UNIVERSIDAD DE CHILE
FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS
DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN

ORÁCULOS DISTRIBUIDOS EN LA BLOCKCHAIN

TESIS PARA OPTAR AL GRADO DE MAGÍSTER EN CIENCIAS MENCIÓN COMPUTACIÓN

MEMORIA PARA OPTAR AL GRADO DE INGENIERO CIVIL EN COMPUTACIÓN

FRANCISCO JAVIER ANDRÉS MONTOTO MONROY

PROFESOR GUÍA:
ALEJANDRO HEVIA

MIEMBROS DE LA COMISIÓN:
INTEGRANTE 1
INTEGRANTE2
INTEGRANTE3

SANTIAGO DE CHILE
MES AÑO

RESUMEN DE LA MEMORIA PARA OPTAR
AL TÍTULO DE
POR: FRANCISCO JAVIER ANDRÉS MONTOTO MONROY
FECHA: MES AÑO
PROF. GUÍA: ALEJANDRO HEVIA

ORÁCULOS DISTRIBUIDOS EN LA BLOCKCHAIN

Este es un resumen muy resumido

Una dedicatoria corta. Por ejemplo, A los creadores de U-Campus

Agradecimientos

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1. Gambling

Gambling is the activity of predicting events and placing a wager on the uncertain outcome of those events, with the intent of winning money or valuable goods. A wager can be put on many different events, in a casino we find randomizing devices as dices, roulette wheels, etc. which are used to get randomize events. In other establishments we can bet on sport events, such as a horse racing, football games, etc. or the minimum temperature in Santiago during this night. Its popularity and the big amounts of money at stake inevitably entails a lot of interest on this activities. Most of the time gambling is heavily regulated and taxed, also it is usual that lotteries are owned by the state.

Internet has been making cheaper to open and operate a casino, even without complying laws from any country. This and the massive internet use, has been moving the gambling industry online[28] [15]. The global Internet gambling market was estimated to be worth US$28.32 billion in 2012 and forecasted to rise to US$49.64 billion by 2017[14]. However, gambling not only takes place in casinos, lotteries or betting sites, it can also involve two or more individuals with no intermediaries. In Chile friends usually bet on their favorites football teams.

Nonetheless all the different ways for placing a bet, all of the mentioned share a common obstacle, participants are required to trust in the other parties to pay if they lose. Even if the bet takes place in a physical casino, where the law can enforce the bet, is not certain the casino will be able to pay after the resolution. We might not be aware of the fact, but every time we place a bet we are implicitly trusting in a third party, either the other player or the bet site. For physical casinos this is usually not a problem, as they are regulated by the law, any misconduct can get the casino to the justice and even get its license revoked. As there is a significant cost on starting a physical casino, them are also encouraged to keep a good reputation, in order to get customers.

Friends usually are trusted people, so trusting them when gambling might not be considered an issue. Also, probably the friendship is at risk if the bet is not paid. Other option is to get a third friend to get the money until the bet result. Online casinos on the other hand are more problematic, there are many knowns scam schemes, as described by Griffiths[16]. And half of the players at this sites believe the providers are cheating on them[22]. However, some of them are subject of government regulation and many have being in the business for several years, this kind of characteristics could help to indicate an online site is trustworthy.

But, what if you would like to gamble in a event that no gambling site offers nor any friend want to? Probably the internet would be the place to look for somebody willing to gamble on this event. Yet, how could you trust the potential person in order to bet with him?

1.2. Cryptocurrency

Digital currency refers to any currency stored and transferred electronically. A subset of the digital currencies is called virtual currencies: them are usually defined[2] as a « *unregulated, digital money, which is issued and usually controlled by its developers, and used and accepted among the members of a specific virtual community*».

Based on the interaction of the currency with currencies outside the community there are three types of virtual currencies: The ones with almost no interaction with the outside money, this is usually the case of video games, where its currency is only valuable within the game. A second type is where the currency can be purchased directly using other currency. Here, we observe an unidirectional flow. The third type is when the flow is bi directional, the users can sell and buy the currency. A cryptocurrency is a bi directional virtual currency, that uses cryptography for security and anti-counterfeiting measures. Virtual currencies are been historically linked to cryptography, the first known investigations [6] to establish a virtual currency where lead by David Chaum, an American cryptographer. However, despite his and others effort (e-gold[1], Ecash[7], DigiCash, LibertyReserve, among others), virtual currencies never where massively adopted.

By late 2008, using a pseudonym, was released a short whitepaper[24] with yet another virtual currency protocol specification. A few months later, during 2009 its implementation was made available as open source code. The main difference with previous implementations was its lack of a central organization, this new coin was completely decentralized. The software started to being run by some early enthusiasts and Bitcoin gave the step from an idea to an usable coin. The first years was the coins were exchanged for free among the community users. However, at some point the community was big enough and its members started to give value to the coin, then the first exchanges from and to other coins started to take place. Bitcoin transitioned into a bi directional flow virtual coin.

Then the first online exchanges between bitcoin and other currencies started to appear, the coin started to gain traction as people outside the community were able to buy and sell coins. As the money became popular, the idea was taken and a whole generation of cryptocurrencies were born. Today the market capitalization of Bitcoin (this is, the amount of money times its value in USD) is over 25,000,000,000 USD.

[1] https://www.wired.com/2009/06/e-gold/

1.3. Gambling using Cryptocurrencies

With cryptocurrencies getting more and more popular, it was only a matter of time until the first sites started to offer started to offer some games of chance an act as online casinos. Where the only difference with a traditional online casino was the currency on which the bet takes place. However, as any other currency online casino, any player who decided to play here is at the mercy of the casino. If the casino does not want or does not have the means to pay, there is nothing the participant can do and its money is lose. More on online casinos at subsection 1.1. The problems described for online casinos using traditional currencies apply in the same way to the new ones.

