The politics of cryptography: bitcoin and the ordering machines

Quinn DuPont
quinn.dupont@utoronto.ca
Faculty of Information, University of Toronto
Keywords: cryptography, bitcoin, computation, code, crypto-anarchism

Abstract

This paper explores the cryptographic aspects of bitcoin. Over two short primers (of advancing specificity) the cryptography in bitcoin is described and contextualized. I offer a description of a full bitcoin transaction to contextualize the terminology and concepts from the primers. I then offer a condensed version of a generic framework for conceptualizing cryptography, as an alternative to Shannon's Mathematical Theory of Communication which will shed light on the politics of cryptography. As a form of praxis I describe my experiences running a bitcoin mining machine. Finally, I critique and reimagine Gilles Deleuze's work on the control society, and suggest that cryptography functions as a "new weapon" in a logic of Order.

It was April 10th, 2013 and the price of a single bitcoin surged past 250 USD on the Mt. Gox exchange. A few months prior I had purchased seven bitcoins for just under $200, now nearing $2000 in value. But, just as fast as the market went up, it came down. I was panicked like an amateur gambler but lulled by my humming money machine, permuting cryptographic codes by the millions every second. The price of bitcoin was being pushed up by a number of factors: mainstream interest, a sustained distributed denial-of-service attack on Mt. Gox (the main bitcoin exchange), and people like me, gambling in the newest crypto-anarchist adventure.

Bitcoin is a fiat currency, like many others (electronic or not), and requires human belief to sustain value. Without government backing fiat currency needs another sustaining ideology—people need to believe that the currency will persist and retain value. Persistence and value are eroded by shaky ideology (e.g., a weak government) and technical flaws (e.g., fraud, counterfeiting, hyperinflation). Bitcoin ideology rests on the general view that cryptography is a panacea, an alchemic universal solvent, or (put crudely) magic fairy dust. The libertarian rhetoric surrounding bitcoin reflects this general ideology. Avoiding technical flaws in bitcoin is (thought to be) largely due to cryptography as well. Counterfeiting and double-spending is prevented as a result of public key cryptography, and hyperinflation is kept in check due to the ability to cryptographically ensure the measured production of money (with a maximum number of coins produced).

Cryptography is easily operationalized—best practices ensure secure information—but still poorly understood. According to a standard account cryptography ensures "information security" or its slightly older cousin, "information secrecy" (see e.g. Kohno et al., 2010). But, how does monetary value arise from information security or secrecy? Security and secrecy are usually understood in terms of social relations (c.f. Bellman, 1979). Modelled in its simplest formulation, a secret is some information that I possess and you do not, while information security might be described more abstractly, as control of information within a relationship. All fine and well, monetary value can surely arise from exclusivity, but what is the function of cryptography here? (Does the monetary value of bitcoin arise from exclusivity?) I argue that cryptography is both central to bitcoin and yet produces a non-secret set of powers for its social effect, which includes monetary value.

To date the most influential conceptualization of cryptography is Claude Shannon's (1945) linked notion of secrecy and information. Prior to his famous Mathematical Theory of Communication (MTC) (Shannon and Weaver, 1948) Shannon had been working on war-time encryption systems. Shannon's cryptographic work stemmed from a long line of influences and prior work (Nyquist, Hartley, Wiener, and so on) (Thomsen, 2009), as well a rich conceptual backdrop spanning from at least the Renaissance (Porta, Bacon, Wilkins, Leibniz and many others) (Cherry, 1953; Geoghegan, 2008). Although only occasionally recognized, the histories of information and cryptography are intimately tied (see e.g., Gleick, 2011). It is telling that Shannon developed his account of cryptography and then his account of information, just as Bacon twinned cryptography and logic, or Wilkins' cryptography and classification.

This cryptographic work set the stage for Shannon's more general, and more rigorous portrayal of information in the MTC. In contrast to much of the engineering work being done on information transmission at that time, Shannon focused on discrete rather than continuous signals (Thomsen, 2009). Drawing on Hartley, Shannon bracketed the issue of meaning, and discussed only *how much* information can pass through a channel. This conceptualization, combined with Nyquist's observation that information transmission obeys a logarithmic rule allowed Shannon to generalize the issue, and show

that information accords to physical properties about the world (modeled as entropy) (Aspray, 1985; Hayles, 1999).

It was in working out the coding issues for cryptography that Shannon developed his theory of information. Left somewhat implicit in Shannon's account, however, was the issue of secrecy (despite being in the title). Shannon argued that his account only pertains to "true" secrecy systems, for which the "meaning of the message is concealed by cipher, code, etc." (Shannon, 1949). Despite this slip about "meaning" Shannon had a clear sense of his MTC as early as 1945 (this sentence remains in the declassified 1949 version, although both versions are very explicit to reject any "psychological" account). Shannon defines cryptography abstractly as the transformation of information from one space to another. Secrecy arises from the "*a priori* probability associated with… choosing that [enciphering] key," which is a function of the statistics of "transformations of one space into another" (Shannon, 1945). In ideal situations secrecy becomes a matter of making guesses in the presence of a stochastic phenomenon.

As conceptualized by Shannon, the cryptography in bitcoin leaves open as many questions as it answers. Shannon's work is conceptually powerful (enabling massive engineering developments), but historically vacuous and philosophically incomplete. Before Shannon "won," the field was less settled (Cherry, 1953); competing conceptualizations existed but none were better prepared than Shannon's for the coming cybernetic and then informatic changes. By identifying Shannon's MTC as the endpoint of cryptographic conceptualization we risk teleological explanations, and make archaeologies of cryptographic mechanism very strange (for the thousands of years of cryptography before the introduction of mathematics—with the Arabs and then Leibniz—what was cryptography?). Even if we accept mathematics as a master logic, how do we account for the human, or even the machinic? And specific to bitcoin, how do these transformations occur? From where does value arise? Perhaps the most central of all, what is the politics of bitcoin itself?

