RAM[103] (ORAM) is an interface between the program and the physical RAM that shuffles memory, so data access patterns are hidden.

In 2013 Apple introduced the idea of a Secure Enclave Processor (SEP) to store and run sensitive data. Since then Intel has created Intel SGX (Software Guard Extensions) that allows private regions of memory inside the CPU. This makes it possible to build a trusted system on top of untrusted software resources, like a malicious operating system or hypervisor. However, it does require trust in Apple and Intel not having put in backdoors.

3.6.4 Trust on the Blockchain

For blockchains, we need a trusted method to communicate with the real world. This is referred to as the ”Oracle problem”. We need a way for smart contracts to access external data sources that they can trust. Smart contracts cannot access the data itself because inconsistent answers and time-outs might shut down the whole network. Many of the use cases for blockchain do not work without a trusted oracle, like a weather report for crop insurance, how much electricity is produced by a solar panel and sold and the outcome of an election on a prediction market.

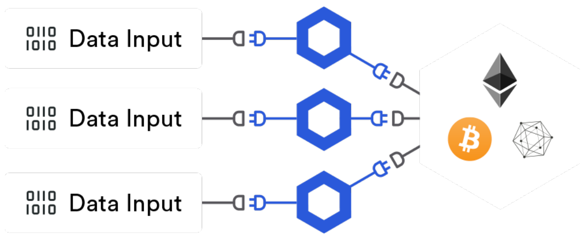


Figure 3.10: Chainlink's blockchain agnostic decentralized oracles[104]

Chainlink[105] develops a decentralized oracle system that uses a reputation system when deciding which sources to trust. If a data source reports a result that deviates too much from the mean value or takes too long to respond, their input will not be included in the aggregate, and they will have less influence on future aggregations. Services will, therefore, be strongly incentivized to provide high availability and performance.

3.6.5 Secure Aggregation

Secure aggregation is a type of secure multi-party computation algorithm where multiple parties aggregate their values without learning anything about other parties values. This can be used to count how many people are in an area without exposing who is there or collecting sensitive medical data.

Trusting a Centralized Actor

To create a secure aggregator, you need two properties to hold. To be able to create a public-private key pair where no one else can now the private key. No one should be able to snoop on the hardware and extract the key. Moreover, the aggregation should only happen if N of the users participate.

Google's federated learning model can be attacked by them if they create $N - 1$ users. We need at least 3 honest users part of the aggregation for t to work. This can be solved by generating random user results (noise) and adding these as part of the aggregation. 3 users would not affect the total result by much if N is 1000 but make it a lot harder to extract user data by diffing the result. Another problem with Google's secure aggregator is that you have to trust that they are running precisely that algorithm and nothing else. I have no way of knowing if they are genuinely using an incorruptible third party for the aggregation without any backdoors. The code is not public and verifiable. Even if it was how can we make sure Google is using that code and not some other code with a backdoor?

Selective denial-of-service attack

If adversaries prevent all honest clients except one from contacting the servers, and they provide all the other values they can differentiate and learn the honest client's values.

To avoid Selective DoS attacks, we can add authentication and only allow authenticated clients to participate. However, API keys leak information to the servers about who is participating. In the Experiments chapter, we address this by using ring signatures to create anonymous authentication.

3.6.6 Federated Learning With Secure Aggregation

Federated learning is suitable for a system where the clients produce vast amounts of data, and the cost of collecting the data is significant. An alternative approach would be to sample the data, but this would make the predictions less precise. For federated learning, Google assumes honest-but-curious actors. We assume a Byzantine environment (trust no-one).

It should be up to the user to decide where they want to store their data. Maybe they want to use the service but do not trust that the cloud provider can store their data securely? Maybe the company have a history of outages or are known to extradite information to the government? The way the data is encrypted should also be transparent to the user. Too many systems rely on outdated encryption or no salts.

Alternatively, a blockchain could be utilized to store the data, but the cost would be much higher due to increased duplication. Cost is an integral part of a system. A system that is secure but too expensive is impractical.

3.6.7 User best-practices

Many, if not most users do not know how their computer works and they do not care. The dream of decentralization is to have everyone host their data, to set up a server, but it is unrealistic that everyone becomes their system administrator. Privacy should not be reserved for the tech-savvy technocratic elite and those who can afford to buy Apple products[106]

Because users do not understand how strong a password should be, we expose them to risk when we allow them to choose weak passwords. We should strive towards creating UI patterns that protect users and expose them to as little risk as possible. It is not sufficient to have a rigorous privacy scheme because if the security becomes too tight, the users will find a way around it.

Chapter 4

Experiments

In this chapter, we implement prototypes to demonstrate how we can achieve better user privacy. Section 4.2 demonstrates how we can recover a lost password without fully trusting a single provider. Section 4.3 shows how we can decouple storage from applications and how blockchains can provide a recovery mechanism when the client experiences data loss. Section 4.4 demonstrates a caching mechanism for a voting protocol that provides authentication, but also plausible deniability for participation.

4.1 Motivation

We want to move storage to the edge and do useful aggregations while maintaining confidentiality; however, having users storing their data in an untrusted environment presents new challenges:

- How will data be recovered if a user forgets their password?
- How will a user be authenticated without a linkable API key?
- What if the user does not have enough storage space on their device to store their data?
- Can we still rely on the cloud, but in a more secure way?

It has become more evident in the last years that companies cannot be trusted to store your personal information. We should move information from these centralized systems to a more decentralized one closer to the user. By decoupling storage from applications, we can have less load on network because data is stored locally or physically close to you. The user experience can become better because of the lower response time. Our trust model should change from trusting large companies to assume that we are in a hostile environment, where no single actor should be trusted alone.

The cloud does not come for free. Lyft plans to spend $300M on AWS in the next 3 years[107] and self-driving cars are estimated to produce between 11-152 TB of data every single day[108]. Application providers benefit from users storing their data and ML models by having lower storage and compute costs. They can also get more high-res data, and we can train neural networks on all data, not just a sample sent to the server. One can also argue that federated learning allows for using more sensitive data for training, which increase the accuracy of the models.