After some time, people started to see some potential on cryptocurrencies to solve some of the trust issues related to gamble. In 2014 Andrychowicz et al. proposed a two party randomized gambling protocol. Players are not required to trust each other in order to gamble, so even if the loser does not behave correctly the honest player, can get its prize. The protocol is not a representation of a casino game, but effectively allows player to gamble on a random event. Also in 2014, a group of Bitcoin enthusiasts started Orisi[2], a distributed oracles system for cryptocurrency contracts. Orisi allows users to access data of the outside world from the blockchain, by using a distributed set of oracles. So instead of trusting in one instance to provide the data, the trust is placed in the majority of several different oracles. More recently, on early 2017, Winsome[3] was released. Advertised as a «*Provably Fair / Trustless Casino*», Winsome is an online casino where wager are placed in a public smart contract posted in the Ethereum's blockchain. So the contract, defining the game, is enforced by the Ethereum protocol. As May 2017, they do offer two casino games, blackjack and `Rouleth`, an online roulette.

Motivated to provide an option to gamble over real world events with untrusted peers. This work proposes a protocol to define the destination of an initial wage between the two player. The decision is taken by a set of oracles, which are being paid also within the protocol to behave correctly.

1.4. Objectives

Design and implement a distributed protocol where real world observations can be used as blockchain transaction inputs.

1.4.1. Specific Objectives

1. Provide a protocol to make possible to gamble with untrusted peers over real world events.

[2]http://orisi.org
[3]https://www.winsome.io

2. Provide the correct economic incentives to the protocol participants to behave correctly, so everyone incentives are aligned.
3. Implement a proof of concept of the designed protocol.
4. Debate of implications and other applications for the designed protocol.

1.5. Methodology

The main phases of this work will be the following:

1. Extensive review of existing proposal and implementations to solve the proposed problem or similar ones. As cryptocurrencies are a recent investigation field, this review must cover literature as well as community gathering places, such as forums and specialized blogs, magazines, etc..
2. Analysis of current solutions to the problem and similar ones.
3. Design and implementation of a protocol to solve the problem. Implementation is considered very important as the current rate of change of cryptocurrencies is considerably fast, validating the protocol within a real implementation is critical.
4. Analysis of the economic incentives of the protocol participants, to ensure protocol viability.

1.6. The Protocol

The main idea behind this work is to eliminate the more single points of trust we can when performing bets. Traditional currencies are produced and controlled at Government's will, so the first decision was to use a currency without a single controller, we chose Bitcoin mainly for two reasons. It is the first and one of the most stable currencies out there, changes are made much slower than other currencies, the market back this claim by making bitcoin the Cryptocurrency with by far the biggest market capitalization. And second, the network supporting bitcoin is much bigger than the ones for other cryptocurrencies. This makes much harder to attack and take control of the currency.

There are two mains phases in the protocol, where the first one is optional and can be replaced at players will:

1.6.1. Oracle's selection

Bitcoin (like most of the cryptocurrencies) includes a scripting language able to control money transferences, well defined and with its execution enforced by the complete bitcoin network. The challenge is to bring data from outside the bitcoin data and reason about it. Our protocol relies on several paid "oracles" to bring this data. As the oracles' output will be used to decide who is the bet's winner, it is a crucial step to avoid a player getting itself or compromised oracles to decide the bet winner. We say this phase is optional as it might be the case both players trust already in a set of oracles.

1. The first step is to compile a list of available oracles, we use as decentralized database for this list the blockchain. Everyone willing to be an oracle can send a transaction to register into the blockchain.
2. The players negotiate some parameters, as the number of oracles to use and the threshold to decide the winner.
3. In order to decide which oracles to use, the oracles need to pick a subset of the available oracles, they do this by running a distributed coin tossing protocol. With this, they can be sure the compiled list is a random subset of the full list. If the list is big enough, the chance of one user controlling the oracles gets smaller. As it would be too expensive to control almost all the oracles in the list.

1.6.2. Bet resolution

This is phase starts after both players agree the bet with and the oracles to be used on it.

1. The players send a transaction to the blockchain with the bet description, including the IDs of the oracles they want to decide the winner. We call this transaction "Bet promise",as the players commit to the bet by placing it. The wage is also on it. The other purpose of this transaction is to invite the oracles to participate in the bet, we make its ID public so they can identify itself and inscribe to participate as oracles.
2. The oracles will see the transaction inviting them to participate in the bet, they will evaluate it and, if they are interested. They will reply with a transaction containing a reference to the "Bet promise" transacation and a small deposit as commitment that they will participate in the process.
3. When the players see the answer from the expected number of oracles, they will send the "Bet" transaction with funds of the bet and the oracles' reward. If not enough oracles reply to the call, a second invitation can be send to a different set of oracles to fill the available spots.
4. As soon as the bet event takes place, oracles are able to collect its payment from the Bet transaction. This payment gets available by making public, -voting- by the winner. After the threshold number of oracles collect its payment, the winner player is able to collect its prize, its private key and the oracle votes are required to get it.
5. After a second timeout, players can take the deposit from the oracles that did not participate in the bet resolution.

The payment of the oracle's is not the only cost of this protocol. There is a not insignificant number of transactions in the protocol, as there is a fee by each transaction in the Blockchain, this makes the protocol more expensive. This is a problem for small bets, the presented protocols is prohibitively expensive for bet of just a few dollars. At least with the current fee costs of bitcoin.

If we step out a little bit, the proposed protocol uses paid oracles to get a binary answer about an event outside the blockchain. This is not useful only for betting on that outcome. The oracles are [4] insensitive to the use given to their answer, they get paid anyway. Further applications of the protocol can be generalized from our proposal. Como por ejemplo.... ?

[4] is this the word?

Chapter 2

Preliminaries

2.1. Hash

Hash is an overload word and it is usually used for a few different things:

2.1.1. Hash Function

Is a function able to map data from arbitraty size to data of a fixed size. The range has only elements of a fixed size, so it's bounded by all the elements of that size. If we represent the data in a binary base, the range is bounded by 2^n where n is the size in bits of the output. The domain of the function is unbounded, so by the *Pigeonhole Principle:*

$$\exists\, \mathrm{i}, j \mid f(\mathrm{i}) = f(j),\ \mathrm{i} \neq j \tag{2.1}$$

We call this a collision, and for most uses of a Hash function are unwanted.

