BRNO UNIVERSITY OF TECHNOLOGY
VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

FACULTY OF INFORMATION TECHNOLOGY
FAKULTA INFORMAČNÍCH TECHNOLOGIÍ

DEPARTMENT OF INFORMATION SYSTEMS
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

IDENTIFICATION OF CRYPTOCURRENCY USERS

IDENTIFIKACE UŽIVATELŮ KRYPTOMĚN

TERM PROJECT
SEMESTRÁLNÍ PROJEKT

AUTHOR **HENRICH ZRNČÍK**
AUTOR PRÁCE

SUPERVISOR **Ing. VLADIMÍR VESELÝ, Ph.D.**
VEDOUCÍ PRÁCE

BRNO 2018

Vysoké učení technické v Brně
Fakulta informačních technologií

Ústav informačních systémů (UIFS) Akademický rok 2018/2019

Zadání bakalářské práce

Student: **Zrnčík Henrich**
Program: Informační technologie
Název: **Identifikace uživatelů kryptoměn**
Identification of Cryptocurrency Users
Kategorie: Web

Zadání:

1. Nastudujte si teorii za nejdůležitějšími kryptoměnami (Bitcoin, Ethereum, Ripple, EOS, Stellar, Cardano) a seznamte se s principy provozu jejich klientů.
2. Seznamte se s frameworky pro web-scraping (např. Lemmiwinks či Scrapy).
3. Dle doporučení vedoucího identifikujte zájmové stránky (primárně sociální sítě, uživatelská fóra) na veřejném Internetu a navrhněte způsob automatizovaného a periodického sběru metadat (adresy, URL, username, aj.).
4. Implementujte řešení s ohledem na to, aby výstupy běhu výsledného programu (programem posbírané identity a jejich metadata) byly dostupné v podobě (ne)strukturovaných dat uložených v databázi.
5. Proveďte testování vašeho řešení, diskutujte možná rozšíření.

Literatura:

- Mahto, D. K., & Singh, L. (2016, March). A dive into Web Scraper world. In *Computing for Sustainable Global Development (INDIACom), 2016 3rd International Conference on* (pp. 689-693). IEEE.
- Narayanan, A., Bonneau, J., Felten, E., Miller, A., & Goldfeder, S. (2016). *Bitcoin and cryptocurrency technologies: a comprehensive introduction*. Princeton University Press.

Pro udělení zápočtu za první semestr je požadováno:

- Body 1 až 3.

Podrobné závazné pokyny pro vypracování práce viz http://www.fit.vutbr.cz/info/szz/

Vedoucí práce: **Veselý Vladimír, Ing., Ph.D.**
Vedoucí ústavu: Kolář Dušan, doc. Dr. Ing.
Datum zadání: 1. listopadu 2018
Datum odevzdání: 15. května 2019
Datum schválení: 30. října 2018

Abstract

The 3rd January 2009 is considered to be the beginning of the cryptocurrency era. This work will deal with one of the ways of obtaining information about cryptocurrency users. It will be done by retrieving it from public websites where users often knowingly or unknowingly publish their cryptocurrency's address for different purposes (donation accounts or online payments which include personal data) as a result of which it is possible to link their crypto account with their real identity. The information is obtained through web scrappings. The result is an implemented application capable of automated collecting of data about the identity of cryptocurrency users and its subsequent storage in a structured database.

Abstrakt

3. január 2009 sa do histórie zapísal ako začiatok éry kryptomien. Vďaka používaniu pseudoným namiesto skutočných mien získali kryptomenu domnelú vlastnsoť anonymity. Táto práca sa bude zaoberať jednou z ciest získavania informácií o uživateloch kryptomien a to ich získavaním z verejných webových stránok, kde užívatelia často vedome, alebo nevedome zverejňujú svoje adresy za rôznym účelom (donačné účty alebo online platby výžadujúce aj osobné údaje) v dôsledku čoho je možné prepojiť ich krypto účet s reálnou identitou. Informácie sú získavané pomocou web scrappingu. Výsledkom je implementovaná aplikácia schopná automatizovaného zberu relevantných dat týkajúcich sa identity uživateľov kryptomien a ich následného ukladanie do štruktúrovanej databázy.

Keywords

Transaction, cryptocurrency, Ethereum, Bitcoin, Dash, Zcash, Monero, Web scraping, Blockchain, Public key, Private Key, Python

Klíčová slova

Transakce, kryptoměna, Ethereum, Bitcoin, Dash, Zcash, Monero, škrabanie webu, Blockchain, verejný kľúč, privátny kľuč, Python,

Reference

ZRNČÍK, Henrich. *Identification of cryptocurrency users.* Brno, 2018. Term project. Brno University of Technology, Faculty of Information Technology. Supervisor Ing. Vladimír Veselý, Ph.D.

Rozšířený abstrakt

Kryptoměny se od doby svého vzniku staly nedílnou součástí společnosti. Z technického hlediska se jedná o distribuovanou, čistě digitální měnu, která postrádá jakoukoliv centrální autoritu. Ta je v tomto případě nahrazena silnou kryptografií a technologií zvanou "blockchain". Blockchain představuje sdílenou distribuovanou databázi obsahující data o každé transakci od počátku sítě. Neexistuje v ní tedy žádný uzel, do kterého by se vkládala důvěra, a měl moc rozhodovat o stavu systému. Pro lidi jsou zajímavé hlavně jejich vlastnostmi, a to především anonymita, dostupnost a decentralizovanost. Transakce žádného z uživatelů nejsou nijak blokovány a vlastnit účet může prakticky kdokoliv. Rovněž zde neexistuje nekontrolovaná inflace, v síti je buď fixní počet mincí, nebo předem stanovená inflace. V každém případě počet mincí v síti v daném čase je jasně dopočitatelný. Tyto a mnohé další aspekty postupně přivedly ke kryptoměnám obrovské množství lidí. Mimo útočiště před finančním dohledem nebo pohodlnějším způsobem ukládání majetku, se kryptoměny taky ukázaly být bezpečným prostředkem pro placení ilegálních služeb, či provozování hazardních stránek, které díky nízkému zvýhodnění a jen minimálním regulacím dokážou poskytnout uživatelům mnohem lákavější nabídky. Neboť transakce jsou zapsány v blockchaine, který je dostupný pro kohokoliv, existuje možnost dohledat si transakce spojené s kterýmkoliv účtem. V momentě, kdy by se tomuto účtu přiřadila reálná identita nebo organizace vlastníka, odkryla by se nejen citlivá finanční informace o hodnotě, kterou daná entita disponuje, ale i celá transakční historie.

Tato práce se bude zabývat jednou z možností získávání informací o uživateli kryptoměn a to jejich získáváním z veřejně dostupných webových stránek jako jsou diskusní fóra, sociální sítě, nebo donační platformy. Jejich uživatelé zde často zveřejňují své adresy za různým účelem jako např. obchod, podvody, dotace, mixéry, hazard a podobně. Aplikace bude schopna prohledávat cílové stránky a detekovat výskyt adresy konkrétních kryptoměn s použitím web scarpingu. Výstupem této aplikace budou strukturovaná data o uživatelích kryptoměn spolu s jejich zařazením do kategorie podle odhadovaného užívání adresy. Tato data by se následně dala použít pro řadu aplikací. Avšak hlavním využitím je informační hodnota, kterou tato data v sobě nesou, jelikož díky přiřazení e-mailové adresy, či uživatelského jména na sociální síti v konečném důsledku do jisté míry obrací vlastnost anonymity či tzv. pseudoanonymity u kryptoměn proti uživateli samotnému. Především u kryptoměn bez pokročilejšího zabezpečení transakcí by bylo možné dokázat zmapovat kompletní historii finančních transakcí daného uživatele. Taková aplikace by mohla najít své uplatnění v potlačování zločinných aktivit spojených s černým trhem či jinými podezřelými transakcemi.

Identification of cryptocurrency users

Declaration

Hereby I declare that this master's thesis was prepared as an original author's work under the supervision of Mr. Ing. Vladimír Veselý Ph.D. The supplementary information was provided by Mr. Ing. Lukáš Zobal. All the relevant information sources, which were used during preparation of this thesis, are properly cited and included in the list of references.

.....
Henrich Zrnčík
May 16, 2019

Acknowledgements

I would like to thank my grandparents for all their support and to my supervisor Mr. Ing. Vladimír Veselý, Ph.D. for his time, goodwill and valuable suggestions.

Contents

Chapter 1

Introduction

Since 2009, when the first Bitcoin's transaction was confirmed, cryptocurrencies gained a worldwide reputation. They brought properties that none of the currencies had brought ever before. Unlike FIAT currencies, there is no more need for giving trust into the third side. There is no more central authority, regulations or arbitrary inflation. Having all these properties was more than enough to attract a wide spectrum of users. Those people were mostly enthusiasts interested in a new way of storing their assets, with the privacy they have never seen before. It did not take long to recognize the huge potential of cryptocurrencies in other sectors (black market or money laundering). Interest was so massive, that in less than 10 years since the birth of Bitcoin, we have more than 1,600 known cryptocurrencies [13]. They were created for various reasons and with knowledge about the strength and weaknesses of their predecessors.

It is important to understand, that these properties of cryptocurrencies are relative. It means that even if a cryptocurrency is decentralized, it truly is only until one entity does not take control over the majority of network[1], transactions are irreversible, but only once block is confirmed. It is the same with anonymity. By using your address publicly, you are probably putting anonymity in vain. Some cryptocurrencies (such as Monero or Zcash) aim for extra privacy. However, most of the cryptos do not consider any obfuscation techniques, and by using your address publicly you risk having your address connected to your true identity. For now, that is the case of most famous cryptocurrencies such as Bitcoin, Litecoin, etc.

This thesis is focused on obtaining information about users, who (aware or not of doing so) reveal their own address publicly. This can happen on forums, social network, or other locations of the public network. But before diving too deep into implementation, we will discuss properties and details of selected cryptocurrencies. Mainly their technical background and origin, all that had to be done, logical and technical barriers which had to be managed, before releasing first cryptocurrency 2.

Knowing what we are looking for and where to find it, we will take a look at widely used web scraping frameworks, and libraries on which application's design and implementation are based. This will be described in chapters Design 4 and Implementation 5. We will discuss the most important parts of the application, their purpose, and functionality there. Finally, we are going to confirm or contradict the usability and scalability of the application, including the validity test of obtained information (addresses, usernames, URLs) in Chapter 6, and discuss possible ways in which, this application can be used, in Chapter 7.

[1] in the case of cryptocurrencies based on Proof of Work, section 2.3

Chapter 2

Cryptocurrency

This chapter is an in-depth description of cryptocurrencies. We will take a closer look at factors behind their origins, the revolutionary way of creating transactions, and the features that have ensured their today's global expansion.

2.1 Origin

The name "cryptocurrency" was obtained by securing transactions with strong cryptography. Their creation is, to some extent, considered as a coincidence, because the initial intention of author/authors of the first cryptocurrency (further called by the pseudonym Satoshi[1]), was to create a "peer-to-peer electronic money system", not a new currency. A common feature for creating any of these services is decentralization that means network[2] in which the authority to decide about the state of the system does not belong to a single node. When `Satoshi Nakamoto` found a way to build a digital decentralized system, there was no further problem to create a cryptocurrency. On January 9th, 2009 Satoshi announced:

"I'm announcing the first edition of Bitcoin, a new electronic cash system that uses peer-to-peer network. It is completely decentralized, without a server or central authority."

With Satoshi's speech, the era of cryptocurrency has began.

2.1.1 Predecessors

Since the early existence of the first societies, there was a need to exchange commodities and services. The gradual development of society has also resulted in new means of exchange and trading. Replacement of linen, grain or livestock for wood and oil became impractical and often led to disagreements. It was necessary to create the first universal medium of payment. First golden and silver coins were incused in Asia 600 years before Christ. It did not take long and the phenomenon spread out in Europe too. Difficulty and lengthy of obtaining gold and silver gave this commodity a generally recognized and approved value. This makes them, the perfect successor of obsolete trade in its time. As most of the known

[1] Till today we do not know who is behind the idea of the First cryptocurrency, probably because of fate that met founders of previous currencies (Bernard von NotHaus, founder of Liberty Dollar, was arrested for "a unique form of domestic terrorism" [15])

[2] It can be any system without a central authority, not just network.

sources were mined out, the deficiencies of the mentioned system started to be apparent. The start of the colonial era and the huge supply of gold from America deferred the end of this system. The system started to collapse with the end of the Renaissance, as banks gradually began using paper money.

Paper money was still managed by specific organizations, not by the state. Growing population and technological advancement increased the need to switch to a financial system, where money could be transferred much more economically. Over the next four centuries, paper money had already proved itself, to be a sufficient successor of incused coins. They were more sophisticated version and, as a rule, they were also covered by a quantity of concrete precious metal. The change occurred only in 1971, since when paper money in the US is no longer covered with gold and other precious metals. This type of money is called FIAT from the Latin word "fiat", which stands for "let it happen". Their biggest disadvantage is they may reach theoretically unlimited numbers, which is among other things, one of the main causes of inflation. Their production, maintenance, storage and security against possible counterfeits is quite costly. Their latest stage of development has been switching to digital form, enabling customers to pay with cards and transfers. Centralized servers maintain information about account statuses. Banks support this trend, as the maintenance of digital money is much easier than physical ones.

2.1.2 Creation of cryptocurrency

In the nineties of the last century, an enormous number of people gained access to the Internet. Because a lot of them believed that the government and corporations were gaining too much control over them, they hoped that the Internet could become a tool for giving people more freedom of information. It culminated into two unsuccessful attempts to create a digital cash system. The effort has calmed down for a while, but mistrust in FIAT, and the financial crisis in 2007 have undoubtedly been a new driving force for further efforts. The breakthrough occurred in the aforementioned year of 2009, with the creation of the first `cryptocurrency`.