4.2 Split-password

Decoupling storage from the service provider creates a recovery problem. The service provider can no longer help the user to recover their data if they lose their password. Forcing users to back up their password on a paper slip and storing it somewhere safe is not ideal, because you might forget where that somewhere safe is, or your house might burn down. Concurrently, service providers should not be trusted to store the password either as this would give them full access to your data.

An alternative to fully trust someone with your password is to split the password into shares and to have independent parties store the shares in semi-trusted locations. One part could be encrypted and stored by the service provider, the second could be stored in your email inbox and the third in your sister's Dropbox[1]. Only by combining all the shares it would be possible to recover the password.

4.2.1 Setup

We created a prototype that demonstrates a split password scheme[2]. When a user creates a password n shares are created so they can recover their password later. The prototype is based on trivial secret sharing, but could also be extended for redundancy by allowing some shares to be lost and still be able to recover the password[109].

To create shares, we generate $n - 1$ random strings with the same length as the password and XOR them with the password to create the last random string.

$$password \oplus random1 \oplus random2 = random3 \tag{4.1}$$

To recover the password we XOR all the random strings together.

$$password = random1 \oplus random2 \oplus random3 \tag{4.2}$$

4.2.2 Analysis

We can observe that we only recover $password$ if all parties cooperate. Knowledge of $k - 1$ shares reveals no information about the password except its length. To recover the password, the client has to gather all the shares. However, how can the parties that store the shares trust that the client really is the client if we cannot authenticate the client? We have to provide a weaker form of authentication, like answering questions that only the client should know, but are also likely to forget.

Because we need all shares to recover the password, all parties must collude to recover the password. This provides good security but is brittle as the shares might be lost or a service provider might refuse to give us their share when requested. This trade-off between security and robustness must be evaluated for each use case.

[1]https://www.dropbox.com
[2]`https://github.com/estensen/split-password`

4.3 Blockchain and IPFS

Data should be stored as close to the user as possible to minimize response time. This is not always possible, due to limited storage space (e.g., mobile phones), so we need somewhere else to store data. It is unrealistic to expect everyone to be their own system administrator and administer a server, but we also do not want to hand over the control to a single service provider. We suggest storing encrypted data in IPFS. IPFS pointers can be stored on a blockchain to simplify synchronization and backup if the user experiences data loss.

We wanted to explore if it is possible to store information on a blockchain and maintain privacy. By encrypting data before storing it on a blockchain, only the ones with the decryption key can make sense of it. Still, there is little reason to use a public blockchain to store your private information, so a permissioned blockchain is used, to increase performance, decrease storage cost and avoid the public even accessing the encrypted data.

The prototype stores data in IPFS and uses a blockchain as a file system. The blockchain is shared with semi-trusted parties for redundancy. If your device dies, you can request the blockchain from the other parties and be sure that you still have access to all your files. The main advantage of using a blockchain to store the pointers is its built-in checksums. When recovering from device loss, you can make sure you get the correct data by comparing root hashes instead of naïvely comparing all elements. Using a permissioned blockchain, we can configure it, so the user node is the leader node by default. The semi-trusted nodes will be used only for data recovery. Because the client is the leader, we do not need a consensus mechanism. If the client signs something, it is considered the truth.

4.3.1 Setup

We created a proof of concept prototype in Go that stores arbitrary data in IPFS[3]. Data is serialized to binary and encrypted before it is uploaded. We encrypt data with AES and use a different salt every time we encrypt data to avoid leaking that the same information is uploaded again. This leads to data duplication hell. The IPFS pointer is added to a blockchain, and the blockchain could be replicated to secondary nodes for backup.

4.3.2 Analysis

The system is centralized around the user and relies on it always being online to share data. If the client goes offline, data cannot be shared because only the client can decrypt the data. Routing all data through the client leads to draining batteries on mobile devices and huge cellular network usage. A user might share photos with a friend and turn off their

[3] `https://github.com/estensen/blockchain`

computer. The friend will then not be able to view the photos. One solution would be to share the decryption key. This is simple but exposes the data forever. The decryption key cannot be unshared. However, sharing the encryption key will make downloads faster as data can be downloaded from multiple IPFS nodes simultaneously.

What do we do if nodes storing the blockchain goes offline? Say we have 3 nodes: user node, insurance node, and hospital node. If the hospital node goes offline and we commit a result, this will only be backed up in the insurance node. If we have disk failure in our user node, we can only recover the last result from the insurance node. If the insurance node behaves maliciously, they can delete the last record. Using symmetric encryption, we are protected against altering data because the insurance company does not have our encryption key. To protect ourselves from a malicious party deleting blocks, we can add more nodes to the system. If we do not trust that the backup nodes act honestly, we can have them stake cryptocurrency and occasionally challenge them by randomly asking for data. If they fail to give us the data, we will automatically get refunded some of our payment from a smart contract.

You could use a simple key-value store instead of a blockchain, but the blockchain has built-in checksum, which makes the recovery fast and straightforward. Instead of downloading both databases and check all key-pairs naïvely you can request the latest hash and compare it. If they match, so does all the keys and values. Then download the blockchain in parallel from all the other nodes, because you can guarantee the integrity with the blockchain.

4.4 Linkable Ring Signatures

Previously data aggregation protocols with authentication, like Prio, leak information about which users participate in the data aggregation because they use API keys for authentication. Using linkable ring signatures (LRS) to not leak information has been suggested before[110] but is not practical to use on a large set of users, due to the time it takes to sign and verify the rings and the proof sizes. This would be very expensive for a voting scheme where users vote regularly. We propose a linkable ring signature scheme where users are partitioned into smaller rings. The ring signatures are used to create secret credentials that are authenticated. If the rings are large enough, we only leak information about participation in two edge cases: If no user in the ring has participated or if all have participated.

