A Primer on Bitcoin

Zachary Waterman

December 23, 2015

Contents

Abstract

This paper seeks to elucidate the often lost particulars of Bitcoin's history, technical components, uses, possibilities, criticisms, and political significance. It does so by delving into the academic literature regarding each of these subjects.

1 How Did Bitcoin Come About?

1.1 History of Cryptography

Cryptography is the practice and study of techniques for secure communication in the presence of third parties called adversaries [35]. More generally, cryptography involves constructing and analyzing protocols that prevent third parties or the public from reading private messages [35].

Until about the the 1970s, cryptography was mainly practiced in secret by military or spy agencies [35]. However, that changed when two publications brought it out of the closet into public awareness: the US government publication of the Data Encryption Standard (DES), a block cipher which became very widely used; and the first publicly available work

on **public-key cryptography**, by Whitfield Diffie and Martin Hellman [35]. In the late 1990s, these ideas coalesced into something like a movement [35].

1.2 Cypherpunk Culture

A **cypherpunk** is any activist advocating widespread use of strong cryptography as a route to social and political change [36]. Originally communicating through the Cypherpunks electronic mailing list, informal groups aimed to achieve privacy and security through proactive use of cryptography [36]. Cypherpunks have been engaged in an active movement since the late 1980s [36].

The tenants of that movement were established in the **Cypherpunk's Manifesto** by Eric Hughes in 1993 [15]. He argued that privacy, which he defines as the indispensable power to selectively reveal oneself to the world, is contingent upon his moment's ability to ensure that each party necessary to a transaction acquire knowledge only of that which is directly necessary for the transaction [15]. Therefore, according to Hughes, free societies require an anonymous transaction system, which would would require powerful cryptography [15].

The more radical members of the movement refer to themselves as **cryptoanarchists**. Following the **Crypto Anarchist Manifesto** of Timothy May, they believe that emerging cryptographic technologies will inevitably lead to the disintegration of governments as people seek ever freer markets and increased privacy [25]. Proponents of this worldview see government regulation as oppressive barbed wire [25].

The **Cypherpunks mailing list** was started in 1992, and by 1994 had 700 subscribers [24]. At its peak, it was a very active forum with technical discussion ranging over mathematic, cryptography, computer science, and political and philosophical discussion [36]. In early 1997, Jim Choate and Igor Chudov set up the Cypherpunks Distributed Remailer, a network of independent mailing list nodes intended to eliminate the single point of failure inherent in a centralized list architecture [36]. This was the first in a long line of cypherpunk inventions, such as BitTorrent, PGP, TOR, SSL/ TLS, OpenSSL, Mojonation, crypto.cat, and Tahou-LAFS [7]. The cypherpunk innovations utilized by Bitcoin will be elucidated later in this paper.

1.3 The Beginnings of Cryptocurrencies

The idea of a **cryptocurrency**, a digital currency using cryptography to secure transactions, manifested itself over and over again on the Cypherpunks list. David Chaum was the first to propose such an idea [17]. Although his eCash startup failed, his initiative inspired a series of similar attempts [17]. For instance, in 1998, Wei Dai, a member of the Cypherpunks list, presented an electronic currency called "b-money [10]." He introduced his research with a statement of approval for Tim May's crypto anarchy [10]. One of the assumptions of the paper was that a cryptocurrency is necessary for the existence of a

crypto anarchic community, because a "community is defined by the cooperation of its participants, and efficient cooperation requires a medium of exchange and a way to enforce contract[10]." Moreover, he asserted that the "b-money" system could replace the need for a government-issued currency [10].

Wei Dai's "b-money" plan shares many key technical characteristics with Bitcoin [10]. But the proposal was admittedly "impractical [10]." Bitcoin solved the problems of "b-money" with a synthesis of several cryptographic innovations [17]. "B-money" is recognized on the official Bitcoin website (originally registered by Satoshi himself) as a precursor to Bitcoin, and has been identified in law journal articles as its intellectual parent [17]. However, Wei Dai's "b-money" was explicitly connected to the cypherpunks and crypto anarchy, Bitcoin's creator never endorsed either ideology. By 2008, the year that the Bitcoin white paper was published, the cypherpunks mailing list was past its heyday, having been declared dead by its founding member John Gilmore in 2001 [17]. The Bitcoin white paper did not appear in the Cypherpunks list, but rather the Cryptography mailing list [17].

1.4 Hurdles to Designing a Functional Cryptocurrency

1.4.1 Byzantine General's Problem

Reliable computer systems must handle malfunctioning components that give conflicting information to different parts of the system [23]. This situation can be expressed abstractly in terms of a group of generals of the Byzantine army camped with their troops around an enemy city [23]. Communicating only by messenger, the generals must agree upon a common battle plan [23]. However, one or more of them may be traitors who will try to confuse the others [23]. The Byzantine Generals problem is to find an algorithm to ensure that the loyal generals will reach agreement [23].

Any successful algorithm will guarantee the following:

1. All loyal generals decide upon the same plan of action.

   The loyal generals will all do what the algorithm says they should, but the traitors may do anything they wish [23].

2. A small number of traitors cannot cause the loyal generals to adopt a bad plan.

   A small number of traitors can affect the decision only if the loyal generals were almost equally divided between the two possibilities, in which case neither decision could be called bad [23].

The **Byzantine Generals Problem** is deceptive in that it appears simple. Its difficult arises from the fact that if the generals can send only oral messages, then no solution will work unless more than two-thirds of the generals are loyal [23]. An oral message is one whose contents are completely under the control of the sender, so a traitorous sender can transmit any possible message [23]. Such a message corresponds to the type of message

that computers typically send to one another and is problem effective cryptocurrencies must address elegantly [23].

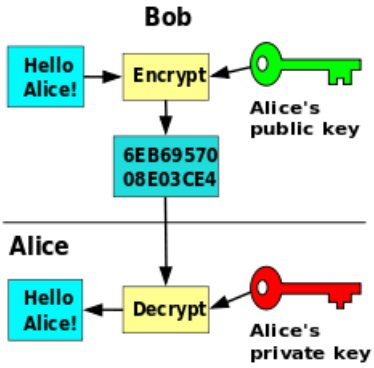
1.4.2 Double-Spending Problem

A **double-spending attack** is a successful attempt to first convince a merchant that a transaction has been confirmed, and then convince the entire network to accept some other transaction; the merchant would be left with neither product nor coins, and the attacker will get to keep both [31]. This is a problem of synchronization - there needs to be some universally accepted signal indicating that some transaction is final and that no conflicting transaction can ever be accepted [31]. Given two conflicting transactions, it does not really matter which of them will be accepted, as long as there is a way to know that one transaction has been accepted and can no longer be reversed [31]. Bitcoin's solution of **Proof of Work** to this thorny problem will be discussed later in this paper.