This paper will first offer a largely non-technical account of bitcoin cryptography using accepted (engineering) terminology. Over two short primers (of advancing specificity) the cryptography in bitcoin will be described and contextualized. I offer a description of a full bitcoin transaction to contextualize the terminology and concepts from the primers. I then offer a condensed version of a generic framework for conceptualizing cryptography, as an alternative to Shannon's that will shed light on the politics of cryptography. I then return to my introductory story of praxis and describe my experiences running a bitcoin mining machine. Finally, I critique and reimagine Gilles Deleuze's work on the control society, suggesting that cryptography functions as a "new weapon" in a logic of Order.

Cryptography primer

Before I turn to the cryptography used in bitcoin it will be helpful to get some of the basic terminology and functionality of standard cryptography out of the way. This account, while conceptually incomplete, will start the process of translating engineering language into more philosophically salient terms.

The cryptography used in bitcoin is not unusual or exemplary, in fact, there are no cryptographic innovations in bitcoin (in computer security terms, this is a virtue of the system). Bitcoin uses a standard SHA-256 hashing algorithm. This hashing algorithm is put to some seemingly strange uses, but nothing unique to the history of cryptography since the development of public key cryptography in the 1970s.

Introductory primer

For the vast majority of the history of cryptography, its development from substitution ciphers and code systems to polyalphabetic and keyed algorithms, the encrypting mechanism was unitary. With the invention of public (or asymmetric-) key cryptography in the 1970s it became possible to create a system that "split" the cryptographic key. (Asymmetric-key cryptography was initially invented in 1973 at the Government Communication Headquarters in the UK by James Ellis, Clifford Cocks, and Malcom Williamson but kept secret; it was then publically re-invented in 1976 by Whitfield Diffie and Martin Hellman.) Splitting the cryptographic key ushered in new uses for cryptography, and was well-timed for the coming advance of the Internet and popularization of point-to-point electronic communication.

In this short introductory primer I will describe three related sets of technology: symmetric-key ("traditional" or "classical") cryptography, asymmetric (public) key cryptography, and hashing algorithms (an offshoot of asymmetric-key cryptography, but because it features centrally in bitcoin I will spend some extra time on its exegesis). It should be noted that this primer accepts the problematic distinctions of contemporary cryptography (and there are many, although usually unacknowledged), as well as the accepted technical nomenclature. In the end, this article will go some way to implicitly problematize these accepted divisions, although I acknowledge that an extended critique is still wanting.

What all forms of cryptography share is the syntactic transformation of symbols. While this conceptualization is generally accepted by the cryptographic community, I will later complicate this definition by offering a different conceptualization based on the same basic principles but offering a much needed semantic aspect. In simple "code" systems (often called a "nomenclator") a letter, word, or entire phrase may be replaced with an alternative (the substitution presumably kept private between the two communicating parties) (Kahn, 1967). The basic principle is captured by so-called substitution ciphers, which exchange one letter for another in a deterministic manner. Implied is a simple but profound characteristic of all cryptography—that cryptography *requires discrete symbols ordered* in various ways. Later, we'll see that this fact is philosophically pertinent and the crux of the politics of bitcoin.

From this mechanism more complicated forms of cryptography were invented. Polyalphabetic cryptography uses multiple alphabets for the substitution (or what might be called transposition within a larger combinatoric space), sometimes jumping from one alphabet to another according to an agreed-upon secret "key." In more recent usage the key became analytically separated from the cryptographic algorithm as a result of "industrial" uses of cryptography that require reusable ciphers. When cryptography shifted from a craft to a mechanized application the algorithm became fixed, and the key became the mutable, secret part.

Cryptographic keys (or "passwords") are often misunderstood as being necessary and primary to cryptography. In fact, the key is no more central than the set of transformations (and should be characterized as part of the system of transformation). For much of the history of cryptography no notion of cryptographic "key" existed. For purposes of cryptanalysis ("codebreaking") a key is only as valuable as knowledge of the corresponding mechanism or set of transformations. Yet, in recent years cryptographic ideology has come to espouse the belief that the best practice is to keep the key secret and the mechanism public (working on the assumption that the mechanism will eventually be discovered and reverse-engineered).

Symmetric-key cryptography employs the same key for encryption and decryption, so the shared secret item is identical among parties. The obvious downside is that to maintain secret communications all parties must ensure that the shared key is kept private from any so-called adversaries. Modern forms of symmetric-key cryptography work on digital bits, either encrypting the bits one at a time (or, more realistically, encrypting byte by byte or in prescribed bit-length "words"), or grouping the bits into blocks (and adding padding as needed so that each block includes the same number of bits). The former is known as a stream cipher, the latter a block cipher. Other non-cryptographic features may be present in a modern cryptographic system, such as error detection, compression, and so on.

The symbolic transformations in symmetric-key cryptography are fundamentally the same as that of asymmetric-key cryptography—in fact, symmetric-key primitives can be used to build asymmetric-key systems. The sole (but critically important) difference between symmetric-key and asymmetric-key cryptography is that rather than sharing a single (unitary) secret key, asymmetric-key cryptography uses a binary key, in which the two parts are linked and both are required to decrypt and encrypt. By splitting the key into two linked parts one part can be kept secret, while the other is made public (the parts are typically linked mathematically—e.g., prime factorization or calculating the discrete logarithm—but any suitable mechanism could be used). The private key should not be easily deducible from the public key, yet the public key should be easily deducible from the private key (using so-called mathematical trap-door functions). A trap-door function such as exponentiation modulo is based on current mathematical knowledge that it is easy to calculate the remainder when a number is raised to some power, divided by another (the modulus), yet, if given all the information other than the exponent, it is very difficult to solve for the exponent (i.e., it is slow to compute the discrete logarithm). Only when the secret key is possessed is it easy to open the trap door, otherwise the calculation is slow (but certainly tractable).