Hash functions are used for many things: File comparison, instead of comparing files bit to bit, the image of a Hash Function can be compared instead; Hash-Tables, this allows quick lookups for the elements; Find similar records, by using a Hash Function that produces similar images for similar pre-images, etc..

2.1.2. Image of a Hash Function

If not stated otherwise, will use the word "Hash" to denote the image of some data using a Hash Function.

2.1.3. Cryptographic Hash Function

This refers to a special class of Hash Functions the Cryptography has defined to be suitable for its use on cryptographic applications. The main property this functions are designed to is to be "one way" functions, this means its infeasible[1] to invert.

In an ideal cryptographic function, the most efficient way to find one of the preimages is a brute-force search[2]. We call this property *preimage-resistance*. It is also important for this ideal function to be *collision resistance*, this means it is infeasible to find any two distinct inputs x, x' with the same image, i.e., such that $h(x) = h(x')$.

When using this ideal function producing a (second) preimage requires 2^n operations, and producing a collision requires at least $2^{n/2}$ operations[27].

2.2. Digital Signatures

The idea of "Digital Signature" was introduced in 1976 by Diffie and Hellman in "New Directions in Cryptography"[10]. In this work is also introduced what they called "Public Key Cryptosystem", were enciphering and deciphering operations use different keys, E and D, such that computing D from E is computationally infeasible. Today this pair are widely used and are known as Public Key (PK) and Secret Key (SK).

The public key cryptosystem, or asymmetrical cryptography was created to solve one important problem of symmetrical systems[3]: It is imposible to start a secured communication in an insecure channel without previously exhange of a key using a secure channel. To stablish a secure communication within an insecure channel participants makes its PK publicly available to the others. Anyone willing to talk to another participant must cipher its message using the public key of the receiver, this way the only one able to decipher the message is the intended receiver.

A digital signature, as its name indicates, is a mechanism to provide protection against third paty forgeries. It must be easy to for anyone to recognize as authentic but impossible for anyone but the signer to produce it. This is specially challenging since any digital signal can be easily copied.

It works within the public key cryptosystem the signer uses its SK to produce a signature over the message to sign, and anyone with the signer PK and the message can determine the validity of the signature.

The most important property of a digital signature is that does not matter how many pairs $<$message, signature$>$ has a third party seen, it does not make it easier to generate a

[1] We say something is computational infeasible when even it is computable, it will require far too many resources to do it.

[2] Also known as exhaustive search, it consists of enumerating all the potential solutions and to check which of them satisfies the predicate

[3] As opposed to the asymmetrical one, this system uses the same key to cipher and decipher the messages.

signature for a new message.

2.3. Ecash

Digital currencies have been a research topic since at least 1983 when David Chaum[6] introduced Blind Signatures. A form of digital signature where the content of the signed message is blinded, so the entity signing the message do not get to see it. This technique was used to provide untraceable payments in a cash system where however anybody can check the signature is valid.

The field has been an active topic since then, in the academy and as business intent. Many research has been published proposing new schemes and cryptographic primitives[26][5] [4][1][21].

In 1990 David Chaum founded DigiCash, which developed an early electronic payment based on blind signatures, payments using the software were untraceable by the issuing bank or any third party, including the government. However the company is not able to beat credit cards in the electronic commerce and files its bankruptcy in 1998.

In 1996 e-gold allowed its user to buy electronic money ("grams of gold"[4]to backed by precious metals held by the company.[17] The users can buy, sell and transfer the ownership of the metals over the Internet. In 1999 the *Financial Times* described e-gold as the only electronic currency that has achieved critical mass on the web. However its success contributed to its demise. It was used for fraud, phishing, cyber crime gangs, etc.. Law enforcement agencies began to characterize e-gold as the favourite payment system for criminals and terrorists[5]. By 2007 the justice start to seizure e-gold balances that ended up with the suspension of the service.

Falta algo para cerrar... ayuda!

2.4. Bitcoin

Bitcoin is the first fully distributed cryptocurrency made publicly available, it was proposed in 2008 by Satoshi Nakamoto (a pseudonym) [24]. The same author shared as open source code a implementation of the protocol in January 2009. And the protocol has being running since then.

Nevertheless, Bitcoin is not the first idea of electronic cash. The idea of electronic cash has been present within the cryptographic community since at least 1983, when Chaum [6] proposed a system for anonymous payments. And the attempts kept going for other three decades, hundreds of paper have been published with improvements of e-cash schemes[3]. So,

[4]Also of platinum, silver, etc..

[5] "Feds out to bust up 24-karat Web worry". NY Daily News. 2007-06-03. Retrieved 2017-07-27

why is Bitcoin so popular and achieved the notority that three decades of academic research on the field could not achieve?

Barber et al.[3] suggest a few key points to explain why was Bitcoin the first electronic currency to take off.

1. No central point of trust. Bitcoin is a fully distributed system, there are no trusted entities in the system. The only asumption is that the majority of the network participants are honests. Every previous proposal had a central trusted entity for critical tasks, as preventing double spending and coin issuance.
2. Predictable money supply. The money supply is minted at a defined and transparent rate, defined from the begining of the protocol.
3. Transaction irreversibility. Bitcoin transactions quickly become irreversible. This is a big difference with credit cards, where chargebacks has been using largely to commit frauds.

Bitcoin has not stopped to gain massive popularity and attention from the press. Mainly because its market capitalization (over USD 36 000 000 000), and some illegal activities it has been using to as ransom to retrieve victim's data encrypted for malicious software, or as exchange medium in one of the most famous online black market, closed in 2013 by the FBI.

The main technical advance in Bitcoin is its database, the `blockchain`[12][25]. The blockchain is a distributed database formed by an always growing list of blocks, where each block contains the data to be stored, a timestamp and a link to a previous block. Its fully distributed nature allows bitcoin to lack a central authority.