2 Cryptographic Innovations Used in Bitcoin

2.1 Public Key Cryptography

Public-key cryptography refers to a set of cryptographic algorithms that are based on mathematical problems that currently admit no efficient solution – particularly those inherent in certain integer factorization, discrete logarithm, and elliptic curve relationships [40]. It is computationally easy for a user to generate a public and private key-pair and to use it for encryption and decryption [40]. The strength lies in the "impossibility" (computational impracticality) for a properly generated private key to be determined from its corresponding public key [40]. Thus the public key may be published without compromising security [40]. Security depends only on keeping the private key private [40]. Public key algorithms, unlike symmetric key algorithms, do not require a secure channel for the initial exchange of one (or more) secret keys between the parties [40].



Because of the computational complexity of asymmetric encryption, it is typically only used for short messages, typically the transfer of a symmetric encryption key [40]. This symmetric key is then used to encrypt the rest of the potentially long and heavy conversation [40].The symmetric encryption/decryption is based on simpler algorithms and is much faster [40].

Message authentication involves hashing the message to produce a "digest," and encrypting the digest with the private key to produce a digital signature [40]. Thereafter

anyone can verify this signature by (1) computing the hash of the message, (2) decrypting the signature with the signer's public key, and (3) comparing the computed digest with the decrypted digest [40]. Equality between the digests confirms the message is unmodified since it was signed, and that the signer, and no one else, intentionally performed the signature operation ? presuming the signer's private key has remained secret. The security of such procedure depends on a hash algorithm of such quality that it is computationally impossible to alter or find a substitute message that produces the same digest - but studies have shown that even with the MD5 and SHA-1 algorithms, producing an altered or substitute message is not impossible [40]. The current hashing standard for encryption is SHA-2 [40]. The message itself can also be used in place of the digest [40].

Public-key algorithms are fundamental security ingredients in cryptosystems, applications and protocols [40]. They underpin various Internet standards, such as Transport Layer Security (TLS), S/MIME, PGP, and GPG [40]. Some public key algorithms provide key distribution and secrecy (e.g., Diffie?Hellman key exchange), some provide digital signatures (e.g., Digital Signature Algorithm), and some provide both (e.g., RSA) [40].

2.2 Digital Signature

Digital signature schemes are designed to provide the digital counterpart to handwritten signatures [18]. A digital signature is a number dependent on some secret known only to the signer (the signer's private key), and, additionally, on the contents of the message being signed [18]. Signatures must be verifiable - if a dispute arises as to whether an entity signed a document, an unbiased third party should be able to resolve the matter equitably, without requiring access to the signer's private key [18].

The original Digital Signature Algorithm (DSA) was specified in a U.S. Government Federal Information Processing Standard (FIPS) called the Digital Signature Standard [18]. Its security is based on the computational intractability of the discrete logarithm problem (DLP) in prime-order sub groups of $\mathbb{Z}_p$ [18].

A digital signature is a mathematical scheme for demonstrating the authenticity of a digital message or documents [37]. A valid digital signature gives a recipient reason to believe that the message was created by a known sender, that the sender cannot deny having sent the message (authentication and non-repudiation), and that the message was not altered in transit (integrity) [37]. Digital signatures are commonly used for software distribution, financial transactions, and in other cases where it is important to detect forgery or tampering [37].

2.3 Elliptic Curve Cryptography

Elliptic curve cryptosystems (ECC) were invented by Neal Koblitz and Victor Miller in 1985 [18]. They can be viewed as elliptic curve analogues of the older discrete logarithm (DL) cryptosystems in which the subgroup of $\mathbb{Z}_p$ is replaced by the group of points on an

elliptic curve over a finite field [18]. It is a form of public key cryptography. Some public key algorithms may require a set of predefined constants to be known by all the devices taking part in the communication [26]. Domain parameters in ECC are examples of such constants [26]. Public key cryptography, unlike private key cryptography, does not require any shared secret between the communicating parties but it is much slower than the private key cryptography [26].

The mathematical operations of ECC is defined over the elliptic curve $y^2 = x^3 + ax + b$, where $4a^3 + 27b^2 \neq 0$ [26]. Each value of the a and b gives a different elliptic curve [26]. All points (x, y) which satisfies the above equation plus a point at infinity lies on the elliptic curve [26]. The public key is a point in the curve and the private key is a random number [26]. The public key is obtained by multiplying the private key with the generator point G in the curve [26]. The generator point G, the curve parameters a and b, together with few more constants constitutes the domain parameter of ECC [26]. One main advantage of ECC is its small key size ([26]. A 160-bit key in ECC is considered to be as secured as 1024-bit key in RSA [26].

Elliptic curve cryptography is an approach to public-key cryptography based on the algebraic structure of elliptic curves over finite fields [38]. ECC requires smaller keys compared to non-ECC cryptography (based on plain Galois fields) to provide equivalent security [38]. Elliptic curves are applicable for encryption, digital signatures, pseudo-random generators and other tasks [38].

2.4 Hashing

Secure hash algorithms are iterative, one-way functions that can process a message to produce a condensed message called a *message digest* [29]. These algorithms enable the determination of a message's integrity: any change to the message will, with a very high probability, result in a different message digest [29].

Each algorithm can be described in two stages: preprocessing and hash computation [29]. Preprocessing involves padding a message, parsing the padded message into m-bit blocks, and setting initialization values to be used in the hash computation [29]. The hash computation generates a *message schedule* from the padded message and uses that schedule, along with functions, constants, and word operations to iteratively generate a series of hash values [29]. The final hash value generated by the hash computation is used to determine the message digest [29].

Hashing algorithms differ most significantly in the number of bits of security that are provided for the data being hashed - this is directly related to the message length [29]. When a secure hash algorithm is used in conjunction with another algorithm, there may be requirements specified elsewhere that require the use of a secure hash algorithm with a certain number of bits of security [29]. For instance, if a message is being signed with a digital signature algorithm that provides 128 bits of security, then that signature algorithm may require the use of a secure hash algorithm that also provides 128 bits of security[29].

2.4.1 SHA-256

SHA-256 is a particularly popular secure hashing algorithm. It may be used to hash a message, M, having a length of ℓ bits, where $0 \leq \ell < 2^{64}$ [29]. The alrogithm uses 1) a message schedule of sixty-four 32 bit words, 2) eight working variables of 32 bits each, and 3) a hash value of eight 32-bit words [29]. The final result of SHA-256 is a 256-bit message digest [29].

The words of the message schedule are labelled $W_0, W_1, ..., W_{63}$ [29]. The eight working variables are labeled **a, b, c, d, e, f, g, and h** [29]. The words of the hash value are labeled $H_0^{(i)}, H_1^{(i)}, ..., H_7^{(i)}$, which will hold the initial hash value, $H^{(0)}$, replaced by each successive intermediate hash value (after each message block is processed), $H^{(i)}$, and ending with the final hash value, $H^{(N)}$[29]. SHA 256 also uses two temporary words, T_1 and T_2 [29].