Our contribution is to point out how ring signatures could be used to create a new public-private key-pair that is anonymous but authenticated. The initial setup cost is high, but all subsequent requests would only require the server to verify the signature from a single key, which in comparison to a large ring signature, is almost free.

The registration phase for the user consists of generating an ECC key pair and signing it with a linkable ring signature of size n. The user sends a message with the new key pair and the signature to a server, which authenticates the validity of the signature and checks that that specific user has not previously generated an anonymous key pair. Linkability is fast to verify as we can add tokens from verified signatures to a set and check for membership instead of naïvely checking against all previous ring signatures.

4.4.1 Setup

We implement linkable ring signatures in Go and measure the execution time for signing and verifying signatures against normal ECC signatures[4]. We would like to know how practical it is to use rings with many public keys and estimate the time saving from caching an anonymous key pair over always authenticating with a ring signature. We assume an ideal world where all citizens have a public key which they can use to authenticate themselves. Users that are authenticated for a given service have their public key added to a public repository. We also assume all users can reliably download all the public keys.

We use ECC over RSA as the size of the signature is smaller, and they are faster to verify. All tests were run on a MacBook Pro (15-inch 2017) on macOS Mojave beta 6 with a 2.8 GHz Intel Core i7 processor and 16 GB RAM. The measurements were done 10 times, and the averages over these measurements were used. From the theory, we assume the signing and verification will scale linearly.

4.4.2 Results

Ring Size	Sign (s)	Verify (s)
10	0.011(0.0010)	0.0097(0.0011)
100	0.086(0.0089)	0.087(0.0066)
1,000	0.73(0.010)	0.73(0.012)
10,000	7.9(0.16)	7.9(0.23)
100,000	80(2.4)	80(2.1)

Table 4.1: Average measured Sign and Verify times in milliseconds with standard deviation in parentheses

For comparison, we benchmarked signing and verifying a message with a standard ECC signature. The average time for signing was 0.00013(0.000018) s and for verification 0.00000016(0.00000012) s.

[4]https://github.com/estensen/linkable-ring-signatures

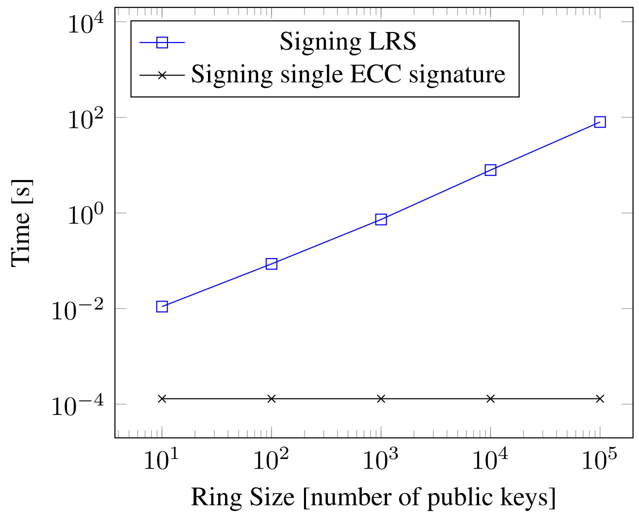


Figure 4.1: Time savings when caching LRS instead of signing the LRS every time data is sent.

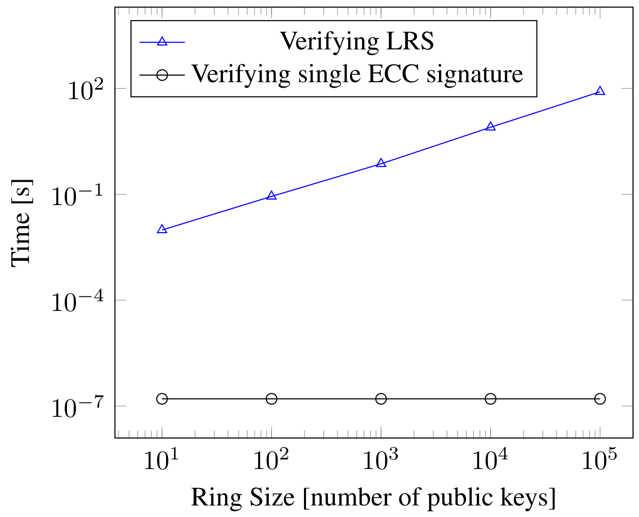


Figure 4.2: Time savings when caching LRS instead of verifying the LRS every time data is received.

4.4.3 Analysis

We can observe from Figure 4.1 and Figure 4.2 that the larger the ring size is, the more we gain from caching. For a ring size of 1,000, signing is approximately 5,600x faster and verifying approximately 4,500,0000x faster. This is not to say that our prototype is perfect. It could very likely be optimized further.

As both signing, verification, and storage scales linearly splitting a ring into 10 would decrease signing, verification, and storage by a factor of 10. The total number of signatures would be constant, but the signatures would be smaller. By caching the newly generated key-pair, we observe that verification is reduced while providing the same privacy guarantees.

Before you leak a secret, you should make sure to use many public keys in your ring. It should not be possible to track down all the participants and force them to expose their private keys to learn who a signer is. The overall response time experienced by the client is essential. We argue that the one-time setup can sacrifice user experience to provide security.

4.5 Future Work

Here we lay out some suggestion and improvements for future improvements in our experiments.

4.5.1 Split-password

We did not find a good way of providing weak authentication so we can retrieve our shares if we lose our password. Answering questions, you might forget the answer to might not be the best idea. Answers should be hard to guess, but also hard to forget or mix. Instead of answering easy-to-guess questions like ”what was your mother’s maiden name” or ”what was the first street you lived in” research should determine which question are easy to remember and hard to forget.