2.4.1. Transaction

Bitcoin works with accounts where coins can be stored, this accounts are identified with an address[6], for this reason some times both words are used interchangeably. The address is not private, and people share theirs when willing to receive money.

It represents a hash of a public key. Therefore, the owner of the account is however control the private key of the address. Bitcoin uses Elliptic Curve Digital Signature Algorithm (ECDSA)[7] to ensure the owner of an address is the only one able to spend its content. Getting a new address is free, it only requires to generate a random byte string and get a ECDSA private key from it. ECDSA allows the public key derivation from the private key. Then, by hashing the public key the address is obtained. Many libraries and most of the bitcoin wallets implement the algorithms to generate new addresses.

A transaction is the only way to move bitcoins from one account to other one, it is

[6] In the wire, an address is a 25-byte value, for human consumption we usually see it in its encoded representation of base 58, resulting in a string of 25-33 characters.

[7] ECDSA is a digital signature algorithm using Elliptic Curve Cryptography. It is an asymmetric scheme, with a private and public key. In bitcoin transactions are secured with a private key signature and validated using the public one.

...	
Num Inputs	Num Outputs
$Input_0$	$Output_0$
$Input_{...}$	$Output_{...}$
$Input_{n-1}$	$Output_{m-1}$

Figure 2.1: Simplified Transaction

basically a list of accounts where the money is pulled from, the inputs. And the outputs, a list of accounts the bitcoins are going to, a simplified view in the figure 2.1.

There is only one exception to this rule, the miner that builds each block is allow to send money to his account from nowhere, this is called the generation transaction and its amount is defined in the protocol. This is the only way bitcoins are generated.

A transaction input points to a previous unspent output and proves it has the right to spend that output. An output contains[8] the address of the account is transferring the money to, so any input signed using the private key of that address has the right to spend the output. This implies that the money from an account must be spent in the same amount the money was received. If an output of $\$10BTC$ is received, when trying to spend it, the same amount must be spent. If willing to spend just a portion, a second output is created and send to the same account.

2.4.2. Blockchain

It works as the bitcoin's ledger, it keeps record of all transactions and coin generation that had ever taken place in the protocol. It is completely distributed and public, anybody can participate and get a copy of it. This makes simple to prevent double spending and be sure the received coins are valid, as anybody can examine where each coin came from.

As any other distributed system, the blockchain must resolve the consesus problem [13]. Get all the participants to agree on the data. This is a fundamental problem to any distributed system. In the the blockchain anybody with an internet connection can be part of the protocol, so solving this problem is quite challenging. Some authors argue the blockchain is the first practical solution to the Byzantine Consensus problem [23] [29].

Proof of work is the algorithm used by the bitcoin blockchain to seek consensus. Each entity trying to add data to the database must prove it has done some required work. This algotithms was designed originally to fight the email spam, by requiring the sender of an email to prove a small work was done in order to send the email[11]. It works by using a hard to calculate, but easy to check function. This way the receiver or the mail server can easily check if the sender did the required work, however this work was much harder. The difficulty of a work is defined by the amount of computational power required to get it done.

[8]This is a simplification, not every output work this way, detailed at subsection .

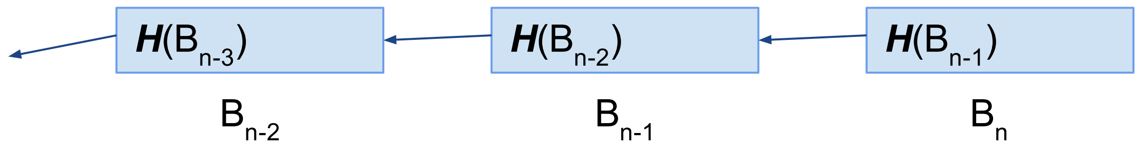

Figure 2.2: Blocks linked to each other in the blockchain.

The atomical piece in the blockchain is the block. Each valid block carries transactions of the protocol and a proof of work. So every entity trying to get a valid block into the database need to collect transactions and solve the puzzle to get a valid proof of work for its block. This process is called mining, therefore the entities trying to get a valid block are called miners. A block is linked to the previous one, as show by the figure 2.2.

The mining process is like playing the lottery, tickets are distributed among every miner until someone gets the winner one. The number of tickets each miner gets is proportional to the work he is doing, so a miner doing more work will get a bigger chance to win the lottery and mine a block. However any miner can win.

Of course, in the real implementation there are no tickets issued to the miners. Proof of work consists in building a block with a hash under a threshold value, so the miners should reorder and change the block until the hash fulfill the requirement. There is not a known algorithm to do this in a better way than brute force, so the only method to get a hash that mets the criteria is to try with different block configurations, there are also some bytes of nonce, a timestamp and transactions to be changes to get different hashes.

Once block is produced, all the others miners need to delete the transactions added by the block from the one they are building and update the link to the new last block. And they start to mine a new block. By design a block must be produced every 10 minutes, so the work required to mine a block is a ajusted periodically to met this goal.

The structure of a Bitcoin block is show in the figure 2.3, the fields with the gray background represents the block header, the data hashed to get the block's hash. The transactions are indirectly hashed in the Merkle Root[9].

As expected in a protocol with many participants, there are times were more than one block is generated with the same parent (figure 2.4), this is call a fork.

In order to achieve consensus, the protocol determines that the chain with more work[10] on it is the active chain. So when a fork happen there are two active chains, while having a non unique active chain miners will try to mine in any of the candidates with the same work. A block mined on one of the branches will decide which is the active one because it adds more work to the chain. However the situation that originated the fork can repeat itself and prevent to have one consensus branch, this is very unlikely[9] to happen during a long time.

[9] A `Merkle Tree` is a tree in which each non leaf node is labeled with the hash of its children's labels. In the block each transaction is mapped into a tree leaf. So the root of this tree hashes all the transactions

[10] The amount of work in a chain is the sum of the difficulty of every block on it.