Cryptocurrencies can, to a certain extent, derive their origin from classical, commonly used money. Both types of currencies have a certain fictitious, people-given value. Both serve as exchange mediums. They differ only in definitions, which is also a purposeful way to protect cryptocurrency assets from control. Cryptocurrency, unlike money, is not officially funded by the government and is not able to accept the physical form. They have found their popularity in the fact that once a cryptocurrency is released, there is no way to interfere with their quantity or value, or at least not without majority agreement. In other words, there is no central node in the cryptocurrency that could increase the amount of money in circulation, as can central and commercial banks do in the case of FIAT currencies. Cryptocurrencies have either a fixed number of units (Bitcoin's 21,000,000) or fixed inflation (Ethereum allows for the creation of new 18,000,000 units annually) [4]. Uncontrolled inflation cannot occur in cryptocurrencies, whenever some important event takes place, at least more than half of the network's voting power[3] must come to an agreement. These events take place for example when a network needs to decrease the size of the smallest unit, or anyhow change consensus.

[3] Does not matter whether it is Proof of Work or Stake

2.2 Functioning

From the technical point of view, each payment network needs two things: 1) the authority that will take care of the problem-free running of the system, and 2) the database. In the case of most of the mentioned networks, the authority is secured by centralization, the network has a superior central node that has the authority to decide with unlimited power. In the case of cryptocurrencies, the central node is missing, which implies that the decision-making process is in the user's hands. We are talking about a decentralized network. By centralization, we can also mean the existence of a huge data-center that takes care of storing all the information. However, cryptocurrencies are based on a network of computers that use a peer to peer distributed topology. Distributed network means that resources such as memory or computational power are distributed among countless nodes. In "peer to peer" communication models, the individual nodes are mutually equivalent. Since all nodes are equivalent and distributed over the entire network, we can speak about a decentralized, distributed system, or a network without any central authority. Topologies are shown in Figure 2.1.

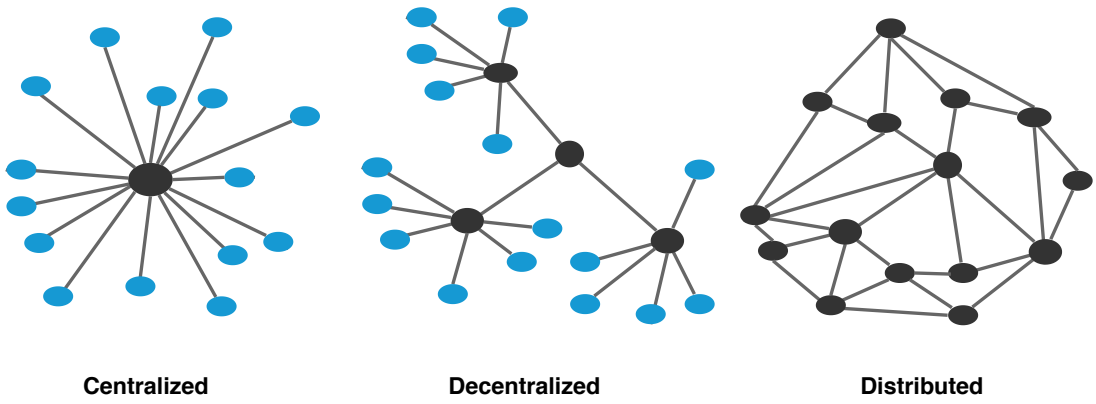

Figure 2.1: Network topology - centralized, decentralized and distributed

One of the many advantages of systems without one central node is that there is no more one truly irreplaceable place, and because of that, chances of bringing the system down are significantly reduced [20]. This and other threats for cryptocurrency are shown in section 2.2.2. The following section describes the database, and delegation of authority to write data into it 2.2.1.

2.2.1 Blockchain

`Blockchain` is still a new term among the public and it is often misinterpreted [16]. From its description, we know that it is a special kind of decentralized database, with possibly thousands of constantly synchronized copies across the network. The true meaning of it is something completely beyond that. It does have various meanings for different groups of people – from a tool for a decentralized world to the new generation of the Internet. We will look at it as a distributed, decentralized, encrypted database. The first two traits have already been explained, and encrypted means a way how is the trustworthiness guaranteed. Blockchain consist of `blocks` which hold data and some additional analytic or management information. Whenever the new block is added, it connects itself to the previous one via

a hash function. Blocks than create a backward linked list of blocks. This way it appears that blocks are chained one with another. This is where its name originated from and also the main reason why data in blocks are irreversible. We will discuss encryption and security in the following subsection 2.2.2. For better understanding, take a look at Figure 2.2.

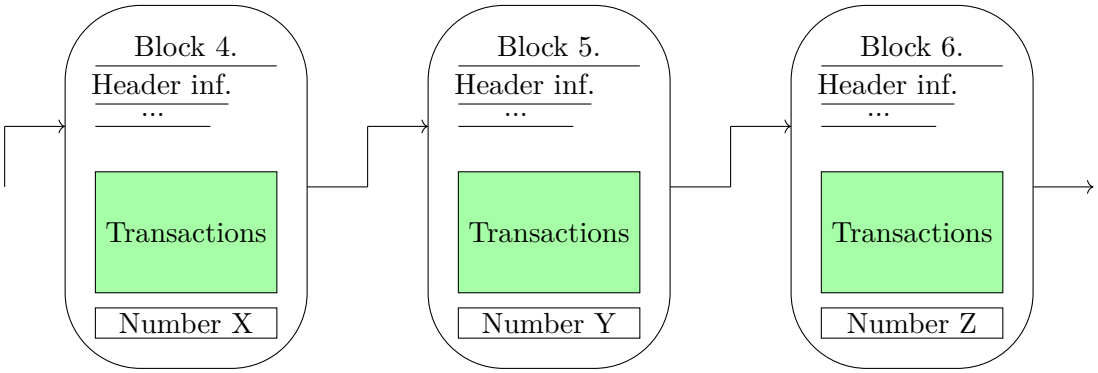

Figure 2.2: simplified scheme of Blockchain

Every single block of data consists of three main parts: Header, Data (transactions) and one extra value, which will be important in the following sections. Data are made of transactions, every transaction has to have at least three main information: 1) addresses of the sender, 2) receiver, and 3) amount of currency to be sent. Since the blockchain inception, every single valid transaction is stored in it and it is being visible for everyone. The core idea behind storing data this way is that when we know about every single confirmed transaction in this network, we can easily get all missing information. Let us see the example of blockchain data from Table 2.1

Sending account	Reciving account	Amount
X	1	50.00
1	2	20.00
1	3	25.00
3	2	15.00

Table 2.1: Transactions

The moment we clear blockchain data from all noise, we get something that is similar to simple excel table[4]. We will think of this table as of a list with all transactions which have ever been made within our network. As you can see on the first line, there is information about account number 1, which received fifty tokens from account X. This symbolizes, account number 1 got these tokens from the system. The second row shows us the user 1 who is sending twenty tokens to user 2. Third and fourth rows can be interpreted accordingly.

Algorithm which decides whether a transaction is possible or not, can be a slightly different, depending on concrete system/cryptocurrency (for example in case of Bitcoin, you have to send all the tokens you got, so in order to send only part of it, you have to provide a second address where the remaining coins will be deposited). Generally, it is based on summarizing all incoming and out-coming transactions. After repeating the same procedure with other transactions from Table 2.1, we obtain the final state as described in Table 2.2.

[4]Microsoft Office Excel, https://office.live.com/start/Excel.aspx

Account	Amount
1	5
2	10
3	35

Table 2.2: Example results

Representation of information from Table 2.2 demonstrates the convertibility of transaction from the blockchain into a table that stores the final balance of accounts. This representation offers way, how to determine whether or not is transaction feasible. This is exactly the way, how is the possibility of executing transactions verified. Rows in Table 2.2 may represent wallets of concrete users. There is a widely spread opinion, the possibilities, blockchain brings for transactions is the same that had the Internet for information [16]. What is discussed here is just the simplified demonstration, as it describes the basics of blockchain 1.0, whereas newer version already exist[5]

2.2.2 Secure

The trust into third sides is replaced with the use of `asymmetric cryptography`. Every user is being given two keys, a public and a private one. `Public key` is visible for everyone else in the network, and is mainly used to verify the signature generated by `private key`. Whereas private key is something no one but the owner should have access to. We use our private key to sign the transaction with the function called `Sign`. The function takes our private key and applies it on a message (in this case transaction) in order to produce a signature. For just a slightly different input, the function produces a different signature. This means every single signature we generate this way is different. The receiver of this message then uses a message, signature and public key to verify, if the signature is correct, and no one did manipulate with it on the way. Public and private keys are paired but there is no way to deduct private key from the public key. So to summarize it, this is the way how to confirm the second side that I know the password without revealing it to this second side [31]. Sure, there are plenty of other ways how to cheat the system, from sophisticated approaches, like trying to spend the same amount two times, to brute force, when a possible attacker tries to break the signature using guessing repeatedly.

2.2.2.1 Miners

title `miner` is used for those nodes, which keep the integrity of the network, by verifying transactions. This includes creating new blocks and adding them to the blockchain. A miner verifies transactions, organizes these transactions into a block, which he distributes throughout the network. This part happens very quickly and does not consume too much energy, but miner still has to persuade others about the validity of his work. The most well-known approach is `Proof of Work`, which is based on solving something called puzzle. To speak plainly, the better hardware you have, the higher chance to mine new block. The second approach is `Proof of Stake`. the Creator of a new block is chosen in a deterministic way, depending on its stake[6]. Term "miner" is replaced with "forger".
As a reward for this services `miner/forger` which add a new block to the blockchain, may

[5]Etherum based on Blockchain 2.0 provides additional extensions such as smart contracts [9]
[6]Stake is wealth, node poses with

append one more record, with bearing information about a certain amount of coins given to him. This is the spacial transaction from Table 2.2, also known as generating transaction[7]. Individual cryptocurrencies have a different approach to these rewards, reducing it every year like Bitcoin 2.3.2, or provide the network with a new constant amount of tokens every year like Ethereum 2.3.3.

2.2.2.2 Byzantine generals

Cryptocurrency's network is `trustless`. If two entities want to communicate, they do not put their trust into the third side. Instead of it, it is a network alone who keeps its integrity and functionality. The only problem is, how to guarantee consensus within the network. In order to keep the trust of people, the blockchain needs to be provably valid, immutable, and consistent across all the nodes [10]. The best way, how to provide these traits, is to validate every single new block which is to be added. In other words, we have to guarantee fully consensus. This means the problem of multiple entities have to come into the agreement, despite the distance between them, before critical actions are taken. It was first introduced as the "Byzantine Generals problem" in 1982.

The idea of `Byzantine Generals` is following. We have a group of generals, with their armies besieging the city. In order to conquer the city, armies have to cooperate and synchronize their attack. The problem is that if you want to cooperate, you have to send messages to each other, and come to the conclusion. In order to get your message to some of the generals, messengers have to pass throw the city so your message can be captured, and delivered with changed content. It is the same from the opposite side. Once you get a message how can you determine, whether it is original or not? The solution which was brought by Satoshi 2.1 and the idea behind it is called Proof of Work 2.2.2.3. Consider the situation, you want to send the message „Night Attack", and you know that there is a special formula in Byzantine army saying that the message is correct if it gives five zeroes after applying hashing function called *ByHash*. Knowing that message **NightAttack** will not give you five zeroes, you can add numbers to the end of the message, and always check if the result begins with five zeros. This process may take some time, but after giving enough tries, you will get wanted results. Once message **NightAttack1784** is received, the army can easily verify, whether the result of function *ByHash*, applied on the message starts with five zeroes, and consider it to be a reliable message. In case the enemy had changed the message to **DayAttack1784**, the result of function *ByHash* almost certainly will not give five starting zeros. Even if the city somehow knew the function *ByHash*, it would have to spend a long time guessing a new correct offset. The geniality of this recline in fact that it is gonna take a long time, to produce a reliable message, as you have no better option than guessing offset, while confirmation, is an almost instant process. The only way for the city to persuade us, about the validity of the message, is to produce more falsifications than all other generals combined. In other words, in order to convince generals in the long term, the city has to produce at least 51% of messages. Depending on the pseudo-casualness of our function and required number of zeros, we can adjust the difficulty of creating new messages, while confirmation of validity is affected only slightly. The main part of this solution is to prove that we did invest a great amount of energy to provide a solution. The following sections describe, how Bitcoin and other cryptos applied this idea in their protocols.

[7] this transaction is the only way, new tokens are created, in cryptorrucency without inflation.

2.2.2.3 Proof of Work

Consider terminology from the previous section. generals with their armies represent miners devoted to network, whereas the city represents a possible threat, hackers/attackers. A string that needs to be added to the concrete message, in order to get a valid result is called `nonce` (random number meant to be used only once), the only difference is that functions used in cryptocurrencies are called `SHA` (Secure hash algorithm) not `ByHash`. Result of this function is then checked, if it satisfies conditions, this condition is similar to condition from previous sub-section 2.2.2.2 that first X digits must be zeroes, where X can be changed according to the needs of the network. We can think for example about `SHA-1 algorithm`, which produces a 160-bit long result, by assigning 1 to X we declare that every hash starting with zero is valid. Increasing this value by one means 2 times less valid results. Cryptocurrencies like Bitcoin 2.3.2 use this mechanism, as a tool to keep new block comings approximately every ten minutes. The chance of hitting the correct combination reminds us of the lottery. Increasing value X means that every single miner in network will probably spend more time guessing this correct number, and therefore spends more energy, and because of fact that this guessing process runs on every miner, electric consumption is alarming[8]. This direct proportion between computing power and the chance of gaining a reward led to the creation of Mining farms, which is a data center, technically equipped with extremely high computational power. To get rid of these two disadvantages: excessively high consumption of energy and centralized mining pools[9], cryptocurrencies have to use a different approach.