4.5.2 Blockchain and IPFS

The prototype works well for isolated user data. Data is private and immutable. However, collaboration on data that change is more complicated, especially in real-time. Conflicts are annoying and often requires a user to manually resolve them if collaborators do not agree beforehand who is going to edit the file[111]. Conflict-free Replicated Data Types (CRDTs) have been suggested as a way to merge concurrent modification[112].

4.5.3 Linkable Ring Signatures

The presented linkable ring signature scheme relies on the hardness assumptions of classical cryptography and is vulnerable to quantum computers. Further research the efficiency of lattice-based linkable ring signatures. Do a cost analysis of using serverless functions to verify and check the linkability of linkable ring signatures. The cost will be an important factor when deciding how many public keys are needed in a ring signature to provide good enough anonymity.

Chapter 5

Related Work and Conclusion

This chapter discusses related work and evaluates the research goals together with the experiments.

5.1 Related Work

Ring signatures are well-studied and have many applications, including cryptocurrencies and whistleblowing. Current research focuses on creating smaller signatures[113, 114] and making them quantum-resistant[115, 116]. Creating smaller signatures is especially important for blockchains, where storage is expensive. Logarithmic-size ring signatures have been created, but no benchmarks were published, and the extra work with pairings, encryption, and signatures look expensive, so the decreased size comes at a cost. Lattice-based cryptography holds a great promise for post-quantum cryptography[117], and schemes for ring signatures have been suggested.

Textile[1] is building a decentralized programmable iCloud on top of IPFS. As of know, they have built a photo app and a note app.

OrbitDB[2] a serverless, P2P distributed database. A database like this could replace the blockchain as a means of backing up the IPFS pointers.

5.2 Purpose

The purpose of this thesis was to explore techniques to increase user privacy. We presented three prototypes that demonstrate techniques that enhance user privacy.

The research questions are repeated here for reference:

> **RQ1**: How can we increase user privacy?
>
> **RQ2**: Can we use a blockchain to store personally identifiable information?

5.3 Conclusion

5.3.1 Towards User Privacy

Throughout the project, we have answered **RQ1** by exploring privacy-preserving techniques. None of these are silver bullets; all have their weaknesses. Defending against increasingly sophisticated attackers is a constant battle that has to be taken seriously. Best-practices has to be followed, but best-practices also change. Hash functions are broken and have to be replaced. However, the best defense against leaking data is not to store data at all. Until now, the practice has been to store data because it might be valuable to mine in

[1] https://www.textile.io/
[2] https://github.com/orbitdb/orbit-db

the future. With the GDPR, this practice is no longer allowed; companies must disclose the purpose for all data they are storing and processing.

Low-Hanging Fruits to Improve Privacy

It is hard to give general guidelines as companies are in different situations. Some companies have security teams, and others have a single developer. Still, security and privacy should be a top priority, and some things should be on the agenda no matter the company size or resources. We suggest some low-hanging fruits:

- Do not store more PII than strictly necessary
- Do not make anonymous data public. Chances are it is pseudonymous, not anonymous
- Pseudonymize data before analyzing it
- Do not allow users to choose weak passwords
- Use end-to-end encryption where possible
- Create a plan for being GDPR compliant
- Do not implement your own cryptography

5.3.2 GDPR, the Breaker of Chains

While the GDPR is a step towards better user privacy, the laws are arguably confusing. The GDPR is unclear about the right to explainability and courts has yet to rule in either direction. Too much privacy can also be problematic. If ML on PII is outlawed in the EU, the US and China might get an unfair technological advantage in the race to become AI superpowers.

This is especially evident with blockchains. Blockchains were not designed for the centralized world the GDPR was made for. Is everyone that is running an Ethereum node a data processor? They are more neutral in the way that they have no say in what should and should not be processed because they might not know what they are processing. On public blockchains, data is public and stored everywhere, not only inside the EU. We should not restrict ourselves by geographical borders, but create tools that can be used by everyone. Build a better future together where everyone has privacy, freedom of speech, and the possibility of participation.

Answering **RQ2**, we conclude that you should most often not use a blockchain to store PII. By adding something to a blockchain, it cannot be removed, and the risk of something being leaked in the future often does not justify the benefits. However, if you must use a blockchain, it can be useful to ask oneself these questions:

- Is data integrity and transparency necessary?
- What type of blockchain will be used?
- What data is stored on the blockchain?
- Can the users demand that their data is redacted and how will this be done?
- If data is encrypted who controls the keys?

Bibliography

[1] E. Hughes, “A cypherpunk’s manifesto.” `http://www.activism.net/cypherpunk/manifesto.html`. [Online; accessed 1 February 2019].

[2] “The universal declaration of human rights.” `http://www.un.org/en/universal-declaration-human-rights`. [Online; accessed 1 February 2019].

[3] “Two years after WannaCry, a million computers remain at risk.” `https://techcrunch.com/2019/05/12/wannacry-two-years-on/`. [Online; accessed 20 May 2019].

[4] “Baltimore ransomware nightmare could last weeks more, with big consequences.” `https://arstechnica.com/information-technology/2019/05/baltimore-ransomware-nightmare-could-last-weeks-more-with-big-consequences/`. [Online; accessed 20 May 2019].

[5] “The 100 largest companies in the world by market value in 2018 (in billion U.S. dollars).” `https://www.statista.com/statistics/263264/top-companies-in-the-world-by-market-value/`. [Online; accessed 20 May 2019].

[6] F. Roesner, T. Kohno, and D. Wetherall, “Detecting and defending against third-party tracking on the web,” in *Presented as part of the 9th USENIX Symposium on Networked Systems Design and Implementation (NSDI 12)*, (San Jose, CA), pp. 155–168, USENIX, 2012.

[7] B. Ur, P. G. Leon, L. F. Cranor, R. Shay, and Y. Wang, “Smart, useful, scary, creepy: Perceptions of online behavioral advertising,” in *Proceedings of the Eighth Symposium on Usable Privacy and Security*, SOUPS ’12, (New York, NY, USA), pp. 4:1–4:15, ACM, 2012.