Such a setup offers several interesting possibilities. The key used to encrypt a message is not the same key used to decrypt it, so therefore I may encrypt a message with your public key, in which only you can decrypt (using your linked private key). Configured this way, I can send you a message that only you can read (akin to placing a piece of mail in a publicly-accessible but locked mailbox), and you can do the same for me by using my public key, ensuring confidentiality. (More complicated three-pass schemes are possible too, in which I encrypt a message and pass it to you, then you encrypt the already-encrypted message and pass it back, I then decrypt my encryption and pass it back to you, at which point you can finally decrypt your encryption and read the message—successfully transferred in public while remaining encrypted.) Similarly, if I encrypt a message with my private key and send you both my message and the encrypted message (or a "digest" of it), your ability to decrypt the encrypted message with my public key ensures that the message is authentic (non-repudiated, or, guaranteed to be mine). These features work to permit secret message communication without ever requiring a secret communication channel, or to ensure that a message has not been changed. In a world where the communication channel is necessarily open to eavesdropping asymmetrical-key cryptography performs a kind of magic trick: secret messages over public channels without ever requiring the prior transmission of secret information.

A related application of asymmetric-key cryptography is the hash function. A hash function is a set of permutations that transform some data into a fixed-sized output (a digest), which changes considerably given even a slightly different input datum. When a hash function uses cryptographic mechanisms to create the digest it can be used to ensure the message is not repudiated (its integrity can be verified), which is especially useful for creating compact, easily transmittable "fingerprints" of data. Similarly, by

computing a hash for a password the hash may be stored in place of the secret password, and authenticated against the digest representation instead of the actual password (allowing the digest to be captured by an adversary without losing secrecy).

An aside: it should be noted that because asymmetric-key cryptography relies on a *ratio* of effort for trapdoor or one-way functions—that it is relatively harder to reverse the function than it is to compute it—given enough time and effort asymmetric-key cryptography is quite susceptible to cracking. Unlike symmetric-key cryptography, in which it is possible (although unwieldy) to create mathematically-guaranteed secrecy, asymmetric-key cryptography can at best make it much more difficult to reverse the cryptographic function than it is to perform the encrypting function. The ratio of cryptanalyse-to-encryption may be 1000/1 or 1,000,000/1, but with enough time all asymmetric-key cryptographic functions succumb to cracking. Due to this known weakness, digests of passwords are typically also protected by non-cryptographic access mechanisms. If cryptographic digests are known to be exposed to the public it is typically only a matter of time until they will be cracked, so good security practice is to require new passwords and corresponding digests for compromised data stores. Given the advent of GPU cracking (and soon, powerful repurposed bitcoin mining machines) hashed passwords are cryptanalysed rapidly, even as much as 60-90% of a corpus within an hour, according to a recent cracking "game" conducted by technology website Ars Technica (Goodin, 2013).

The hash algorithm used in bitcoin is SHA-256, a protocol for hashing in the Secure Hash Algorithm 2 family that outputs 256 bit digests. SHA-256 is composed of a simple set of logic transformations configured to perform the necessary mathematical trap-door function required by the asymmetric-key cryptography. Thus, described in terms of levels of abstraction from the basics of digital computation to a full ASCII-encoded hash digest, the SHA-256 hash algorithm operates as such: electromagnetic flux is discretized on a clock-cycle; bits are then transformed using logical operations performed by transistors; the binary representation is collected into 64 bit "words" that function as high-level data structures (integers); mathematical trap-door/one-way functions are constructed from the set of transformations (+, and, or, xor, shr, rot), with the entire algorithmic structure corresponding to the Merkle–Damgård padding and compression scheme; the resulting 256 bit hash digest may be encoded into a 64 ASCII-encoded character string for portability and human-readability. In sum, SHA-256 is the process of successively interpreting electric-magnetic flux as a series of different ordering mechanisms or techniques. The primary ordering is the first: from flux to binary, and once this discretization is accepted as real (by fiat) the ordering techniques are limited only by human imagination.

Advanced primer

The fundamental cryptographic algorithm used in bitcoin is SHA-256, however, the conceptual utility of this hashing algorithm as used in bitcoin draws on a recent history of previous academic and practical developments. Of the numerous developments, the most significant and relevant are: Ralph Merkle's hash-trees (patent filed in 1979), David Chaum's blind signatures (1982), Adam Back's hashcash proof of work system (1997), Wei Dai's b-money scheme (1998), Nick Szabo's bit gold concept (1999), and Hal Finney's reusable proofs of work (2004).

In addition to having a hand in the invention of public key cryptography, Ralph Merkle developed a system for efficiently verifying large data structures through a tree structure of hash digests (Merkle, 1982). As described above, a hash digest can be used to verify the non-reputability of a datum, but for

large data structures it would be extremely time-consuming to perform a hash function on every datum. Merkle realized that by organizing hash digests into a tree structure (where each node is a hash digest) it is possible to compute the hash digest for only the top-most node (while authenticating the left and right nodes) rather than every node, to ensure non-reputability. Hash trees are commonly used to ensure data integrity, but when used with cryptographic hash functions every prior message is checked for authenticity (none of the messages can be faked).

Blind signatures are another result of public key cryptography being used in unexpected ways. In Chaum's original conceptualization for blinded signatures, payment systems with the anonymity of cash but the security of digital money were the intended target (Chaum, 1982). By using public key cryptography Chaum proposed a system that ensured 1) the inability of third parties to determine information about the payee, 2) the ability of individuals to provide proof of payment, and 3) the ability to stop payments when needed (in cases of theft). Chaum envisioned a digital equivalent of paper envelopes lined with carbon paper. By writing a signature on the outside of the envelope a second "blind" signature is duplicated on the inside. In Chaum's example of authenticated secret voting, the blinded signature is sent to the elector, removed from the envelope, signed by the elector, and mailed back to the voter in a new envelope (thus only the elector views the signature). If a voting dispute arises the signatures can be authenticated against the signatures on the envelopes, but each vote remains anonymous.

