NeuCoin: the First Secure, Cost-efficient and Decentralized Cryptocurrency

Version 1.1
March 25, 2015

Kourosh Davarpanah
kourosh@neucoin.org

Dan Kaufman
dan@neucoin.org

Ophelie Pubellier
ophelie@neucoin.org

Preface

The NeuCoin Project hopes that this white paper will spark an honest, fact-based debate on the pros and cons of the two main systems used to provide security for peer-to-peer cryptocurrencies: proof-of-work and proof-of-stake.

NeuCoin believes that its own proof-of-stake design fully solves the security and centralization problems with earlier proof-of-stake coins, as well as the mounting cost and increasing centralization problems faced by proof-of-work systems. As such, NeuCoin is the first peer-to-peer cryptocurrency, regardless of technology, that is secure, cost-efficient and decentralized in the long run.

The NeuCoin Project is inviting all members of the cryptocurrency community to make comments, suggest edits, and point out potential flaws in the paper. NeuCoin bounties will be rewarded by the NeuCoin Code foundation for all constructive input, both positive and critical (click here for details).

Note that this paper does not discuss consensus algorithms based on trusted nodes (the systems used by Ripple and Stellar) because while that protocol has similar benefits to those of proof-of-stake, it requires users to trust third parties and beyond that, depends on trustworthy third parties to support the network by operating nodes, which may or may not happen over time.

Also not covered in this paper is the enormous side benefit derived from not having to distribute the currency supply to proof-of-work miners to cover their operating costs. Capitalizing on this opportunity, NeuCoin distributes the currency strategically to all the participants who increase its utility and value. Please visit NeuCoin's wiki[1] for a summary of NeuCoin's distribution and user adoption plans.

[1] neucoin.org/en/wiki

Abstract

NeuCoin is a decentralized peer-to-peer cryptocurrency derived from Sunny King's Peercoin, which itself was derived from Satoshi Nakamoto's Bitcoin. As with Peercoin, proof-of-stake replaces proof-of-work as NeuCoin's consensus mechanism, effectively replacing the operating costs of Bitcoin miners (electricity, computers) with the capital costs of holding the currency. Proof-of-stake also avoids proof-of-work's inherent tendency towards centralization resulting from competition for coinbase rewards among miners based on lowest cost electricity and hash power.

NeuCoin increases security relative to Peercoin and other existing proof-of-stake currencies in numerous ways, including: (1) incentivizing nodes to continuously stake coins over time through substantially higher mining rewards and lower *minimum stake age*; (2) abandoning the use of coin age in the mining formula; (3) causing the *stake modifier* parameter to change over time for each stake; and (4) utilizing a client that punishes nodes that attempt to mine on multiple branches with duplicate stakes.

This paper demonstrates how NeuCoin's proof-of-stake implementation addresses all commonly raised "nothing at stake" objections to generic proof-of-stake systems. It also reviews many of the flaws of proof-of-work designs to highlight the potential for an alternate cryptocurrency that solves these flaws.

Contents

1 Introduction

NeuCoin's technical ambition is to create a cryptocurrency that is secure, cost-efficient and decentralized in the long run. This paper explains why NeuCoin chose proof-of-stake over proof-of-work to achieve these goals.

Over time, the cryptocurrency community has generally become aware of several drawbacks of Bitcoin that spring from its proof-of-work design, including:

1. the prospect of higher transaction fees in the long run in order to maintain security
2. the increasing centralization and corporate control of mining
3. the divergence of interests between miners and Bitcoin holders

These flaws are mounting as Bitcoin matures, and they go against Satoshi's vision of an open, decentralized network maintained by its participants. They also undermine the principal economic benefit of cryptocurrency - very low transaction fees.

Many in the community believe that Bitcoin's network effects are strong enough to prevent an alternate cryptocurrency from achieving wide consumer adoption, even if it were technically superior to Bitcoin and solved its shortcomings. Others believe that a strong alternative to Bitcoin's consensus mechanism can be built.

"There is more than one way in which a distributed ledger system can work, and remuneration would have to be designed in such a way as to incentivize honest participation in the system without leading to socially inefficient over-investment in transaction verification" (*Source: Bank of England* [3, p31.])

Proof-of-stake's advantages over proof-of-work

Proof-of-stake, the most widely considered alternate to proof-of-work, fits this bill perfectly. There are two fundamental differences between proof-of-stake and proof-of-work. First, in proof-of-stake, miners compete for newly issued coins based not on the amount of electricity and computing resources spent, but rather on the number of coins owned. This crucial difference effectively eliminates the operating costs incurred in proof-of-work mining, replacing them with the capital costs of holding coins. Second, coinbase rewards (called *coinstake* rewards in proof-of-stake) are typically not a fixed amount (as in 25 per block in Bitcoin) but proportionate to the number of coins held by the miner and the period of time during which the coins used to stake have remained idle. As such, they are akin to "interest payments" on the miner's coin holdings.

Based on these two differences, proof-of-stake completely solves the three problems with proof-of-work that were presented above.

1. With virtually no operating costs in proof-of-stake, transaction fees can be far lower than in proof-of-work in both the short run and long run, and regardless of transaction volumes. The only reason they are above zero is to prevent transaction spam.
2. Proof-of-stake doesn't suffer from gradual centralization as proof-of-work does, because all proof-of-stake miners earn the same rate of return on their coins (the "interest rate") regardless of computing hardware or electricity costs.
3. There can be no misalignment between miners and coin holders, since they are by definition one and the same.

Proof-of-stake also has security advantages over proof-of-work when it comes to resisting 51% attacks from large hostile actors - those whose purpose would be to destroy the network as opposed to simply seeking financial gain. In proof-of-stake, an attacker would have to acquire 51% of the total coin supply (the value of which would be destroyed in an attack), versus deploying 51% of total computing power in proof-of-work (which computing power could be re-deployed elsewhere after the attack). Once a proof-of-stake coin achieves a material value, it would be extremely expensive for an attacker to buy up a large percentage of all coins.

Bitcoin supporters' rational response to proof-of-stake

It would be economically irrational for large Bitcoin owners to be supportive of or even open-minded about an alternative cryptocurrency design that purported to solve Bitcoin's main flaws and beyond that, enabled distribution to all the participants who helped increase its utility and value. Bitcoin supporters want all demand for digital currencies to be channeled to Bitcoin only. Moreover, they trust that if enough time passes before a strong challenger arises, Bitcoin's network effects will have grown to the point where it simply won't matter if someone builds a better mousetrap.

The party line among Bitcoin supporters is that in proof-of-stake, there's "nothing at stake". What they mean is that since proof-of-stake mining does not consume any outside resources (electricity, computing power), miners have no costs, so nothing prevents them from endlessly trying to commit double-spends, or mining on multiple branches, no matter how low the odds of success. Since there is "no cost" to behaving maliciously, proof-of-stake systems are unsecure and can't even reach consensus. They ask, "how can you have security without paying anything for it?"

What proof-of-work proponents are neglecting to see is that proof-of-stake security does have a cost: the capital cost of acquiring and holding coins. The brilliance of proof-of-stake is that it turns all coin owners into security providers, and requires any would-be attacker to purchase a large amount of the currency to attempt an attack, which would be an attack on his own wealth.

Besides ignoring the reality of capital costs, proof-of-stake critics are also prone to depicting scary-sounding attack vectors against proof-of-stake - grinding through the blockspace, rewriting history with old private keys, long-range, pre-programmed double spends - without explaining the details of how these attacks would be conducted or demonstrating mathematically that they have more than an infinitesimal chance of success. The truth is, these attack vectors do represent valid areas of concern, so it is interesting to ask why the critiques are always theoretical and never concrete.

Perhaps by leaving the critiques abstract, without even mentioning Peercoin, NXT, Bitshares, or Blackcoin, etc., with all their different parameters, the point is made that proof-of-stake's "nothing at stake" flaw is fundamental and nothing can be done to fix it. Or it could simply be that analyzing the odds of success of a given attack vector against a specific proof-of-stake implementation isn't worth the effort. Besides, none of the existing proof-of-stake coins have published substantial rebuttals to the various critiques.

NeuCoin hopes that this paper - first by frankly describing proof-of-work's own serious flaws, and second, by explicitly addressing the generic critiques of proof-of-stake - will force proof-of-stake critics to engage in a real debate about the pros and cons of proof-of-work versus proof-of-stake.

NeuCoin's design answers the "nothing at stake" argument

To choose the parameters and features of its own design - indeed to even pursue a proof-of-stake design in the first place - NeuCoin began by researching and mathematically modeling all the potential attack vectors against a proof-of-stake cryptocurrency: double spends, history revisions, grinding attacks, and preprogrammed attacks. In the end, NeuCoin believes that it was able to architect a proof-of-stake design that defeats all these attacks and addresses all "nothing at stake" issues.

There are three critical elements to its design (explained in detail in section 3.2)[2]:

1. **High mining incentives:** NeuCoin provides much higher mining rewards than existing proof-of-stake currencies in order to maximize the percentage of coins being mined at all times. The odds of success for all attack vectors in proof-of-stake are based on the percentage of staked coins that the attacker controls, so it is paramount to maintain a high percentage of all coins staking across time, which existing proof-of-stake coins fail to do. To further bolster mining participation, NeuCoin reduced *minimum stake age* to one day (from 30 days in Peercoin) and abandons the use of coin age as a factor influencing the probability of generating a block.
2. **Redesigned *stake modifier*:** NeuCoin chose to adapt a version of BlackCoin's *stake modifiers*, which floats over time, rather than use Peercoin's design, which permanently fixes the *stake modifier* after the initial *modifier interval*. NeuCoin chose this design because Peercoin's design is susceptible to preprogrammed long-range attacks (described in section 3.3.4). Moreover, the *modifier interval* and *selection interval* parameters were substantially adjusted to minimize the threat of grinding (described in section 3.3.3).
3. **Duplicate stake punishment:** NeuCoin uses a client version developed by Michael Witrant aka "sigmike" (core developer of Peercoin and Technical Advisor to NeuCoin) that not only detects duplicate stakes so that honest nodes can reject them, but also punishes nodes that broadcast duplicate stakes by rejecting all blocks broadcast by the dishonest miner.

The majority of this paper is devoted to describing the general principles of proof-of-stake, reviewing NeuCoin's own design, and demonstrating mathematically how NeuCoin addresses the "nothing at stake" objections and foils all attack vectors.

But before diving into that technical material, the paper will first discuss Bitcoin's proof-of-work design and its inherent problems, which are creating a clear opportunity for a competitor that can solve them.

[2]This summary of NeuCoin's changes to existing proof-of-stake implementations uses technical terminology that will be unfamiliar to those not well versed in proof-of-stake. These terms will be explained in detail in section 3.1.

2 Proof-of-work and Bitcoin

2.1 How proof-of-work secures the Bitcoin network

The public has a grand view of how Bitcoin - the digital currency with no banks or governments backing it - is secured. Bitcoin "miners" are envisioned to be computer experts racing against each other to solve complex math problems using cryptographic algorithms on powerful computers. In the public's imagination, the security is driven by the complexity of the math problems, the cryptography, the network's computing power, and the army of computer expert miners.