[8] A. Acquisti, I. Adjerid, and L. Brandimarte, “Gone in 15 seconds: The limits of privacy transparency and control,” *IEEE Security and Privacy*, vol. 11, pp. 72–74, July 2013.

[9] “Leveling the Playing Field.” https://www.timetoplayfair.com/. [Online; accessed 4 June 2019].

[10] “Dear Mark. I am writing this to inform you that I shall not comply with your requirement to remove this picture.).” https://www.aftenposten.no/meninger/kommentar/i/G892Q/Dear-Mark-I-am-writing-this-to-inform-you-that-I-shall-not-comply-with-your-requirement-to-remove-this-picture. [Online; accessed 20 May 2019].

[11] “Facebook, Google and Twitter told to do more to fight fake news ahead of European elections).” https://techcrunch.com/2019/01/29/facebook-google-and-twitter-told-to-do-more-to-fight-fake-news-ahead-of-european-elections/. [Online; accessed 20 May 2019].

[12] “Facebook, Google and Twitter told to do more to fight fake news ahead of European elections).” https://ooni.torproject.org/post/2019-china-wikipedia-blocking/. [Online; accessed 21 May 2019].

[13] “The State of Internet Censorship in Egypt.” https://ooni.torproject.org/post/egypt-internet-censorship/. [Online; accessed 21 May 2019].

[14] “The State of Internet Censorship in Venezuela.” https://ooni.torproject.org/post/venezuela-internet-censorship/. [Online; accessed 21 May 2019].

[15] “Defend Assange Campaign.” https://twitter.com/defendassange/status/919247873648283653?lang=en. [Online; accessed 2 June 2019].

[16] “Banking Blockade.” https://wikileaks.org/Banking-Blockade.html. [Online; accessed 21 May 2019].

[17] “The bankers’ blockade of WikiLeaks must end.” https://www.theguardian.com/commentisfree/2011/oct/24/bankers-wikileaks-free-speech. [Online; accessed 21 May 2019].

[18] “RAPPORT D’INFORMATION.” http://www.assemblee-nationale.fr/15/pdf/rap-info/i1624.pdf. [Online; accessed 21 May 2019].

[19] “May calls again for tech firms to act on encrypted messaging.” https://www.theguardian.com/technology/2018/jan/25/theresa-may-calls-tech-firms-act-encrypted-messaging. [Online; accessed 21 May 2019].

[20] “Australia’s ban on encryption could endanger us all.” https://www.technologyreview.com/f/612562/this-is-how-australias-ban-on-encryption-could-endanger-us-all/. [Online; accessed 21 May 2019].

[21] “Data protection: Rules for the protection of personal data inside and outside the EU..” https://ec.europa.eu/info/law/law-topic/data-protection_en. [Online; accessed 4 June 2019].

[22] “APPA-WP29 GDPR factsheet.” https://ec.europa.eu/newsroom/article29/item-detail.cfm?item_id=614208. [Online; accessed 1 June 2019].

[23] “What rules apply if my organisation transfers data outside the EU?.” https://ec.europa.eu/info/law/law-topic/data-protection/reform/rules-business-and-organisations/obligations/what-rules-apply-if-my-organisation-transfers-data-outside-eu_en. [Online; accessed 21 May 2019].

[24] “Digital Single Market – Communication on Exchanging and Protecting Personal Data in a Globalised World Questions and Answers.” http://europa.eu/rapid/press-release_MEMO-17-15_en.htm. [Online; accessed 1 June 2019].

[25] “Google ordered to remove UK conviction from search results.” https://www.ft.com/content/7ab9d224-3f29-11e8-b7e0-52972418fec4. [Online; accessed 1 June 2019].

[26] “Is there a ’right to explanation’ for machine learning in the GDPR?.” https://iapp.org/news/a/is-there-a-right-to-explanation-for-machine-learning-in-the-gdpr/. [Online; accessed 24 May 2019].

[27] S. S. Adams, I. Arel, J. Bach, R. Coop, R. Furlan, B. Goertzel, J. S. Hall, A. V. Samsonovich, M. Scheutz, M. Schlesinger, S. C. Shapiro, and J. F. Sowa, “Mapping the landscape of human-level artificial general intelligence,” *AI Magazine*, vol. 33, 2012.

[28] “Revealed: 50 million Facebook profiles harvested for Cambridge Analytica in major data breach.” https://www.theguardian.com/news/2018/mar/17/cambridge-analytica-facebook-influence-us-election. [Online; accessed 21 May 2019].

[29] “Making face recognition less biased doesn’t make it less scary.” https://www.technologyreview.com/s/612846/making-face-recognition-less-biased-doesnt-make-it-less-scary/. [Online; accessed 21 May 2019].

[30] “Amazon scraps secret AI recruiting tool that showed bias against women.” https://www.reuters.com/article/us-amazon-com-jobs-automation-insight/amazon-scraps-secret-ai-recruiting-tool-that-showed-bias-against-women-idUSKCN1MK08G. [Online; accessed 21 May 2019].

[31] P. Baran, “On distributed communications networks,” *IEEE Transactions on Communications Systems*, vol. 12, pp. 1–9, March 1964.

[32] "Question on the terms 'distributed' and 'decentralised'." https://ethereum.stackexchange.com/questions/7812/question-on-the-terms-distributed-and-decentralised. [Online; accessed 23 May 2019].

[33] "What's the difference between distributed and decentralized in bitcoin-land?."

[34] "The Meaning of Decentralization." https://medium.com/@VitalikButerin/the-meaning-of-decentralization-a0c92b76a274. [Online; accessed 23 May 2019].

[35] "Privacy." https://en.bitcoin.it/wiki/Privacy. [Online; accessed 24 May 2019].

[36] "Remittance Prices Worldwide." https://remittanceprices.worldbank.org//sites/default/files/rpw_report_march_2019.pdf. [Online; accessed 24 May 2019].