2.2.2.4 Proof of Stake

Just like proof of work, the purpose of this proof is to establish consensus on the blockchain, but the way to accomplish that is completely different. When we think about the proof of work, we can see that there are plenty of disadvantages decreasing its usability. Mainly big dependency on electric power, and mining pools, which disclaim the idea of decentralization. Proof of Stake tries to fix these problems by replacing the previously mentioned process of a lottery, by choosing forgers. These forgers can add this block and others only vote if this new block is valid or not. The choice of this forger is based on his stake, thus coins he uses to guarantee his honest ground. The bigger your stake is the higher the probability of being chosen you have. So there is no more connection between how powerful hardware you have, and the impact you have on the network. The problem with energy is also eliminated as a result of choosing only a small group of forgers, instead of having every single miner guessing. On the other hand, there are new problems, such as "rich become richer"[10], or "blocking". Therefore we can only hardly compare these two approaches and choose one which would be better. More details are shown in Table 2.3.

2.2.3 Principles

There are several properties, every cryptocurrency needs to have, in order for it to be called a cryptocurrency. These traits/properties are results of approaches and mechanisms

[8] It is hard to calculate exact consumption of electric power, but today sources agree that Bitcoin consumes at least, as much electricity as small countries.

[9] More than 70% of new blocks were approved by pools located in China [19]

[10] Without further upgrades of choosing forgers, those forgers without significant resources get hardly ever chance to forge new block.

Consensus Mechanism	Pros	Cons
Proof of Work	tested and reliable, mathematically simple, clean,	mining pools , 51% attack energy consumption,
Proof of Stake	needs far less hardware and energy, validator answers with his assets	rich get richer , problems with inicial decentralization,

Table 2.3: Difference between proofs

that founders of concrete cryptocurrency choose. Therefore we often see that some of the cryptocurrencies satisfy these properties only partially, and they are often described by the volume of following these properties.

2.2.3.1 Decentralization

Is also probably one of the most disputable topic bounded with cryptocurrency. To have a decentralized network, we have to be sure that there is not one entity that can easily control the whole network, this power has to be distributed all around. Prove of Work or Stake together with the idea of rewarding honest nodes helps accomplish this idea, just the same as the fact that everyone in the network can start mining/forging. In the ideal world, we would have these miners roughly evenly distributed across the earth. In cryptocurrencies such as Bitcoin, this idea worked perfectly in the beginning, but mining with laptop evolved and was slowly replaced by mining using graphics card, and within few steps evolved into big mining pools and farms built on an application-specific integrated circuit -ASIC. The arrival of these super miners has evolved into a serious threat for decentralized networks, as it meant the end for most smaller miners, with no such powerful hardware.

A good example is the current distribution of hashing power in Bitcoin's network, where more than 80% of the mining power is now concentrated in Chine. Mining centralization in China is one of Bitcoin's biggest issues at the moment. Seven out of ten biggest mining pools are located in China, and even when they are not part of the same company, they are located under the same government, and thus if they connected together, they would easily overcome the network with their 80% of computing power.

2.2.3.2 Trustless

Every single form of currency before the first cryptocurrency required a central authority, and therefore in order to use it, we had to put our trust into their hands. In cryptocurrencies, this trust is replaced with `cryptographic proof` [10]. New transactions are validated using signatures and added into blocks which are afterward added into the blockchain. Process of adding blocks liable to consensus mechanisms such as Proof of Work and Proof of Stake. With everyone having an exact copy of the ledger, we no longer need to put this trust into the hands of a single entity- bank, or organization. These mechanisms have proved themselves to be safe enough. To accomplish unfair intention, you would need huge computing power or already have significant estates in concrete currency at your disposal.

2.2.3.3 Immutable

Once a block is added to the blockchain, it can never be changed or destroyed. Or at least, it is really difficult to do so. It is the same for transactions, once it is approved nobody

can undo them. To do so, we would need to create blockchain from begging, and therefore this action would again need enormous computing power. Now revising the example about Byzantine generals 2.2.2.2, there was a possibility to convince generals that message is correct by generating messages with changed content, but still fulfilling constraints in a higher rate than other, honest generals. In a single moment, the network can consider more streams of messages to be correct. The transaction becomes immutable once it is added to the blockchain, not at the moment, node broadcast it to network.

2.2.3.4 Anonymity

In order to own a cryptocurrency, the user does not have to provide any personal information that means there is no link between owner and address. This property of pseudo-anonymity gives users a whole bunch of possibilities. The most discussed are those in illegal areas. cryptocurrencies are a perfect means of money on Silk road[11]. and other black markets, they can help with money laundering and tax evasions.

Even though cryptocurrencies use `pseudonyms`, and therefore your real identity does not need to be connected with your account, once you reveal you are the owner of the specific account, your privacy is becoming feigned. Consequences of doing so are multiplied by fact, every transaction stored in the blockchain can be trucked and thereby, both your assets and complete history of transactions, are revealed with a bit of effort. There is actually a whole bunch of ways how your anonymity can be compromised, with posting your address publicly on some forum or used with an internet purchase being just the top of the iceberg.

2.2.3.5 Others

There are actually more of these traits, and the final number of these properties is subject of many discussions, but those mentioned above, mainly(decentralization, anonymity, Immutability) are most demanded. Some of them are the following:

- Free of Permission
- Borderless
- Instantaneous

2.3 Best known cryptocurrency

As we have already mentioned in previous section 2.1, since first cryptocurrency, new one are created almost daily. Despite they are all cryptocurrencies, there is a bunch of differences between them, mainly considering their specific characteristics, and how they stand with principles mentioned previously. Now we will take a closer look on some of them.

2.3.1 Terminology

In order to fully understand individual cryptocurrency, we must firstly move through bit of terminology. Terms such as public and private key have already been described but there are few more.

[11] Silk road is an anonymous and illegal online market, also known as a platform for selling illegal drugs [36]

Private Key - with this key you sign the cryptocurrencies you send to others, only you alone should know your private key.

Public Key - the public key is used to generate an actual cryptocurrency address. Together with the private key they are making a pair. This means that if somebody knows your public key he can send you tokens, nothing more.

Address - a string of alphanumerical characters, a user can share with anyone, who wants to send him tokens. It is created from public key via one way hashing functions.

Wallet - is nothing more but hardware, software or even piece of paper that stores public key, private key, and our balance. All these kinds of storing your information, determine the volume of how much safe, user-friendly and available your wallet will be.

2.3.2 Bitcoin

Bitcoin is the first cryptocurrency ever made, as you probably already have noticed within the process of the description of cryptocurrency, mainly because there are plenty of references on Bitcoin. We did that because descriptions of these two big concepts are connected. All the other cryptos are somehow based on its principles and findings.

Bitcoin is invented by Satoshi and its token is called also Bitcoin. First Bitcoins were mined on the third of January 2009 by Satoshi alone [12]. Since then about seventeen and a half million were mined. Bitcoin has only a finite number of coins (this token is also called BTC), and it is 21 000 000. Back in 2009, a reward for one mined block was 50 BTC. This reward is decreased by half every with every 210000th mined block[12].

As was mentioned in section Miners 2.2.2.1, everyone can become a miner, with more hashing power and same difficulty of puzzle[13], we could possibly start solving new blocks much faster and therefore, every 2016th block the difficulty of puzzle is changed in order to keep time between two added block approximately 10 minutes.

Combination of Proof of Work and a network of Bitcoin's size and its hashing power requires an enormous amount of electric power. According to Alex de Vries, a bitcoin specialist at PwC estimates that the current global power consumption for the servers that run bitcoin's software is a minimum consumption of 22 terawatt-hours per year—almost the same as Ireland [14].

Bitcoin is considered to be a cryptocurrency with a low level of anonymity, as details about transactions do not have any kind of protection (sender and receiver are visible). This so-called weakness, is partly removed by using CoinJoin[14]. Using this method consist in combining multiple payments into a single transaction so it is more difficult to track down certain communication.

2.3.2.1 Bitcoin client

Each node runs a specialized piece of software called a Bitcoin Client [33]. A client has to synchronize itself with the rest of the network. It connects with the user's "Wallet" and updates it whenever it is needed. This "Client" or "node" can run in two modes, depending on what functionality you want.

[12] Some sources also provide information that this decrease takes place every four years, this means completely the same information, when we consider that process of adding a single block takes approximately 10 minutes

[13] process of guessing the nonce

[14] Coin Join: https://bitcointalk.org/?topic=279249

- **Lightweight client/ Full nodes** - perform work of miners 2.2.2.1, their task is to download every block and transaction, then check them against Bitcoin's consensus rules. In order to become Bitcoin miner, you have to firstly synchronize with a network (you will need around 100 GB of space), once you are done with it you can start performing as Bitcoin's miner/ full node.
- **Lightweight client/ Thin clients** - or in case it is called from a web browser, it is called web client or mobile client. Manages only information relevant to you. These clients communicate with the blockchain, but they download only a fragment of blockchain.

Official implementation of Bitcoin client is open-source and is named Bitcoin Core[15]. There are also some alternative implementations such as Gocoin[16] or Armory[17] however, these requires giving our trust into third parties.

2.3.3 Ethereum

Also known as cryptocurrency of the second generation is blockchain based technology, which enables users to build and deploy decentralized applications. Currency is called `Ether` (ETH), Functionality of this mechanism is a bit different from „normal" cryptocurrency[18].

2.3.3.1 Smart contracts

Blockchain has built-in programming language, which allows a feature called Smart Contract. These Smart contracts are then stored in the blockchain, so everyone can see the source code, and therefore examine the condition, but only a creator can delete them. The main idea of this all is that with bitcoin and other cryptocurrencies of the first generation, we did first steps to avoid trusting third sides, but the first generation only allowed for monetary transactions, there was no way to add conditions to those transactions [6]. This allows users to get rid of the dependency on a third side even more[19], this is also known as if-then premise-based system. To sum them up, smart contracts help us exchange money, property, shares, or anything of value in a transparent, conflict-free way while avoiding the services of a middleman [7].

2.3.3.2 Ethereum Virtual Machine

Virtual machine responsible for executing all smart contracts mentioned in the previous subsection 2.3.3.1. To do so, three important things must be maintained. Contracts must be isolated, deterministic and terminable. To accomplish these traits founders of Etherium had to solve a problem with endless loop and keep code deterministic. The first problem is solved by a metric called „gas" every single operation cost some of this gas, once you run out of it, a contract is killed. The second problem meant that some functionality has to be

[15] https://bitcoin.org/en/download

[16] https://www.crunchbase.com/organization/gocoin#section-overview

[17] https://www.bitcoinarmory.com/

[18] part of Dr. Gavin Wood words on address of Ethereum: "Ethereum is decentralized computer which is not really good nor fast, but has a global characteristic, cannot be stuck and is available everywhere you have access to network".

[19] with smart contracts you can define timing, penalties and other rules of transactions, even self-destruction of contract

pressed, this is done by providing a new language (Providity), so one has to learn a whole new language in order to create a smart contract on Ethereum.

2.3.3.3 Gas & Miners

This resource called gas is used as a metric for computing power required to execute some transactions. This gas can be bought for ETH, and spend by a predefined table on concrete instructions obtained in a transaction. Forbye preventing from an endless loop, there is one more reason for this gas concept, and it attracts more miners with this opportunity of earning something extra. There are two possible options, that can happen during the execution of the operation. If we run out of gas we return to the state before an operation, gas stay spend. In the second scenario, we provided enough gas and the rest of it is returned. Problem is that miners prefer transactions with a lower amount of gas because there is a higher probability they will not have to return anything.

2.3.3.4 Ethereum client

These clients run the Ethereum Virtual Machine and are written in following languages. Ethereum foundation is supporting various client implementation. This helped to find and fix some of bugs connected with consensus. Secondly it is safer to have more implementation of client, specially if one does not work properly. Some of these implementations :

- **C++:** TurboEthereum, Webthree
- **Go:** Go Ethereum
- **Python:** Pyeth
- **Java:** EthereumJ

all of mentioned clients[20] are at the moment considered to be fully working.

2.3.4 Dash

As Bitcoin was just the first application of digital currency, it had some flaws, which other cryptocurrencies are trying to remove. Probably the biggest one is lack of anonymity[21]. Dash, what stands for Digital cash[22], is built on Bitcoin's core but is aimed to be more like cash system, brings features such as enhanced privacy and quick transactions. What is more transmission's fees are small even considering terms of cryptocurrencies [21].
Dash has a finite number of tokens - eighteen million, the last coin is expected to be mined around 2300. For now, there are around seven and a half million coins in the network. The difficulty of mining block is adjusted for every single block, in order to keep the time between two added block around two and a half minute [38]. Structure of reward for the mined block is the following:

- **45%:** for miner.

[20] https://github.com/ethereum/wiki/wiki/Clients,-tools,-dapp-browsers,-wallets-and-other-projects

[21] This lack of anonymity is meant in terms of cryptocurrency, and the possibility of tracking transactions

[22] Dash was released in January 2014 as "Xcoin" than also known as "DarkCoin", what was not the best idea, because of DarkNet and on 25 March 2015 the name was changed to "Dash„"

- **45%:** for master nodes.
- **10%:** for future network funding projects[23].

What makes Dash more specific, Dash has some additional features which can network provide via the existence of special kind of node called "MasterNode",

2.3.4.1 Masternodes

These nodes are the second tier of classic nodes, we know from previous cryptocurrency 2.3.2 [37]. Difference between master nodes and other nodes is in their rewards, services and requirements[24]. All of them are shown in Table 2.4.

	Master node	Node
Reqierements	dedicated IP address, 1 000 DASH, max. hour/daily unattached	Hardware, running client, space
Service	*PrivateSend*, *InstantSend*, Mining	Mining
Reward	approximately 2 DASH weekly, Vote Transaction Fee	Transaction Fee

Table 2.4: MasterNodes and Nodes in Dash

2.3.4.2 PrivateSend

`PrivateSend` is designed to improve the privacy of transactions, it does so by swapping among users which breaks the history of transactions. This process of mixing is provided by a master node, and you have to ask for it specifically. By using this service we can hide the association between us and recipients [18].