While the mutability of binary digits is useful for much computing, a system of ecash requires the opposite quality: money needs to be made solid, slow, and non-replicable. Originally proposed and developed by Adam Back (1997) for limiting email spam, hashcash uses two facts of public key cryptography: non-reputability of hash digests, and the computational difficulty of finding a hash "collision." Due to the fact that it is impossible to predict the outcome of a hash function on an arbitrary input (with current knowledge of the mathematical underpinnings of asymmetric-key cryptography used in hash functions), but easy to verify the results, a challenge-response mechanism can be created to require "work" (computational effort). By arbitrarily requiring a specified output of a hash function (say, the first 20 or more bits of the hash digest must be zeros), the sender can establish a "difficulty" threshold. The only way to find a hash digest with a specified output is to compute the hash of a different input value over and over until the result meets the necessary difficulty, and since the result can be verified easily the receiver of the hash function does not need to repeat the computational work to verify that the sender expended a set amount of work. For its original purpose of limiting email spam, the requirement to perform work when sending email would make sending email slow and computationally expensive, thus, sending a single email would result in a modest slow-down, but sending millions would become nearly impossible (or at least would require many expensive computers). In such an email system any email that was sent without corresponding evidence of computational work would not pass verification by the receiver (which is a quick calculation, in comparison to performing the original hash calculation), and would be discarded.

As part of the discussion of applications of Back's hashcash proposal on the Cypherpunks mailing list Wei Dai proposed a system of currency generation using the mechanism of hashcash (Dai, 1998). (At this point the anarchist/libertarian undercurrents are completely at the fore, Dai starts his proposal, "I am fascinated by Tim May's crypto-anarchy"). Dai applies Back's hashcash mechanism to establish an anarchist world in which cryptography functions for the "medium of exchange" and as a way to "enforce contracts," without the interventions of a government (Dai, 1998). The creation of bmoney is

accomplished by requiring a specified amount of computational work (which anyone can perform) and when verified by the community a collective ledger book is updated, awarding the worker the specified funds. In the bmoney proposal exchange of funds is accomplished by collective bookkeeping (authenticated with cryptographic hashes), and contracts are enforced through the broadcast and signing of transactions with digital signatures (i.e., public key cryptography).

Hal Finney (2004) extended the bmoney and hashcash proposals by suggesting a formalization of the proof of work mechanism, a scheme that permits the reuse and exchange of proof of work tokens (hash digests). With these extensions it became possible for Nick Szabo (1997) to conceive a system that accurately calculates the "difficulty" of proof of work for the purpose of money generation, and to allow the generated money (hash digests) to be exchanged and reused.

How bitcoin functions

When the pseudonymous programmer Satoshi Nakamoto proposed bitcoin in 2008 it built on the crypto-anarchist developments from the following two decades (Nakamoto, 2008). In terms of invention, bitcoin introduced a modest change to the bmoney, bit money (and other) proposals already in existence. Rather than require a single collective ledger of transactions, or awkwardly share the ledger among parties, Nakamoto suggested that a "blockchain" contain all transactions (including generated money by "miners" performing cryptographic proofs of work). The blockchain is a Merkle hash tree of transactions. For each transaction the mining servers verify the hash digests that result from transactions (incentivised to perform computational work by being awarded money for successfully performing proofs of work). The transactions are verified by the miners and grouped into "blocks"; once the top node of each block is verified a specified amount of work is required to seal the block and win the resulting money.

A full round of a transaction is described in the following. Money is stored in a wallet, which is a unique hash digest generated by each user (and any number of wallets are possible). To send you money I first digitally sign the transaction request with my private key (that is, I perform asymmetric-key encryption on the transaction request data). Using my public key, the network can verify my transaction request. The transaction request is sent (via peer-to-peer communication protocols) to the network and then bundled with others into blocks every ten minutes. Each block includes a hash digest of the previous blocks (arranged in a hash tree), a hash digest of the current block, and a "nonce". A nonce is added input that (when hashed) results in a radically different output. Only when an output value meets a certainly "difficulty" (verification of proofs of work) will a block be considered authenticated (the difficulty is specified by requiring at least *n* leading zeros in the hash digest output, set by the protocol to regulate the speed of block generation). As each subsequent block is verified the previous blocks fall further down the hash-tree, with the newer hashes contingent on the previous hash digests' value. In this way any fraudulent changes to the blockchain are instantly discovered (and rejected).

It is improbable to create fraudulent transactions because as time goes on, and block upon block is verified by the miners, fraudulent transactions would require changing every subsequent block, at a rate greater than the (legitimate) network of miners. A fraudulent block would only be accepted if the alternative blockchain was longer, which would require performing more proofs of work than the legitimate network. (Of course, this is the ideal scenario. Any competing network of greater computational power could best the legitimate one, and therefore human vagaries of collusion and

consolidation come to play. This has actually happened, when due to a technical bug the blockchain became "forked," and was only reset when a cabala of powerful mining pools colluded to switch to the "corrected" blockchain.)

Conceptualizing cryptography

Do the existing conceptualizations of cryptography make sense for bitcoin? Bitcoin transactions can be modelled as a Shannon communication channel, but what role does secrecy play? It has been alleged (by crypto-anarchist proponents of bitcoin) that bitcoin is anonymous and privacy-ensuring, yet nothing further could be true! The fundamental innovation in bitcoin as compared to the prior hashcash, bit gold, and bmoney schemes is the blockchain. The blockchain is necessarily public and amounts to a bookkeeper's dream, recording *every* transaction. Once a "real" identity is linked to a hash digest every transaction is recorded and publically visible. If there is any secrecy or privacy it results from the social fact that I may choose to keep my hash digest private (that is, not associate any personal information with the hash digest). Even this breaks down in practice, as any "adversary" of whom I engage with can expose identifying information about a particular hash signature. For example, a much-lauded example of bitcoin use is for buying mail-order drugs. If the drug-seller released his or her mailing database it would be trivial to connect my hash digest with my mailing address. Similarly, if I used the same wallet to perform innocuous transactions (perhaps filling it with funds converted from other currencies), and then went on to purchase mail-order drugs, *any* release of my identity associated with that hash digest would reveal *all* of my transactions.