The digital currency community knows that the reality of Bitcoin mining is more prosaic. Systems administrators connect computers to the internet and run a software program that races to perform a function as fast as it can, the only purpose of which is to generate a pseudo-random number. The "complex math problem" is in fact just the pseudo-random number generator purposely designed to use a lot of computing resources. The owner of the first computer to generate a number below a certain threshold - a "proof of work" - gets to add a block to the *block chain*[3] and is rewarded with free Bitcoin.

The "proof-of-work" in Bitcoin mining is actually just "proof" that a miner did the "work" of running the software program and using electricity and computing power. It is proof that the miner incurred costs and spent money. Consensus security in Bitcoin is based not on the complexity of the math problems or the advanced cryptography, but on the amount of electricity and computing power spent by the miners.[4] The more resources spent, the more secure the network. Cutting the spending cuts the security. **Cost equals security.**

This may not sound so impressive to the public and mainstream media, but in fact it was the first practical - and brilliant - solution to a very difficult problem. To help explain Bitcoin's proof-of-work solution we will quote from a paper by Bitcoin researcher Andrew Poelstra[5].

Securing a peer-to-peer cryptocurrency requires that nodes of a distributed asynchronous network (the size of which is unknown and the nodes of which are anonymous) reach a consensus on the time ordering of messages.

As Poelstra writes[9]:

"The reason that this consensus is needed is called the double-spending problem. That is, in any decentralized digital currency scheme there is the possibility that a spender might send the same money to two different people, and both spends would appear to be valid. Recipients therefore need a way to be assured that there are no conflicts, or that if there are conflicts, that the network will recognize their version as the correct one. A distributed consensus on transaction ordering achieves this: in the case of conflict, everyone agrees that the transaction which came first is valid while all others are not."

In the case of Bitcoin, Poelstra continues:

"It can be mathematically proven that given only an asynchronous network it is impossible to achieve distributed consensus in a cryptographically guaranteed way[6]. Bitcoin achieves

[3]The *block chain* is the public ledger of all Bitcoin transactions that have ever happened.

[4]Bitcoin and Mooncoin (a cryptocurrency hoped to be used on the moon) both use the same technology and design (since Mooncoin is a fork of Bitcoin); the only difference is that there is very little hash power securing Mooncoin.

[5]We quote from Poelstra's paper entitled "Distributed Consensus from Proof of Stake is Impossible" - the proof-of-work community's favorite critique of proof-of-stake - which we rebut in section 3.

the impossible by weakening its requirement from cryptographic guarantee to a mere economic one. That is, it introduces an opportunity cost from outside of the system (expenditure on computing time and energy) and provides rewards within the system, but only if consensus on an unbroken transaction history is maintained." "To accomplish this, Bitcoin provides a way to prove, for each candidate history, (a) that opportunity cost was forfeited, and (b) how much. This is a so-called proof-of-work. Furthermore, the work proven includes that of all participants who worked on the history. The consensus history is the one with most total work (at least as far as it has propagated through the network our weak synchronicity requirement means that the consensus on the most recent part of the history is uncertain). Since the consensus history is the only one containing spendable rewards for work done, this means (a) that provers have an incentive to work on the same history that other provers are, and (b) individual provers can't take control of the history because they need their peers' contribution."

The technical specifics of creating a "proof-of-work" are actually quite simple. A miner must find a tuple that satisfies the following equation:

$$hash(blockheader, nonce) \leq target$$

The *blockheader* is a data structure that contains information about the block, which itself encapsulates information about the Bitcoin ledger and transactions. The *target* is a value shared by the network, adjusted so that on average new blocks are mined every ten minutes. The *nonce* is an integer.

The only way for a miner to add a block is to be the first one to find a hash below the target. The only way to do that is to try his luck as many times as possible by incrementing the nonce and checking the newly computed hash. The rate at which a miner generates blocks is proportional to the computing power he deploys relative to the rest of the network.

Bitcoin's proof-of-work system provides great security so long as no hostile actor controls 51% or more of the aggregate computing power in the network. If any entity did control a majority of the network's hash power, it could alter transaction history, or effectively shut the network down by refusing to record new transactions.

Since there are no barriers to entry in mining, the system reaches an equilibrium where mining expenses are roughly equal to mining revenues. That is, miners continue to add incremental hash power so long as the expected revenues are higher than the incremental expenses (where "expected" revenues are the product of the coinbase reward and the probability of earning it).

Bitcoin enjoys high levels of security today because the high price of Bitcoin incentivizes miners to deploy a huge amount of hash power[6]. In effect, security in Bitcoin and all proof-of-work designs is 100% dependent on and correlated to the costs incurred by miners generating their proofs of work. In the next section, we consider the implications of a system where its security is a function of its operating expenses.

[6]The hash rate deployed on March 12th, 2015 on the Bitcoin network was 391,367,666 GH/s or 391 PH/s *source:* `http://www.blockchain.info/charts/hash-rate`), ie more than 5,000 higher than the combined power of the top 500 supercomputers that exist in the world (*source:* `http://www.top500.org/statistics/list/`)

2.2 Problems with proof-of-work

2.2.1 Costs of security and transactions in Bitcoin

It is commonly believed that the principal economic benefit of Bitcoin and digital currencies in general is their cost advantage over legacy payment systems and their potential to offer lower cost transactions.

As Marc Andreessen put it in his widely read article Why Bitcoin Matters:

"Put value in, transfer it, the recipient gets value out, no authorization required, and in many cases, no fees. That last part is enormously important. Bitcoin is the first Internet-wide payment system where transactions either happen with no fees or very low fees (down to fractions of pennies). Existing payment systems charge fees of about 2 to 3 percent - and that's in the developed world. In lots of other places, there either are no modern payment systems or the rates are significantly higher."

But does Bitcoin really enable extremely low-cost transactions? When one contemplates hundreds of thousands of computers racing against each other to perform billions of pseudo-random computations per second - "proofs of work" - in hopes of generating a low enough number to be rewarded free Bitcoin, the value of which is equal to the entire network's costs it is hard to imagine that this is a cost-efficient system.

Figure 1: A technician tending a warehouse full of mining rigs dedicated to running a hashing function that outputs a pseudo-random 256-digit number.

As discussed above, Bitcoin's network security is based on the total hash rate[7] of all the miners in the network. And the aggregated hash rate deployed by miners is strictly a function of financial payments made to miners. Hence, network security is directly determined by payments to miners. For security to stay high, payments to miners must stay high.

Miner payments are actually composed of two components: coinbase rewards (the new Bitcoin that is minted every block and rewarded to the miner who creates the block) and transaction fees. From Bitcoin's inception in January 2009 through November 2012, coinbase rewards equalled 50 Bitcoin per block. In November 2012, they were cut to 25 per block,

[7]The hash rate is a unit of measurement corresponding to the number of calculations per second a computer can perform. The hash rate of the Bitcoin network equals the combined hash rates of all computers mining Bitcoin.

which will continue until roughly July 2016, when they will be cut in half again to 12.5 per block. This halving continues every 4 years indefinitely, so that coinbase rewards will decline to 0.2 Bitcoin in 25 years and to practically zero in 50 years.

Here is the simple formula for computing the value of miner payments each day:

$$\textit{daily payment} = \textit{coinbase reward} + \textit{transaction fees}$$

$$= [\textit{coinbase per block}] \times [\textit{blocks per day}] \times [\textit{Bitcoin price}]$$

$$+[\textit{number of transactions per day}] \times [\textit{Average transaction fee}]$$

Using actual numbers from March 6th 2015[8] and an average *block time* of 10 minutes: coinbase reward of $25 \times 144 \times \$271$ plus transaction fees of $96,611 \times \$.04$, which equals: $\$975,600 + \$3,864 = \$979,464$

Today fully 99.6% of the payments to miners are composed of newly created Bitcoin and just 0.4% is in the form of transaction fees. The actual cost per Bitcoin transaction today is \$979,464/96,611, or \$10.14.

Yes, those hundreds of thousands of mining rigs performing pseudo-random computations to create proofs of work end up costing a lot for every transaction they process.

Bitcoin end users today are shielded from (and oblivious to) these high costs because the high *coinbase* rewards today pay for the costs of the system. But what will happen over time as the *coinbase* rewards keep getting cut in half? Where will the miner payments come from that provide Bitcoin's security? Looking back at the formula, security will decrease unless:

1. the price of Bitcoin rises
2. the number of transactions rises
3. the price of transaction fees rise

Bitcoin supporters maintain that both the price of Bitcoin and transaction volumes will multiply 100- or even 1,000-fold from present levels to keep transaction fees low and security high in the long run. Looking at last year's numbers, those assumptions seem rather optimistic.

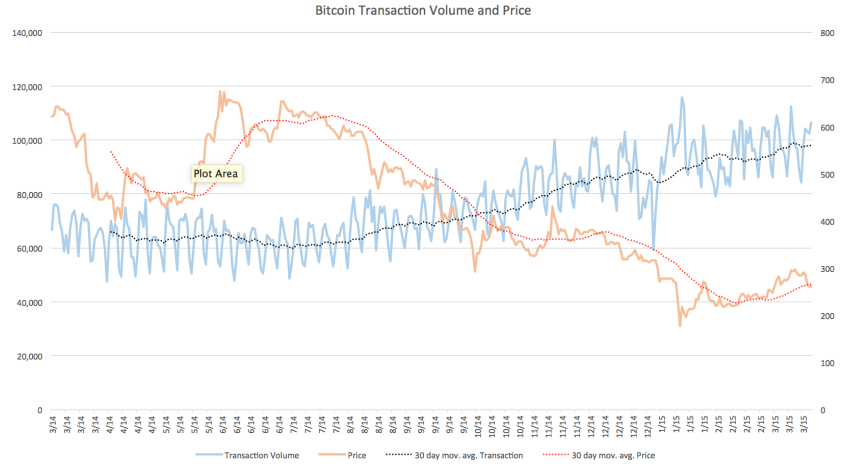

Figure 2: Bitcoin transaction volume and price from March 2014 to March 2015 (*source: blockchain.info*).

[8] *source:* `http://www.blockchain.info`

Meanwhile, even if user demand for transactions were to soar, there is the problem that Bitcoin's current block size doesn't allow transaction volumes of more than 5-10X today's levels. As Gavin Andresen, the Bitcoin Foundation's Chief Scientist (and Bitcoin's original core developer) put it:

"Once we get to one megabyte we've got to make blocks bigger. If we don't, transaction fees will just rise and rise and rise to a point where only rich people can afford to transact on the Bitcoin network... How is going to be the tricky bit. It's going to be hard to get consensus on exactly how far we should raise the block size."[9]

Getting consensus with Bitcoin miners who have veto power over changing the block size (or making any other change to the Bitcoin protocol requiring a hard fork) will indeed be "tricky". Miners may prefer smaller block sizes and higher transaction fees. Moreover, in its six years of existence, Bitcoin has not been able to achieve the necessary consensus to actually implement a hard fork a single time.

In conclusion, it is very hard to agree with Marc Andreessen that Bitcoin enables "no or very low transaction fees (fractions of pennies)"[10]. Secure, low cost transactions are incompatible with a system whose security is a direct function of its cost.