After preprocessing is completed, each message block, $M^{(1)}$, $M^{(2)}$, ... , $M^{(N)}$, is processed in order, using the following steps [29].

1. Prepare the message schedule, $\{W_t\}$:

$$W_t = \begin{cases} M_t^{(i)} & 0 \leq t \leq 15 \\ \sigma_1^{\{256\}}(W_{t-2}) + W_{t-7} + \sigma_0^{\{256\}}(W_{t-15}) + W_{t-16} & 16 \leq t \leq 63 \end{cases}$$

2. Initialize the eight working variables, **a, b, c, d, e, f, g, and h**, with the $(i-l)^{st}$ hash value:

$$\begin{aligned} a &= H_0^{(i-1)} \\ b &= H_1^{(i-1)} \\ c &= H_2^{(i-1)} \\ d &= H_3^{(i-1)} \\ e &= H_4^{(i-1)} \\ f &= H_5^{(i-1)} \\ g &= H_6^{(i-1)} \\ h &= H_7^{(i-1)} \end{aligned}$$

3. For $t = 0$ to 63:

$$T_1 = h + \sum_{1}^{\{256\}}(e) + Ch(e, f, g) + K_t^{\{256\}} + W_t$$
$$T_1 = \sum_{0}^{\{256\}}(a) + Maj(a, b, c)$$
$$h = g$$
$$g = f$$
$$f = e$$
$$e = d + T_1$$
$$d = c$$
$$c = b$$
$$b = a$$
$$a = T_1 + T_2$$

4. Compute the i^{th} intermediate hash value $H^{(i)}$.

$$H_0^{(i)} = a + H_0^{(i-1)}$$
$$H_1^{(i)} = b + H_1^{(i-1)}$$
$$H_2^{(i)} = c + H_2^{(i-1)}$$
$$H_3^{(i)} = d + H_3^{(i-1)}$$
$$H_4^{(i)} = e + H_4^{(i-1)}$$
$$H_5^{(i)} = f + H_5^{(i-1)}$$
$$H_6^{(i)} = g + H_6^{(i-1)}$$
$$H_7^{(i)} = h + H_7^{(i-1)}$$

5. After repeating steps one through four a total of N times (i.e., after processing $M^{(N)}$), the resulting 256-bit message digest of the message, M, is:

$$H_0^{(N)} \| H_1^{(N)} \| H_2^{(N)} \| H_3^{(N)} \| H_4^{(N)} \| H_5^{(N)} \| H_6^{(N)} \| H_7^{(N)}$$

2.4.2 Hashcash

Hashcash was originally proposed as a mechanism to throttle systematic abuse of unmetered internet resources such as email in 1998 [3]. The **hashcash** CPU **cost-function** computes a token which can be used as a **proof-of-work** [3].

A **cost-function** should be efficiently verifiable, but expensive to compute (Back). In the context of cost functions, **clients** refer to users who must computer a **token** (denoted T) using a cost-function MINT() which is used to create tokens with a **server** [3]. The term **mint** refers to the cost-function because of the analogy between creating tokens and minting physical money [3].

The functions are sorted by the amount of work, w, that the user will have to expend on average to mint a token (Back). In addition, hashcash is computed relative a service name, s, to prevent tokens minted for one server being used on another [3]. The hashcash function is defined as follows:

$PUBLIC:$	hash function $H(\cdot)$ with output size k bits
$\tau \leftarrow MINT(s, w)$	**find** $x \epsilon_R [0,1] \times \mathbf{st} H(s \| x) =_w 0^k$
	return (s, x)
$\nu \leftarrow \text{VALUE}(\tau)$	$H(s \| x) =_v 0^k$
	return v

Hashcash is a non-interactive, publicly audit-able, trapdoor-free cost function with unbounded probabilistic cost [3]. The hashcash cost-fuction is based on finding **partial hash collisions** on all the 0 bits in k-bit string 0^k [3]. The fastest algorithm for computing partial collisions is brute computational force [3].

The server needs to keep a double spending database of spent tokens, to detect and reject attempts to spend the same token again [3]. To prevent the database growing indefinitely, the service can include the time at which it was minted [3]. This allows the server to discard entries from the spent database after they have expired [3]. Hashcash is the foundation of Bitcoin's **proof-of-work** mechanism.

2.5 Pretty Good Privacy

Pretty Good Privacy (PGP) is a data encryption and decryption computer program that provides cryptographic privacy and authentication for data communication [39]. PGP is often used for signing, encrypting, and decrypting texts, e-mails, files, directories, and whole disk partitions and to increase the security of e-mail communications[39]. It was created by Phil Zimmermann in 1991[39].

PGP encryption uses a serial combination of hashing, data compression, symmetric-key cryptography, and finally public-key cryptography; each step uses one of several supported algorithms [39]. Each public key is bound to a user name and/or an e-mail address [39]. The first version of this system was generally known as a web of trust to contrast with the X.509 system, which uses a hierarchical approach based on certificate authority and which was added to PGP implementations later [39]. Current versions of PGP encryption include both options through an automated key management server[39].

3 The White Paper

3.1 What is Bitcoin?

Bitcoin was invented by Satoshi Nakamoto, who published the invention on 31 October 2008 in a research paper called "Bitcoin: A Peer-to-Peer Electronic Cash System".[34] It was implemented as open source code and released in January 2009 [34]. Bitcoin is often called the first cryptocurrency, although prior systems existed [34]. Bitcoin is more correctly described as the first decentralized digital currency [34].

It is a currency both created and tracked through a decentralized peer-to-peer network [17]. The Bitcoin protocol is an open source protocol available through several different clients–i.e., the basic code is nonproprietary, and allows for many different kinds of software to be created to interface with it [17].

3.2 Political Context

Bitcoin was introduced to the world in the context of the 2008 financial crisis [17]. The language used in the white paper itself is apolitical, declining to even use the "government" or "sovereign," instead setting up Bitcoin as an alternative to a system controlled by "financial institutions [27]." This careful language is in stark contrast to Bitcoin's predecessors proposals to facilitate true crypto anarchy [17]. In the moment of financial crisis in 2008, Satoshi instead focuses his skepticism and anger towards the banks [17]. Still, despite his animosity towards the financial system and its entrenched interests, Satoshi did not seem interested in using Bitcoin as a weapon against the government, unlike the cypherpunks [17].