	0	1	2	3	4	5	6	7
0	Magic no				Blocksize			
8	Version Number				Hash Previous Block			
16	Hash Previous Block (cont)							
40	Hash Previous Block (cont)				Hash Merkle Root			
48	Hash Merkle Root (cont)							
72	Hash Merkle Root (cont)				Timestamp			
80	Target difficulty				Nonce			
88	Transaction counter and Transactions.							
...								

Figure 2.3: Block Structure

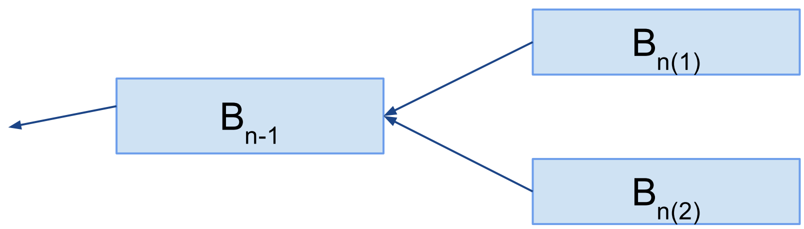

Figure 2.4: A fork in the blockchain.

The chain structure gives a chronological order to the transactions in the protocol, so it makes clear to check if a transaction is valid. Any participant willing to probe the validity of a given transaction needs to evaluate the script (see section 2.4.3), and check the block where the output being spend is stored up to the current block and see if the money was already spent in a different transaction.

2.4.3. Script

When sending money, there is a little more than we saw at section 2.4.1. In an input (figure 2.5) there is more than a signature, and at each output (figure 2.6) also more than an address.

	0	1	2	3	4	5	6	7
0	Previous Tx Hash							
32	Previous Tx Output index				Script Length[1-9 bytes]			
	Script / scriptSig [<Script Length> bytes]							
	sequence_no							

Figure 2.5: Wire format of an Input.

	0	1	2	3	4	5	6	7
0	Value							
8	Script Length [1-9 bytes]							
	Script / scriptPubKey [<Script Length>bytes]							

Figure 2.6: Wire format of an Output.

An output does not send money to a given address, but defines how the money can be spent. Currently there are two formats in use. The most used is called "Pay To Public Key Hash" (P2PKH)[11]. And "Pay To Script Hash" (P2SH).

As the figure 2.6 shows, the output have a script on its wire representation, this script is written in a small stack based language. It is read from left to right and it is purposefully not Turing-complete. The script is evaluated using the scriptSig as input. If the transaction willing to spend this output provides a valid[12] scriptSig, the output is available to be spend. This is how a P2PKH script looks like:

```
OP_DUP OP_HASH160 <pubKeyHash> OP_EQUALVERIFY OP_CHECKSIG
```

It receives two values as input: <pubKeyHash> and <sig>. And the execution is explained in the table 2.1.

The scriptPubKey in a pay to script hash transaction is even simpler:

```
OP_HASH160 <scriptHash> OP_EQUAL
```

Whis script is pretty simple, it takes the first element of the stack, calculates its hash and compares it with '<scriptHash>'. The first element in the stack is however a complete script, and after checking it hashes to the expected scriptHash, it will be evaluated with its required input. This implies the scriptSig now holds the script and its signature:

```
<sig> >script>
```

A execution of a sample P2SH is show in the table 2.2.

A P2SH transaction allows different conditions to redeems its outputs, a complete list of the operations supported by the Bitcoin scripting language can be found at the Bitcoin wiki: https://en.bitcoin.it/wiki/Script.

[11] The key hash is the address of an account

[12] A script is considered valid if after its execution the value in the top of the stack is True.

Table 2.1: Script evaluation to check a P2PKH transaction.

<table>
<tr><th>Stack</th><th>Script</th></tr>
<tr><td colspan="2">Constants from scriptSig are copied to the stack.</td></tr>
<tr><td><pubKey>
<sig></td><td>OP_DUP OP_HASH160 <pubKeyHash> OP_EQUALVERIFY OP_CHECKSIG</td></tr>
<tr><td colspan="2">OP_DUP copies the top element from the stack.</td></tr>
<tr><td><pubKey>
<pubKey>
<sig></td><td>OP_HASH160 <pubKeyHash> OP_EQUALVERIFY OP_CHECKSIG</td></tr>
<tr><td colspan="2">The hash of the top element is calculated.</td></tr>
<tr><td>H(<pubKey>)
<pubKey>
<sig></td><td><pubKeyHash> OP_EQUALVERIFY OP_CHECKSIG</td></tr>
<tr><td colspan="2">The destination address is moved to the stack.</td></tr>
<tr><td><pubKeyHash>
H(<pubKey>)
<pubKey>
<sig></td><td>OP_EQUALVERIFY OP_CHECKSIG</td></tr>
<tr><td colspan="2">The destination address is compared with the Hash of the Public Key (PK) provided by the sig Script. This check the provided Public Key is the one from the intended receiver.</td></tr>
<tr><td><pubKey>
<sig></td><td>OP_CHECKSIG</td></tr>
<tr><td colspan="2">Using the already verified PK, the script checks the transaction was signed using the corresponding Private Key. This step secures the transaction from tampering an proves it was sent by the private Key controller.</td></tr>
<tr><td>True</td><td></td></tr>
</table>

Table 2.2: Script evaluation of a P2SH transaction.

<table>
<tr><th>Stack</th><th>Script</th></tr>
<tr><td colspan="2">The serialized script and its input are copied to the stack.</td></tr>
<tr><td>{<pubkey> OP_CHECKSIG}
<sig></td><td>OP_HASH160 <scriptHash> OP_EQUAL</td></tr>
<tr><td colspan="2">The serialized script hash is calculated.</td></tr>
<tr><td>H({<pubkey> OP_CHECKSIG})
<sig></td><td><pubkeyHash> OP_EQUAL</td></tr>
<tr><td colspan="2">The expected hash is pushed to the stack.</td></tr>
<tr><td><pubKeyHash>
H({<pubkey> OP_CHECKSIG})
<sig></td><td>OP_EQUAL</td></tr>
<tr><td colspan="2">Then the hashes are compared, and if they match, the script is deserialized and evaluated.</td></tr>
<tr><td><sig></td><td><pubkey> OP_CHECKSIG</td></tr>
<tr><td colspan="2">The public key is pushed to the stack.</td></tr>
<tr><td><pubKey>
<sig></td><td>OP_CHECKSIG</td></tr>
<tr><td colspan="2">The provided signature is validated using the public key.</td></tr>
<tr><td>True</td><td></td></tr>
</table>

2.5. Previous Work

There are several attempts to provide information to the blockchain from the outside, by the way they gather the data we divide them in “Distributed Oracles”, where the data is gathered by a group of third party participants. Or as “Data Feeds”, where the data is provided by a centralized party, using some techniques to authenticate the data.