This could be disappointing to some, but maybe the greatest benefit of Satoshi's vision - of a decentralized network maintained by its participants without third party financial intermediaries - is not low costs but rather the decentralization itself.

2.2.2 Increasing centralization

In Bitcoin's early years, mining was highly decentralized among thousands of individuals using consumer-grade computers - and this true peer-to-peer nature was considered one of Bitcoin's core benefits: anyone was able to participate in making the *block chain* secure, removing the need for a central authority to validate and guarantee transactions. This was Satoshi's original vision. But as Bitcoins came to have a material value, Bitcoin mining has turned into a highly competitive business, with no barriers to entry, huge economies of scale, and only one dimension along which to compete: cost.

As a result, slowly but surely from 2010 through mid-2013, but after that in a landslide, hobbyist and small-scale miners have gotten knocked out of business. It's a very simple dynamic. When a miner with access to capital enjoys lower overall costs than the competition, he simply buys more computer power, driving up the difficulty of earning mining rewards, and knocking the least efficient competitors out of business. Lower costs can either take the form of lower cost hash rate or lower cost electricity. Bitcoin mining is rapidly becoming controlled by a handful of companies with tens of millions of dollars of the most efficient *ASICs*[11], operating from facilities with the very lowest electricity and cooling costs on the planet (e.g. Iceland and northern Sweden).

Centralization of proof-of-work mining is problematic for three reasons:

First and foremost, decentralization is one of the primary tenets of Bitcoin: a prime reason that Satoshi created it and its most attractive feature to many of its holders. Decentralization

[9] Gavin Andresen's keynote at Bitcoin 2014 in Amsterdam in May 2014

[10] Meanwhile, Marc Andreessen's venture capital firm is the largest investor in Ripple, a Bitcoin competitor with a $1.1 billion market capitalization which does not use proof-of-work mining and thus can offer near zero transaction fees in the long run.

[11] *Application-specific integrated circuit.* Mining ASICs are specifically designed to compute hashes and have made mining with a personal computer obsolete.

is what enables the removal of pesky, intrusive, and costly middlemen. As Bitcoin's increasing centralization becomes clearer and clearer to the crypto-community, many of its supporters may abandon it in favor of a more decentralized solution.

Second, centralization of mining represents a severe security risk. Any entity (or entities working together) that controls 51% or more of the network's computing power can seriously harm the network. As Gavin Andresen put it: "One of the things a 51% attacker can do is prevent any transactions or new blocks from anybody besides themselves from being accepted, effectively stopping all payments and shutting down the network"[1].

Third, when there are only a few, highly-capitalized entities that control the network, the entire network becomes susceptible to government control through regulation of these few entities. Governments often allow individuals to freely carry out certain activities among themselves but strictly regulate businesses that facilitate those same activities. As such, while governments might take a hands off approach to tens of thousands of geographically dispersed miners processing transactions for peers, they could decide to heavily regulate the few giant miners that dominate transaction processing.

The centralization of Bitcoin mining was not what Satoshi or many Bitcoin supporters wanted, but it's a fact. As a consolation, at least one would expect those powerful miners that control the *block chain* to do what's best for Bitcoin. Let's explore that.

2.2.3 Divergence of interests between miners and coin holders

In the early days, when mining **was** decentralized, miners and Bitcoin owners were mostly the same people, so miner interests and Bitcoin holder interests were basically the same.

Over time though, Bitcoin miners have become a distinct group from Bitcoin holders. Most miners sell the Bitcoins they earn right away in order to pay for energy costs, to recover their investment in hardware before their mining equipment becomes obsolete, and to invest in more advanced next generation *ASIC* chips.

The dominant miners today are large companies run by corporate executives backed by institutional investors looking to take them public. As such, they should be expected to behave just like other large companies: dedicated to maximizing their profits and minimizing their legal and regulatory risks, and looking at developing additional revenue streams, such as opportunities to sell their data or offering different pricing tiers for faster transactions.

As of today, the interests of corporate miners and Bitcoin holders are still relatively well aligned, because both groups have the same primary, overriding goal - to increase the price of Bitcoin. Miners share this goal with Bitcoin holders because today, 99.6% of miner revenues are in the form of newly created Bitcoin.

However, as one looks into the future, as miner revenues shift away from *coinbase* rewards to transaction fees, miners and holders will no longer share the same primary goal. Bitcoin holders will still want to maximize the value of Bitcoin. Miners will want to maximize their revenues from transaction fees and other sources.

Transaction fees are just one issue. As mentioned earlier, miners also have veto power over any changes to Bitcoin's protocol. It is hard to foresee how the dynamics will play out between Bitcoin's miners, core developers and holders, but given miner control over transaction fees and protocol changes, it's safe to assume that the interests of the large corporations that control Bitcoin mining will take precedence.

2.2.4 Summary of Bitcoin's proof-of-work problems

Bitcoin is not cost-efficient, it is becoming more and more centralized, and the miners who control the network have diverging interests from Bitcoin holders.

As Vitalik Buterin, inventor of Ethereum, put it in December 2014: "What Bitcoin is doing is paying $600M/year [...] on a 5 of 10 multisig. Ultimately there are maybe 5 or 10 [...] mining companies that control the entire network. [...] It's this incredibly inefficient protocol where miners are literally competing to see who can waste the most resources the fastest and on the other hand it's not getting us much decentralization."[12]

The preceding section on Bitcoin's and proof-of-work's problems was not meant to argue that Bitcoin is doomed, or even that it will not remain the leading cryptocurrency for the foreseeable future. Bitcoin was the pioneer and enjoys substantial network effects stemming from its large community of investors, developers and companies that are building out its ecosystem and creating meaningful value propositions for users.

However, Bitcoin's serious and growing problems should be kept in mind when evaluating the viability of alternate cryptocurrency designs. In particular, these real problems should be weighed against the hunch-based arguments and folklore that have been used to claim that other designs are unviable or even outright "impossible" because they are "not secure enough", or "not decentralized enough". With that in mind, let's now review the main alternative to proof-of-work: proof-of-stake.

3 Proof-of-stake

Note: This section uses Peercoin[7] as its primary reference, rather than other leading proof-of-stake implementations such as NXT or Bitshares, for two reasons:

- *Peercoin was the first proof-of-stake cryptocurrency to launch and publish source code (2012, versus 2014 for NXT and Bitshares). It has received the most discussion and attention and has been forked the most times*
- *Peercoin was the proof-of-stake implementation that NeuCoin chose to fork, for the reasons above and because the NeuCoin team felt most confident in using Peercoin as the basis for its own secure proof-of-stake design.*[13]

Introduction to proof-of-stake

In proof-of-work, the task of reaching consensus on transaction history is delegated from coin holders to miners, who spend computing power and electricity, but need not own any coins. In proof-of-stake systems, the task of reaching consensus is not delegated to third parties; it is done by the coin holders (the "stake" holders) themselves.

In proof-of-stake, mining nodes generate blocks in proportion to their stake in (i.e. ownership of) the currency itself. Every second, every *stake* has a certain probability to generate a block and this probability is proportional to its size. The hashing operation is done over

[12] https://www.youtube.com/watch?v=qPsCGvXyrP4 at 12:50

[13] This does not mean that NeuCoin does not believe that NXT and/or Bitshares could also be viable implementations to achieve adequate security in a proof-of-stake design, just that NeuCoin determined that Peercoin was the safest, most direct path.

a limited search space (specifically one hash per unspent transaction output, or *UTXO*, per second) instead of an unlimited search space as in proof-of-work.

Instead of providing security based on **operating costs** as in proof-of-work, proof-of-stake provides security based on the **capital costs** of investing in the currency.[14]

This one critical change from proof-of-work is what allows extremely low transaction fees in the long run - without relying on enormous growth in transaction volume or the price of tokens.

The second major difference between proof-of-stake and proof-of-work mining is the formula for computing mining rewards. Instead of using a fixed reward (e.g. 25 Bitcoin per block), proof-of-stake miners generally earn a rate of return on the **number of coins** being staked and the **amount of time** they have been idle[15] (akin to an interest rate).

Acting together, these two changes allow a proof-of-stake cryptocurrency to remain decentralized in the long run. Miners have no potential to gain advantages over each other by investing in powerful, specialized computers or by accessing lower cost electricity. In addition, a miner who owns 10% of the currency supply will not earn appreciably higher returns over time than a miner who owns 0.01% of the currency supply - they both earn the same interest rate even if the larger coin holder receives payments more frequently.[16]

3.1 How Peercoin's proof-of-stake works in contrast to proof-of-work

In proof-of-work coins like Bitcoin, miners use computing power to extensively modify the input of the hash. By design, the nonce creates a quasi-infinite search space to find a block header that produces a valid proof of work.[17]

In Peercoin, a miner's chances of creating a block are based on his stake alone regardless of computing power. This is accomplished by limiting miners to the rate of one hash per second per stake.

Thus, to create a valid "proof of stake" and generate a block, a miner must:

- Control coins (technically, an unspent transaction output, or *UTXO*) that have been idle for a minimum time period (the *minimum stake age*)
- Find a *kernel* (the equivalent of the block header in proof-of-work) that satisfies Peercoin's mining formula[18]:

$$hash(kernel) \leq target \times balance\ of\ UTXO \times time\ weight$$

[14]Some Bitcoin proponents insist that the ownership of currency cannot be considered a "cost". They should consider the opportunity cost of converting fiat money (which can earn interest) into digital currency and the risk one takes of devaluation while he holds the digital currency. If converting fiat currency into digital currency were "costless", people should be willing to do so even without the promise of gains.

[15]Time "idle" means the amount of time that has elapsed since the staked coins last earned a *coinstake* reward or were sent from one address to another.

[16]The larger coin holder has only one small advantage: he receives more compounded interest, which is insignificant in comparison to the overall *coinstake* rewards.

[17]Even without the nonce, which is used in practice to "grind" through a large number of block headers to find a suitable hash, a miner could very well obtain the same result by modifying a number of other parameters.

[18]In Peercoin's mining formula, *balance of UTXO* $\times$ *time weight* is sometimes written as *coins* $\times$ *age*, or as just *coinage*. In Peercoin, *time weight* is assigned a value of 0 for 30 days following the time that stake received a *coinstake* reward or was moved from one address to another, and then grows from a value of 1 to 60 over the following 60 days, where it is capped.

In the mining formula, the *target* is the same for the entire network and is readjusted at every block. The *balance* of the *UTXO* is the size of the stake. The *time weight* relates to the amount of time since the coins were touched. Note that in contrast to proof-of-work, where all miners are hoping their hashes fall below a given shared target, proof-of-stake miners are hoping their hashes fall below a value that is unique to each stake (the larger the stake, the easier to create the proof).

The *kernel* is a data structure composed of the following parameters:

- *nTimeTx*: current timestamp (incremented every second)
- *nTxPrevTime*: Timestamp of the *UTXO*
- *nPrevoutNum*: Output number of the *UTXO*
- *nTxPrevOffset*: Offset of the *UTXO* inside the block
- *nTimeBlockFrom*: Timestamp of the block which provided the *UTXO*
- *nStakeModifier*: a 64-bit string seeded from the *block chain*[19]

The current timestamp (*nTimeTx*) modifies the *kernel* each second to provide one chance per second per *UTXO* to generate a block.