3.3 How does it work?

Payments are made from one **virtual wallet** to another, using cryptographic innovation of **private/public keys** to assign ownership [17]. The 'coin' is transferred by digitally signing a **hashed** previous transaction with their private key and with the intended recipient's public key [17]. The broadcast of the signature on top of the has constitutes a transaction, which then becomes part of the decentralized accounting system created by the protocol

[17]. Due to the frequency of transactions and number of nodes, not all nodes will agree on what the transaction history is at a given time; however, over time, parts of the time-stamped transaction history (called the "**block chain**") are "resolved" and transactions are "finalized" as nodes accept what becomes the one true transaction history–what is referred to as the "longest blockchain [17]."

Each node (that is, a running instance of the client) "**works**" on the latest block of transaction history in the blockchain, repeatedly applying **the cryptographic hash function SHA-256** to the data in order to come up with a desirable result, also known as "**satisfying the proof-of-work** [17]." This is done by arriving at a hashed string beginning with a certain number of 0s (this number is sensitive to the rate of mining, and is automatically changed to tweak the "difficulty" of mining) [17]. The block is incrementally adjusted with a **nonce**, a small number of characters at the end that unpredictably changes how the hash comes out, until the target is reached [17]. When the target is reached, it is broadcast to the entire network [17]. The other nodes can verify that this is the correction "solution" to the "problem," and then proceed to accept this hashed block as the latest resolved block in the chain [17]. They then move on to **work** on the next block of transactions, using the latest resolved as a starting point[17].

The resolution of blocks within this decentralized accounting system is also the mechanism by which new bitcoins are released into the economy [17]. Bitcoins are not released by a central authority, but are rather "minded" by individual users according to the protocol [17]. The node (or pool of nodes) that first resolves the block is rewarded with a set number of bitcoins [17]. At the moment, the base mining reward is set at 25 BTC, with optional bounties collected from those seeking to process large transactions (which make the block larger) [17]. The **work** to resolve blocks is computationally difficult, and takes enormous processing power [17].

3.4 Security

The mechanism of **public/private key** transactions allow any node in the Bitcoin network to transact with any other node, transcending any geographic barrier [17]. The mechanism of public broadcast of the timestamped transaction history prevents **double-spending** of coin in transactions [17]. Transactions thus take 10 to 60 minutes to be verified by the network [17]. If a user attempts to **double-spend** one of those transactions will simply fail to go through as the blockchain becomes resolved [17]. In order to commit fraud upon the currency (forge the currency, or rather, alter the transaction history), an attacker must control the majority of CPU power in the entire network [17].

Bitcoin has the capacity to be anonymous, but in general, anonymity is not currently a prominent design feature [17]. Although it is possible to transact in a way that obscures one's identity, because the Bitcoin network necessarily broadcasts the history of all transactions ever made, analysis of this data can unveil revealing information about particular nodes and their transactional activities [17].

3.5 Monetary Policy

The protocol contains a number of economic rules that serve as monetary policy enacted in code [17]. These rules are "enforced collectively by the network," meaning that if the network came to a consensus about modifying the rules, these policies could change. The current Bitcoin monetary policy is directed primarily at controlling inflation [17]. Bitcoin has a hard limit of currency at about 21 million bitcoins [17]. This limit is estimated to be reached in 2030 [17]. To ensure transaction ease, each bitcoin is divisible up to 8 decimal places [17]. The mining reward is reduced over time at a set rate [17]. The code controls for the rating of mining by automatically increasing or relaxing the difficult of the **proof-of-work** in response to the rate of generation of blocks [17]. Interestingly, since the reward is given to the first miner to resolve the block, the protocol has resulted in a computer processing arms race, which has birthed massive "mining rigs" equipped with specialized computer chips specifically designed for Bitcoin mining [17].

3.6 Proof of Work

3.6.1 What is it?

Proof-of-work (PoW) is a principle to artificially impose transaction costs in the absence of a payment system [5]. The idea is to "charge" the requester of a service with the effort to present a solution to a problem that is much harder to solve than to verify [5]. In this way, PoW can help to ration access to services which would other be abused and solve the double-spending problem [5].

3.6.2 Relevance to Bitcoin

Bitcoin is a particularly innovative example of this principle. In a system set up by the Bitcoin protocol, users constantly participate in a lottery, and each user's chance of winning is proportional to the computing power he is willing to invest [5]. The task is to modify a document until its hash is of a particular structure [5]. In parallel, users publicly announce transactions of bitcoins, thereby expressing their intention to transfer a certain amount of this currency to another user [5]. The lottery is designed in such a way that every win, as a side-effect, returns a timestamp of all transactions in an atomic operation [5]. Users accept this timestamp after validating that the PoW has been delivered [5]. Then, the next round of the lottery starts [5]. The sequence of timestamps forms a history of transactions [5]. In the case that competing histories emerge (which is easily possible in a peer-to-peer network), users believe in the history for which the most PoW has been delivered [5]. This quickly resolves the conflict [5]. Altering any transaction timestamped in the past requires redoing all work that has been done afterwards [5]. As this becomes more unlikely the longer a transaction has been timestamped, manipulation is prevented and users collectively agree on a single history of transactions, thereby determining bitcoin

ownership [5].

The goal of Bitcoin is to replace trust in financial institutions with trust in PoW, thereby eliminating the need for large financial institutions and with it the fees they charge [5]. However, running the Bitcoin system is not free either [5]. As computational power is required for the PoW, hardware has to be acquired, powered, and maintained [5] .

3.6.3 51% Attack Potential

The Bitcoin system is secure as long as no single party controls more than 50% of the network's computing power [5]. If a party did, it could redo work of the past and ultimately outperform the honest part of the network [5]. While digital signatures on transactions prevent arbitrary manipulations, the attackers would have the power to double-spend their own Bitcoins, thereby generating profits for themselves, or to prevent transactions from being timestamped, thereby undermining the trust in the system [5]. The viability of the system depends on the assumption that nobody will ever gain this power [5].

4 Current Applications

4.1 Proof of Existence

4.1.1 What is Proof of Existence?

It is a service to anonymously and securely store an online distributed proof of existence for any document [28]. Documents are not stored in a database or in the bitcoin blockchain, so private data cannot be accessed by others [28].

All that is stored is a cryptographic digest of the file, linked to the time in which it was submitted [28]. In this way, the user can later certify that the data existed at that time [28]. This is the first online service allowing one to publicly prove that they have certain information without revealing the data or themselves, with a decentralized certification based on the bitcoin network [28].

The key advantages are anonymity, privacy, and getting a decentralized proof which can't be erased or modified by anyone (third parties or governments) [28]. Document's existence is permanently validated by the blockchain even if this site is compromised or down, so it does not depend or need to trust any central authority [28]. All previous data timestamping solutions lack this freedom[28].

4.1.2 Common Uses

1. **Demonstrating data ownership without revealing actual data** One can publicly reveal the digest and if conflict arises you can prove you had the data that generates the digest. This is useful for copyrighted material, patents, etc.[28]

2. **Document timestamping** One can prove certain data exists at a certain moment of time [28]. As the bitcoin blockchain is used to store the document proof, one can certify the existence of thier document without the need of a central authority [28]. The computing power of the whole bitcoin network is used to certify the data[28].