We also add “Trustless Distributed Casino” as a solution to a similar problem, it does not provide information about the real world, but it does perform backed by the ethereum blockchain, any participant has entire power over its money at any point, however is limited to gambles on random events.

2.5.1. Distributed oracles

Orisi

Orisi[20] is a distributed system for bitcoin smart contracts that relies in multiple oracles to bring information from outside of the blockchain. It allows its users to transfer money from one address to another when a condition is met.

Both players agree on 7 oracles to be used to decide the transfer, usually chosen from "The Oracle List", a curated list with oracles. But could also be chosen from any other place the players want. Then, a multisignature address is generated to store the money while the bet takes place. A multisignature address is defined by `m` addresses and a required number `n` (`n` $<$ `m`) of them to sign. A valid signature for a multisignature address is generated by using at least `n` out of the `m` addresses defining it.

The multisignature address generated will store the money until the oracles decide where the transaction goes. To avoid the oracles sending the money to themselves the multisignature transaction include the address of the receiver, so we want a $1 + (n \text{ of } m)$, where the extra signature is from the receiver. As this kind of transaction is not considered standard [13], Orisi uses a bigest multisignature address, where instead of using n out of m oracles, it adds more receiver keys. Requiring $m + 1$ signatures of $2m$ - $n + 1$. With this configuration the oracles are not able to move the money by themselves, and at least one signature from the receiver is required.

2.5.2. Trustless distributed casino

Winsome.io

In may 2016 Rouleth[18] was launched as a distributed application on the ethereum network. Offering its players a "provably-fair", real money roulette. Later, in early 2017, also using the ethereum network "BlockJack" was launched, the first playable blackjack game on the Ethereum mainnet.

Winsome.io is the instance where these games are enclosed, it offers unique advantages over traditional casinos (physical and virtual), like trusless, and complete control over the funds the entire time while playing. It does work in a distributed fashion using smart contracts, publicly availables for everyone's scrutiny.

Winsome.io provides its users trustless gambling over random events, by using the ethereum network as backend. It have been quite successful, it is one of the most popular descentralized applications on the Ethereum Network.

2.5.3. Secured data feeds

Oraclize

Oraclize[19] provides an interface for using data fetched from a web site in the ethereum blockchain, it works with arbitrary URLs or queries in certain web services, as "Wolfram

[13] Non standard is recognized as a valid transaction by everyone, however by the time this article was written only about the 5% of the mining power will not process it. Including this transaction in the blockchain will take on average much more time than a standard one.

Alpha" [14]. It provides an Authenticity Proof of the data gathered, so the user can check the data provided by the interface was generated by the source and have not been tampered.

Town Crier

Town Crier[30] is an authenticated data feed system for the ethereum blockchain, as oraclize it works as a bridge between web feeds, and the blockchain. It uses an Intel technology called "Software Guard Extensions"[8], than provide some execution guarantees of the software executed by hardware protected areas. This protects the execution of the data feed even with the the host OS, BIOS or any other piece of the machine compromised.

[14] Wolfram Alpha is a knowledge engine, able to answer queries rather than provide links to data sources, as a search engine does.

Chapter 3

The Protocol

3.1. Oracles

The first issue to solve when making decisions over events in the world is to define who track and define the outcome of said event. In our day to day we get information about events from a variety of sources. The television, an internet portal, our eyes among many others. Any protocol willing to make decisions over events needs a source for those events.

In order to keep the protocol decentralized we define its data source as a set of entities, called oracles. The decision is made by the oracles voting on the outcome of the event. In this scheme the decision does not rely on a centralized entity, but in a group of them.

Oracles are rewarded when provide the correct answer to resolve the bet. We define the correct answer as the one gave by at least m of the n oracles where $\lfloor \frac{n}{2} \rfloor \leq m$. When providing the incorrect anwswer oracles do not get paid, and when giving both answer they get penalized because its misbehavior. This gives strong economic incentives to the oracles to answer as they expect the other ones are going to answer. Explicar enseguida lo de los incentivos?

3.2. Players

Players are the ones wagering the money, we define its number to 2, however when gambling there could be more than two players. Usually we will refer to them as “Player A” and “Player B”. Players are the ones paying the bet, the oracles and the transactions fees. The one who predicts correctly the outcome of the event they are betting on takes the prize.

3.3. The Protocol

Our protocol is divided into two parts, where the second one is the actual bet and the first one is an optional step to select the oracles.

3.3.1. First part: Oracle Selection

The oracles are a key piece in the protocol, as they get to decide who get the prize money. The first part of the protocol defines a way to select them in a trustless way. The idea is quite simple, players selects from a list of oracles a subset to participate on its bet. With a big enough list, selecting randomly from it reduces the chances from any of the participants to influence on the selection. citation required? However, there are a few key properties in our protocol to reduce the chance of influencing their selection.

Oracle list

In order to get a trustworthy list, we define a few key properties: It must be a decentralized list; anybody willing to be an oracle can inscribe itself; and must be visible for both of the players.

As we saw in sub section , we already have a public distributed database to store information. We use the blockchain to keep the list of oracles, this provides tampering protection, a public database and a distributed source for the list. In order to let anybody inscribe to be an oracle, the inscription is a simple transaction generated by the oracle sent to the blockchain.