So that every stake performs differently, the *kernel* needs to include information specific to the stake. The timestamp of the *UTXO* (*nTxPrevTime*), its output number (*nPrevoutNum*) and its offset in the block which provided its first confirmation (nTxPrevOffset) fulfill that role. The reason why all three of them are included is simply to reduce the chance of nodes having the same *kernel* (which could result in them generating blocks at the same time).

If all miners could be trusted to behave honestly, the above parameters would be sufficient. However, such a design would have three serious vulnerabilities:

1. First, miners could mine on multiple chains to avoid the chance of having their blocks *orphaned*, which would undermine the ability of the network to reach consensus.

2. Second, it would allow users to pre-compute future proofs-of-stake by calculating hashes using future *timestamps*. This could encourage miners to only mine their stakes at the time they were likely to earn a *coinstake* reward, severely curtailing the mining participation rate over time. It could also allow miners to attack the network by colluding.

3. Third and most problematic, users could gain advantage over other miners by using computational power to "grind" through *kernels* to identify virtual *UTXOs* that would mine very quickly or at some specific time in the future. If it were profitable to gain advantage using computational power, then the system would degrade into a proof-of-work situation, with all of its shortcomings.

To correct the first issue, Peercoin built a feature into its client called the *duplicate stake detection mechanism* that detects when miners attempt to mine on multiple chains and rejects their blocks mined with duplicate stakes.[20]

[19]The generation mechanism of the *stake modifier* and its role in proof-of-stake mining are described in details in the following paragraphs.

[20]Proof-of-stake critics raise the concern that there could be a tragedy of the commons caused by large numbers of selfish miners colluding and altering their clients to both transmit and accept duplicate stakes, preventing the network from reaching consensus. The NeuCoin team does not share this concern. This sort of selfish behavior could only yield negligible gains but would destroy the currency if it became widespread. Regardless, NeuCoin's modification to Peercoin's client completely addresses the concern of nodes mining on multiple chains by penalizing those that attempt to do so. See section 3.2 for further discussion.

Peercoin attempted to address the second and third issues at the protocol level by including a parameter which included the timestamp of the block which provided the *UTXO* (*nTimeBlockFrom*). It also included the current proof-of-stake *difficulty* from which the network's target is derived. These features were intended to obfuscate the future performance of the stake, but did not perform as well as necessary.

Recognizing that Peercoin was still vulnerable to the precomputation of future proofs and grinding attacks, seven months after Peercoin's launch, creator Sunny King introduced a final parameter, called the *stake modifier* (*nStakeModifier*). The next two sub-sections explain this complex parameter, which is widely misunderstood in the digital currency community at large and even among proof-of-stake's own proponents and detractors.

Peercoin's *stake modifier* parameter

In brief, the *stake modifiers* for a given *UTXO* is calculated using bits from blocks that are generated **after** the miner receives the *UTXO*. Furthermore, each *stake modifiers* modifies a large group of stakes - all of them that have their first confirmation in a six-hour window in the case of Peercoin.

Given this design, a miner cannot precompute a future proof when he first receives a *UTXO* because he does not yet have the *stake modifiers*, one of the required parameters of the *kernel*.[21].

As for how the *stake modifiers* impacts attempts at grinding through the blockspace, there are two variants to consider:

1. **Dishonest miners working on the main chain.** The *stake modifiers* completely thwarts this attack. Recall that without *stake modifiers*, the miner could grind through various virtual *UTXOs* to find a *kernel* that performs well with a timestamp in the near future. With the advent of *stake modifiers*, given that (a) *kernels* cannot be hashed without *stake modifiers*, and (b) that *stake modifiers* cannot be computed for some period after a miner receives a *UTXO*, there is no way to grind through virtual *UTXOs* to find a high-performing *kernel*.
2. **Dishonest miners working on their own forks.** This case, in which miners have complete control over the block space on their own fork, represents the most virulent form of the "grinding attack" referred to by proof-of-stake critics. The threat is that attackers could grind through different *stake modifiers* to try to find those that enable the attacker to perform better than the rest of the network. A detailed discussion of this attack, along with how NeuCoin's design addresses it, is presented later in section 3.3.3. For now it will be noted that the *stake modifiers* was designed with the express intent of making this attack impractical by limiting the room to maneuver that an attacker has even operating on his own branch. More specifically, the *stake modifiers*:

   (a) is the only parameter through which an attacker could grind;
   (b) cannot be predicted;
   (c) modifies a large group of stakes at once, so that attackers cannot grind through stakes one at a time. He is stuck having to grind through *stake modifiers* in hopes of finding one that enables a large group of stakes to perform better than the

[21] In the case of Peercoin, the *stake modifier* for a stake can be calculated 9 days after receiving the stake. At that point in time, a user can compute when in the future the stake is likely to generate a proof-of-stake.

network as a whole. This will be shown to be near impossible unless the attacker owns a near-majority of all staked coins.

Mechanics of *stake modifier* generation in Peercoin

NOTE: The material in this sub-section can be skipped by non-technical readers. It is necessary for following the later discussions regarding how NeuCoin redesigns Peercoin's stake modifier to defeat preprogrammed attacks and counter the grinding attack.

The mechanics of generating the *stake modifier* parameter are complex. They are explained below in order to provide a basis for understanding what would be involved in a grinding attack, in which an attacker grinds through *stake modifiers* in an attempt to find one that enables a large number of his stakes to outperform all the stakes in the rest of the network.

In the case of Peercoin, the *stake modifier* is a 64-bit number that can be computed 9 days after the *UTXO* is first received. Hence, there are 2^{64} (approximately 10^{19}) possible *stake modifiers* that an attacker could grind through.

A new *stake modifier* is computed every *modifier interval* (*nModifierInterval*), which is 6 hours in Peercoin. The *stake modifier* is computed for the first block of this 6-hour long window, and all the other blocks within this window inherit the same *stake modifier*. Since Peercoin's *block time* is 10 minutes, there are approximately 36 blocks that share each *stake modifier*. All of the stakes first confirmed within any of these 36 blocks inherit the same *stake modifier*.

The 64 bits in the *stake modifier* are generated by first selecting 64 blocks from a time window (*nSelectionInterval*) that begins 9 days before the beginning of the *modifier interval* (at the time the miner receives the *UTXO*) and ends when the *modifier interval* begins. The selection of the blocks happens over 64 incrementally longer windows contained in the *selection interval*. One block is selected from each time window.

The graph below depicts the 64 subdivisions of the *selection interval*.

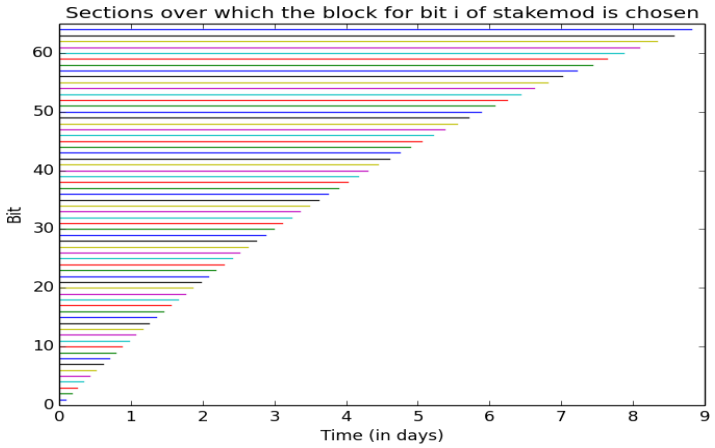

Figure 3: Subdivisions of the *stake modifier selection interval* over which the block contributing to the i-th bit is chosen.

Each of the 64 selected blocks contributes one bit to the *stake modifiers* as described below.

The first bit is computed as follows:

- Within the shortest time window in the chart above, for each of the ~ 12 blocks therein:
  - The hash of the *kernel* used to create the block (the *hashProof*) and the previous *stake modifiers* are concatenated
  - The resulting string is then hashed
- All the (~ 12) resulting hashes are compared
- The block which provided the lowest hash is selected and the least significant bit of this hash is selected

To compute the second bit:

- The second shortest time window is considered
- The previously selected block is ignored
- We go through the same steps we went through for the first bit. This time the block is selected from among ~ 24 blocks

This process is repeated across 64 time windows until the 64 bits have been generated.

The complexity of the generation mechanism was required in order to create a strong interdependency between the parameters of the *UTXO* used in the *kernel* and the *stake modifiers* as well as between the consecutive *stake modifiers*.

Mining rewards and transaction fees

The entire technical discussion of Peercoin above related to how miners create "proofs of stake" in order to be able to generate blocks. Here we briefly consider the financial rewards of mining.

As mentioned earlier, miners in Peercoin earn rewards (*coinstakes*) that are akin to interest rates. The precise mining reward formula is as follows:

$$coinstake = balance\ of\ UTXO \times \frac{days\ idle}{365} \times 1\%$$

Balance of UTXO is the number of coins, *days idle* is the amount of time since the coins last earned a *coinstake* reward or were transferred from one address to another (technically, it equals the current timestamp minus the timestamp of the *UTXO*).

There are two interesting things to note. First, miners do not earn any transaction fees, which are equal to .01 Peercoin per transaction. Rather than paying these fees to miners, Peercoin elected to actually destroy them. Its rationale was to partially offset the growth of the currency supply due to *coinstake* rewards. Peercoin's creator Sunny King put a very high priority on achieving a long term inflation rate close to zero.

But the much bigger point about Peercoin's reward formula is that miners only earn a 1% annual rate of return on their staked coins. As we will see later, this low rate is a serious problem.

Is Peercoin's proof-of-stake design secure?

In other words, does Peercoin answer the "nothing at stake" critiques? Does it both (a) prevent attackers from altering history and (b) maintain consensus on the order of transactions on a single *block chain*?

Before answering this question, let's recall the context. Bitcoin itself is not cryptographically secure[6]. It only tries to be "economically secure" in the sense that an attacker would have to spend more than he could gain through an attack. Furthermore, it is fully acknowledged that any actor controlling 51% of Bitcoin network's hash power could effectively shut Bitcoin down. In addition, Bitcoin has some flaws - inherently high operating costs, increasing centralization, and diverging interests between miners and coin holders - which Peercoin would correct - so perhaps some loss of security in Peercoin relative to Bitcoin would be an acceptable tradeoff.

Even with all of these considerations, our conclusion would still be that Peercoin's design is only somewhat secure - and significantly less secure than that of Bitcoin. The rationale for our conclusion will be explained in the following section 3.2, where we review all of the changes that NeuCoin has made to Peercoin's design. In our opinion, the factor that detracts the most from Peercoin's security is that its design fails to incentivize a large proportion of coins to be staked at any one time. It also suffers from issues related to its use of coin age in block generation. Lastly, it fails to foil attempts to precompute proofs of stake due to the static nature of its *stake modifier* design.

Whether or not Peercoin's creator Sunny King would agree with our specific conclusions is unknown. It is likely, however, that he shares the view that Peercoin's design is not currently "secure enough". This can be inferred from the fact that three years after its creation, Peercoin still uses centrally broadcast checkpoints that prevent any possible changes to the part of *block chain* earlier than the checkpoint. The checkpointing is done several times per day by way of Sunny King digitally signing the *block chain*.