2.3.4.3 InstantSend

Dash adds new blocks every 150 seconds, so that means if you have paid for something, the seller had to wait a long time for the first confirmation of this transaction. `InstantSend` is achieved by master node network quorum. The transaction is broadcast to the network and then locked within a master node. Master node than reject all blocks with transactions containing these coins. This can accelerate payment to 1.3 seconds approximately.

2.3.5 Zcash

This protocol is also based on Bitcoin. The relationship between these two protocols can be compared to one between HTTP and HTTPS. Each user is provided with two types of address, `z-addr` which is fully private and `t-addr` which stands for transparent, similar to a Bitcoin address, it is up to user which one he is gonna use, so there are actually four

[23]These project are voted and only masterNodes have vote, https://app.dashnexus.org/proposals/leaderboard

[24]Concept known as Proof of Service

variations of sender-receiver address pair[25]. To secure the information about the value we have to use *z-addr* at least for the receiver.

Zcash uses a process called shielding, to mask value, receiver, and sender. It is done by zero-knowledge proof what means, the way how one person, can proof he knows some information without revealing this information. Zcash uses zk-SNARKs [3] to accomplish this.

There is an official Zcash client built for Linux users. Community members have created modified versions of it for MacOSX and Windows. You can also opt for a third-party wallet application.

Commonly used option to store Zcash is the Multi-Currency wallet Rahakott. The Zcash Company maintains the Linux command-line reference client, zcashd[26] There is also a fully implemented client written in Rust[27]

2.3.6 Monero

It was launched on April 14. 2014, as a fork of Bytecoin. Monero is a secure, private and untraceable currency, built on the Cryptonote protocol [23]. It uses a proof of Work mechanism called `CryptoNight` and does not require as big computing power as Bitcoin, mainly in order to encourage smaller miners to join the network. By now more than 16.5 million of initial 18.4 million coins were mined, and mining will go until May 31, 2022. Further on there will be a permanent production of 0.3 XMR every minute. A new block should be added every two minutes, this is accomplished by recalculating difficulty for every single block.

The main difference between Monero and Bitcoin lies once more in privacy. Monero aims to make transactions unlinkable and untraceable. This is achieved by a combination of these three approaches.

- **Ring Signature:** a signature which is created by group of signers and is used to sign a transaction. From perspective of third party, each participant is possible sender. To form a ring signature we use our account keys and selected public keys. This feature helps hide the origin of the transaction. Size of this ring can be changed according to need of users.
- **Ring Confidential Transactions:** an addition from 2015, with using this the amount we send in transaction becomes invisible for all but a sender and receiver. It is done by encryption using combination of transaction public view key and recipient's private view key.
- **Stealth address:** an onetime address generated from a recipient's public address. Transactions sent to the same receiver are sent to a unique address. The receiver gets their funds using their wallets private view key which scans the blockchain, and show them as transparent transactions. This approach helps hide recipient's funds as they re no longer located within user's wallet and are no longer linkable with user.

[25] One year from start of Zcash more than 95% of transactions had both addresses fully transparent, and therefore fully visible.

[26] Zcash wallet: https://z.cash/wallets/

[27] https://github.com/paritytech/parity-zcash

Monero has many confirmed clients[28] such as lightWallet or MyMonero which does not need ani installation, and is owned and operated by Riccardo Spagni, one of the Monero Core Team members.

For mining, there is also a full Monero client which has been recently renamed to Monero with decent GUI.

2.3.7 Litecoin

The Litecoin Network went live on October 13, 2011. It is basically a fork[29] of the Bitcoin Core client with changes which were made in order to make it lightweight. Litecoin has a four times bigger final supply of coins(LTC), than Bitcoin does and also adds block four times faster (84 million coins and 2.5 minutes). The same metric is used for halving block reward that is done every 840 000 blocks. Litecoin also uses Proof of work, but it does not use SHA256 algorithm but one called `Script`. This change was done because of Bitcoin's problem with the centralization of hashing power as a result of mining pools. Scrypt is called a "memory hard problem" since the main limiting factor isn't the raw processing power but a memory [8]. These puzzles can no longer be solved as a parallel problem but a serial one. This gives advantage to the hands of common users and therefore help keep network decentralized.

There are several implementation of Litecoin wallet such as Electrum-LTC[30] and Litecoin has also two minor implementations of a full-node, ltcd and Litecoin Core Since LC is the full Litecoin client, the initial synchronization will take time and space as it downloads the full blockchain. .

2.3.8 DogeCoin

Billy Markus from Oregon developed This cryptocurrency started as a joke. Its named after meme of Shiba Inu[31]. This currency is supposed to be taken rather a friendly community than a serious cryptocurrency. and so the main characteristic of Dogecoin is to stay as user-friendly as possible, as it derives basics properties without extra keen on anonymity[32] or any other key part of cryptocurrency. Block is added every minute, it uses Script algorithm and Dogecoin has also one of the lowest fees available.

There are two major wallets for storing Dogecoin, Ledger and Jaxx[33].

2.3.9 Summary

Table 2.5 shows information about their technical aspects. From their description, it is obvious that most of the traits come from their predecessors and are have changed only some of the key properties.

[28] official clients: https://monero.org/monero-clients/
[29] Practically more compact version of Bitcoin also called digital silver
[30] Electrum-LTC :https://electrum-ltc.org/
[31] https://dogecoin.com/
[32] Actually there is also cryptocurrency called DOGP which stands for DogeCoin Private
[33] official site: https://jaxx.io/downloads.html

Crypto-currency	**Protocol**	**POW algorithm**	**Block time /min**	**Coin**	**Total& stable supply**	**main feature**
Bitcoin	Bitcoin	SHA256	10	BTC	21M -	First crypto-currency
Ethereum	Ethereum	Ethash	0.5	ETH	- 18M/Y	Smart contracts
Dash	Bitcoin	X11	2.5	DASH	18M -	InstantSend PrivateSend
Zcash	Bitcoin	Equihash	2.5	ZEC	21M -	Shielding
Monero	CryptoNote	CryptoNight	2	XMR	- 157788/Y	Strong privacy
Litecoin	Bitcoin	Scrypt	2.5	LTC	84M -	lightweigh Beitcoin
Dogecoin	Bitcoin	Scrypt	1	DOGE	- 5 256M/Y	Joke

Table 2.5: Technical summary

Chapter 3

Sources of data and Web Scrapping

This chapter takes a close look on, how information is stored and accessible on the most known network. When we think of data and information, for sure the very first thing we think of, is the web. For better understanding, we will discuss HTML and its importance for the united representation of data and also process called web scraping, and protocols that it uses and known frameworks, such as Scrappy 3.3.5 or Selenium 3.3.6.

3.1 Public network

Although the Internet began its way as a military project, it slowly grew into bigger web[1] which linked mainframes at universities, government agencies and other important institutes [30]. Over more than the last forty years, the Internet became an inseparable part of today's society it forever changed the way how we perceive the information. Since beginning many new improvements were made such as TCP/IP model or DNS protocol, which in cooperation with all protocols we know today, allows connect more smaller networks, bring unity to formatting packets and allows millions of users communicate with knowing almost nothing about what the Internet actually is or how does it work. On top of this world wide web which allows users to start manipulating it using only browser and typing www. Word wide web is built over TCP IP protocols and consists of four main parts.

- textual format to represent all the data also known as Hyper Text Markup Language (HTML)
- simple transport protocol called Hyper Text Transfer Protocol (HTTP)
- Client to display and possibly edit these information (browser)[2]
- Server, which store these documents.

3.1.1 Data representation

HTML stands for hypertext markup language and its very first version is based on was based on SGML (Standard Generalized Mark-up Language)[3]. In 1989 Tim Berners-Lee

[1] Network built by a research team under the conduction of US defense department, "the Advanced Research Projects Agency" (ARPA) also called ARPANET

[2] the first Web browser called WorldWideWeb

[3] https://www.w3.org/MarkUp/SGML/

invented something, that we know as world wide web. As was already mentioned, with the arrival of the Internet we had suddenly the opportunity to communicate with stations and people from all around the world. But even with the Internet, the only way how to share data and link them was to make them available as files that could be downloaded. Instead of this Berners-Lee suggested that we could actually link the text into files themselves. He imagined this as a web of information with cross-references from one source paper to another one. This would mean that you could display another paper with relevant information, as one can directly contain this reference.

The idea was that the language was independent of the formatter[22], (usually browser) and display data in the intended content. To do so he had to set several rules on how to represent these data. As was already mentioned HMTL is based on SGML and uses some its language construction, such as pair tags and some concrete tags such as paragraphs or headers, but the idea of using hypertext links was purely new.

3.1.1.1 Hierarchy

The way how these tags are organized reminds us of a tree. It comes from the ability to nest HTML elements within one another. As we look at the structure of the page we will see that each element is related to another element. This is called a parent-child-sibling relationship [32]. These relations are for example parent, siblings, first child and so on. This tree starts with HTML and head or body tags.

3.1.2 Data transfer

3.1.2.1 Communication's protocol

To ensure the services of the world wide web, we have to also provide a way how to transfer data. This is mainly done by HTTP[4], what is set of rules on how to transfer all graphic, images, sound, and other multimedia files on web.
The initial version had no version number, but it has been added later, so we can find it in official documentation as HTTP/0.9. This version has only one method called GET with one parameter - path to the resource [25]. The following request can return only the file itself.

```
1 GET /Example.html
```

Version HTTP 1.0 brings many new options and information, for instance, version or user agent. While in previous version answer contained only the requested document, from version 1.0 every respond contains status code, which allows the browser to understand what is the reaction of the server.

- HTTP GET request and all optional data are in the URL. Serverreturn data or error message according to the result of processing URL and appended parameters
- HTTP POST this method is safer than GET and contains information in the body rather than in URL.
- HTTP HEAD works the same as URL but server returns only header information rather than whole content.

[4] Hypertext transfer protocol https://www.ietf.org/rfc/rfc1945.txt

Every request sent to a server generates a response, which are identified by its number consisting of three digits. From the first number we can identify the type of response (redirection, success, errors, etc.) next two digits give us more info about a particular type of response and what causes that. The five categories of HTTP's response are identified by the following order

- **1xx:** response contains information about possibility to continue sending requests or switching to another protocol.
- **2xx:** Represent all variety of successful responses. The most common response code from this category is 200, which basically stands for succeeded request.
- **3xx:** Redirection of the original request, code 301 when the page was moved or 303, when the client is redirected to a new URI.
- **4xx:** Indicate all sorts of client-side errors. For example, 400 is returned in case of bad request syntax, 404 if a page is not found.
- **5xx:** Error on server's side. Most common in terms of web scraping is 503 indicating that the server is overload.

Having this united identification of responses allows a web-based application to perform profusely adaptable running with a straightaway representation of eventual problems.

3.1.2.2 Communication's ensuring

With growing community of users and usability of web, and increasing demand to use web (including HTTP) also for sensitive data, since plain HTTP was not secured, (any of network's nodes between two endpoints could possibly be monitoring traffic and subsequently read this data), there was a need for some kind of encryption, which came with HTTPS (http over secured socket layer). HTTPS keeps information safe as it encrypts them on their way from server to browser and also backward. There are basically two types of encryption. Sender firstly encrypts data with a key, then send them via a network. When the receiver gets these data he uses his key to decrypt them. There are two types of encryption and both of them are used in order to maintain security in the network.

- **Symmetric encryption -** both ends of connection use the same key. Usually used between end station (computer) and router, as you can easily control both of these nodes.
- **Asymmetric encryption** - use two keys, secret one and public one. This is done because sending encrypted data is pointless as long as you are firstly sending a key to unlock them. By using asymmetric encryption we are using key pair composed of the mentioned public and private key, and the main idea behind them is when one key encrypts data only the second one can decrypt those data. Since the server owns one key and user another one, the connection is secured.

Both HTTPS and HTTP are publicly and widely used services, and therefore run on well-known ports[5] (HTTP 80, HTTPS 443). The latest version HTTP 2.0 became an approved standard in 2015. By now more than 50% of web use https and this number is rising as privacy is valued more and more.

[5] https://www.iana.org/assignments/service-names-port-numbers/service-names-port-numbers.xhtml

3.2 Web scrapping

3.2.1 Motivation

Over the last few decades web developed into the form, we can see today. Problem with deficiency of information no longer exists, as by typing a few keywords into a browser, you can possibly get thousands of web pages with the requested content. The second important factor is that even though there is not a lack of information, they will be probably not distributed across the network, depending on authors or providers of information. Consider that you want to buy a book, tickets, a holiday or anything else. Google search will find the best results for you according to what these web pages proclaim, not the best result (by price, time, quality or any other parameter). What leads to numbers of results without effective value and time-consuming searching, especially if we want to aggregate information from tens or hundreds of web pages. Having some tool which would collect these data automatically instead of doing so manually is exactly what web scraping is about.

3.2.2 Description

The simplest definition would be to call Web scraping (Web Data Extraction, Screen Scraping, Web Harvesting etc.) the process of extracting data from website, to be clear in practise is this term used as gathering data from website with by using some other form than interacting with server's API 3.3.7. This process of "scrapping„ is performed using a bot[6] which sends a number of request to server after what it extracts data out of the contents and then eventually saves them [39]. Typically the structure of web scraper consists of the following components:

- **crawling**: this part presents sending of HTTP requests for concrete URL or URLs by following its pattern. It also includes managing cookies, which may be demanded from the server side. Afterward website content is download and these data passed to the data extractor.
- **extractor**: from the fetched response, we firstly extract target data, which are subsequently parsed with using of relevant technique, mostly regular expressions or AI[7]
- **pipelines**: convert these data into the structured format as they always do not fit into the required format. Usually only smaller changes with data or selection of those fitting our requirements. This part also often use regular expressions.
- **Database**: saving data into database or another place[8].