[37] "UN World Food Programme uses Parity Ethereum to aid 100,000 refugees." https://www.parity.io/un-world-food-programme-uses-parity-ethereum-to-aid-100-000-refugees/. [Online; accessed 24 May 2019].

[38] "Mastodon Instances." https://instances.social/list/advanced#lang=&allowed=&prohibited=&users=. [Online; accessed 2 June 2019].

[39] "Slow Software." https://www.inkandswitch.com/slow-software.html. [Online; accessed 20 May 2019].

[40] "Make Data Useful." https://web.archive.org/web/20081117195303/http://home.blarg.net/~glinden/StanfordDataMining.2006-11-29.ppt. [Online; accessed November-24-2018].

[41] V. Zakhary, F. Nawab, D. Agrawal, and A. El Abbadi, "Global-scale placement of transactional data stores," in *Proceedings of the 21th International Conference on Extending Database Technology, EDBT 2018, Vienna, Austria, March 26-29, 2018.*, pp. 385–396, 2018.

[42] "The Stringify Service is Shutting Down." https://www.stringify.com/stringifyshuttingdown/. [Online; accessed 20 May 2019].

[43] "A second spring of cleaning." https://googleblog.blogspot.com/2013/03/a-second-spring-of-cleaning.html. [Online; accessed 20 May 2019].

[44] "Why was Mailbox shut down?." https://help.dropbox.com/mailbox-shut-down. [Online; accessed 20 May 2019].

[45] M. Stevens, E. Bursztein, P. Karpman, A. Albertini, and Y. Markov, "The first collision for full sha-1," *IACR Cryptology ePrint Archive*, vol. 2017, p. 190, 2017.

[46] R. L. Rivest, A. Shamir, and L. Adleman, "A method for obtaining digital signatures and public-key cryptosystems," *Commun. ACM*, vol. 21, pp. 120–126, Feb. 1978.

[47] E. T. Arjen K. Lenstra, Thorsten Kleinjung, "Universal security; from bits and mips to pools, lakes – and beyond." Cryptology ePrint Archive, Report 2013/635, 2013. https://eprint.iacr.org/2013/635.

[48] V. S. Miller, "Use of elliptic curves in cryptography," in *Advances in Cryptology — CRYPTO '85 Proceedings* (H. C. Williams, ed.), (Berlin, Heidelberg), pp. 417–426, Springer Berlin Heidelberg, 1986.

[49] A. Shamir, "How to share a secret," *Commun. ACM*, vol. 22, pp. 612–613, Nov. 1979.

[50] M. Naehrig, K. Lauter, and V. Vaikuntanathan, "Can homomorphic encryption be practical?," in *Proceedings of the 3rd ACM Workshop on Cloud Computing Security Workshop*, CCSW '11, (New York, NY, USA), pp. 113–124, ACM, 2011.

[51] C. Gentry and D. Boneh, *A fully homomorphic encryption scheme*, vol. 20. Stanford University Stanford, 2009.

[52] "The 773 Million Record "Collection #1" Data Breach." https://www.troyhunt.com/the-773-million-record-collection-1-data-reach/. [Online; accessed 27 May 2019].

[53] S. Goldwasser, S. Micali, and C. Rackoff, "The knowledge complexity of interactive proof systems," *SIAM Journal on computing*, vol. 18, no. 1, pp. 186–208, 1989.

[54] N. Bitansky, R. Canetti, A. Chiesa, and E. Tromer, "From extractable collision resistance to succinct non-interactive arguments of knowledge, and back again," in *Proceedings of the 3rd Innovations in Theoretical Computer Science Conference*, pp. 326–349, ACM, 2012.

[55] "What are zk-SNARKs?." https://z.cash/technology/zksnarks/. [Online; accessed 27 May 2019].

[56] E. Ben-Sasson, I. Bentov, Y. Horesh, and M. Riabzev, "Scalable, transparent, and post-quantum secure computational integrity.," *IACR Cryptology ePrint Archive*, vol. 2018, p. 46, 2018.

[57] "Building Identity-linked zkSNARKs with ZoKrates." https://medium.com/zokrates/building-identity-linked-zksnarks-with-zokrates-a36085cdd40. [Online; accessed 27 May 2019].

[58] B. Bünz, J. Bootle, D. Boneh, A. Poelstra, P. Wuille, and G. Maxwell, "Bulletproofs: Short proofs for confidential transactions and more." Cryptology ePrint Archive, Report 2017/1066, 2017. https://eprint.iacr.org/2017/1066.

[59] D. Chaum and E. van Heyst, “Group signatures,” in *Advances in Cryptology — EUROCRYPT ’91* (D. W. Davies, ed.), (Berlin, Heidelberg), pp. 257–265, Springer Berlin Heidelberg, 1991.

[60] R. L. Rivest, A. Shamir, and Y. Tauman, “How to leak a secret,” in *PROCEEDINGS OF THE 7TH INTERNATIONAL CONFERENCE ON THE THEORY AND APPLICATION OF CRYPTOLOGY AND INFORMATION SECURITY: ADVANCES IN CRYPTOLOGY*, pp. 554–567, Springer-Verlag, 2001.

[61] J. K. Liu, V. K. Wei, and D. S. Wong, “Linkable spontaneous anonymous group signature for ad hoc groups,” in *Information Security and Privacy* (H. Wang, J. Pieprzyk, and V. Varadharajan, eds.), (Berlin, Heidelberg), pp. 325–335, Springer Berlin Heidelberg, 2004.

[62] C. Newcombe, T. Rath, F. Zhang, B. Munteanu, M. Brooker, and M. Deardeuff, “How amazon web services uses formal methods,” *Commun. ACM*, vol. 58, pp. 66–73, Mar. 2015.

[63] “The guy who blew up Parity didn’t know what he was doing.” https://www.reddit.com/r/CryptoCurrency/comments/7beos3/the_guy_who_blew_up_parity_didnt_know_what_he_was/?st=jeydhos6&sh=b4457e3a. [Online; accessed 27 May 2019].