Even if bitcoin was configured in such a way that it did preserve privacy, perhaps employing a blind signature mechanism yet still managing to guard against double-spending, would "information secrecy" still be the correct way to conceptualize how cryptography is being used? I argue that it would not, and that bitcoin is not the only use of cryptography that cannot be cashed out in terms of information secrecy. The history of cryptography is much longer and richer than usually thought and intersects with many domains, including the development of scientific practice, modern classification theory, linguistic and semiotic developments, hermetic and cabalalistic practice, and discrete mathematics. For reasons of brevity this history cannot be told here—although it very much still needs to be done—but these various uses highlight the fact that cryptography has long been used for non-secret purposes.

Instead of focusing on a singular use of cryptography, I introduce a framework in which secrecy becomes only one of the possible (unlimited) uses of cryptography. The framework conceptualizes cryptography in terms of uses or powers (e.g., secrecy, hermeneutical exploration of the natural world, philosophical languages), and analytical requirements. So long as a proposed use of cryptography corresponds to the analytical requirements it can be legitimately called cryptography. Thus, cryptography is broad enough to include bitcoin (economics), but also for example, Jim Sanborn's cryptographic art *Kryptos* (aesthetics), or John Wilkin's (1668) attempt to develop a philosophical language (logic/classification/method).

Within the framework (here only briefly described), cryptography is a derivative notational system sustained by second-order metaphor. These requirements rely on Nelson Goodman's (1976) careful articulation of a notational system, but rather than a notational system with reference to the real, full-blooded world, cryptography works only from mark to mark (a mark can be an utterance, textual inscription, etc.). Goodman argues for a set of syntactic and semantic requirements for a notational

system that (to speak precisely) prescribes a "digital" system. The discreteness requirements of Goodman's notational system ensure that no ambiguity can enter the system: notational symbols univocally refer to their referents. These requirements are (roughly) unambiguity, disjointedness, differentiation. For Goodman, many encoding schemes potentially qualify as notational, such as binary, Morse code, musical notation, dance notation, and so on (it is important to note that these notations only count as proper notational systems on Goodman's terms if the semantic requirements are also met, which is not typically how these systems are understood as operating).

By adapting Goodman's notational system to apply only to marks (a derivative system) the syntactic and semantic requirements obtain from mark to mark. While the syntactic requirements are relatively unproblematic, the semantic requirements are unusual and require further elucidation. I argue that the semantic properties are sustained through a second-order functioning of metaphor. While the non-cryptographic sense of metaphor employed in typical signification (of which I have no account for, preferring a pluralism) is complicated, rich, and profoundly productive in human language, this second-order metaphor is comparatively weak and narrow. Borges (2000) offers a helpful comparison by invoking "mere shuffling" as a weak form of metaphor,

> *Any metaphor, as beguiling as it may be, is a possible experience, and the difficulty lies not in its invention (a simple thing, attained by the mere shuffling of fancy words) but in achieving it in a way that astonishes its reader.*

A second-order metaphor between of mark to mark is the (semiotic) substitution of one mark for another, maintained in some kind of memory (perhaps held in neural memory, inscribed on paper, or stored in a digital computer). For example, the second-order metaphor of the mark "A" for "D" is legitimately an encryption of "A" (so long as the rest of the notational system maintains the Goodmanian properties of unambiguity, disjointedness, and differentiation).

With these requirements any number of notational systems may count as cryptography (binary, compression algorithms, source code compilers, and so on), and yet they may have very different uses. I acknowledge that this broad account results in the (possibly) odd suggestion that "code" may become cryptographic. I see this to be a virtue of the account. For example, the difficulty of cryptanalysing a compressed file is often equal to cryptanalysing "proper" cryptographic systems. Likewise, we often equivocate on the plurality of meanings of "code". Yet, cryptography and code do not always collapse. As Lacan rightly notes, language is not code (quoted in Hayles, 1999), but to the extent that code parasitically refers beyond its self (like language), then it too is not cryptography. Only once the initial semiosis of signified to sign occurs, with the necessary violence of abstraction, is the notational system potentially in place and the new writing becomes cryptographic.

With this framework in place we are finally freed to interpret cryptographic systems on their own terms (not just their efficacy of "secrecy"), and to evaluate the effects accordingly.

Bitcoin praxis

Before moving on to a description of bitcoin's effects and uses in light of this framework I will offer an example of bitcoin praxis. In many ways the bitcoin miner is at the heart of bitcoin cryptography, since it creates money and verifies transactions. Starting in 2013, I engaged in the practical operation of a bitcoin miner. What follows is a description of this bitcoin praxis.

As described above, the bitcoin algorithm uses the SHA-256 method of computing hash digests. While any computing mechanism could in theory calculate a SHA-256 hash, there are certain reasons why conventional Central Processing Unit (CPU) mining is now rarely used. With even the fastest modern CPUs running software designed to take advantage of multithreaded computation the number of hashes computed per second is low compared to other technologies, and because the ability to "win" the awarded money for a successfully verified block is in competition with other miners, an arms race is always at hand.

CPUs are usually designed to manage and switch computational tasks, and take care of a variety of sub-processes, which makes a CPU ideal for general computing but inefficient for performing the same type of simple calculation repeatedly. Commercially-available Graphics Processing Units (GPUs) are designed to be relatively free from the management of resources and thus (when appropriately programmed using low-level software) are able to perform repeated calculations much faster than CPUs. Additionally, GPUs are designed to work in parallel, so while a multi-core, multithreaded CPU may be able to perform a certain amount of work in parallel, a modern GPU can perform thousands of computations in parallel. Some GPUs are ideally suited to perform SHA-256 calculations because they have been designed to perform XOR logic in a single step, rather than the two steps (or cycles) needed for other devices (the SHA-256 algorithm relies extensively on XOR transformations). For real-world comparison, on a slightly aging (2008) Mac Pro computer I was able to perform roughly 30 Kh/s (kilo hashes per second, or thousands of hashes per second) using the CPU (a server-grade 3GHz quad core Intel Xeon processor). When I installed a modern mid-level gaming video card (AMD Radeon HD 5850) with a dedicated GPU the same machine was able to perform roughly 350 Mh/s (millions of hashes per second), using only the GPU for calculating the hashes.