3. **Checking for document integrity** If ones stores a proof of their document and later re-uploads it, the system will only recognize it if it is completely and fully the same document [28]. The slightest change, and the system will recognize it is different, giving one the security that certified data can't be changed[28].

4.1.3 Technical Foundations

The document is certified via embedding its SHA256 digest in the bitcoin blockchain[28]. This is done by generating a special bitcoin transaction that encodes/contains the hash via an OPRETURN script[28]. This is a bitcoin scripting opcode that marks the transaction output as provably un-spendable and allows a small amount of data to be inserted, which in this case is the document's hash, plus a marker to identify all transactions [28].

Once the transaction is confirmed, the document is permanently certified and proven to exist at least as early as the time the transaction was confirmed [28]. If the document hadn't existed at the time the transaction entered the blockchain, it would have been impossible to embed its digest in the transaction [28]. This is because of the hash function's property of being second pre-image resistant [28]. Embedding some hash and then adapting a future document to match the hash is also impossible (due to the pre-image resistance of hash functions)[28]. This is why once the bitcoin blockchain confirms the transaction generated for the document, **its existence is proven, permanently, with no trust required**[28].

4.2 Smart Contracts

4.2.1 Origin of Smart Contracts

The idea of smart contracts goes way back to 1994, nearly the dawn of the World Wide Web itself [9]. That's when Nick Szabo, a cryptographer widely credited with laying the groundwork for bitcoin, first coined the term "smart contract [9]." At core, these automated contracts work like any other computer program's if-then statements [9]. They just happen to be doing it in a way that interacts with real-world assets [9]. When a pre-programmed condition is triggered, the smart contract executes the corresponding contractual clause [9].

Szabo's original theories about how these contracts could work remained unrealized because there was no digitally native financial system that could support programmable transactions [9]. It defeats the purpose of smart contracts if a bank still has to manually authorize the release and transfer of money[9]. "One big hurdle to smart contracts is that

computer programs can't really trigger payments right now," says Phil Rapoport, Ripple Labs' director of markets and trading[9].

The advent and increasingly widespread adoption of bitcoin is changing that, and as a result Szabo's idea has seen a revival [9]. Smart contract technology is now being built on top of bitcoin and other virtual currencies?what some have termed "Bitcoin 2.0" platforms[9]. Because bitcoin is itself is a computer program, smart contracts can speak to it just like they would any other piece of code[9]. The puzzle pieces are falling into place. A computer program can now trigger payments[9].

4.2.2 Current Commercial Offerings

There are currently two major open source projects working on smart contracts, both of which have taken big leaps forward this year[9]. One is called Codius and the other is Ethereum[9]. Codius was developed by Ripple Labs, which also created its own digital currency called Ripple[9]. Codius aims to be interoperable between a variety of cryptocurrency, such as Ripple and bitcoin, although it is managed by the private company[9].

5 Criticism

5.1 Environmental Concerns: How Much Energy Is Required?

In order to test whether Bitcoin could be a viable replacement for all other currencies, given the energy required to run a PoW network of that scale, a multistep procedure is necessary [5]. First, an estimate of the volume of all transactions taking place anywhere in the world ought to be calculated [5]. Second, find an estimate of the transaction costs incurred in a system using a centralized currency as a fraction of that volume [5]. Third, calculate the properties of a PoW network limited to the total transaction costs of this centralized currency [5]. Finally, using those dimensions, the total electricity required, and therefore the CO_2 produced, are calculated[5].

5.1.1 Calculating Total Transaction Volume

The Bank of International Settlements (BIS), located in Switzerland, publishes annual statistics on payment, clearing, and settlement systems [13]. The reports provide data for 23 countries, among them the United States, Germany, China, Japan, France, the United Kingdom, Russia, Italy, Canada [13]. The transactions taking place in these countries are used as a proxy for those of the entire world [5]. This appears justified as the economically strongest countries of the world are included [5]. In the latest report (Bank for International Settlements, 2011), the BIS reports the total yearly transaction volume for all 23 countries, categorized by method of payment [13]. Figures for 2010 can be found in table 1.

Table 1: Total value of transactions by payment instrument in US dollars, 2010 [13]

Payment Instrument	Transaction Volume (USD)
Credit Transfers	3.74E+14
Direct Debt	4.37E+13
Check	9.36E+13
E-Money	1.80E+10
Card Payments (Debit/Credit)	9.45E+12

Recall that the centralized system, to which the PoW network is compared, is an electronic card payment system, whose transaction costs will be estimated as a certain fraction of the volume [5]. Such systems are used to process small, cash-like transactions [5]. Therefore, large credit transfers between corporations, professional investors, or nations should not be included in the estimate, for applying transaction fees of an electronic card payment system to them would be unreasonable [5]. Consequently, the item credit transfers are excluded [5]. Direct debit on the other hand will usually represent cash-like transactions and is therefore included [5].

The next important item in the list is check, which is also a famous payment instrument. Looking at the data, one can see that more than 75% of the check transaction volume stems from China and the United States [13]. To identify if these transactions are cash-like one has a look at the average volume per transaction [13]. For China, it amounts to more than 52000 USD per transaction, which indicates that the volume stems in large parts from very huge transactions [13]. Thus, Chinas volume is excluded from the estimate [5]. For the United States however, the average value amounts to only 3000 USD [13]. This indicates that, while there will surely be large transactions, there will also be a number of small, cash-like ones [5].Therefore, 50% of the check transaction volume of the United States are included [5].

All other check transactions are included [5].. This also applies to all E-Money transactions as well as card payments, delivering a transaction volume of 9.03E+13 USD in 2010 [5]. Transactions for which actual cash is being used are not included yet [5]. For these, there is no known reliable data source [5]. However, statistics from the BIS report the total volume of ATM cash withdrawals for 2010, which amounts to 4.09E+12 USD [13]. Assuming that most of this cash is spent only once and then deposited into a bank account before being withdrawn again, it is used as an estimate for cash transactions [5]. Adding it, one ends up with a total transaction volume of 9.44E+13 USD [5].

5.1.2 Calculating the Total Transaction Costs of Centralized Payment

Given an estimate of the transaction volume of all cash-like transactions taking place anywhere on the world, one ought now to begin estimating the cost of these transactions charged by the financial system to the real economy [5]. As a reference, the electronic

payment system is used, as for such systems market prices are known[5]. In general, there are two different types of them, the first of which is a credit card system. Typically, credit card fees paid by merchants in different countries can vary around 1-3 percent of the transaction volume [33]. However, pricing of credit cards is subject to considerable suspicion [5]. Retailers have filed numerous lawsuits in the past as they believe institutions issuing these cards exaggerate their costs [8]. Thus, a credit card system is not the ideal reference for comparison with a PoW network. In addition, credit cards are not only used for ordinary payments, but also offer a credit function, i.e., the card holder buys products on credit and pays for them later [5]. This is an additional service not offered by a PoW network like Bitcoin [5]. Naturally, these services charge a credit risk premium [5]. This further strengthens the belief that costs of a credit card system are inadequately high for the purpose of this analysis [5].