[64] C. Dwork and A. Roth, “The algorithmic foundations of differential privacy,” *Found. Trends Theor. Comput. Sci.*, vol. 9, pp. 211–407, Aug. 2014.

[65] L. Sweeney, “Simple demographics often identify people uniquely,” *Health*, vol. 671, 01 2000.

[66] K. Bonawitz, V. Ivanov, B. Kreuter, A. Marcedone, H. B. McMahan, S. Patel, D. Ramage, A. Segal, and K. Seth, “Practical secure aggregation for privacy preserving machine learning.” Cryptology ePrint Archive, Report 2017/281, 2017. https://eprint.iacr.org/2017/281.

[67] C. Song, T. Ristenpart, and V. Shmatikov, “Machine learning models that remember too much,” *CoRR*, vol. abs/1709.07886, 2017.

[68] M. Fredrikson, E. Lantz, S. Jha, S. Lin, D. Page, and T. Ristenpart, “Privacy in pharmacogenetics: An end-to-end case study of personalized warfarin dosing,” in *Proceedings of the 23rd USENIX Conference on Security Symposium*, SEC’14, (Berkeley, CA, USA), pp. 17–32, USENIX Association, 2014.

[69] U. Erlingsson, V. Pihur, and A. Korolova, “Rappor: Randomized aggregatable privacy-preserving ordinal response,” in *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*, CCS ’14, (New York, NY, USA), pp. 1054–1067, ACM, 2014.

[70] C. Dwork, M. Naor, O. Reingold, G. N. Rothblum, and S. Vadhan, “On the complexity of differentially private data release: Efficient algorithms and hardness results,” in *Proceedings of the Forty-first Annual ACM Symposium on Theory of Computing*, STOC ’09, (New York, NY, USA), pp. 381–390, ACM, 2009.

[71] C. Dwork, K. Kenthapadi, F. McSherry, I. Mironov, and M. Naor, "Our data, ourselves: Privacy via distributed noise generation," in *Advances in Cryptology - EUROCRYPT 2006* (S. Vaudenay, ed.), (Berlin, Heidelberg), pp. 486–503, Springer Berlin Heidelberg, 2006.

[72] "Rappor (Randomized Aggregatable Privacy Preserving Ordinal Responses)." http://www.chromium.org/developers/design-documents/rappor. [Online; accessed 28 May 2019].

[73] Z. Ding, Y. Wang, G. Wang, D. Zhang, and D. Kifer, "Detecting violations of differential privacy," in *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, CCS '18, (New York, NY, USA), pp. 475–489, ACM, 2018.

[74] H. Corrigan-Gibbs and D. Boneh, "Prio: Private, robust, and scalable computation of aggregate statistics," *CoRR*, vol. abs/1703.06255, 2017.

[75] "Testing Privacy-Preserving Telemetry with Prio." https://hacks.mozilla.org/2018/10/testing-privacy-preserving-telemetry-with-prio/. [Online; accessed 28 May 2019].

[76] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," 2008.

[77] K. Delmolino, M. Arnett, A. Kosba, A. Miller, and E. Shi, "Step by step towards creating a safe smart contract: Lessons and insights from a cryptocurrency lab," vol. 9604, pp. 79–94, 02 2016.

[78] "Merkle Tree." https://en.wikipedia.org/wiki/Merkle_tree#/media/File:Hash_Tree.svg. [Online; accessed 2 June 2019].

[79] N. Szabo, "Formalizing and securing relationships on public networks," *First Monday*, vol. 2, no. 9, 1997.

[80] I. Nikolic, A. Kolluri, I. Sergey, P. Saxena, and A. Hobor, "Finding the greedy, prodigal, and suicidal contracts at scale," *CoRR*, vol. abs/1802.06038, 2018.

[81] "Fees for the United States." https://www.kickstarter.com/help/fees?country=US. [Online; accessed 31 May 2019].

[82] M. J. Fischer, N. A. Lynch, and M. S. Paterson, "Impossibility of distributed consensus with one faulty process," *J. ACM*, vol. 32, pp. 374–382, Apr. 1985.

[83] S. Basu, D. Easley, M. O'Hara, and E. Sirer, "Towards a functional fee market for cryptocurrencies," *Available at SSRN 3318327*, 2019.

[84] "The Old Fee Market is Broken, Long Live the New Fee Market." http://hackingdistributed.com/2019/01/22/doing-fees-right/. [Online; accessed 16 May 2019].

[85] "Proof of Stake FAQ." https://github.com/ethereum/wiki/wiki/Proof-of-Stake-FAQ. [Online; accessed 2 June 2019].

[86] D. Boneh, J. Bonneau, B. Bünz, and B. Fisch, "Verifiable delay functions," in *Annual International Cryptology Conference*, pp. 757–788, Springer, 2018.

[87] "Bitcoin P2P e-cash paper." http://www.metzdowd.com/pipermail/cryptography/2008-November/014814.html. [Online; accessed 31 May 2019].

[88] "CryptoKitties craze slows down transactions on Ethereum." https://www.bbc.com/news/technology-42237162://jesper.borgstrup.dk/master-thesis-report.pdf. [Online; accessed 30 May 2019].

[89] K. Croman, C. Decker, I. Eyal, A. E. Gencer, A. Juels, A. Kosba, A. Miller, P. Saxena, E. Shi, E. G. Sirer, *et al.*, "On scaling decentralized blockchains," in *International Conference on Financial Cryptography and Data Security*, pp. 106–125, Springer, 2016.

[90] "Decentralized Payment Systems: Principles and Design." https://dtr.org/wp-content/uploads/2019/01/2019-01-16-Decentralized-Payment-Systems-Principles-and-Design.pdf. [Online; accessed 30 May 2019].