Oracle registration

Registration transaction In order to register themself into the list, oracles need to send a transaction to the blockchain. This transaction must include the address to be used, this is the oracle identifier, and the protocol indentification. We defined our own string as protocol indentification, but different protocols can use different strings to define other lists.

There is no required deposit for registration, however the transaction fee must be paid when sending the transaction. Some may argue than a higher price to register an oracle will decrease the chances of an individual controlling the majority of the list. If that is the case, increasing the cost by adding a required a unspendable output does not require any change in the transaction link to the tx, as the unspendable output already exist. Adding a deposit spendable by an address will require a new output, but the idea remain the same.

Compiling oracle list

There are a few parameters players must agree in order to select oracles from the blockchain list. First they decide the period of time[1] they will consider oracles from. Some participants might want to avoid recently registered oracles, as they might have an higher chance to be controlled by the other player. Others might argue too old oracles are likely to be inactive, in order to avoid oracles registered long time ago.

Second they decide the list to get the oracles from, and if they want to filter out oracles that paid less than b bitcoins on fees at registration time. Finally they decide the number of oracles to use, also they can decide to select a few more than the required oracles, anticipating one or more of the selected oracles will not reply to the invitation.

Once they decide the filter and which blocks to use for retrieve the oracles, both players can compile the same list of availables oracles. Then they just need to decide which of them to use.

Oracle selection

Both players need to have the certainty the election from the list is random. In order to achieve this property we use a protocol originally proposed to flip coins over the phone[**blum1983coin**]. Today this algorithm is mostly known as "Coin Tossing", and lives in a subfield of cryptography called Multi Party Computation. Multi party computation, or secure multi party computation aims to provide protocols for computing public functions and gets its results while participants keeps their input private.

The idea of the Coin Tossing we use is to get a random bit, as neither of the players trust the other to select the bit randomly, both players select a bit and the they XOR it with the other one. This way, does not matter how the other bit was chosen, the result is random. There is one important restriction when using this protocol, the bit must be chosen before knowing the value of the other, otherwise if one bit is known the second one can be selected in order to get the desired outcome. If we were physically together we would write down the bit in a paper, wait for the other player to write his and then reveal both bits and perform the XOR. However we would like to run this algorithm through the phone or in this case the computer. The idea is the same, but instead of writting down into a paper, players "commit" to the value they just chose randomly by sending to the other player a "commitment". This commitment binds the player to the value calculated, without revealing it. Once both players receive the other's commitment, they send the bit they chose. Both check the commit with the commitment received previously, and if they match, the protocol outputs a random bit. Otherwise, a player tried to cheat and the protocol aborts as there is no way to calculate a random bit.

If we have a list both players agree with, and we can also produce random bits, selecting a number of oracles from the list is a trivial exercise.

[1] Measured as a range of blocks in the blockchain.

3.3.2. Second part: The bet

Bet Promise

Once players decided the oracles to use, whether using the *Oracle Selection* part or by any other mean. They need to agree in the terms of the bet, the event and who is the winner on each outcome, the time available for the oracles to answer, money required by each player, fees of the oracles, deposit required to the oracles, etc.. Once all the bet parameters are set, they are serialized and its hash is calculated, players puts all the money required to run the protocol, including fees and the prize into a transaction. The selected oracles, bet hash and a method[2] to get the transaction full description are appended to this transaction in plain text and the transaction is sent to the blockchain. We call this transaction "*Bet Promise*", as it is a commitment from both players to the bet.

Figura de la tx

The commitment consists in paying the transaction fees[3], and to move all the money required by the BET into the control of both players. Any other transaction with this money needs the approval from both, however in the case one of the player dissapear and does not participate anymore in the protocol, most of it is return Hablar de la plata que va a los oraculos y es irrecuper

Oracle Inscription

Oracles invited to participate needs to retrieve the Bet description as instructed in the *Bet Promise* transaction and decide whether to participate or not. When an oracle does not participate, players decide how to select a new one, a waiting list is recommended to be selected in the first part of the protocol. When it decides to participate, it builds and send to the players Explicar como se comunica the "Oracle Inscription" transaction.

This transaction is the oracle's commitment with the bet, and its acceptance from the players. The inputs come from the Oracle's deposit and the players joint account, some of it goes to an account controlled by the oracle and the players. The portion left goes to any of the players if the oracle does not give the correct answer or goes back to the oracle after the bet is done, we call this output "two answers penalty", as it is a penalty for the oracle when it misbehaves.

This transaction also binds the oracle anwers, at this step the oracle generates two random strings to be used as answers, one for each possible outcome. This strings are to remaing secret until the oracle answers the event's outcome. However the hashs of this strings are included in the *two answers penalty* output, if a player gets to know both answers can spend this output.

[2]For instance an URL to a website with the bet description. Oracles are responsible to check the description fetched with the hash provided in the transaction.

[3]This money is gone forever.

Bet

Once the required oracles are inscribed to participate the "Bet" transaction is built by the players. The input contains what is remaining from the original players contributions to the prize, controlled by the players' joint account plus most[4] of the oracles deposit, controlled by the corresponding oracle and both players. Because of this, the transaction must be signed by both players and all the oracles.

The first two outputs are the prize, divided in two, so in case there is no answer from the oracles half of the prize returns to each player. Each of them getting back almost all the money initially spend, some is lose on fees and oracle payments. Esto depende de los valores de las variables. On the other hand, if the oracles get to an agreement and the bet is resolved on time. Using the oracles' answer the winner player can spend both outputs.

The next n outputs are the oracle payments, they are spendable by the corresponding oracle plus any of its answers. Spending this output requires to make public the oracle's answer. This way the oracle is guaranteed to get its payment when responding on time, and players know the oracle only gets pay when it answers. If the transaction is not spent in a defined time ($\mathrm{T}_{\mathrm{reply}}$) we say the oracle didn't answer and the money can be spend by the players joint account. This prevents the oracle of taking the money after not responding in the required time.