The great thing about *checkpointing* is that even proof-of-stake's fiercest critics will concede that it does secure the currency against attacks. The downside of checkpointing is that it is a form of centralization. It is not the case that the entity that performs the checkpointing has the power to control the *block chain*, but it is the case that all users of the currency are relying on this "trusted entity" to provide a necessary layer of security.

Unfortunately for Peercoin, checkpointing has severely hurt market acceptance of the currency partly because the checkpointer is the anonymous creator Sunny King himself. There are obvious and near-fatal problems with trusting an anonymous person with a necessary security function. What happens if that anonymous person dies or becomes incapacitated, sells off his or her stake in the currency, or simply loses interest?

3.2 NeuCoin's proof-of-stake design

NeuCoin believes that its own proof-of-stake design is fully secure - and will not need to use checkpoints. This section describes all of the principal changes that NeuCoin made to Peercoin's design and explains how they increase security and address Peercoin's vulnerabilities. Following this section, the paper will review all of the attack vectors against proof-of-stake designs and demonstrate how NeuCoin's design stands up to these attacks.

NeuCoin significantly modified six aspects of Peercoin:

1. **Mining reward rates:** NeuCoin dramatically increased *coinstake* rewards for mining in order to maximize the percentage of coins being mined at all times, which is the bedrock of security in any proof-of-stake cryptocurrency. NeuCoin's rewards start at a 100% annual interest rate and decline steadily over a 10-year period to a 6% rate - versus just 1% per year in Peercoin.
2. **Minimum stake age:** NeuCoin's design uses a 1.6 day *minimum stake age* versus 30 days in Peercoin. This also has the effect of increasing mining participation.
3. **Role of coin age in the mining equation:** NeuCoin does not utilize coin age (*day-weight*) in the mining equation as a factor for determining the probability of generating a block. This change also increases mining participation.
4. **Block time**: NeuCoin uses a *block time* of 1 minute, versus 10 minutes in Peercoin, which improves user experience and enhances security against some attack vectors.
5. **Stake modifier:** NeuCoin chose to adapt BlackCoin's *stake modifier*, which floats over time, rather than Peercoin's, which permanently fixes the *stake modifier* after the initial stake interval for a given set of *UTXOs*. NeuCoin chose this design because it believes Peercoin's design is susceptible to preprogrammed long-range attacks (described in section 3.3.4). Modifier interval and *selection interval* were substantially adjusted relative to both BlackCoin and Peercoin in order to reduce the effectiveness of grinding through *stake modifiers*.
6. **Duplicate stake punishment:** NeuCoin uses a client version developed by Michael Witrant, aka "sigmike" (core developer of Peercoin and Technical Advisor to NeuCoin), that not only detects duplicate stakes so that honest nodes can reject them, but also punishes nodes that broadcast duplicate stakes by rejecting all blocks broadcast by the dishonest miner. This revision completely addresses the concern that proof-of-stake designs cannot reach consensus due to miners mining on multiple forks.

Mining reward rates: 100% annual rate declining to 6% per year

The key to high security in all proof-of-stake coins is maintaining a high percentage of coins staking over time. Therefore, it is critical to provide high compensation to miners for staking their coins.

One severe weakness of Peercoin is that it suffers from extremely low mining participation, with typically less than 10% of the currency supply staked at any one time. This means that an attack requiring control of 51% of the staked coins only requires control of 5.1% of the total currency supply. There are several factors that contribute to the low mining rates in Peercoin, but the principal one is miniscule rewards for mining: just a 1% annual return. The

small amount of mining that does take place in Peercoin is most likely driven more by selfless efforts to support the currency than by the financial rewards of mining.[22]

In contrast, NeuCoin provides very high compensation to miners, especially during the early years following its launch. Specifically, NeuCoin's *coinstake* rewards begin at a 100% annual rate and decline in a linear fashion over ten years after which they reach a 6% annual rate, where they remain indefinitely. These high reward rates will incentivize a large number of miners to stake a large number of coins. In addition, these high reward rates create incentives for cloud mining operators to provide services to NeuCoin holders not interested in mining themselves, where both the mining operator and the NeuCoin holder can receive substantial rewards.

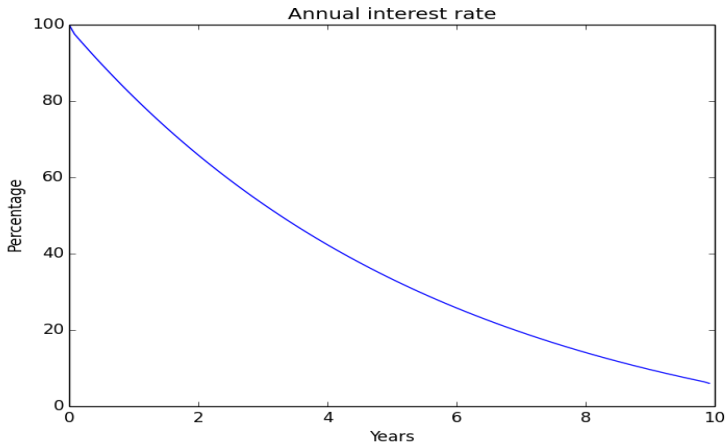

Figure 4: Evolution of the annual interest rate during the first 10 years.

Some observers might consider NeuCoin's high reward rates to cause unacceptably high inflation. In fact, NeuCoin utilizes high inflation deliberately in its early years to compensate users for holding and staking the currency while its value, utility and security are lowest and its risk is highest.

Given the mining reward rate of 6% from year 10 onwards, and assuming a 50% coin staking rate, long-term currency supply growth is projected to be 3% per year. Given that this is slower than fiat currency monetary growth (which averages the rate of economic growth plus the rate of inflation), in the long run there will be a mild deflationary effect relative to fiat money. This rate of inflation is highly preferable to outright deflation for any digital currency that aspires to serve as an actual unit of account and means of exchange for commerce as opposed to merely a store of value.

Minimum stake age: 1.6 days

On top of high mining rewards, the second important modification NeuCoin made to increase mining participation was to reduce *minimum stake age* to 1.6 days, versus 30 days in Peercoin. This change significantly decreases the number of coins that are ineligible for mining because

[22]This issue of low mining incentives is even worse for NXT than for Peercoin. NXT has no *coinstake* rewards at all. Instead, the only financial reward for mining is to earn transaction fees, which amount to just $26 worth per day at NXT's market price on March 17, 2015.

they have done so recently (more recently than the *minimum stake age*). It also reduces the barrier to entry associated with having to wait for such a long time before being eligible to start mining with any given stake.

It would have been preferable to have a *minimum stake age* of even less than one day - one hour for instance. For the same reasons listed just above, an even lower *minimum stake age* would have tended to increase the mining participation rate even further. Unfortunately, at a certain point, lower *minimum stake ages* can begin to increase the risk of a grinding attack (discussed in section 3.3.3). NeuCoin believes that 1.6 days represents the best tradeoff.[23]

Coin age: zero effect

NeuCoin does not utilize coin age (*day weight*) in the mining equation as a factor for determining the probability of generating a block. NeuCoin's mining equation is simply:

$$hash(kernel) < target \times balance\ of\ UTXO$$

versus Peercoin's mining equation: $hash(kernel) < target \times balance\ of\ UTXO \times time\ weight$

In Peercoin, *time weight* was the parameter that reflected coin age, equalling 0 up to the *minimum stake age* (30 days) and then growing from a value of 1 to 60 over days 31 to 91, at which level it is capped.

NeuCoin's deletion of the *time weight* parameter brings two significant benefits for security but at the cost of a loss of decentralization. Let's consider the benefits first.

First, *time weight*'s deletion removes the opportunity for attackers to *supercharge* their stakes. An attacker can keep his node offline until he reaches the maximum *time weight* of 60 (after waiting 91 days from receiving the stake) and then carry out an attack with far more block generating power than he would normally have. This *supercharging* effect is especially potent if an attacker were to use it in combination with a preprogrammed, long-range attack, described in detail in section 3.3.4.

Time weight's deletion also tends to increase mining participation. Allowing stakes to accumulate *time weight* for block generation encourages miners to keep their nodes offline and come back online after accumulating significant *time weight* to very quickly collect the reward. This behavior is compounded with Peercoin's 30 day *minimum stake age*, which requires nodes to wait 30 days before earning a *coinstake* reward. But at 31 days, a miner's odds of earning a reward are very low because he is competing with nodes that have high *time weight*. Rational (albeit selfish) behavior would be to wait 90 days before mining, at which time the reward will likely be earned quickly. Obviously, the percentage of coins being staked over time suffers greatly if even dedicated miners find it advantageous to leave their nodes offline the majority of the time.

As mentioned above, there was also a downside to disregarding coin age in the mining equation - a loss of decentralization.

In fact, using coin age in the mining equation is a centerpiece of Peercoin's design. The benefit of the time-weight parameter is that it significantly reduces the amount of time it takes **very small coin holders** to mine blocks. This goal was clearly a top priority for Peercoin. Recall Peercoin's mining equation, where time-weight = 0 for the first 30 days

[23] In fact, one of NeuCoin's top research priorities is finding ways to prevent grinding attacks with lower mininum stake ages, which would allow NeuCoin to reduce *minimum stake age* to 12 hours or less.

and then grows from values of 1 through 60 over the following 60 days. Now consider two holders of Peercoin, one who has owned 1,000 Peercoins for 90 days while the other has owned 50,000 Peercoins for 31 days. Due to time-weight's role in Peercoin's mining equation, the small coin holder actually has a higher chance of mining than the large coin holder with only 1/50th of his stake. This benefit was undoubtedly a major selling point for Peercoin when it was introduced to the cryptocurrency community August 2012, which then consisted almost entirely of proof-of-work miners. Many of these miners must have become frustrated by the fact that it could take years for a small miner to earn a coinbase reward in Bitcoin. All that said, it must be recalled that all miners in NeuCoin do still benefit from coin age in the *coinstake* reward equation, which is the same as it is in Peercoin, (except for the fact that in NeuCoin the interest rate starts at 100% and declines to 6%, versus a flat 1% in Peercoin):

$$coinstake = balance\ of\ UTXO \times \frac{days\ idle}{365} \times x\%$$

Consider two NeuCoin miners with the same number of coins (balance of *UTXO*). Miner A is lucky and receives one *coinstake* after 45 days (*days idle* = 45) and a second *coinstake* 15 days later (days idle = 15). Miner B receives only a single *coinstake* after 60 days (*days idle* = 60). How much better off is Miner A over Miner B? Miner B's single *coinstake* was **almost** equal to the sum of the two *coinstakes* received by Miner A, but not quite. The reason for the discrepancy is that Miner A earned a tiny bit of incremental compounded interest: 15 days of interest on the first *coinstake* reward he received on day 45.[24] In addition to this slightly higher amount of total *coinstake* rewards, Miner A also had a small benefit in terms of liquidity. That is, Miner A could have sold his coins after 45 days without sacrificing any accumulated coin age (equivalent to accrued interest). Miner B had to wait an extra 15 days.