While 350 Mh/s may seem like considerable computational power—and it is, especially for the corollary purpose of password cracking—newer technologies have all but obsoleted GPU bitcoin mining. For the last several years more dedicated bitcoin mining individuals have purchased Field Programmable Gate Array (FPGA) devices that are tailored to perform these sorts of computational tasks, doing so much more quickly and with less power consumption. By the end of 2012 the newest type of bitcoin mining device entered the commercial market, eclipsing even FPGA devices in terms of speed and power efficiency. These Application-Specific Integrated Circuit (ASIC) devices are custom-designed for bitcoin mining and thus do so with remarkable speed and power efficiency. As of 2013 there are commercially available bitcoin miners available for $150USD that perform 5 Gh/s (billions of hashes per second) and use only 30 watts of power (compared to an average video card's consumption of 100-150 watts), with more expensive versions performing hundreds or even thousands of Gh/s.

Efficient bitcoin mining is only possible using specially-built software that is tailored to take advantage of built-in hardware capabilities. On GPUs the programming language used to write the portion of software that performs the hash calculations is typically Compute Unified Device Architecture (CUDA) or Open Compute Language (OpenCL), whereas the high-level software that controls the input/output, networking, and display of graphical user interfaces can be written in any suitable programming language. Contemporary cryptography algorithms are highly repetitive, requiring round after round of simple logic transformations, just like digital signal processing, big data and science computing, and gaming (computing millions of polygons). For this reason, cryptography shares many of the technologies and advances with these computing fields.

Result

That the economy is a "slave to the algorithm" is nothing new (Slater, 2013); dominant capital is now almost exclusively run through digital trading software, and much of the developed world's "cash" passes digitally direct from bank to merchant through debit or credit machines at point-of-sale terminals. Cash money already seems quaint: the stuff of slightly unscrupulous transactions (a manual trade exchanged for tax-free cash payment), or downright illegal transactions (purchasing drugs). Cryptography is sought as the way to modernize these old practices. So strong is the belief in cryptography that a popular rallying cry on (unconsciously libertarian) software developer forums such as Hacker News is "encrypt everything." Julian Assange reports that "the universe believes in cryptography" (Assange et al., 2012)—so it is only natural that our old fashioned human practices eventually catch up.

What the anarchist hype does tell us, however, is that bitcoin is an ideal site for the investigation of a politics of cryptography. Far from being simply "neutral", bitcoin technology reflects a politics in its history, construction, and use. There are a number of potential ways to understand the politics of cryptography: as a social phenomenon new articulations to the flow of capital become visible, legal structures change and ossify, and so on (Castells, 1996; Lessig, 2006). Accepting the lessons from the last few decades of STS scholarship a more fruitful way might be to attempt to let the technology speak on its own terms. Established models in this realm include Winner's "Do artifacts have politics?" (1980) and most literature informed by Heidegger (Feenberg, 1999; Introna, 2009). Latour's generally-accepted co-constitution of technology (Latour and Woolgar, 1986; Latour, 1999, 1993) brings us one step closer to a machinic description. Finally, the new materialists, Harman's Object Oriented Ontology (2009), Bennet's *Vibrant Matter* (2010), and Bogost's *Alien Phenomenology* (2012), offer radical reformulations of the politics inherit in technologies.

While drawing on these authors I take my point of departure with Gilles Deleuze's short "Postscript on the societies of control" (1992). In this work Deleuze highlights the fact that Foucault's genealogical models are transient, and suggests that our contemporary model (adopted from Burroughs and Virilio) is characterized by control. Deleuze charges his readers not to "fear or hope [for]" these mechanisms but instead to "look for new weapons" (Deleuze, 1992, p. 4). I argue that cryptography is one such new weapon in the control society.

Deleuze summarizes Foucault's periodization of history by reflecting on the "transience of the model" or what I might call the contingent nature of history. In *Discipline and Punish* (1979) Foucault argued that the sovereign society was replaced by the disciplinary one, a transition occurring roughly at the dawn of the eighteenth society. By pressing the lessons from *Order of Things* (2002) into the periodization of *Discipline and Punish* I see that the sovereign society operated through a logic of Similitude, and the disciplinary society operated through a logic of Order. According to Deleuze, the next shift occurs at the "outset of the twentieth" century, towards a control society (Deleuze, 1992, p. 3). The mechanisms of the disciplinary society were gradually instituted, then accelerated through World War II which caused the "environments of enclosure" to suffer a "crisis" (Deleuze, 1992, p. 3).

Deleuze argues that the control society is characterized as a modulation, rather than the "molds, distinct castings" of the disciplinary society (Deleuze, 1992). Each society is characterized by a kind of language: the disciplinary society is *analogical* and the control society is *numerical*. That we have entered a society of control is a brilliant (and I think correct) insight, but Deleuze seems to have erred by arguing for a

control society characterized by numerical, or modulating logic. I do not take this to be a point of interpretive error; Deleuze's characterization is doubly clear. In the English translation "modulation" evokes the Latin *modus*, meaning measurement, accomplished by numbers ("numerical"). Foucault spends considerable effort in *Order of Things* showing how measurement and number are a central part of the logic of the classical era, summed up in the term "order". If the control society can be cashed out in terms of order, what has changed?