The second type of electronic payment system is a debit card system. In contrast to credit cards, payments made with a debit card are transferred immediately (within days) from the user?s bank account to the recipient [5]. It does not provide a credit function . Also, the transaction costs are considerably lower as compared to credit cards [5]. Therefore, a debit card system is used for comparison to the PoW network, as it comes closest to a frictionless centralized payment system free of bells and whistles that have no counterpart in the PoW network [5]. The debit card system used in Germany, called electronic cash, charges merchants 0.3% of the transaction volume as a fee [5]. Applying this to the transaction volume estimated previously, one ends up with total transaction costs of 2.83E+11 USD for an imaginary world in which all transactions are processed with a centralized system comparable to debit cards [5].

5.1.3 Calculating the Size of a Global PoW Network

In the following step, the transaction cost of a centralized system is used as a budget to size a PoW network [5]. The first and most important question is what the components of that network should be [5]. Today?s Bitcoin network is run to a large extent by individuals [5]. They use their PCs and other equipment to create new blocks. However, once block creation becomes a commercial activity, it is unlikely that any kind of equipment owned by individuals could compete with specialized hardware as it is found in data centers [5]. Thus, creating blocks would quickly become unprofitable for them, leaving the market to professional players[5].

On the other extreme, economies of scale could result in only a very few data centers, owned by a small number of organizations that cover the entire market of PoW delivery [5]. Such a setting would clearly not be the decentralized currency envisioned by the Bitcoin pioneers [5]. However, given that the fate of the global financial system rests on the security of this system, the risk that somebody gained control over these data centers could under no circumstances be acceptable [5]. Moreover, as the organizations running the data centers might collude to exploit the system, one would be forced to trust them

[5]. For these reasons, in this scenario it is assumed that there is scale between these two extremes [5]. More precisely, a large number of independent data centers distributed all over the world is assumed, each small enough to implement a governance regime with effective checks and balances [5].

Having defined the main building block of the PoW network, we now analyze the typical cost structure of a data center. According to Belady (2007), there are three main components.

- Acquisition cost: The cost of acquiring hardware. Typically amounts to about 25% of the total cost [6].

- Energy cost: The cost of electricity required to run the data center. Typically amounts to about 30% of the total cost [6].

- Infrastructure cost: The cost of providing the environment in which the data center is run (e.g., the rent for the building). Typically amounts to about 45% of the total cost [6].

Of particular importance for this analysis are the costs for electricity, as these will be the main driver for pollution [5]. The analysis therefore pursues the following course in the construction of the PoW network [5]. An estimate of computing power is achieved with respect to consumed electricity, under the assumption that particularly energy efficient hardware is used [5]. At the same time, the fact that such hardware might have higher acquisition cost is disregarded [5]. This means the data centers of the network are energy efficient but still have the above mentioned cost structure [5].

It is worth noticing that the costs of the communication infrastructure required to operate the system are not taken into account [5]. For instance, Kaminsky (2011) expresses doubt that the decentralized architecture of the current Bitcoin network could scale up to the size of our PoW network [20]. Other authors already investigated the notion of scalability and provide suggestions how a large-scale system could be designed [4]. For this calculation, it is assumed that smart protocols could overcome all problems [5]. However, as we do not know how such a solution might look like, we do not estimate any communication costs for now and focus solemnly on effort for delivering PoW [5]. With a total budget of 2.83E+11 USD, of which 30% are being spent for electricity, an electricity budget of 8.49E+10 USD is assumed [5]. The next task is to determine the amount of electricity that can be produced [5]. Again, current market prices are used[5].

Several countries do have very low electricity prices but are rather unimportant with respect to global energy production [5]. Considering only countries with a major output of electricity, Russia has the lowest (0.10 USD/kWh) [5]. Consequently, this value is used as a lower bound for the cost of electricity [5]. Given the electricity budget of 8.49E+10 USD, a total of 8.49E+11 kWh is available for computation [5]. This equals 3.06E+18 Ws (watt-seconds) after a unit conversion [5]. Note that this is the amount of electricity available per year, since it is derived from previously calculated annual transaction costs [5].

5.1.4 Calculating the Computational Power of the PoW Network

It is now appropriate to estimate the computing power that could be sustained over the year [5]. An important point is how computing power should be measured for this purpose [5]. Typically, high performance computing power is measured in floating point operations per second (FLOPS) [5]. Delivering PoW in the Bitcoin system however heavily relies on computing hashes, for which integer operations are required [5]. Technically, the FLOPS measure should be replaced with a more appropriate metric such as megahash per second [5]. Unfortunately, due to the widespread adoption of FLOPS, data on other metrics is not directly available [5]. Therefore, it is adopted for this analysis [5].

In the most recent ranking (November 2011), the most efficient system is an experimental computer from the IBM Blue Gene/Q project at IBM ? Rochester, achieving 2026.48 MFLOPS/W [5]. The performance of an average green supercomputer however is not even close to this value [5].

For the ?average? energy-efficient computer used to construct the hypothetical PoW network, a value of 181.77 MFLOPS/W is appropriate [5]. It equals the median of the Green500 supercomputers [5]. Note that MFLOPS/W means million floating point operations per second per watt, i.e., million floating point operations per watt- second [5]. To keep the notation clear, the efficiency from now on is expressed in terms of (floating point) operations per watt-second (Ops/Ws) [5].

Multiplying the estimate for energy-efficiency (1.82E+8 Ops/Ws) with the available electricity delivers a total of 5.56E+26 Ops that can be achieved per year [5]. Divided by the 3.15E+7 seconds of a year, the computing power of the PoW network is 1.76E+19 Ops/s [5].

5.1.5 Calculating Total Environmental Impact

Given the annual power consumption of 3.06E+18 Ws, it is now straightforward to estimate the environmental impact of the PoW network [5]. The environmental impact will be measured in terms of CO_2 emissions. To start with this, information on how energy is being produced is required [5]. Such data can be obtained from the International Energy Agency, which provides statistics on the worldwide production of electricity in 2009 with respect to the energy carrier [1].

Knowing the relative importance of each energy carrier, one can combine this information with corresponding average CO_2 emissions. Computing an average of the CO_2 emissions, weighted by the relative importance of each carrier, delivers an average CO_2 emission of about 718 gram/kWh or, after a unit conversion, of 1.99E-7 kg/Ws [5].