3.2.3 Pros & Cons

Since web scrapers can do almost everything a human can, but in a much shorter time and without man being involved, as the script does all the work, web scraping has proved itself to be useful in many sectors. Generally speaking, we can see the use of scrapers most in the following areas.

[6] or web crawler, spider, scraper. This term is also flexible and different frameworks use different words.
[7] AI stands for Artifical Intelligence, in this case mainly neural networks
[8] Data can be stored to a file in .json .xml .csv format also

- **Research:** mainly done by scientific or research organizations.
- **Market Analysis:** includes the gathering of any form of information about users, such as their phone, email or any other personal details.
- **Price Monitoring:** scraper gather data about a specific product from more websites and compare them in order to offer the best price or any other needed information

Major companies such as Google or Yahoo have their own crawling in engines to provide the functionality of finding desired web content. Before you can start indexing web you have to crawl its content and therefore so we can assume that if web page wants to be visible it has to count with few bots, visiting its content. Part of the web which is not indexed, as the searching engine does not have access to it is called deep Web[9]. What is more, a web page can forbid some of the crawlers from crawling their content, as crawlers follow rules in the robot.txt file on which we will take a closer look in the next section 3.2.3.1.

Despite some positive scrapping application, there are many cases when scraping is considered as parasitic, usually giving too many benefits on the host's side. That unsurprisingly means that websites also often fight to protect their data using various technological, legal or ethical ways and means.

- **Terms of services :** includes a note about scraping being prohibited.
- **Changing structure of inner .html .css[10] documents often:** spiders may download the content of web page, but would not be able to get desired data. This is the simplest method as it will make bigger harm than a few rewritten selectors.
- **Behavior analysis:** there are some different aspects how bot and human behave on the website, especially considering the frequency of request, and one user does not perform the same tasks all the time, or in the same time period. Taking look at requests from specific IP address, we may deduce the pattern of the spider.
- **IP reputation :** this is done manually. Ip addresses are filtered and only those, who posses no threat are provided with an answer. This method is mostly used to filtrate spam but may.
- **Honeypots:** the links which are not visible for users but only for spiders[11], by opening the link the spider is discovered. These links have usually the color of the background or have displayed property in CSS set to None, which makes them invisible for regular users.
- **Progressive challenges:** web scraping is still weak in terms of downloading data from dynamic web, therefore in order to make scraping harder we can add extra javascript, ajax and also include extra cookies which can also serve as a barrier for bots.
- **Legal steps:** this topic may go even deeper as in recent years there were plenty of law cases, which in the end resulted mostly in favor of users. Some of the most known cases

[9] Today is widely assumed that deep Web makes up around 90% of the Internet [24]

[10] Cascading Style Sheets: https://www.w3.org/Style/CSS/Overview.en.html

[11] Spider is name how is, the script for scratching concrete website or websites called, this name was given according to fact that many spiders can work in parallel and spanning a large area of the "web"

are eBay v. auction compiler Bidder's Edge[12] method [17], or Feist Publications v. Rural Telephone Service CO., which ended with scraping and republishing telephone listings facts, established by The US Supreme Court[13].

It is important to understand that none of these steps cannot prevail possible spiders from success, they only make scrapping more difficult and time consuming, and therefore the more of these measures we set, the more skilled programmer and effort it takes to scrape the site.

Steps which scrapers make in order to not to be caught as a bot, can be simply described as quest to look as much as regular user or group of users as you can. This can be done with help of few techniques which practically eliminate all sorts of protection which the web site dispose with.

- **Irregular time between requests:** we do not use a full speed of crawling and make spider slower. Adding some extra random time period between requests gonna help our spiders gain even more human like behaviour.
- **crawling pattern:** bots follow logic and therefore there is some pattern which is always repeated. Human do not have tend to repeat things in exactly same order. Therefore bot must not have some easily visible pattern of behavior. We may possibly write more spiders for the same page, in order to hide our true aims.
- **Rotating IPs:** firstly have to create a pool of IPs addresses, and then randomly choose a new one for every request or set of requests. By using this method, it appears that requests did not come from the bot but from a group of people, most common way of doing this is via proxy servers[14].
- **Spoofing User Agent:** since HTTP 1.0 requests contain user agent description in it headers. This was originally invented in order to provide specific operating systems and devices with better-formatted data, in other words, every program. including browsers, search engine web crawlers have this specification. Sending many requests with the same user agent makes bot absolutely visible. This field can be changed and replaced by a user agent of the browser, or any other commonly used such as Google bot's user agent name [35].
- **Avoiding honeypot traps:** Any traffic initiated by the honeypot means the system has most likely been compromised and the attacker is making outbound connections [29]. Honeypot still has some weaknesses such as they have some specific characteristics, and once discovered honeypot may become of a weak part of the network. Also, there is still a wide misunderstanding and not clear definition about this term, which decrease the spread of this security tool.
- **following file robots.txt 3.2.3.1:** this is also one of the aspects according to which we distinguish[15] between spiders and crawlers.

[12] Case over republication of auction data. ,eBay, Inc. v. Bidder's Edge, Inc., 100 F. Supp. 2d 1058 (N.D. Cal. 2000).

[13] https://caselaw.findlaw.com/us-supreme-court/499/340.html

[14] Proxy servers act on behalf of a Client's computer, from the view of network Proxy server is considered to be the endpoint of communication.

[15] spiders ignores robots.txt identifies itself as a browser and sometimes even execute javascript.

3.2.3.1 robots.txt

This text file is located in the root of the website and contains rules wrote in specific syntax which describes, how the bots are allowed to work with this website. This file is usually first to be download by bots. By following these rules, we have full certitude our bot will not be persecuted, or blocked. A simple example shows the syntax.

```
1 #all user-agents have no access
2 User-agent:*
3 Disallow:/
4
5 User-agent:Agent1
6 Disallow:/home
7 Allow:/users
8 Crawl-delay:5
```

only Agent1 is allowed to crawl data but only from users directory and he can make one request every 5 seconds, all the other user agents have no access. This delay is the main option of how to prevent bots from causing DOS[16]. Many web sites have a similar structure of the robots.txt file because they want to grant extra rights to Google's crawler.

3.3 Frameworks, Tools and Libraries

As was mentioned in the previous subsection3.2, the whole process of scraping is made of four main important parts, and there are several tools that can help us to write these procedures without investing too much time and effort.

3.3.1 XPath

XPath is a language that describes how we can locate element, group of elements, or any branch of data which are organized in Extensible Markup Language (XML). As both XML and HTML are markup language XPath is serviceable also for searching in HTML.

XPath uses syntax which allows us to write simple instructions, which simply represent direction and selection inside the hierarchy tree. The result of this selection may be a new tree, group of elements or value. We can use a built-in function, values of attributes, the existence of children any many others expressions.

It is a good practice to write XPath selectors with relative paths, because an absolute way consist of too many nodes, and change in any of them may make our XPath selector useless. Also, try to avoid using the order of elements. We want to write these expressions as generally as is possible.

XPath is usually used by programs and libraries which are oriented around aggregation, presentation, and collection of information such as web scrappers.

Another approach is to use CSS selectors, but as many web pages use bootstrap, changes in these structures are more likely to happen.

[16] Denial of service, caused by an enormous number of requests from bots.

3.3.2 Regular expressions

Also called regex or rational expression is a special text string that defines a search pattern. They may seem quite complicated on the first look, mainly as these strings often seems like nonsense. In reality, regular expressions can search almost for everything, URL matching, supporting search, matching email addresses, extracting substrings, including cryptocurrency addresses. In the area of data extraction, they are practically priceless, as they simplify otherwise complicated matching patterns.

We can use them in almost every well-known programming language including C++, Java, and Python.

3.3.3 Beautiful Soap

Its name comes from poem sung in Alice's Adventures in Wonderland [24]. This python library is mostly used to navigate and parse once downloaded data in HTML or XML as it provides simple, idiomatic ways of navigating, searching, and modifying the parse tree [26]. If a given document does not contain any mistakes, a parsed structure looks the same, but in case there are some broken parts, Beautiful Soup uses heuristics to fix them. There are also two variants of selectors, via CSS and XPath. BS coverts sometimes hardly readable HTML to python object and makes selection easy to use even for somebody, who see HTML for the first time. The newest version is Beautiful Soap 4. Although using this library is quite straightforward, it is not the best choice for more complex projects[17].

3.3.4 Portia

This tool does not need any programming skills. It is created by Scrapinghub and it is possible to use for free via a hosted version. Instead of writing code, Portia only repeats actions which user performed, so it is really useful to crawl ajax powered websites. The biggest pros are that it is user-friendly and does not require any programming skill, and also filters the page. User is allowed to use CSS or XPath for selection of data. The biggest disadvantage is a big time consumption in comparison with other web scrapers [40].

3.3.5 Scrapy

Open source and collaborative web crawling framework, and is built on top of Python's library. It can always be used with other libraries, BS4, or Selenium. One of the strongest sides of Scrapy is that projects have their own predefined structure. Once we understand its architecture it is easy to manage, even when the project becomes larger. Scrapy also does have its own shell, which is a powerful debugging tool. Same as beautiful Soup, Scrapy has a great tolerance towards broken HTML structures.

3.3.5.1 Architecture

labelssub:scrappy Architecture of srapy is shown in Figure 3.1. Main components are described in following lines.

- **Spiders:** custom classes written by users, they define, how one website or group of them will be scrapped, and also how to parse received data. Every spider has

[17] Library itself is often used with Request library and smaller projects which scrapes only a few web sites

name, *starting_urls*, what is list[18] of URLs, (web pages which content is requested as first), and implemented method *parse(self, response)*, which is called at the moment response is received[19].

- **Engine:** this component is in-between for other parts of the architecture, is mainly responsible for communication between components.
- **Scheduler:** enqueues requests from the engine and returns the first one, when engine asks. Requests may also have priority what is key that scheduler uses if there is more than one request.
- **Downloader:** handles the sending and receiving of data.
- **item pipeline :** final procedures that usually stores, updates or deletes data which came there as a result of spider's selection.

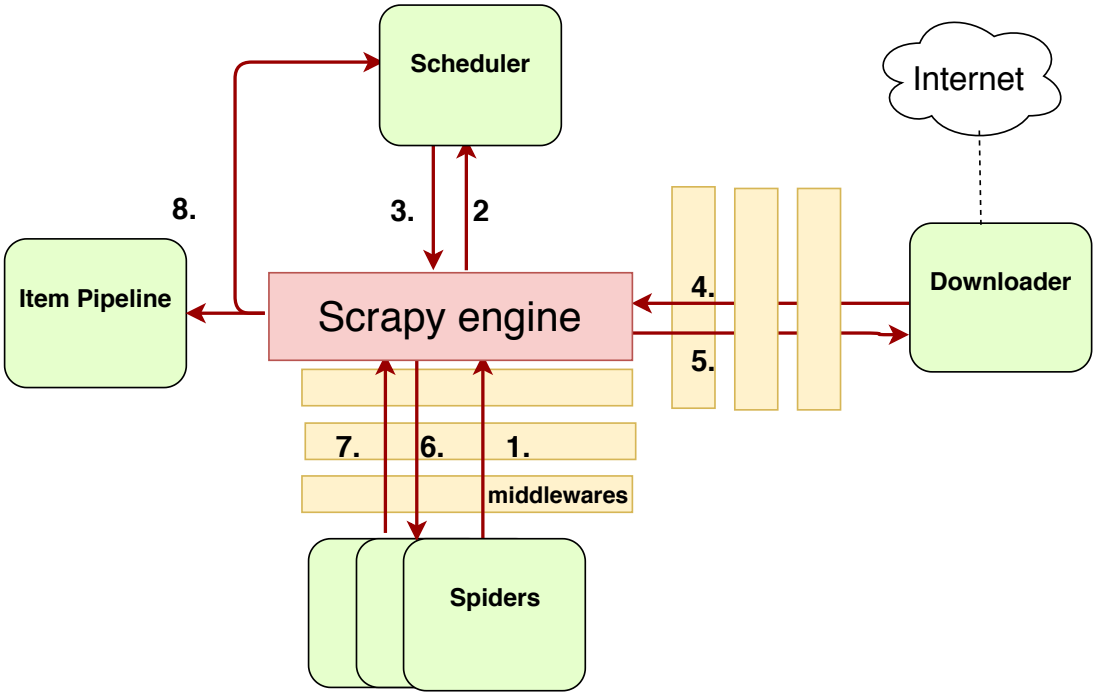

Figure 3.1: Scrapy architecture

We can follow the flow of control in scheme[11]:

1. Spider sends an initial request to the engine.
2. Engine schedules request.
3. Once a request meets conditions scheduler return first request to the engine.
4. Request is being sent to the downloader.
5. Once a page is downloaded response is returned to the engine.

[18]list is python structure.
[19]Scrapy is event-driven framework

6. Engine sends a response to the spider.
7. Spider parses the response and returns item/s or new Request/s.
8. Item is sent to pipeline (usually to store to Database) and possible new requests to scheduler

This was a simplified version showing the flow of control required to handle one request. Because of fact, Scrapy's architecture is built with Twisted, an event-driven networking framework, it handles requests concurrently what allows to perform a high volume of requests per minute. Once a request is sent Scrapy does not wait for a response but send another[20].

3.3.5.2 Settings

labelssub:settings By creating new Scrapy project we also create a file called settings.py which contains a bunch of variables. These variables represent settings that are applied for every single spider in the project. Settings determine the way our scraper works. These settings have an impact on different parts of the project, as we may set the order in which are called pipelines, middlewares, or prevent the spider from visiting file robot.txt before sending the first request for data. In generally some of the most common areas, we may affect from Scrapy's settings file are the following: Performance, ending conditions proxying, media download, crawling style. As the application does use Scrapy framework we return to most of these settings in the Chapter Design 4.