Foucault's examples in *Order of Things* tell a compelling story of order: several fields of study develop in this period ("general grammar, natural history, and the analysis of wealth"), all which require a *mathesis* ("a universal science of measurement and order"), but none use the explicit techniques of mathematics (the "algebraic method"), nor mechanism (Foucault, 2002, p. 63). Foucault summarizes the relationships as such, "the analysis of wealth is to political economy what general grammar is to philology and what natural history is to biology" (Foucault, 2002, p. 182). According to Foucault, the disciplinary society is as numerical and modulating as ever, it is only the prior sovereign society that is analogical in the sense of having a logic of similitude.

Perhaps "we have never been modern" (Latour, 1993), and that there is no discontinuity between the disciplinary society and the control society (or perhaps the control society does not exist). Part of the challenge may be Foucault's examples, items so new in terms of social evolution that we are still working through the implications. The history of cryptography tells us that while cryptography is a general writing system—as old as writing itself—its shifting structure maps well on to Foucault's characterization in *Order of Things*: from Similitude to Order. Renaissance cryptography was, on my interpretation, necessarily a mechanism of discretization and order, but was *used* for purposes of similitude. Even as late as Athanasius Kircher (often thought as a man out of his time) cryptography functioned as simulacra, as a media instrument (Stolzenberg, 2001; Wilding, 2001; Zielinski, 2008). The cryptographic media instruments paralleled developments in classification and linguistics (the "Universal" or "Philosophical" language movement), which Foucault (circuitously) dealt with in *Order of Things*. As we move past Renaissance history of cryptography Order takes over, discretizing, shifting, arranging, and creating identities (Cheney-Lippold, 2011).

If Deleuze is correct that we are now in a control society, but mistakenly characterized it with a disciplinary logic of modulation and numerical control, what is the new logic of control? As contemporary designers and graphic artists might say, what is the "authentically digital" in this new control society? I argue that a "new weapon" of the control society—ubiquitous cryptography—functions on a logic of order. Cryptography is an ordering machine: it orders each successive realm it captures.

Cryptography functioned as an ordering machine in the disciplinary society too, but it was a technology genuinely ahead of its time. With mechanization, and ultimately, digital electrification cryptography shifted from occasional military application to ubiquitous commercial application. Cryptography now functions infrastructurally and sits invisibly behind most of the world's communications (machinic "data" transmissions such as encrypted Netflix TV streams or Trusted Computing modules, but also human correspondences such as email communications). As bitcoin and other electronic cash systems become prevalent, cryptographic ordering will also become more entrenched in the economic realm (it already functions invisibly at electronic point-of-sale machines, automated bank tellers, financial trading, and so on). For bitcoin, the specific ordering is the logic of SHA-256 described above: discrete symbols arranged

through a collection of logical transformations, built block by block of irreversible containers with strong identity parameters (necessarily discrete and unambiguous). The hash digests ("fingerprints") are then organized into a tree structure, recalling Deleuze and Guattari's famous charge, "we're tired of trees" (1987, p. 15). Finally, the hash tree is sent through peer-to-peer networks, succumbing to a logic of collusion and virality.

I will now return to the questions asked at the outset of this paper, how do cryptographic transformations occur, where does value arise from, and what is the politics of bitcoin itself? As part of an engineering narrative I offered a description of SHA-256 hash functions. The transformations from "one space to another" that Shannon articulated contain less mathematical idealism than previously hoped. On my account SHA-256 is a very concrete transformation of marks to marks, assembled in complex ways. The mathematics of public-key cryptography are still very much part of the story of SHA-256, but end up being a contingent factor, not constitutive of cryptography itself. Even the shuffling of bits in logic tables at the core of the SHA-256 algorithm require human (semiotic) sustenance. In a world without humans there may still be logic and mathematics, but there cannot be cryptography.

The value of bitcoin arises from its politics, and politics from its value. Ordering mechanisms are some of the most potent technologies we have. As discovered in Fordism, but also contemporary practices of intermodal containerized shipping and logistics the ability to control otherwise continuous phenomena is vastly valuable. If the pre-cryptocurrency metaphor for dominant capital was frictionless "flow" (like water), the new metaphor should be the rapid growth of crystalline structures. Cash flows, but bitcoin gets containerized and moved with even greater efficiency.

I have not offered a "position" on cryptography, for these answers are still unsettled. In the early days of modern computing Norbert Wiener worried about human autonomy becoming subject to machinic ordering (Hayles, 1999; Wiener, 1965). The idea of becoming subject to our semiotic technologies is built right into our language about their functioning. Code is powerful because it represents, that is, it both "re-presents" or makes something present again, and "stands for" or "substitutes" (Prendergast, 2000). Famously, Rousseau worried about the consequences of political representation, fearing any dictatorial relationship where we permit others to "stand in" for us. Similarly, Heidegger calls representation the master category of modern thought because it forces the division of subject and object (Heidegger, 2002; Prendergast, 2000).

More concretely, our cryptographic technologies are at once Privacy Enhancing Technologies (PETs) and also weapons in the commonplace cyberwars amongst developed nations. Even prosaic questions are ambiguous. Does Google increase my privacy by encrypting my Gmail communications, now open to only the machinic display of advertising? Am I better off by having a cryptographically-secure boot sequence for my computer (and thus preventing the installation of "competitor" operating systems, like Linux)? Is it a "feature" that bitcoin transactions are cryptographically irreversible? In this new control society there is no second-guessing your economic decisions, and no need to involve messy legal and political authorities, since code has become law in frighteningly efficient ways (Lessig, 2006).

This paper responds to Deleuze's call to "look for new weapons." I showed how the traditional conceptualization of cryptography is wanting, which has deeply coloured beliefs about the functioning and effects of bitcoin. I argued that cryptography is not necessarily about information secrecy (obviating anarchist hopes), and in particular that this conceptualization is unhelpful for understanding bitcoin. In its place, a broader conceptualization took its place, built upon a generic framework for understanding

cryptography. This framework suggests that cryptography is indeed politically ambiguous, amounting to "just" a machinic ordering. Critiquing and reimaging Deleuze's notion of a control society I suggested that cryptography is a powerful "new weapon," functioning as a mechanism of order. As an eminently cryptographic system, bitcoin is at the forefront of advancing this ordering logic to the economic realm.