In sum, in abandoning time-weight, NeuCoin chose the benefit of higher security at a cost of creating small advantages for large miners over smaller miners, which would tend to lessen decentralization. However, the final thing that should be noted is that only small miners will experience this disadvantage to a material degree. Given NeuCoin's parameters for *minimum stake age* and *block time*, a miner owning 1/50,000 of the total currency supply would be expected to earn a *coinstake* once per month.[25]

Block time: one minute

NeuCoin chose a 1 minute *block time* primarily to benefit users who appreciate one relatively quick (though not very reliable) confirmation for transaction processing. In addition, having a shorter *block time* does increase security for some types of attacks.

The probability for a transaction to be reversed by an attacker depends on its number of confirmations (i.e. how deep the transaction is in the *block chain*). The more confirmations, the more blocks the attacker needs to create on his competing fork.

In the case of a simple double spend - or any attack that attempts to alter transaction history without doing anything to improve performance such as grinding or supercharging stakes - shorter *block times* reduce the attacker's odds. In these cases the attacker is hoping

[24] To illustrate how small a factor this is, consider the **effective** annual interest rate of a 10% simple annual interest rate compounded at different intervals: compounded quarterly: 10.38%; compounded monthly: 10.47%; compounded weekly: 10.51%; compounded daily: 10.52%

[25] Assuming $\sim$ 85% of the total number of coins mine at any given time. If the percentage were to decrease, the considered miner would earn a *coinstake* more often.

to get lucky over a short period of time thus he is hurt by having to create more blocks in a given time period.

However, in the case of a grinding attack or a deep reorganization, where the attacker has to build a very long chain, his only hope is to be able to consistently create blocks faster than the main chain (because luck won't help over a long time horizon). In these cases, *block time* is irrelevant.

The reason that NeuCoin did not choose an even shorter *block time* is that the *block chain* fork rate (the probability that a miner's block is orphaned) increases when *block time* decreases. With a 10 minute *block time*, the fork rate is approximately 1.9%. With 1 minute, it is closer to 17%. Going down to 10 seconds would cause a fork rate of roughly 70%[5].

Reaching consensus in these conditions would be difficult and inefficient, as there would be near constant reorganizations. Given that getting orphaned has only limited impact on miner returns in proof-of-stake, NeuCoin considered 1 minute a good tradeoff between fast transaction confirmations and *block chain* fork rate.

Stake modifier: dynamic; 200 minute *modifier interval*; 1.6 days *selection interval*

As described in section 3.1, Peercoin's *stake modifier* is generated 9 days (the *selection interval*) after the first confirmation of a given stake. Although the *stake modifiers* is not computable at the time the stake is received, once it is generated it does not change over time. As a result, miners can predict when they will be able to mine in the future with a given stake. This could reduce mining participation over time, but more critically, it opens the door to the preprogrammed double spend attack, described in section 3.3.4.

BlackCoin addressed this vulnerability of Peercoin's "static" *stake modifier* in their August 2014 hardfork. BlackCoin's protocol update changed the *stake modifier*'s behaviour in two ways:

1. It caused a given stake to use different *stake modifiers* over time. Rather than acquiring a static *stake modifier* generated over the initial *selection interval* following a stake's first confirmation, stakes utilize the *stake modifier* seeded from the most recent *selection interval*.[26] (BlackCoin's *selection interval* being 5.87 hours).
2. As a consequence, all the stakes being mined at any one time use the same *stake modifier*. The *stake modifier* changes at every *modifier interval*, 10 minutes in the case of Blackcoin. Hence, every 10 minutes a new *stake modifier* is generated based on blocks from the preceding 5.87 hour window.

Put simply, BlackCoin's update caused the *stake modifier* to become dynamic, to shift over time. This change made it impossible to precompute when in the future a *UTXO* would generate a block.

In addition, BlackCoin's choices of values for *modifier interval* and *selection interval* enabled BlackCoin to have a much lower *minimum stake age* than PeerCoin. Having a lower *minimum stake age* tends to cause a higher mining participation rate.

Unfortunately for Blackcoin, however, their choice of a 10 minute *modifier interval* made them more susceptible to a grinding attack (reviewed in section 3.3.3) than need be the case.

[26] Note that this does not mean that when a stake is received, the user can already compute its *stake modifier*. Indeed, since the stake is not allow to mine for *minimum stake age* (which is larger than the *selection interval*), none of the blocks that will be used to compute the first *stake modifier* that will be included in the *kernel* exist at that point.

The potential benefits an attacker can gain by grinding through *stake modifiers* depends on the number of blocks contained in the *modifier interval*. In essence, during this period, the attacker must be able to find more blocks than the rest of the network. With BlackCoin's 10 minute *modifier interval*, the attacker must grind through *stake modifiers* until he finds one that enables him to generate ten blocks faster than the rest of the network. NeuCoin believes that this is not an adequate defense against a grinding attack (discussed in section 3.3.3).

NeuCoin chose a *modifier interval* of 200 minutes and a *selection interval* of 1 day. As will be shown in section 3.3.3, these parameter choices would require an attacker using grinding to acquire at least 31% of the staked coin supply. Choosing a *modifier interval* of 400 minutes would have increased the required coins for a grinding attack up to 36%. 800 minutes would have raised the requirement up to 40%. The reason NeuCoin chose 200 minutes and not more is that longer *modifier intervals* require longer *minimum stake age*. And as discussed above, longer *minimum stake ages* lead to lower mining participation rates.

NeuCoin is continuing to research the inter-relationship between *modifier interval* and *minimum stake age*. Its goal is to find improvements to its design that would further diminish the advantage that an attacker could gain from grinding, ideally to the point where the attacker would need the same 51% of the staked coins that he would need in any other attack vector. An additional defense mechanism NeuCoin is considering is to modify the client so that it does not accept reorganizations deeper than h hours worth of blocks. Since an attacker necessarily accumulates a significant lag when creating a fork, this would prevent the possibility of any malicious reorganization of the transaction history.[27]

Punitive duplicate stake detection mechanism

The main branch of "nothing at stake" critiques against proof-of-stake relates to a lack of security against an attacker seeking to alter transaction history. The second branch, which is more subtle, argues that proof-of-stake systems will not be able to reliably maintain consensus on transaction history. This critique contends that since miners bear no operating costs, there is nothing to prevent them from mining on multiple chains. They would do so in order to avoid the possibility of their blocks getting orphaned. If this behavior were to become widespread, it could create a situation in which many competing forks of similar length coexist, making it impossible to identify a single fork as the main chain.

To address this concern, Peercoin built a feature into its client called the "duplicate stake detection mechanism" that detects and rejects blocks mined with duplicate stakes, which works as follows:

- When a node receives a block, it checks to ensure that the proof used to create the block has not also been used to create another block previously received by the node (a "duplicate" stake). If the node detects the duplicate use of a stake, it simply discards the second block received[28].
- All nodes that use the unmodified client act this way. Therefore, a block created with a duplicate stake will not propagate through the network.

[27]The downside of this defense is that new nodes would have to rely on trusted nodes from whom they download the *block chain*. The NeuCoin Code Foundation, which is controlled by NeuCoin holders, would at least be an excellent entity to fulfill this role.

[28] Note that the reason nodes are able to detect users mining on two chains is because the proofs used to mine on both chains are the same. This is a consequence of the design of the *kernel* which doesn't include the hash of the previous block or the root of the Merkle tree.

Proof-of-stake critics make the point that since this mechanism is implemented only at the client level, nothing stops greedy nodes from running modified clients that would allow mining with duplicate stakes on all possible forks. Even though honest nodes would reject their blocks built with duplicate stakes, if other greedy nodes also modified their clients, the greedy nodes could collude by accepting each other's blocks built with duplicate stakes.

NeuCoin does not consider this scenario at all realistic. The potential gains from acting dishonestly are too miniscule, and the only way the miniscule gains are earned is when this antisocial behavior becomes widespread, which would seriously diminish the value of the currency.[29]

Nevertheless, NeuCoin solves this theoretical problem completely by using a client that **penalizes** nodes that attempt to mine using duplicate stakes. Michael Witrant (aka "sig-mike"), core developer of Peercoin and Technical Advisor to NeuCoin, has adapted Peercoin's duplicate stake detection mechanism and turned it into a **punitive** duplicate stake detection mechanism. Quite simply, when a node receives a block generated with a duplicate proof-of-stake, it discards not only the second block but also the first one received using the duplicate stake. Therefore, whenever a miner tries mining on multiple chains, all of his blocks get rejected by honest nodes. In this system, dishonestly mining on multiple chains yields no possibility of greater rewards (rather the guarantee of lower rewards).

3.3 How NeuCoin's design prevents attacks on transaction history

The proof-of-work community has dismissed proof-of-stake because its miners don't have operating costs, so they have "nothing at stake". Given this root issue, proof-of-stake critics have described four ways that attackers could alter transaction history: simple double spends, history revisions, grinding attacks, and preprogrammed attacks. Each of these attack vectors are summarized briefly below. Following the summaries, the remainder of the paper will probe each of these attack vectors in detail: explaining how they work against proof-of-stake designs in general, describing how NeuCoin's own design stands up to them, and demonstrating their odds of success with mathematical models.

The various ways that miners could try to modify transaction history in proof-of-stake can be broken down into four categories:

1. **Simple double spend**

   The widely and casually held view in the proof-of-work community - that it is "costless" to attack a proof-of-stake currency, so attackers will keep trying until they succeed - would suggest that proof-of-stake designs are vulnerable to simple double spend attacks. Proof-of-stake critics with deep knowledge of the design might quietly concede that this simple attack wouldn't work, but they have allowed the proof-of-work community to

[29] Given that proof-of-stake rewards are variable (akin to an interest rate), the only possible gain from mining on a secondary chain would be the compound interest earned on one *coinstake* reward to the next that had been lost due to a block getting orphaned. This de minimus return would only be earned in the unlikely event that a block mined on a duplicate stake were propagated to another dishonest miner who created a legitimate block on top of the illegitimate one. Critics should consider the behavior of Bitcoin miners, who could increase their chances of creating blocks that do not get orphaned by not including any transactions at all in their blocks. A 1mb block has a $\sim 13\%$ higher chance of getting orphaned than an empty one due to network lag issues[2]. To compensate, the block's transaction fees should equal at least $25 \times 13\% \sim 3.2BTC$. In reality, the number is approximately $\sim 0.5BTC$. Yet miners do not mine empty blocks because doing so would hurt the whole network and value of Bitcoin.

believe otherwise. Therefore, the double spend attack will be explained in depth first to correct this widely held misconception, and secondly to demonstrate the basic mechanics of an attack on transaction history in proof-of-stake, which remain the same in more sophisticated attacks.

2. **History revision using old private keys**

   A more serious critique concerns the possibility of rewriting transaction history using "old private keys" (technically, using stakes that are no longer unspent at present time, or in layman's terms: using coins that are no longer owned). Moreover, the old stakes could have been owned by a third party (who since sold them). The disconcerting upshot is that an attacker never needs to own the currency at all (so it really is "costless"). This section will show that when creating a fork at an earlier point in transaction history with old private keys, the attacker will be starting the attack too many blocks behind to ever be able to catch up.