The last n outputs are a withholding for the same amount of the oracle payment, if the oracle gives a wrong answer for the oucome this money goes to the real winner, this way we take the payment out from the oracle. If the oracle behaves as expected, this money is spendable by the oracle some time after ($\mathrm{T}_{\mathrm{undue}}$) the bet is resolved.

They have three options to

HAblar de la comunicacion

[4]The other portion of this deposit was used in the *two answers penalty* output in the *Oracle Inscription* transaction.

Chapter 4

Conclusion

4.1. Conclusion

Bibliography

[1] Ross Anderson, Charalampos Manifavas, and Chris Sutherland. Netcard—a practical electronic-cash system. In *International workshop on security protocols*. Springer, 1996, pages 49–57.

[2] Europen Central Bank. Virtual Currency Schemes. 2012. URL: `https://www.ecb.europa.eu/pub/pdf/other/virtualcurrencyschemes201210en.pdf` (visited on 01/03/2017).

[3] Simon Barber, Xavier Boyen, Elaine Shi, and Ersin Uzun. Bitter to better—how to make bitcoin a better currency. In *International conference on financial cryptography and data security*. Springer, 2012, pages 399–414.

[4] Jean-Paul Boly, Antoon Bosselaers, Ronald Cramer, Rolf Michelsen, Stig Mjølsnes, Frank Muller, Torben Pedersen, Birgit Pfitzmann, Peter De Rooij, Berry Schoenmakers, et al. The esprit project cafe—high security digital payment systems. *Computer security—esorics 94*:217–230, 1994.

[5] David Chaum. Achieving electronic privacy. *Scientific american*, 267(2):96–101, 1992.

[6] David Chaum. Blind signatures for untraceable payments. In *Advances in cryptology*. Springer, 1983, pages 199–203.

[7] David Chaum, Amos Fiat, and Moni Naor. Untraceable electronic cash. In *Proceedings on advances in cryptology*. Springer-Verlag New York, Inc., 1990, pages 319–327.

[8] Victor Costan and Srinivas Devadas. Intel sgx explained. *Iacr cryptology eprint archive*, 2016:86, 2016.

[9] Christian Decker and Roger Wattenhofer. Information propagation in the bitcoin network. In *Peer-to-peer computing (p2p), 2013 ieee thirteenth international conference on*. IEEE, 2013, pages 1–10.

[10] Whitfield Diffie and Martin Hellman. New directions in cryptography. *Ieee transactions on information theory*, 22(6):644–654, 1976.

[11] Cynthia Dwork and Moni Naor. Pricing via processing or combatting junk mail. In *Annual international cryptology conference*. Springer, 1992, pages 139–147.

[12] Liam Edwards-Playne. The invention of the blockchain. 2013. URL: `https://medium.com/@liamzebedee/the-invention-of-the-blockchain-fe25be0caebc` (visited on 05/24/2017).

[13] Michael J Fischer. The consensus problem in unreliable distributed systems (a brief survey). In *International conference on fundamentals of computation theory*. Springer, 1983, pages 127–140.

[14] Sally M Gainsbury, Alex Russell, Robert Wood, Nerilee Hing, and Alex Blaszczynski. How risky is internet gambling? a comparison of subgroups of internet gamblers based on problem gambling status. *New media & society*, 17(6):861–879, 2015.

[15] Mark Griffiths and Andrew Barnes. Internet gambling: an online empirical study among student gamblers. *International journal of mental health and addiction*, 6(2):194–204, 2008.

[16] MD Griffiths. Crime and gambling: a brief overview of gambling fraud on the internet. *Internet journal of criminology*, 2010.

[17] Sarah Jane Hughes, Stephen T Middlebrook, and Broox W Peterson. Developments in the law concerning stored-value cards and other electronic payments products, 63 bus. *Law*, 237:237–40, 2007.

[18] Hrishikesh Huilgolkar. Introducing winsome.io. 2017. URL: `https://blog.winsome.io/introducing-winsome-io-86ba703f33c` (visited on 05/24/2017).

[19] John Ferlito John Ferlito Robert Lord. Oraclize docs. 2016. URL: `http://docs.oraclize.it/` (visited on 07/09/2017).

[20] Tomasz Kolinko and the Orisi team. Orisi white paper. 2014. URL: `https://github.com/orisi/wiki/wiki/Orisi-White-Paper` (visited on 06/07/2017).

[21] Anna Lysyanskaya and Zulfikar Ramzan. Group blind digital signatures: a scalable solution to electronic cash. In *Financial cryptography*. Springer, 1998, pages 184–197.

[22] John L McMullan and Aunshul Rege. Online crime and internet gambling. *Journal of gambling issues*:54–85, 2010.

[23] Andrew Miller and Joseph J LaViola Jr. Anonymous byzantine consensus from moderately-hard puzzles: a model for bitcoin, 2014.

[24] Satoshi Nakamoto. Bitcoin: a peer-to-peer electronic cash system. 2008.

[25] John Naughton. Is blockchain the most important it invention or our age? 2016. URL: `https://www.theguardian.com/commentisfree/2016/jan/24/blockchain-bitcoin-technology-most-important-tech-invention-of-our-age-sir-mark-walport` (visited on 05/24/2017).

[26] Tatsuaki Okamoto and Kazuo Ohta. Universal electronic cash. In *Annual international cryptology conference*. Springer, 1991, pages 324–337.

[27] Bart Preneel. Analysis and design of cryptographic hash functions. PhD thesis. Katholieke Universiteit te Leuven, 1993.

[28] Bhiru Shelat and Florian N Egger. What makes people trust online gambling sites? In *Chi'02 extended abstracts on human factors in computing systems*. ACM, 2002, pages 852–853.

[29] Felix Sun and Peitong Duan. Solving byzantine problems in synchronized systems using bitcoin, 2014.

[30] Fan Zhang, Ethan Cecchetti, Kyle Croman, Ari Juels, and Elaine Shi. Town crier: an authenticated data feed for smart contracts. In *Proceedings of the 2016 acm sigsac conference on computer and communications security*. ACM, 2016, pages 270–282.