References

Aspray, W.F., 1985. The Scientific Conceptualization of Information: A Survey. Annals of the History of Computing 7, 117–140.

Assange, J., Appelbaum, J., Müller-Maguhn, A., Zimmermann, J., 2012. Cypherpunks. OR Books, New York.

Back, A., 1997. hash cash postage implementation. Cypherpunks.

Bellman, B.L., 1979. The Paradox of Secrecy. Human Studies 4, 1–24.

Bennett, J., 2010. Vibrant Matter: A Political Ecology of Things. Duke University Press, Durham N.C.

Bogost, I., 2012. Alien Phenomenology, or, What It's Like to Be a Thing. University of Minnesota Press, Minneapolis.

Borges, J.L., 2000. Borges: Selected Non-Fictions. Penguin, New York.

Castells, M., 1996. The Space of Flows, in: The Rise of the Network Society. Blackwell Publishing, Oxford, MA, pp. 376–428.

Chaum, D., 1982. Blind Signatures for Untraceable Payments, in: Rivest, R.L., Chaum, D., Sherman, A.T. (Eds.), Presented at the Advances in Cryptology Proceedings of Crypto 82, Plenum, pp. 199–203.

Cheney-Lippold, J., 2011. A New Algorithmic Identity. Theory, Culture & Society 28, 164 –181.

Cherry, E., 1953. A history of the theory of information. Information Theory, IEEE Transactions on 1, 22–43.

Dai, W., 1998. PipeNet 1.1 and b-money. Cypherpunks.

Deleuze, G., 1992. Postscript on the Societies of Control. October 59, 3–7.

Deleuze, G., Guattari, F., 1987. A Thousand Plateaus: Capitalism and Schizophrenia. University of Minnesota Press.

Feenberg, A., 1999. Questioning Technology. Routledge, New York.

Finney, H., 2004. RPOW - Reusable Proofs of Work. Cypherpunks.

Foucault, M., 1979. Discipline and Punish: The Birth of the Prison. Vintage Books, New York.

Foucault, M., 2002. The Order of Things: An Archaeology of the Human Sciences. Routledge, New York.

Geoghegan, B.D., 2008. Historiographic Conceptualization of Information: A Critical Survey. IEEE Annals of the History of Computing 30, 66–81.

Gleick, J., 2011. The Information: A History, a Theory, a Flood, 1st ed. Pantheon Books, New York.

Goodin, D., 2013. Anatomy of a Hack: How Crackers Ransack Passwords Like "qeadzcwrsfxv1331" [WWW Document]. Ars Technica. URL http://arstechnica.com/security/2013/05/how-crackers-make-minced-meat-out-of-your-passwords/ (accessed 6.3.13).

Goodman, N., 1976. Languages of Art: An Approach to a Theory of Symbols. Hackett Publishing, Indianapolis.

Harman, G., 2009. Prince of Networks: Bruno Latour and Metaphysics. re.press, Melbourne, Australia.

Hayles, N.K., 1999. How We Became Posthuman: Virtual Bodies in Cybernetics, Literature, and Informatics. University of Chicago Press.

Heidegger, M., 2002. The Age of the World Picture, in: Young, J., Haynes, K. (Trans.), Off the Beaten Track. Cambridge University Press, Cambridge UK, pp. 57–85.

Introna, L.D., 2009. Ethics and the speaking of things. Theory, Culture and Society 26, 25–46.

Kahn, D., 1967. The Codebreakers: The Story of Secret Writing. Macmillan, New York.

Kohno, T., Ferguson, N., Schneier, B., 2010. Cryptography Engineering: Design Principles and Practical Applications. Wiley Publishing, Inc., Indianapolis, IN.
Latour, B., 1993. We Have Never Been Modern. Harvard University Press.
Latour, B., 1999. Pandora's Hope: Essays on the Reality of Science Studies, 1st ed. Harvard University Press.
Latour, B., Woolgar, S., 1986. Laboratory Life: The Construction of Scientific Facts, Reprint. ed. Princeton University Press.
Lessig, L., 2006. Code v2. Basic Books.
Merkle, R.C., 1982. Method of providing digital signatures. 4309569.
Nakamoto, S., 2008. Bitcoin P2P e-cash paper. Cypherpunks.
Prendergast, C., 2000. The Triangle of Representation. Columbia University Press, New York.
Shannon, C., 1945. A Mathematical Theory of Cryptography (No. 20878). Bell Labs, New Jersey.
Shannon, C., 1949. Communication Theory of Secrecy Systems. Bell System Technical Journal 28, 656–715.
Shannon, C., Weaver, W., 1948. A Mathematical Theory of Communication. Bell Syst. Tech. J 27, 379–423.
Slater, J.B. (Ed.), 2013. Mute Vol. 3 #4 - Slave to the Algorithm. Mute.
Stolzenberg, D. (Ed.), 2001. The Great Art of Knowing: The Baroque Encyclopedia of Athanasius Kircher. Stanford University Libraries, Florence, Italy.
Szabo, N., 1997. Formalizing and Securing Relationships on Public Networks. First Monday 2.
Thomsen, S.W., 2009. Some evidence concerning the genesis of Shannon's information theory. Studies in History and Philosophy of Science 40, 81–91.
Wiener, N., 1965. Cybernetics. MIT Press.
Wilding, N., 2001. "If You Have A Secret, Either Keep It, Or Reveal It": Cryptography and Universal Language, in: Stolzenberg, D. (Ed.), The Great Art of Knowing: The Baroque Encyclopedia of Athanasius Kircher. CADMO, Firenze, Italia, pp. 93–103.
Winner, L., 1980. Do Artifacts Have Politics? Daedalus 109, 121–136.
Zielinski, S., 2008. Deep Time of the Media: Toward an Archaeology of Hearing and Seeing by Technical Means. The MIT Press.