[91] "The Dark Side Of Blockchain: Electricity Consumption." https://cleantechnica.com/2018/12/08/the-dark-side-of-blockchain-electricity-consumption-blockchain-report-excerpt/. [Online; accessed 30 May 2019].

[92] "On sharding blockchains." https://github.com/ethereum/wiki/wiki/Sharding-FAQ. [Online; accessed 30 May 2019].

[93] "How Many Transactions Per Second Can Stellar Process?." https://www.lumenauts.com/blog/how-many-transactions-per-second-can-stellar-process. [Online; accessed 3 June 2019].

[94] "XRP." https://ripple.com/xrp/. [Online; accessed 3 June 2019].

[95] "Who Scales It Best? Inside Blockchains' Ongoing Transactions-Per-Second Race." https://cointelegraph.com/news/who-scales-it-best-inside-blockchains-ongoing-transactions-per-second-race. [Online; accessed 3 June 2019].

[96] "EOS: An Architectural, Performance, and Economic Analysis." https://www.whiteblock.io/library/eos-test-report.pdf. [Online; accessed 3 June 2019].

[97] "The Tangle." https://assets.ctfassets.net/r1dr6vzfxhev/2t4uxvsIqk0EUau6g2sw0g/45eae33637ca92f85dd9f4a3a218e1ec/iota1_4_3.pdf. [Online; accessed 3 June 2019].

[98] "The Tangle: an Illustrated Introduction." https://blog.iota.org/the-tangle-an-illustrated-introduction-4d5eae6fe8d4. [Online; accessed 2 June 2019].

[99] "Nano: A Feeless Distributed Cryptocurrency Network." https://nano.org/en/whitepaper. [Online; accessed 2 June 2019].

[100] D. Cash, P. Grubbs, J. Perry, and T. Ristenpart, "Leakage-abuse attacks against searchable encryption," in *Proceedings of the 22Nd ACM SIGSAC Conference on Computer and Communications Security*, CCS '15, (New York, NY, USA), pp. 668–679, ACM, 2015.

[101] "List of data breaches." https://en.wikipedia.org/wiki/List_of_data_breaches. [Online; accessed 30 April 2019].

[102] P. Grubbs, M.-S. Lacharité, B. Minaud, and K. G. Paterson, "Learning to reconstruct: Statistical learning theory and encrypted database attacks," in *IEEE Symposium on Security and Privacy (S&P) 2019*, 2019.

[103] O. Goldreich and R. Ostrovsky, "Software protection and simulation on oblivious rams," *J. ACM*, vol. 43, pp. 431–473, May 1996.

[104] "Chainlink: Connected Consensus on Ethereum." https://blog.chain.link/chainlink-live-ethereum-mainnet-connected-consensus/. [Online; accessed 2 June 2019].

[105] "ChainLink A Decentralized Oracle Network." https://link.smartcontract.com/whitepaper. [Online; accessed 31 May 2019].

[106] "Google's Sundar Pichai: Privacy Should Not Be a Luxury Good." https://www.nytimes.com/2019/05/07/opinion/google-sundar-pichai-privacy.html. [Online; accessed 28 May 2019].

[107] "Lyft S-1." https://www.sec.gov/Archives/edgar/data/1759509/000119312519059849/d633517ds1.htm. [Online; accessed 16 May 2019].

[108] "Autonomous and ADAS test cars produce over 11 TB of data per day." https://www.tuxera.com/blog/autonomous-and-adas-test-cars-produce-over-11-tb-of-data-per-day/. [Online; accessed 29 May 2019].

[109] J. Kurihara, S. Kiyomoto, K. Fukushima, and T. Tanaka, "A new (k,n)-threshold secret sharing scheme and its extension," in *Proceedings of the 11th International Conference on Information Security*, ISC '08, (Berlin, Heidelberg), pp. 455–470, Springer-Verlag, 2008.

[110] "Private, trustless and decentralized message consensus and voting schemes." https://jesper.borgstrup.dk/master-thesis-report.pdf. [Online; accessed 29 May 2019].

[111] "Local-first software: You own your data, in spite of the cloud." https://www.inkandswitch.com/local-first.html. [Online; accessed 20 May 2019].

[112] M. Shapiro, N. Preguiça, C. Baquero, and M. Zawirski, "Conflict-free replicated data types," in *Proceedings of the 13th International Conference on Stabilization, Safety, and Security of Distributed Systems*, SSS'11, (Berlin, Heidelberg), pp. 386–400, Springer-Verlag, 2011.

[113] M. Backes, N. Döttling, L. Hanzlik, K. Kluczniak, and J. Schneider, "Ring signatures: Logarithmic-size, no setup — from standard assumptions." Cryptology ePrint Archive, Report 2019/196, 2019. https://eprint.iacr.org/2019/196.

[114] B. Libert, T. Peters, and C. Qian, "Logarithmic-size ring signatures with tight security from the ddh assumption," in *Computer Security* (J. Lopez, J. Zhou, and M. Soriano, eds.), (Cham), pp. 288–308, Springer International Publishing, 2018.

[115] W. A. Alberto Torres, R. Steinfeld, A. Sakzad, J. K. Liu, V. Kuchta, N. Bhattacharjee, M. H. Au, and J. Cheng, "Post-quantum one-time linkable ring signature and application to ring confidential transactions in blockchain (lattice ringct v1.0)," in *Information Security and Privacy* (W. Susilo and G. Yang, eds.), (Cham), pp. 558–576, Springer International Publishing, 2018.

[116] C. Baum, H. Lin, and S. Oechsner, "Towards practical lattice-based one-time linkable ring signatures," in *Information and Communications Security* (D. Naccache, S. Xu, S. Qing, P. Samarati, G. Blanc, R. Lu, Z. Zhang, and A. Meddahi, eds.), (Cham), pp. 303–322, Springer International Publishing, 2018.

[117] D. Micciancio, "Lattice-based cryptography," *Encyclopedia of Cryptography and Security*, pp. 713–715, 2011.