3. **Grinding attack**

   The grinding attack - where the attacker uses computational power to grind through *kernels* in hopes of discovering ones that enables him to outperform the main chain - is the most serious threat to proof-of-stake coins. This paper has already discussed how Peercoin attempts to defeat this threat through the *stake modifier* and also discussed the changes that NeuCoin made to Peercoin's *stake modifier* parameters. The upcoming section on the grinding attack will show how NeuCoin's design makes grinding cost-prohibitive, requiring the grinding attacker to acquire a minimum of 30% of the staked currency in order to be successful (versus requiring 51% of the staked currency without grinding).

4. **Preprogrammed long-range attack**

   This attack involves putting together a collection of stakes that will perform very well in a specific time window in the future (for instance a year or more). This is a very potent theoretical attack against Peercoin that stems from the fact that the *stake modifier* of a given stake is static. NeuCoin's changes to the *stake modifier* parameters completely neutralize this attack vector.

3.3.1 Simple double spend

A "simple double spend" attack consists of spending coins and recovering them after the counterparty has given a product or value in exchange for the coins. To successfully commit a double spend the attacker must be able to rewrite the transaction history. Specifically, he must be able to fork the main *block chain* and extend his side branch until it is longer than the main branch which the rest of the network is working on.

A double spend attack on a proof-of-stake coin consists of the following steps, described in a way that a non-technical person can follow:

1. First of all, the attacker must own coins. A portion X of these will be used for the double spend itself while the rest (Y) will be used to fork the *block chain* and attempt to extend this side branch until it is longer than the main branch which the rest of the network is working on.

2. The portion of coins Y must be split into a very large number of smaller stakes to maximize the attacker's chances of success. (See the subsection "Role of *minimum stake age* in attacks" below for an explanation.)
3. The attacker broadcasts a transaction to the network sending X coins to the merchant he plans to attack.
4. The merchant waits for some number of confirmations to assure himself that the transaction is valid. In Bitcoin and Peercoin, the standard practice is to wait for 6 confirmations, or approximately 60 minutes, since their blocktime is 10 minutes. We will assume that the merchant waits the same 60 minutes to be assured that the transaction is valid. Since NeuCoin's *block time* is 1 minute, this equals 60 confirmations)[30].
5. While the merchant waits for his confirmations, the attacker secretly forks the main chain and begins building a side branch on the block prior to the one in which the X coins sent to the merchant was first confirmed. In his side branch, the attacker includes a conflicting transaction sending the same X coins back to himself.
6. In order for his attack to succeed, the attacker must build 61 blocks on his side branch faster than the rest of the network put together builds 60 blocks on the main chain.
   (a) To help conceptualize this task, let's assume that the attacker purchased 10% of the total amount of currency being staked by the network to use for building his branch.
   (b) Since the *block time* in NeuCoin is one minute, the target is set so that the entire network - all of the stakes being mined - should on average generate one block every one minute.
   (c) Hence, the expected time for the attacker to a generate a single block on his side branch is 10 minutes (since $Y = 10\%$).
   (d) Meanwhile, the expected time for the rest of the network (collectively staking 90% of the currency) to generate a block on the main chain is 1.1 minutes.
   (e) To summarize, the attacker must build 61 blocks faster than the rest of the network builds 60 blocks, but his expected time to build **each** block is 10 minutes versus an expected time per block of 1.1 minutes for his competitor.
   (f) Intuitively, it's clear that this is a very difficult task for the attacker, but it is hard to fathom just how difficult it is. The actual odds that the attacker will win this race are 1 chance in $\sim 10^{35}$, or 1 in 100,000,000,000,000,000,000,000,000,000,000,000[31]
7. To conclude this illustration, in the virtually impossible event that the attacker did succeed in building 61 blocks faster than the rest of the network, after the merchant sent the attacker the product, he would broadcast his side branch, invalidating the transaction he sent to the merchant and replacing it with the transaction he sent back to himself.

Given that this would-be attacker in the example above had to start his attack by purchasing 10% of the total currency supply in order to have these infinitesimal chances of success,

[30] For extremely high-value transactions, merchants would typically wait for several hours' worth of confirmations to increase their security.

[31] The formula used to produce this result is below.

it is obvious that this attack will never happen in the real world. It should also be pointed out that it's not as though the attacker had the chance to earn a huge reward.

His upside was simply to defraud a merchant of X NeuCoins (not the large amount Y used for the attack). And if the attacker were attempting to defraud the merchant of a large dollar value (say $10,000 or more), the merchant would have waited for far more than one hours' worth of confirmations. Following our example above, if the merchant waited for two hours (120 confirmations) rather than one, the attacker's odds of success would decrease from $\sim 10^{-35}$ all the way to $\sim 10^{-69}$.

To model the double spend attack mathematically[8], we consider an attacker with a portion p of the mining coins trying to attack a merchant who waits n confirmations. The probability to succeed is:

$$P[\textit{double spend}] = 1 - \sum_{i=0}^{n} \frac{\lambda^i e^{-n\frac{p}{q}}}{i!} \Big[1 - \big(\frac{p}{q}\big)^{n-i}\Big]$$

with p the portion of mining coins owned by the attacker, $q = 1 - p$ the portion of mining coins of the rest of the network, $\lambda = n\frac{p}{q}$ the expected number of blocks generated by the attacker (modeled as a Poisson process) assuming the rest of the network took the average expected time per block to create n blocks.

p	1 conf. (1mn)	10 conf. (10mn)	60 conf. (60mn)	120 conf. (120mn)
1%	0.020	1.3×10^{-16}	6.1×10^{-95}	6.8×10^{-189}
5%	0.10	1.3×10^{-09}	5.0×10^{-53}	4.4×10^{-105}
10%	0.21	1.2×10^{-06}	4.4×10^{-35}	3.5×10^{-69}
20%	0.42	0.0011	1.5×10^{-17}	3.7×10^{-34}
30%	0.63	0.042	3.7×10^{-08}	2.3×10^{-15}
40%	0.83	0.36	0.0083	0.00010
50%	1	1	1	1

Table 1: Probabilities of successfully completing a double spend in NeuCoin base on the portion of staked coins used (p) and the number of confirmations required by the attack victim.

Role of *minimum stake age* in attempts to revise transaction history

Some critics have the impression that an attacker could beat the odds presented above by sending his coins back to himself again and again until he finds parameters that enable him to mine more blocks than the main chain. These critics are neglecting to consider the *minimum stake age*.

NeuCoin's *minimum stake age* of 1.6 days requires that a given stake must remain idle for 1.6 days before its allowed to mine[32]. This foils an attacker who tries to repeatedly send

[32]In practice, this means that a proof using a timestamp within the *minimum stake age* period would produce an invalid block which would be rejected by the network

his stake back to himself searching for parameters that enable him to mine more blocks than the main chain.

The analysis above demonstrated that the odds of success decrease exponentially the larger the number of blocks that the attacker must create relative to the main chain. (For example, an attacker using 10% of the staked currency has a $\sim 10^{-6}$ chance of building 10 blocks faster than the rest of the network builds 10 blocks, but only $\sim 10^{-69}$ chance of building 120 blocks faster than the rest of the network builds 120 blocks).

Due to the *minimum stake age* , an attacker who hopes to send his coins to himself repeatedly looking for an advantage by definition starts his attack with 1.6 days disadvantage (in average 2,304 blocks) compared to the main chain. To illustrate the effect of *minimum stake age* , lets consider an attacker with 10% of the staked currency. Such an attacker would have a $\sim 10^{-35}$ chance of mining 60 blocks faster than the main chain on his first attempt, prior to re-sending his stake back to himself. On subsequent attempts however, the attacker cannot create any blocks during the first 1.6 days. Therefore, when the attacker's stake finally starts generating blocks, the rest of the network has in average generated $1.6 \times 24 \times 60 \times (100\% - 10\%) \sim 2073$ blocks. Therefore, the attacker must catch up with the main chain despite this initial lag. His chances to succeed are less than $\sim 10^{-500}$. [33]

So this simplistic version of "grinding" through stakes by re-sending them to oneself is defeated by the *minimum stake age* . The more sophisticated version of grinding not through stakes but through stake modifiers is discussed in section 3.3.3.

Besides defeating simple grinding attacks, the *minimum stake age* also has the side effect of requiring an attacker attempting a double spend to split his stake into a very large number of very small stakes. As reviewed above, the *minimum stake age* prevents a stake from mining for 1.6 days after it has generated a block. Therefore, the attacker must split his stake into:

- at least the number of confirmations used by the merchant plus one.
- to increase his odds, a much greater number than the number of confirmations so his probability to mine successfully does not drop materially after mining each block.

[33] We model this mathematically by considering an attacker attempting to reorganize the transaction history by repeatedly trying to fork the main chain and controlling a portion p of the mining coins at the beginning of the fork.

The arrival of the blocks on the fork is modelled as a Poisson process. At time t after the beginning of the attack, the expected number of blocks the attacker will have created is:

- $\frac{pt}{\tau}$ for the first fork.
- $\frac{p(t-T_{MSA})}{\tau}$ if $t \leq T_{MSA}$ (with τ the *block time* and T_{MSA} the *minimum stake age*) for all subsequent forks.

During the same time period, miners working on the main chain will have generated, on average, $(1-p)\frac{t}{\tau}$ blocks. Therefore, the probability that the attacker's fork is longer than the main chain at time t is:

For the initial fork:

$$P\Big[Poiss\big(p\frac{t}{\tau}\big) \geq (1-p)\frac{t}{\tau}\Big] = \Gamma\Big((1-p)\frac{t}{\tau}, p\frac{t}{\tau}\Big)$$

For the subsequent forks:

$$P\Big[Poiss\big(p\frac{max(t-T_{MSA},0)}{\tau}\big) \geq (1-p)\frac{t}{\tau}\Big] = \Gamma\Big((1-p)\frac{t}{\tau}, p\frac{max(t-T_{MSA},0)}{\tau}\Big)$$

where Γ is the regularized incomplete gamma function, τ is the average *block time* and T_{MSA} is the *minimum stake age*.

Consider an attacker planning a double spend attack on a merchant who waits for 60 confirmations. Now consider three scenarios:

- Scenario A: the attacker splits his stake into 100 *UTXOs*.
- Scenario B: he splits his stake into 10,000 *UTXOs*.
- Scenario C: he splits his stake into 1,000,000 *UTXOs*.

In all scenarios, his chances of generating his first block in a given timestamp are the same. This may seem counter-intuitive. It would seem that his chances of generating a block with 10,000 *UTXOs* would be higher than it would be with 100 *UTXOs*. However, one must recall that the chances of mining a block are directionally proportional to the size of the stake. So in scenario A each of the 100 *UTXOs* have a 100x greater chance of creating a block than the 10,000 *UTXOs* in scenario B.

Now consider what the scenarios would look like after the attacker has created 20 blocks: Scenario A: attacker has 80 *UTXOs* left with which he is trying to generate 40 more blocks. Note that he has 80% of his original stake available to mine. Scenario B: attacker has 9,980 *UTXOs* left with which he is trying to generate 40 more blocks. Note that he has 99.8% of his original stake available to mine. Scenario C: attacker has 9,999,980 *UTXOs* left with which he is trying to generate 40 more blocks. Note that he has 99.9998% of his original stake available to mine.