3.3.6 Selenium

Selenium is a web testing library that is mainly known for its big usability, even if the website does contain a lot of javascript functionality. This tool was originally developed for website testing. Selenium does not have our browser and in order to perform the tasks, it has to use a third party's browser (chrome, opera, etc.). By running it you can literally see navigate it throw the browser so the user can watch the code being executed.

3.3.7 API

Speaking Technically, API stands for Application Programming Interface. At some point or another, most large companies will eventually provide some kind of API. From whatever reason, it may be for their customers, or for internal use. One of the benefits of having API, is that it provides sufficient enough way to gather data for most of the cases when we would otherwise consider pure web scraping. To provide a response on API request is for server much easier as these data are in JSON or XML format, without plenty of HTML, etc. That subsequently leads to fever requests to the server from bot, as it can get all the data within a few responses.
For users, it may come handy in case we do not need streams of responses, or data which server may not want to provide. Although using APIs isn't generally considered web scraping, it may be useful in most cases. Especially because of the fact that using server's API usually takes few lines and does need such a big investigation of site, its structure or, possible threats for our spiders not mentioning that we no longer depend on the DOM

[20] Unless is not supposed to, and should wait before next sending the next request, as there are options to throttle volume of requests.

structure of a page. To sum it up, we should always consider using API before starting writing spiders.

Chapter 4

Design

In this chapter, we will move through the design of the application which can automatically scan part of the public network (popular forums and websites), and extract information about users of cryptocurrency.

4.1 Data resources

Because of the numerous properties, Cryptocurrencies are implied part of many areas on the web. In most cases, users do not want their real identity to be linked to address. Besides excluding black markets (mostly they are not part of the public web), we may also exclude services with possibly illegal character as well. Services as exchanges and wallets are directly connected with users assets, and therefore, also do not share much information about users. This left us with donation platforms, forums, gambling sites, mining pools, and social media.

4.1.1 Forums

Cryptocurrency's forums are in constant change as the market constantly evolves. New changes, technologies, and even cryptocurrencies are announced daily, which logically leads to numbers of forums specialized on some of these areas or currencies.

- **Masters of Crypto**[1]**:** is a new Bitcoin forum community which tries to bring newest and the most useful tips, trends, upcoming launches and so on. It also encourages users to post material by paying them according to the relevance of their posts.
- **Bitcointalk**[2]**:** is created by Satoshi Nakamoto, one of the first cryptocurrency forums and is available in many languages. Bitcointalk is built exclusively on Bitcoin[3]. In fact, Bitcointalk is a larger cryptocurrency forum, than the rest of them combined.
- **Cryptorum**[4]**:** Is one of smaller starting forums,

[1] https://mastersofcrypto.com/forum/

[2] https://bitcointalk.org/

[3] There are also some threads for other currency but their size is just a fraction compared to Bitcoin's thread.

[4] https://cryptorum.com/

4.1.2 Charity

2017 was a record year for crypto donations [5] since 2009, there was slow but considerable growth in this area. In the nutshell, cryptocurrency donations work almost the same way as usual donations (we choose an organization, project, hide or show our identity). However, there are some extra advantages of using crypto, instead of FIAT money. The most important is the trust of users, as crypto donations offer possibilities to see, what is the donation used for, and by using smart contract user can ensure even more funds will be used for agreed purposes, as he can set contract's restrictions.

The Web page https://cryptogivingtuesday.org/ provides a list of donation's platforms and charity organizations that accept Bitcoin and other cryptocurrencies. Whereas mentioned charities only extend opportunities for donations, platforms usually offer transaction history of donation, and names of donors[5].

4.1.3 Gambling

Gambling is also one of the areas, which may cryptocurrency users find interesting. Since everything is powered by smart contracts, and anyone has access to the blockchain, it is not hard to see what is going on behind the scenes. This means that just like in the case of charity, having the opportunity to see into contracts or the possibility to see transactions, may give the site much bigger trustworthy. Gambling platforms usually describe itself as "Provably fair"[6]

Sadly this area does not fit our too much for scraping data, as these platforms usually work with one-time generated addresses, what stays the same even in case of betting web pages. There are only a few gambling websites that provides a list of addresses that are used as a bet deposit. One of such websites is satoshidice[7]

4.1.4 Social media

Approximately 80% of Internet users are also active social media users. The average daily time spent on social is 116 minutes a day [28], these values implicate for us two main things. Firstly, there is no chance to scrape every single page of any social media and not to find a single cryptocurrency address. Secondly, there is no chance to scrape every single page of social media[8]. We can handle complications with further investigation of selected social media, and define local parts of social media, by any means the server provides.

- **Youtube:** YouTube is the 3rd most visited website in the world. Youtube stores videos, comments, profile information, and some additional data. It is heavily loaded with javascript and AJAX and therefore not very easy to scrape. Most of the content is accessible even without logging. As there are not some organization of videos due to their category or something like that, the only relevant option of sorting videos is via searching formula. This is actually exactly the case when can be previously mentioned topics connected with cryptocurrency really useful, as we can use them as keywords in the youtube search engine.

[5] Unless the donor requested to remain anonymous

[6] What means that house does not make users unfairly loose.

[7] https://satoshidice.com/

[8] figuratively speaking, in most cases we have neither enough time nor resources to perform such tasks.

- **Reddit :** Has many nicknames but one of the famous is "The Front Page of the Internet". This title has a straightforward informative value about the size of this social media. Reddit, same as youtube, has a search engine of its own and therefore allows us to navigate spiders using keywords. Reddit is also divided into smaller categories called "subreddits". These two selectors can be also combined.

  Reddit also uses advanced technical tools to prevent web scraping. On the other hand, the server does provide very simple to use API.

4.2 Database

The database of this application needs to store information about users, and addresses they use. In this section, we will describe all the information that we will store and the way how we store them. The database scheme is shown in Figure 4.1 .

- **Owners**: table *Owners* contains information about owner of bigger amount of addresses (organizations).
  - *id_owner*: primary key.
  - *name*: name of owner, for example *crypto.com*.
- **Categories:** The table *Categories* contains information about category of addresses. Address gets its category according to nature/reputation of website it is located on.
  - *id_category*: primary key.
  - *name*: name of category.
  - *color*: this property informs us about severity of category. This is implemented as enumeration.
  - *created_at*: is date of creation.
  - *updated_at*: is date of update.
- **Identities:** The table *Identities* contains information about identity of user.
  - *id_identity:* primary Key.
  - *id_address:* foreign key to table *address*.
  - *source:* information about origin of data. For example Pirate Bay ...
  - *label:* nick, name, email or other identification.
  - *url:* reference on web page, where data were found.
  - *created_at*: is date of creation.
  - *updated_at*: is date of update.
- **Address:** The table *Address* contains information about addresses. .
  - *id_address:* primary Key.
  - *id_owner:* foreign key to table *owners*.
  - *address:* string that represent address.

- *crypto:* information about which currency's exactly this address is.
- *created_at*: is date of creation.
- *updated_at*: is date of update.

Relationships:

- *Address categories :* N to N relation. Address can belong to more categories and other way round. One extra table is needed.
- *Owner address :* 1 to N relation. Owner can have more addresses.
- *address Identities :* 1 to N relation. One address may have more identities.

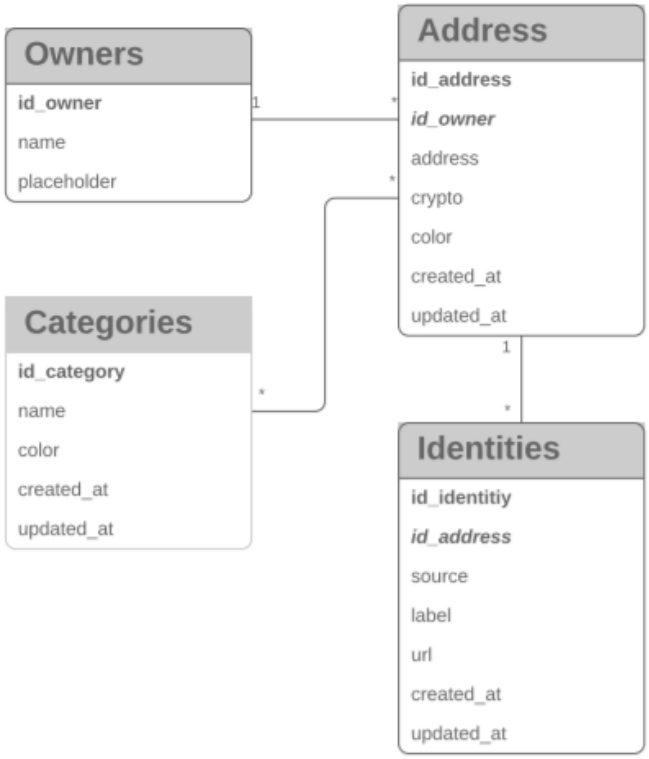

Figure 4.1: Database schema

4.3 Application

The main purpose of this application is to browse across selected parts of the public web and collect information about users of cryptocurrency. The result of this process of searching is a database with addresses, their user's identification and presumed the purpose of using. The application needs to be executed periodically in order to get new information or eventually

update those old. Application extends architecture from Scrapy, which is shown in Figure 3.1. I chose this approach mainly, because of scalability and powerful tools that Scrapy provides to accomplish a suitable flow of concurrent requests. Complete scheme is shown in Figure 4.2.

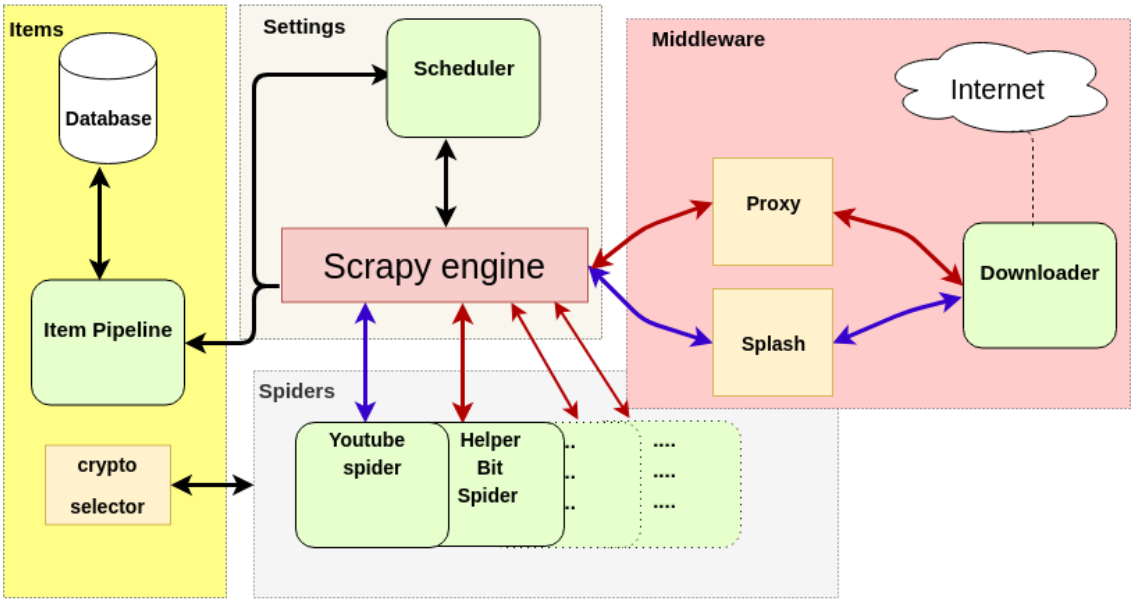

Figure 4.2: Application schema

Individual parts of application's architecture from Figure 4.2 are fully described in following subsections

4.3.1 Spiders

Every spider represents a pattern on how to scrape the concrete web site. When the application starts, every spider sends an initial request to the Scrapy engine, afterward spider only wait to be awakened when a response comes. This type of behavior comes from twisted architecture as these processes have their specific callbacks.

The difference in color between youtube spider and others in Figure 4.2 only indicates that youtube spider needs javascript to be executed and therefore send different type of requests.

Once the response comes, the spider is awakened and starts processing the response. Spider extracts required data by using XPath selectors, and store these data in a class called item, what is class providing container like behavior for extracted data. afterward, the spider may continue sending requests or populate newly created item[9] to the item pipeline.

4.3.2 Middleware

There are two important in-betweens before a request/response reaches its destination.

[9]We may also use a dictionary but items are a more convenient choice as dictionary lack structure.

4.3.2.1 Proxy

A proxy[10] server acts as an intermediary between an endpoint device, in our case between computer which is running this application, and server to which we are sending requests[27]. An advantage that this brings to us, is that our application seems to a server as a group of users, which are all asking only for a fraction of the server's document. Since there is a longer direct connection between our application and server, we are also risking being banned.

But there is also the other side of the coin. Having in-between on the data transfer way means that data we are sending are available for this third side. What is more using proxy may become also time consuming as there is a need to send all traffic firstly to proxy server, and only afterwards to one of endpoints, secondly unless we use paid proxy server we will probably have to wait even longer as free proxy may often use not suitable address and therefore has to repeat request.

4.3.2.2 Splash

Some web pages may execute a lot of javascript, what is something Scrapy cannot handle very well. You can assure yourself of that for example with help of chrome extension Quick Javascript switcher[11]. Now, if you visit web site like Youtube.com and switch javascript off, you, just like your spider, will not be able to see most of the expected elements. Therefore we will use Splash, what is an open source lightweight browser capable of processing multiple pages in parallel. We will run an instance of Splash[12] using docker and communicate by using its HTTP API. Javascript and costumed behavior (to scroll, click, wait...) can be set via Lua scripts[13]. Executing those scripts may be time-consuming, but we would have nothing to parse without having those responses firstly rendered by Splash take some time.