Multiplying this estimate with the power consumption of 3.06E+18 Ws, the PoW network is responsible for a total of 6.10E+11 kg of CO_2 per year [5]. Compared to the total man-made CO_2 emissions due to fuel combustion in 2009, which have been 2.90E+13 kg [1], the PoW network at this scale would increase CO_2 emissions by more than 2.1%[5]. This is about the share of global commercial air traffic [5]. It is assumed that the CO_2

emissions of the centralized system are negligible compared to global emissions, **making the effect of the PoW network a net increase in global CO_2 emissions** [5].

6 Future Possibilities

6.1 Viability as an Intergalactic Currency

It is unlikely that Bitcoin could function effectively as an intergalactic currency without untenable latency in each transaction. The nearest galaxy, Andromeda, is 2,500,000 light-years away. That is simply too far for the signal-carrying waves to travel within a reasonable time span. Data travels at the speed of light, which is 186,000 miles per second. A light year is 5.879e+12 miles. Thus, Andromeda is 1.46975e+19 miles from earth. Simple division finds that it would take a signal traveling from earth 914,569,643 days to reach Andromeda. Humans would need to disrupt death before Bitcoin could become an effective intergalactic currency. Interplanetary transactions, however, would be quite possible.

6.2 Political Implications

Although anarchist and libertarian leanings are prevalent in the bitcoin community, users fall along all points of the political spectrum. Some might seek violent overthrow of the government (e.g., a Blacknet type of future), others hope to simply decouple money from the Federal Reserve. Still, for others, using Bitcoin represents the creation of an insular community that rebels not through violence but through retreat [17]. The project of anonymity and Bitcoin ought to be interpreted as one of evasion of the state [17].

7 Conclusion

Where Bitcoin goes and what it becomes largely depends on what kind of political inclinations win out in its community[17]. If one is satisfied with Bitcoin becoming a mere transaction system, a new PayPal, then the rise of the exchanges and the current hyperdeflation is not as big of a concern [17]. But those who see Bitcoin as intrinsic to anonymous political speech have a vested interest in limiting the power of the exchanges (who retain data key to potential government prosecution and persecution[17]). This may involve solving the volatility issue with price-stabilizing mechanisms, to bring Bitcoin out of the shadow of sovereign currency [17]. For those who use Bitcoin as for purposes of cultural identity that particular development will be most important of all [17]. And finally, resistance or acceptance of regulation by sovereign governments like the United States depends largely on the political leaning of the commentator [17].

8 Glossary

Blockchain: a distributed database that maintains a continuously growing list of data records that are hardened against tampering and revision, even by operators of the data store's nodes. The most widely known application of a block chain is the public ledger of transactions for cryptocurrencies used in bitcoin. This record is enforced cryptographically and hosted on machines running the software.

Hash function: any function that can be used to map data of arbitrary size to data of fixed size. The values returned by a hash function are called hash values, hash codes, hash sums, or simply hashes. One use is a data structure called a hash table, widely used in computer software for rapid data lookup. Hash functions accelerate table or database lookup by detecting duplicated records in a large file. An example is finding similar stretches in DNA sequences. They are also useful in cryptography. A cryptographic hash function allows one to easily verify that some input data maps to a given hash value, but if the input data is unknown, it is deliberately difficult to reconstruct it (or equivalent alternatives) by knowing the stored hash value. This is used for assuring integrity of transmitted data, and is the building block for HMACs, which provide message authentication.

Cryptography: the practice and study of techniques for secure communication in the presence of third parties (called adversaries).[2] More generally, it is about constructing and analyzing protocols that block adversaries;[3] various aspects in information security such as data confidentiality, data integrity, authentication, and non-repudiation[4] are central to modern cryptography. Modern cryptography exists at the intersection of the disciplines of mathematics, computer science, and electrical engineering. Applications of cryptography include ATM cards, computer passwords, and electronic commerce

Byzantine Generals' Problem: An agreement problem (described by Leslie Lamport, Robert Shostak and Marshall Pease in their 1982 paper, "The Byzantine Generals Problem") in which a group of generals, each commanding a portion of the Byzantine army, encircle a city. These generals wish to formulate a plan for attacking the city. In its simplest form, the generals must only decide whether to attack or retreat. Some generals may prefer to attack, while others prefer to retreat. The important thing is that every general agrees on a common decision, for a halfhearted attack by a few generals would become a rout and be worse than a coordinated attack or a coordinated retreat.

The problem is complicated by the presence of traitorous generals who may not only cast a vote for a suboptimal strategy, they may do so selectively. For instance, if nine generals are voting, four of whom support attacking while four others are in favor of retreat, the ninth general may send a vote of retreat to those generals in favor

of retreat, and a vote of attack to the rest. Those who received a retreat vote from the ninth general will retreat, while the rest will attack (which may not go well for the attackers). The problem is complicated further by the generals being physically separated and must send their votes via messengers who may fail to deliver votes or may forge false votes.

Byzantine fault tolerance can be achieved if the loyal (non-faulty) generals have a unanimous agreement on their strategy. Note that there can be a default vote value given to missing messages. For example, missing messages can be given the value ¡Null¿. Further, if the agreement is that the ¡Null¿ votes are in the majority, a pre-assigned default strategy can be used (e.g., retreat).

The typical mapping of this story on to computer systems is that the computers are the generals and their digital communication system links are the messengers.

Proof-of Work (POW) System: an economic measure to deter denial of service attacks and other service abuses such as spam on a network by requiring some work from the service requester, usually meaning processing time by a computer. The concept may have been first presented by Cynthia Dwork and Moni Naor in a 1993 journal.[1] The term "Proof of Work" or POW was first coined and formalized in a 1999 paper by Markus Jakobsson and Ari Juels.[2]

A key feature of these schemes is their asymmetry: the work must be moderately hard (but feasible) on the requester side but easy to check for the service provider. This idea is also known as a CPU cost function, client puzzle, computational puzzle or CPU pricing function. It is distinct from a CAPTCHA, which is intended for a human to solve quickly, rather than a computer.

Electronic funds transfer (EFT): the electronic transfer of money from one bank account to another, either within a single financial institution or across multiple institutions, through computer-based systems and without the direct intervention of bank staff. EFTs are known by a number of names. In the United States, they may be referred to as electronic checks or e-checks.

At the heart of bitcoin is a fundamental innovation: a distributed public ledger. A ledger in accounting is a book that you cannot edit once you have written in it. Instead, if you have made a mistake, the only way to fix it is to add another transaction to the ledger that undoes the error. As we know from accounting fraud, problems arise when people figure out ways to transact without recording it in the ledger or making ex post changes to the ledger (this is why Quickbooks isn't really an accounting system). The bitcoin ledger is the so-called blockchain which uses the fact that there are many copies of it that are broadly distributed combined with a fair bit of math to ensure that once a transaction has been recorded in the blockchain that transaction can not be changed after the fact. There is no other widely used

protocol in the world today that accomplishes this: with bitcoin anyone can make a statement (a transaction) and have this be recorded in a globally visible and fixed ledger.