4.3.3 Items

Items are created only in case, some cryptocurrency address was found. This scan for addresses is made in "crypto selector" by using a set of regular expressions, which are matching the exact format of concrete cryptocurrency.

The moment spider populates the item it is going to item pipeline. The architecture allows us to write a series of pipelines where we can edit the values of items. Scrapy allows us to write functions called processors. These processors are a series of small functions that format data into an expected form (removing white spaces, joining more values together, etc.). This helps us keep pipelines straightforward, as we know that values in an item already meet expected format and therefore do not need to perform this formatting anymore.

In case of item manages to get to the last pipeline, values in the item are inserted into the database, or update the corresponding record in case it already exists (*column updated_at*).

[10] Original meaning of the word is originated from early 15c., proccy, prokecye, "agency of one who acts instead of another".

[11] url:https://chrome.google.com/webstore/detail/quick-javascript-switcher/geddoclleiomckbhadiaipdggiiccfje

[12] As the project will grow, we may run even more instances and distribute requests using load balancer, for example, http://www.haproxy.org/

[13] Lua is designed to be a lightweight embeddable scripting language, https://www.lua.org/start.html

4.3.4 Settings

This part of architecture has two key roles, first is managing all spiders, the number of concurrent requests per domain, ordering and allowing of pipelines and middlewares. There is already a prepared file for those settings, so this application will mainly only override default settings.

The second role is to manage bigger spiders, with a possibly infinite lifetime. As we already discussed in section 4.1, some web page has a size that makes it impossible to scrape its content in relevant time. Therefore we will control this flow from .conf files.

Chapter 5

Implementation

Due to choice of scrapy as main framework for this application, architecture and implementation build on the templates provided by scrapy. Following sections will describe essential parts of implementation and dependencies, application requires. As scrapy alone does depend on several libraries, and there are also few more added, virtual environment is created, in order to keep track of all necessary libraries, thus, only those essential for specific task are mentioned within following sections.

5.1 Configuration

5.1.1 Splash

Communication between spiders and the lightweight browser is enabled using scrapy-splash library[1]. Therefore we need to have a running instance of Splash. This can be done using docker, by typing following command.

```
$ docker run -d -p 8050:8050 --memory=5.2G --restart=always scrapinghub/
    splash:3.1 --maxrss 4000 --max-timeout 500
```

Besides creating Splash's instance, there are also a few more parameters and ports on which communication is performed, the timeout for requests, and limitation of space Splash can use, as Splash uses an unbound in-memory cache eventually consume all RAM and afterward crash completely. By using memory limitations and restart instruction, we are giving docker instruction to restart the instance of Splash, and by doing so clean its cache. Restart may take up to one minute, during this time all sent requests result in timeout warnings, for this case scrapy provides a setting called `RETRY_TIMES`, which allows to every request to be sent multiple times if response code matches expected value.

5.1.2 Concurrence & reliability

Scrapy supports running multiple spiders per process, using the internal API. Every spider located in this project is added to the crawling process. The moment crawling process is called, the scheduler starts receiving requests from all spiders simultaneously.

The following settings/approaches allow the script to perform scraping concurrently without hitting the server too hard or being revealed.

[1] https://github.com/scrapy-plugins/scrapy-splash

- *CONCURRENT_REQUESTS_PER_DOMAIN:* regulate maximal volume of requests sent to single domain.
- *USER_AGENT:* user agent is sent together with our HTTP or https requests, its purpose is to tell a server what the visiting device is. The server can use this information to determine what content to return to the node, that is asking for data or even refuse to return any content. This script identifies itself, as a browser, as some of the aimed web pages refuse to communicate with bot.
- *DOWNLOAD_DELAY:* is time in second how long should downloader wait before sending another request. This value acquires value from a random interval between 0.5 and 1.5 * `DOWNLOAD_DELAY` [2].
- `Different behavior`: there are two implemented types of behavior for browsing youtube, first one wait for all possible data, as some URLs do not need to be crawled so thoroughly, there is also lightweight script, which does not wait for loading and rendering of some parts of the site.
- `Xpath verification test`: more about them in caption 6.

Library scrapy-proxy-pool[2], that provides service of Proxy, is also included but it does not increase the rate of requests as most of requests have to be sent repeatedly. By deffault proxy midleware is not called.

5.2 Crawlers

5.2.1 Spiders

Every single web site has a different structure. Therefore, there is a need to create a specific spider, that can follow its structure. Spider contains all the logic needed to download and extract data. Before writing spider there is a need to spend some time exploring the structure of the target web site.

The situation is slightly more complicated in case of scraping youtube, as we need to define a smaller area within the web site.

File "youtube.cfg" provides the environment, where user can type string which will be passed in form of requests to the youtube search engine. Web page responses with a list of video results and those serve as a substitution for classic *start_urls* list. User may also set, how many results he wants via integer value in range 1 - 100. Besides this user may also enter the name of youtube channel which will be crawled completely. Syntax With example are shown details in Appendix A.

There are three important *parse_functions* in youtube spider.

- *parse_page_full*: requests all data from page.
- *parse_page_comments*: requests only comments.
- *parse_expedition*: requests description and part of comments.

[2] https://github.com/hyan15/scrapy-proxy-pool

Once URLs from video search are provided, the spider can start its search for cryptocurrency addresses. Response from every single URL calls function *parse_expedition()*. The response contains a description of the video and up to fifty comments. The function gets values that are shown in the following pseudo-code.

```
#parse_expedition pseudo code
parse_expedition(self, response):
    ...
    crypto_in_comments = is_crypto_in_comment(response)
    crypto_video = is_crypto_in_video_description(response)
    comment_sum = is_more_than_fifty_comments(response)
    ...
```

The spider now knows if there are some cryptocurrency addresses in the first fifty comments, or in the description and whether or not there are more than fifty comments. According to these values, function decides whether or not is needed to call functions for scraping full page. The biggest advantage of firstly calling *parse_expedition* is, that there are many videos with thousands of comments, but without a single cryptocurrency address, so rendering it would be just a wasting of resources. This approach built on the heuristic that if there is one cryptocurrency address in comments, there will be probably more.

The main part of searching for addresses is implemented as a series of regular expressions. The table with all these expressions can be found in appendix C. Besides regular expressions, there are also some additional checks, such as control of white spaces around an address or partial ambiguity in the case of Bitcoin and BitcoinCash legacy format.

5.2.2 Reddit bot

It is the simplest solution to use API if the server provides one. Reddit provides API (PRAW) which allows bots to download data in JSON like format. In order to use API, module PRAW[3] (Python Reddit API wrapper) has to be imported. This module provides all functionality bot is going to use. Secondly, we have to have an account on Reddit to generate a key for the bot.

Bot uses following classes:

- **Reddit:** instance of this class is returned once we call function Reddit and send username and key as arguments within the function.
- **Subreddit:** Reddit is categorized into subreddits. An instance of this class is obtained from an object of class Reddit (We have to firstly identify subreddits, we want to scrape, Bitcoin, Dogecoin, etc.).
- **Submission:** is class which represents one topic in subreddit. The object is accessible from the Subreddit class.
- **Comments:** object of this class, are accessible from Submission class or another Comment class

[3] https://github.com/praw-dev/praw/tree/master/praw

relations between these classes/objects, are shown in Figure 5.1, bot can iterate over 1000 of submission from every single subreddit and load comment, no matter how deep they are. As is shown in Figure 5.1, comments may plunge one into another.

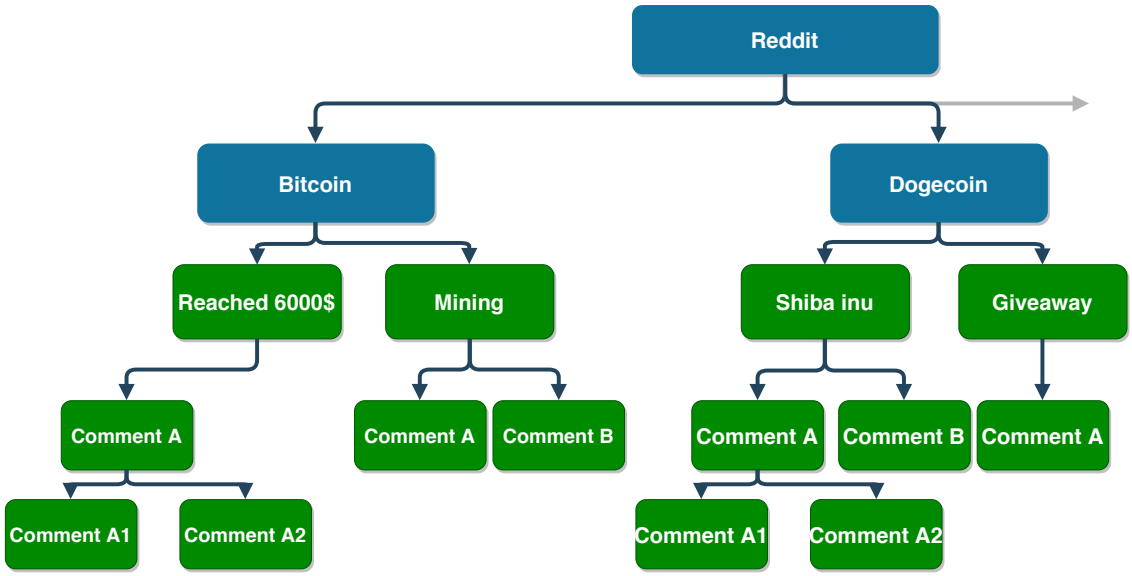

Figure 5.1: Reddit data organization

Subreddits and searching keywords are loaded from `reddit.cfg`, once loading is done, the bot tries every combination of subreddits and keywords. In order to not exceed downloading restrictions, the bot is waiting for two seconds per requested submission.

Chapter 6

Testing

In this chapter, we will take a look at crucial tasks, which have to be done correctly, in order to use this script and data it provides with absolute confidence. Most of the tests are written using selenium. Dependencies and complete guidance to run these tests are included in the README.txt file.

6.1 Verification

6.1.1 XPath

One of the weakest parts of web scraping is that navigation through the web site uses selectors based on HTML structure. Even a smaller change in this structure may break spiders. To test results that script provided is a process, which can prove that the script was working well in the time tests were performed, but there is no assurance about correctness next time. This problem can be removed by using `selenium tests`. By opening page dynamically we can test XPath selectors before starting scraping. This way we can have confidence, that selectors work properly, and eventual errors are not caused because of different page content. This way user is aware of a spider that is not working properly and also which XPath needs to be replaced. The following tests perform this check for concrete sites.

- *TestBinance*
- *TestBithope*
- *TestYoutube*
- *TestConcreteVideo*[1]

Despite Scrapy provides basics set of tests for every function, selenium is a perfect tool for running tests in the web environment. Tests are stored in separated files, and organized per domains, so as the times go on and the project grows, it will be still possible to maintain clarity in file organization. There is also an implemented *TestRunner*, for the purpose of running all these separated tests collectively

[1]There are more tests for youtube as it is the by fat largest website among those used in the project

6.1.2 Address selection

Once spider/bot provides a response and extracts targeted data, the script may start to search for addresses. *TestCryptoAddressValidation* is a unit test that can be used to verify the reliability of the searching engine. It uses fictive message which serves as input to test if the search engine matches all expected addresses. Addresses are assigned to types of cryptocurrency and in the case of the test not ending expected way, it hints user which regular expression is probably not working good.

6.2 Validation

Once the script stops running, we may check, if records stored in the database truly exist on the concrete website. Besides the manual approach, we can test again with the use of selenium. This test joins table address and table identity. Then, it visits the web page on our behalf and looks for it. In the case that it did not find the address, it informs about it and ends test. This way we can check the validity of most of the records from the database, but as this test has to load all these pages and execute custom javascript it may take a long time. The average time of validating 100 addresses is 13 minutes. Still, this test is fully automated and much faster than doing it on our own.

6.3 Performance & Results

Most of the visited websites consist of at most few hundreds page. In that case, the spider ends its work after a few minutes. On the other hand, bot and youtube spiders are slowed down by limit restrictions, or necessity of executing javascript[2]. What is more, the performance was also decreased as we did decrease the number of concurrent requests per domain in order to avoid getting banned. Following Figure 6.1 shows more details about addresses.

[2]including waiting for responses generated by asynchronous javascript (AJAX)

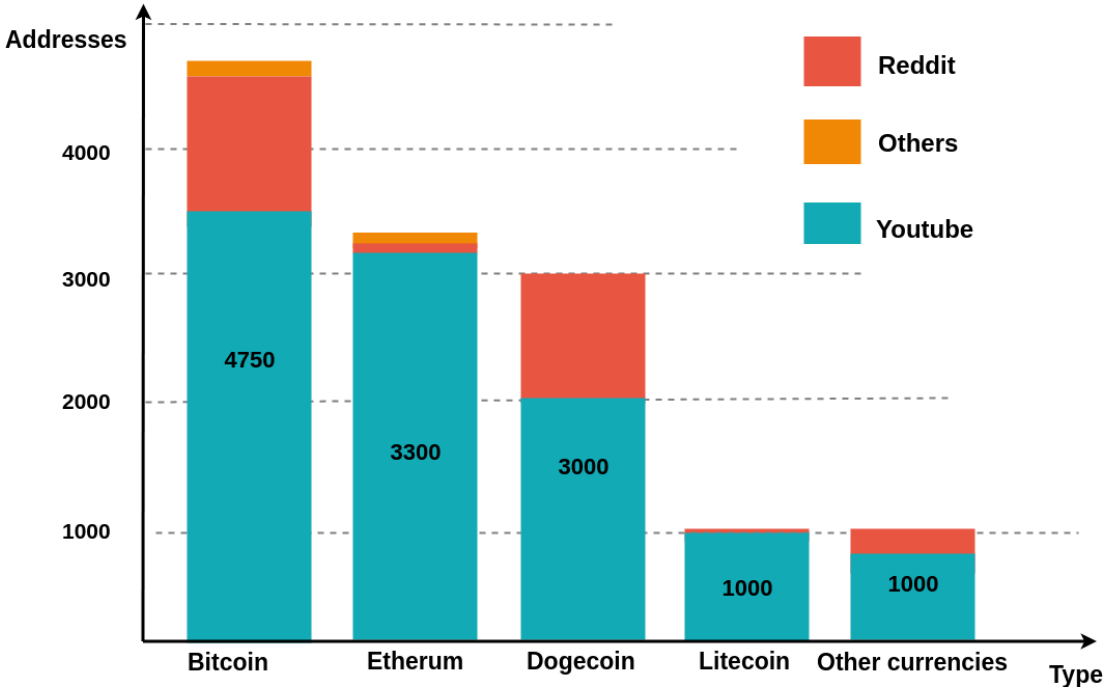

Figure 6.1: Addresses, their types and source

We can clearly see that Bitcoin, Etherum and Dogecoin have much more addresses than 12[3] others cryptocurrency together, what is probably a result of their overall public known. all spiders and bot, were collectively capable of extracting more than 13 000 unique addresses within the first 32 hours.