Double-spending: a failure mode of digital cash schemes, when it is possible to spend a single digital token twice. Since, unlike physical token money such as coins, electronic files can be duplicated, and hence the act of spending a digital coin does not remove its data from the ownership of the original holder,[1] some other means are needed to prevent double-spending.

Mining: the process of adding transaction records to Bitcoin's public ledger of past transactions. This ledger of past transactions is called the block chain as it is a chain of blocks. The block chain serves to confirm transactions to the rest of the network as having taken place. Bitcoin nodes use the block chain to distinguish legitimate Bitcoin transactions from attempts to re-spend coins that have already been spent elsewhere.

Mining is intentionally designed to be resource-intensive and difficult so that the number of blocks found each day by miners remains steady. Individual blocks must contain a proof of work to be considered valid. This proof of work is verified by other Bitcoin nodes each time they receive a block. Bitcoin uses the hashcash proof-of-work function.

The primary purpose of mining is to allow Bitcoin nodes to reach a secure, tamper-resistant consensus. Mining is also the mechanism used to introduce Bitcoins into the system: Miners are paid any transaction fees as well as a "subsidy" of newly created coins. This both serves the purpose of disseminating new coins in a decentralized manner as well as motivating people to provide security for the system.

Bitcoin mining is so called because it resembles the mining of other commodities: it requires exertion and it slowly makes new currency available at a rate that resembles the rate at which commodities like gold are mined from the ground.

Genesis Block The first block of a block chain. Modern versions of Bitcoin assign it block number 0, though older versions gave it number 1. The genesis block is almost always hardcoded into the software. It is a special case in that it does not reference a previous block, and for Bitcoin and almost all of its derivatives, it produces an unspendable subsidy.

References

[1] International Energy Agency. Electricity information. *International Energy Agency*, 2012.

[2] Syed Taha Ali, Dylan Clarke, and Patrick McCorry. Bitcoin: Perils of an unregulated global p2p currency. *Technical Report SEries*, (CS-TR-1470), May 2015.

[3] Adam Back. Hashcash - a denial of service counter-measure. *Cypherspace*, August 2002.

[4] Simon Barber, Xavier Boyen, Elaine Shi, and Ersin Uzun. Bitter to better — how to make bitcoin a better currency. *Palo Alto Research Center, University of California, Berkeley*, 2012.

[5] Jorg Becker, Dominic Breuker, Tobias Heide, Justus Holler, Hans Peter Rauer, and Rainer Home. Can we afford integrity by proof-of-work? scenarios inspired by the bitcoin currency. *Security WEIS 2012*, Workshop on the Economics of Information Security, February 2012.

[6] Christian L Belady. In the data center, power and cooling costs more than the it equipment it supports. *Electronics cooling*, 2007.

[7] Dan Boneh, Xavier Boyen, and Hovav Shacham. Short group signatures. *Advances in Cryptology*, 3152:41–55, 2004.

[8] Terri Bradford and Fumiko Hayashi. Developments in interchange fees in the united states. *Federal Reserve Bank of Kansas City*, April 2008.

[9] Jay Cassano. What are smart contracts? cryptocurrency's killer app. *Fast Company*, 2014.

[10] Wei Dai. B-money. *Weidai.com*, 1998.

[11] Lodewijk Andrè de la Porte. The bitcoin transaction system. June 2012.

[12] Whitfield Diffie. The first ten years of public-key cryptography. *Proceedings of the IEEE*, 76(5), 1988.

[13] Bank for International Settlements. 81st annual board report. *Basel*, 2011.

[14] Pasquale Forte, Diego Romano, and Giovanni Schmid. Beyond bitcoin - part i: A critical look at blockchain-based systems. *Cryptology ePrint Archive*, 2015.

[15] Erich Hughes. A cypherpunk's manifesto. *Activism.net*, 1993.

[16] Edwin Jacobs. Bitcoin: A bit too far? *Journal of Internet Banking and Commerce*, 16(2), August 2011.

[17] Sarah Jeong. The bitcoin protocol as law, and the politics of a stateless currency. March 2013.

[18] Don Johnson and Alfred Menezes. The elliptic curve digital signature algorithm (ecdsa). *Certicom*, 2000.

[19] Luke Patrick Johnson, Nick Gogerty, Joseph Zitoli, and Ahmed Isam. Connecting the blockchain to the sun to save the planet. *SSRN*, December 2015.

[20] Dan Kaminsky. Some thoughts on bitcoins. *Presentation on slideshare*, 2011.

[21] Trevor I. Kiviat. Beyond bitcoin: Issues in regulating blockchain transactions. *Duke Law Journal*, 65(569-608), 2015.

[22] Ahmed Kosba, Andrew Miller, Elaine Shi, Zikai Wen, and Charalampos Papamanthou. Hawk: The blockchain model of cryptography and privacy-preserving smart contracts. *Cryptology ePrint Archive*, 675, 2015.

[23] Leslie Lamport, Robert Shostak, and Marshall Pease. The byzantine generals problem. *SRI International*, 1982.

[24] Robert Manne. The cypherpunk revolutionary: Robert manne on julian assange. *Cryptome*, March 2011.

[25] Timothy C. May. The crypto anarchist manifesto. *Activism.net*, 1994.

[26] Anoop MS. Elliptic curve cryptography: An implementation guide. *Infosecwriters*, 2007.

[27] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system. *N/A*, 2008.

[28] Proof of Existence Website. What is proof of existence? *Proofofexistence.com*, 2012.

[29] Federal Information Processing Standards Publication. Specifications for the secure hash standard. *FIPS Publication*, August 2002.

[30] B. Prashanth Reddy. A review of the bitcoin network security framework. 2010.

[31] Meni Rosenfeld. Analysis of hashrate-based double-spending. December 2012.

[32] Mag. Peter Šurda. The origin, classification, and utility of bitcoin. *SSRN*, May 2014.

[33] Stuart E. Weiner and Julian Wright. Interchange fees in various countries: Developments and determinants. *Federal Reserve Bank of Kansas City*, 2005.

[34] Wikipedia. Bitcoin. *Wikipedia*, 2015.

[35] Wikipedia. Cryptography. *Wikipedia*, 2015.

[36] Wikipedia. Cypherpunk. *Wikipedia*, 2015.

[37] Wikipedia. Digital signature. *Wikipedia*, 2015.

[38] Wikipedia. Elliptic curve cryptography. *Wikipedia*, 2015.

[39] Wikipedia. Pretty good privacy. *Wikipedia*, 2015.

[40] Wikipedia. Public key cryptography. *Wikipedia*, 2015.