[3] Neo, Ripple, Zcash, Snowgem, Burst, Bitcoin Cash, TRON, Monero, Stellar Lumen, Dash

Chapter 7

Conclusion

This thesis' main goal is to collect information about users of cryptocurrency from the public network. The application collects data mainly by the use of web scraping (specifically framework Scrapy). Due to the type of service application that is supposed to provide, it is necessary to know where to search for data. In the thesis, we infer this information from the properties of cryptocurrency. By doing so, I could easily identify areas which include cryptocurrency, speaking plainly donations, gambling or social media, this all is described in section 4.1.

The application solves setback of need for execution of JavaScript, by use of Splash, which may run locally, but application considers it to be intermediate. The results of this thesis are two things – application and database. Usability of the first one lies besides already mentioned, in its architecture and automatized tests which are able to detect errors in XPath expressions. By doing so, they help us get rid of architecture's Achilles heel, and thus allow the application to stay usable long past its expiration.

Usability of data infers from the character of blockchain and pseudo-anonymity of cryptocurrency. Results from the first scrapping show that most of the obtained addresses come from cryptocurrencies, which do little for shielding transactions.

By putting all these factors together, we can use this database to reveal the true identity of owners of accounts, depending on the amount of data, it would be possible to replace addresses in transaction history with true identities and thus have not just information about ownership of an account, but complete transaction history.

This all makes application usable in the financial, criminal or informational sphere. Application is developed as a part of project Tarzan [1] an integrated platform for the analysis of digital data from security incidents.

The application can be extended by adding more spiders or a more sophisticated way of deciding the type of address. For now, it only considers the source site and location within it. In some cases, the application needs to recognize if the user recommends address belonging to another organization, etc. Secondly, add some tools for web archiving, such as Lemmiwinks, which was implemented as part of Serečun Viliam's master thesis[34].

Bibliography

[1] *Integrated platform for analysis of digital data from security incidents.* [Online; visited 12.05.2019].
Retrieved from: http://www.fit.vutbr.cz/units/UIFS/grants/index.php?id=1063

[2] *Settings.* [Online; visited 03.05.2019].
Retrieved from: https://docs.scrapy.org/en/0.16/topics/settings.html

[3] Ben-Sasson, E.; Bentov, I.; Horesh, Y.; et al.: Scalable, transparent, and post-quantum secure computational integrity. Cryptology ePrint Archive, Report 2018/046. 2018. https://eprint.iacr.org/2018/046.

[4] bitcoindata.org: *bitcoindata.org.* [Online; navštívené 9.12.2018].
Retrieved from: https://bitcoindata.org/sk/co-je-kryptomena

[5] Blenkinsop, C.: *Blockchain in Charity, Explained.* [Online; visited 27.04.2019].
Retrieved from:
https://cointelegraph.com/explained/blockchain-in-charity-explained

[6] blockgeeks.com: *ethereum-gas-step-by-step-guide.* [Online; navštívené 24.01.2019].
Retrieved from:
https://blockgeeks.com/guides/ethereum-gas-step-by-step-guide/

[7] blockgeeks.com: *Smart Contracts: The Blockchain Technology That Will Replace Lawyers.* [Online; navštívené 24.01.2019].
Retrieved from: https://blockgeeks.com/guides/smart-contracts/

[8] blockgeeks.com: *What is Litecoin? The Most Comprehensive Guide Ever!* [Online; navštívené 24.01.2019].
Retrieved from: https://blockgeeks.com/guides/litecoin/

[9] Charles Brennan, M. Y. W. L., Brad Zelnick: *research-doc.credit-suisse.com.* [Online; navštívené 20.01.2019].
Retrieved from: https://research-doc.credit-suisse.com/docView?language=ENG&format=PDF&sourceid=csplusresearchcp&document_id=1080109971&serialid=pTkp8RFIoVyHegdqM8EllLNi1z%2Fk8mInqoBSQ5KDZG4%3D

[10] Curran, B.: *blockonomi.com.* [Online; navštívené 21.01.2019].
Retrieved from: https://blockonomi.com/trustless-environments/

[11] Documentation, S.: *Scrapy architecture.* [Online; visited 27.04.2019].
Retrieved from:
https://docs.scrapy.org/en/latest/topics/architecture.html#data-flow

[12] Dominik Stroukal, J. S.: *Bitcoin Penize Budoucnosti*. Liberální institut. 2016. [Online; navštívené 24.01.2019].
Retrieved from:
http://libinst.cz/wp-content/uploads/2017/02/Bitcoin_for_web.pdf

[13] Frankel, M.: Many Cryptocurrencies Are There? Bitcoin, Ethereum, and Ripple are just the beginning. [Online; visited 28.01.2019].
Retrieved from: https://www.fool.com/investing/2018/03/16/how-many-cryptocurrencies-are-there.aspx

[14] G.F: Why bitcoin uses so much energy. *www.economist.com*. 2018.

[15] Gilkes, P.: *Coin World*. [Online; navštívené 9.12.2019].
Retrieved from:
http://www.libertydollar.org/news-stories/pdfs/1166043540.pdf

[16] Gupta, M.: *Blockchain For Dummies*. John Wiley Sons, Inc.. 2017. ISBN 9781119371.

[17] Hirschey, J. K.: *SYMBIOTIC RELATIONSHIPS: PRAGMATIC ACCEPTANCE OF DATA SCRAPING*. [Online; visited 25.01.2019].
Retrieved from: http://btlj.org/data/articles2015/vol29/29_AR/29-berkeley-tech-l-j-0897-0928.pdf

[18] https://medium.com: *Dash Course*. [Online; navštívené 24.01.2019].
Retrieved from: https://medium.com/dash-for-newbies/dash-school-2-how-does-a-blockchain-work-4b84c69faa68

[19] Kaiser, B.: The Looming Threat of China: An Analysis of Chinese Influence on Bitcoin. *arxiv.org*. 2018.

[20] kryptomagazin.sk: *Centralizovaná, Decentralizovaná a distribuovaná sieť*. [Online; navštívené 19.11.2018].
Retrieved from:
https://kryptomagazin.sk/wp-content/uploads/2017/06/image-2-768x415.png

[21] Lee David, L. L.: *Inclusive FinTech: Blockchain, Cryptocurrency and ICO*. World Scientific. 2016. ISBN 9789813272767.

[22] Longman, A. W.: *History of html*. [Online; navštívené 25.01.2019].
Retrieved from: https://www.w3.org/People/Raggett/book4/ch02.html

[23] MANGAL, A.: coincentral.com. What Is Monero (XMR)? | An In-Depth Guide to the Privacy Coin. 2019. https://coincentral.com/what-is-monero/.

[24] Mitchell, R.: *Web scraping with Python collecting more data from the modern web*. OReilly. 2018. ISBN 1491910291.

[25] R. Fielding, U. I.: Hypertext Transfer Protocol. RFC 1945. RFC Editor. May 1996.
Retrieved from: https://www.ietf.org/rfc/rfc1945.txt

[26] Richardson, L.: *Beautiful Soup Documentation*. [Online; visited 27.01.2019].
Retrieved from:
https://www.crummy.com/software/BeautifulSoup/bs3/documentation.html

[27] Rouse, M.: *Proxy Server*. [Online; visited 02.05.2019].
Retrieved from: https://whatis.techtarget.com/definition/proxy-server

[28] Smith, K.: *123 Amazing Social Media Statistics and Facts*. [Online; visited 23.04.2019].
Retrieved from: https://www.brandwatch.com/blog/amazing-social-media-statistics-and-facts/

[29] Spitzner, L.: *Honeypots tracking hackers*. Addison-Wesley. 2002. ISBN 0321108957.

[30] Tarnoff, B.: *How the internet was invented*. [Online; navštívené 25.01.2019].
Retrieved from: https://www.theguardian.com/technology/2016/jul/15/how-the-internet-was-invented-1976-arpa-kahn-cerf

[31] Truschka, J.: *www.systemonline.cz/it-security/asymetricka-kryptografie-v-praxi.htm*. [Online; navštívené 21.01.2019].
Retrieved from: https://www.systemonline.cz/it-security/asymetricka-kryptografie-v-praxi.htm

[32] Underwood, L.: *The HTML Hierarchy: Thinking Inside the Box*. [Online; visited 27.01.2019].
Retrieved from: https://www.htmlgoodies.com/beyond/article.php/11156_3681551_2/The-HTML-Hierarchy-Thinking-Inside-the-Box.htm

[33] Vaidya, K.: Clients , Wallets Storage in Bitcoin. *medium.com*. 2016.

[34] Viliam Serečun: *Automatizovaná rekonstrukce webových stránek*. Master's Thesis. Vysoké učení technické v Brně, Fakulta informačních technologií. 2018.
Retrieved from: http://www.fit.vutbr.cz/study/DP/DP.php?id=21138

[35] webmasters.googleblog.com: *Google official web crawler*. [Online; visited 27.01.2019].
Retrieved from: https://webmasters.googleblog.com/2008/04/crawling-through-html-forms.html

[36] Weiser, B.: *Man Behind Silk Road Website Is Convicted on All Counts*. [Online; navštívené 23.01.2019].
Retrieved from: https://www.nytimes.com/2015/02/05/nyregion/man-behind-silk-road-website-is-convicted-on-all-counts.html?hp&action=click&pgtype=Homepage&module=first-column-region&region=top-news&WT.nav=top-news&_r=0

[37] www.dash.org: *Dash*. [Online; navštívené 24.01.2019].
Retrieved from: https://www.dash.org/masternodes/

[38] www.finder.com: *What is Dash? A step-by-step guide*. [Online; navštívené 24.01.2019].
Retrieved from: https://www.finder.com/dash

[39] www.scrapehero.com: *Beginners guide to web scrapping*. [Online; visited 26.01.2019].
Retrieved from: https://www.scrapehero.com/a-beginners-guide-to-web-scraping-part-1-the-basics/

[40] www.scrapehero.com: *Best Open Source Web Scraping Frameworks and Tools.* [Online; visited 27.01.2019].
Retrieved from: https://www.scrapehero.com/open-source-web-scraping-frameworks-and-tools/

Appendix A

youtube.conf syntax

1. Line starting with # is comment, but # must be the first none white space character in line .
2. empty lines are allowed
3. keyword *end_file* marks end of executable part of file, if not present whole file is considered to be executable
4. There are two types of search request, by: name of video, name of youtube channel
5. Search by video name: (string)[name]: (unsigned int in range 1-400)[deep]
   - Bitcoin Address : 400
   - Crypto Donation : 50
6. Search by channel name: $$$ (string)[name]
   - $$$ Crypto Kety
   - $$$ Bitcoin_Boss
7. Write one request per line
8. If you want to know more look on complete documentation in file youtube.conf

Complete example

```
#code instructs spider to scrape 400 results of searching
   # for video by name Bitcoin
# and to scrape all videos posted by channel Crypto Kings
Bitcoin : 400
# popular crypto channel
$$$ Crypto Kings

end_file
```

Appendix B

Cryptocurrency Regexs

cryptocurrency	regular expression
Bitcoin	**bc1[A-Za-z0-9]{39}|[13][1-9a-km-zA-HJ-NP-Z]{25,33}**
Litecoin	**[3LM][a-km-zA-HJ-NP-Z1-9]{33}**
Dash	**X[1-9A-HJ-NP-Za-km-z]{25,33}**
DogeCoin	**D[1-9A-HJ-NP-Za-km-z]{33}**
bitcoincash	**(bitcoincash:)?[qp][0-9a-z]{41}**
Zcash	**t[a-zA-Z0-9]{34}**
Etherum	**0x[0-9a-fA-F]{39,40}**
Nano	**xrb_[0-9a-zA-Z]{60}**
Monero	**4[0-9a-zA-Z]{94}**
Stellar	**G[A-Z0-9]{55}**
Burst	**(BURST(-[2-9a-zA-HJ-NP-Z]{4}){4}[2-9a-zA-HJ-NP-Z])**
NEO	**A[1-9a-km-zA-HJ-NP-Z]{33}**
Ripple	**r[1-9a-km-zA-HJ-NP-Z]{33}**
Snowgem	**s[1-9a-km-zA-HJ-NP-Z]{34}**
TRON	**T[1-9a-km-zA-HJ-NP-Z]{33}**

Table B.1: Regular expressions implemented in scanning function

Appendix C

The content of CD

- src – source files for application
- tests – tests
- documentation – documentation
- README.txt – install instructions, map of CD
- env – virtual environment
- doc – the thesis
- run_spiders.py – launcher
- scrapy.cfg – scrapy configuration
- database_login.cfg – scrapy configuration
- youtube.cfg – spider configuration
- reddit.cfg – reddit API configuration
- SQL – reddit API configuration

For more information and guidance open file README.txt