Since the probability of success decreases with the attacker's stakes that cannot mine, it is clear that the attacker will want to split his stake into a very large number of smaller stakes in order to increase his chances. That said, we can see from our example that the gains from splitting into smaller and smaller stakes quickly diminish - improving from the use of 80% of the original stake to 99.8% is much more important than the further improvement to 99.9998%.

3.3.2 History revision using old private keys

In the previous section on double spends, it was shown that even in the case where an attacker owned a significant portion of all staked coins, he still faced infinitesimal odds of success. But proof-of-stake critics have pointed out that attackers could actually attempt to rewrite transaction without owning any of the currency. All they would need to do is acquire "old private keys". In other words, they could use old stakes that were previously owned by third parties who had since sold them.

The history revision with private keys attack is in essence similar to the double spend attack. In both cases, the attacker creates a fork and attempts to extend it so that it becomes longer than the main chain.

However, in a double spend attack, the attacker attempted to rewrite history over a relatively small number of blocks, corresponding to the number of confirmations required by the merchant. When trying to rewrite history with old private keys, the attacker necessarily has to start his fork far behind because of the time it takes to obtain keys giving control over a significant portion of coins in the past. In addition, in double spend attacks where the attacker is using a stake that he actually owns (say 20% of all staked coins), the rest of the network with which he competes owns the remaining 80%. However, when using old private keys to 20% of the staked coins, the attacker is competing against not 80% of the staked coins but against 100% of them, because the attacker's old coins are now owned by new parties who mine on the main chain.

Clearly, attempting to rewrite history over a long range makes things much harder. For example, an attacker controlling private keys over 60% of the coins 2 days in the past would have $\sim 10^{-141}$ chance of ever catching up with the network. To be able to rewrite history over a significant period of time (a few days or more), the attacker actually needs to own old private keys giving control over more coins than are currently staking on the main chain.

For our mathematical model, lets consider an attacker owning old private keys giving him control over a portion p of the mining coins at a given point t minutes in the past where he starts his fork.

We model the arrival of the blocks on the attacker's fork as a Poisson process. The expected number of blocks the attacker will be able to create from the start of the block to present time is $\frac{pt}{\tau}$.

During the same period, miners working on the main chain will have generated, on average, $\frac{t}{\tau}$ blocks. Therefore, the probability that the attacker's fork is longer than the main chain at time t after the beginning of the fork is:

$$P\Big[Poiss\big(p\frac{t}{\tau}\big) \geq \frac{t}{\tau}\Big] = \Gamma\Big(\frac{t}{\tau}, p\frac{t}{\tau}\Big)$$

where Γ is the regularized incomplete gamma function and τ is the average *block time*.

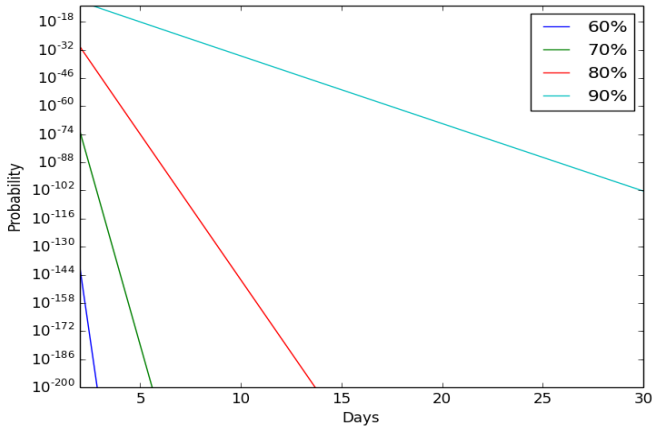

Figure 5: Probability to catch up with the main chain starting d days in the past for an attacker controlling different portions p of old private keys.

Preventing very long-range attacks using old private keys

As stated previously, can represent a threat if an attack is able to get his hands over almost all the private keys at a given point in the past. Since the percentage of coins mining is expected to decrease with time, an attacker with old private keys giving him control over 90% of the old private keys a few years in the past for example will be able to rewrite the entire history over the last three years.

An simple and elegant way to neutralize this attack vector is to forbid nodes from reverting more than a certain number of blocks. If we suppose that it would take at least a few months for an attacker to gather enough old private keys to carry such an attack, nodes could reject any reorganization deeper than say 1-month worth of blocks (43830 blocks).

Such a proof-of-stake implementation is completely secure to all types of attacks and fully decentralized (it doesnt require any kind of centrally broadcasted checkpoints). However, such an implementation differs from Bitcoin in that when setting up a new node, being aware of the protocol (i.e. "longest" chain is valid) and the set of all blocks that have been published is not enough to be able to determine which chain is the one currently accepted by the entire network. Such a node must also be aware of which chain was valid say a month in the past (this idea has been explored by Vitalik Buterin[4]).

Therefore, such an implementation does not allow "trustless bootstrapping" but once a trusted source has provided a newly entering node with the necessary information, the network functions in a fully decentralized manner.

The necessary additional information (which chain was considered valid N blocks ago) can be provided by an wide range of entities (friends, NPOs, corporations). What is critical is that such an entity cannot gain anything from providing erroneous information.

3.3.3 Grinding attack

The next idea that proof-of-stake opponents have speculated about is the possibility of gaining a significant edge against the network by "cheaply searching the blockspace to find blocks that direct history in their favor"[9]. More concretely, they refer to"grinding" through *kernels* in order to improve the performance of his stakes. This attack sounds very compelling to a proof-of-work miner because it is analogous to grinding through nonces as fast as possible in order to generate a proof-of-work.

The rationale behind this attack is that since the inputs of the *kernel* are seeded from the chain the attacker is working on, and since the attacker is working on own branch, he can shape it as he wishes and might be able to find a seed that will allow him to perform significantly better than expected.

In Bitcoin, miners "grind" through block headers **one by one**. Proof-of-stake designs use the *stake modifier* parameter - which modifies a large number of stakes - for the precise reason of preventing attackers from being able to grind through stakes one at a time, even if the attacker is alone on his branch. The only way they can grind through stakes is indirectly, through the *stake modifier*, which affects many stakes at once.

As we will see, this design feature makes the grinding attack impossible unless the attacker owns a very large portion of the total supply of coins being staked.

Step One - gaining control of the *stake modifier*

The first step of the grinding attack is for the attacker to generate blocks in order to gain influence over and eventually control the *stake modifier*. This step takes time and causes the attacker to accumulate a significant lag behind the main chain.

The process of gaining control of the *stake modifier* is very technical, difficult to express and difficult to follow. But the details do not matter. The only important takeaway is that the process takes a significant amount of time (at least hundreds of minutes).

The process is as follows:

- During the first 200 minutes[34] (the first *modifier interval*) of the attack, the attacker cannot grind through *stake modifiers*. He must use the *stake modifier* that was seeded

[34]To keep things simpler we consider that the fork starts at the beginning of a new *modifier interval*

from blocks from the main chain over the preceding *selection interval* (1.6 day, or $\sim$ 2250 minutes).

- When a new *stake modifier* is computed (200 minutes after the beginning of the attack), the new *selection interval* will encompass the 200 minutes of the attack plus the $\sim$ 2050 minutes preceding the attack. Based on how many blocks the attacker managed to generate during the first *modifier interval*, the attacker now has some chance of controlling a small minority of the 64 bits of the new *stake modifier*. He will at most control 1 bit of the *stake modifier*. (Assuming that the attacker controlled 20% of the total staked currency, he would most likely control $\sim$ 0.15 bit.) Even with 1 bit, the attacker has only gained the ability to grind through 2 *stake modifiers*. Obviously, this will not allow him to gain any advantage over the rest of the network.
- The attacker repeats the same process over the next 10 *modifier intervals* (2,000 minutes), after which the entire *selection interval* will no longer stretch further back in time than the beginning of the attack.
- The attacker is waiting to have sufficient bits under his control in order to be able to grind through a material number of *stake modifiers*. His final goal is to have control over all 64 bits. This can be done in no less time than $\sim$ 1.6' day (the *selection interval*). If the attacker owns at least $\sim$ 3% of the total staked currency, he could expect to control the *stake modifier* at the end of this 1.6 day period.

Step two - grinding through *stake modifiers*

The object of the preceding subsection was simply to demonstrate that the attacker's fork is hundreds of minutes behind the main chain.

In determining whether a grinding attack will succeed or not, it is not material whether the attacker's chain is 400 minutes behind the main chain or 2,250 minutes behind. The only way to be able to catch up from "far" behind the main chain is to be able to **consistently** outperform the rest of the network.

In the following paragraphs, we will detail the attack in order to assess what portion of the mining coins an attacker must control in order to succeed with a grinding attack. While it's obvious that an attacker controlling more than half of the mining coins would succeed, it is unclear to what extent grinding would lower this threshold. Proof-of-stake critics instinctively assume that grinding through the block space would make it possible to revise history with a minute fraction of the coins. It will be shown that a very substantial portion of the staked coin supply is necessary.

In order to describe the grinding attack, we consider an attacker working alone on a fork and therefore having complete control over the block space. This allows him to modify the hash of blocks he creates[35] in order to set the bits of a given *stake modifier* as he desires. For the sake of simplicity, we consider that the attack starts when the attacker has complete control over the current *stake modifier*.

In order to outperform the rest of the network and succeed in a grinding attack, **the attacker must be able to create more blocks than the network is expected to create during the *modifier interval*.**

Here are the steps:

[35] By incrementing the nonce for example.

1. The attacker splits his stake into a very large number of *UTXOs*.
2. The attacker will then search the *stake modifier* space and for every possible *stake modifier*, compute the hash of the *kernel* of every one of his *UTXOs*. Since the *stake modifier* is a 64-bit number, there are 2^{64} (roughly 10^{19}) possibilities for the attacker to grind through.[36]
3. If the attacker is able to find a 64-bit string that allows him to generate more blocks than the rest of the network, he succeeds. He then proceeds to create the corresponding *stake modifier*.

Grinding Illustration

Consider an attacker that owns 20% of the total supply of staked coins taking the steps listed above.

The heart of the grinding attack is that the attacker must be able to create more blocks than the network is expected to create during the *modifier interval*. If an attacker owns 20% of the staked coins, the rest of the network would be expected to generate 160 blocks (80% times the 200 blocks in the 200 minute *modifier interval*). Therefore, the attacker needs to generate 161 blocks in 200 minutes.

The attacker splits his stake into a very large number of stakes. Let's say 1 million stakes.

The attacker will then search the *stake modifier* space and for every possible *stake modifier* he computes the hash of the *kernel* of every one of his 1 million *UTXOs*. Since the *stake modifier* is a 64-bit number, there are 2^{64} (roughly 10^{19}) *stake modifiers* multiplied by 1 million *UTXOs* for the attacker to grind through.

The attacker is hoping to find a single *stake modifier* for which at least 161 of his 1 million *UTXOs* create a proof within the *modifier interval*.

With 20% of all staked coins, the attacker will on average find 40 blocks within this interval. The following graph shows the probability mass function for an attacker with 20% of the coins over the *modifier interval*.

With 20% of all staked coins, the attacker will on average find 40 blocks within this interval. The following graph shows the probability mass function for an attacker with 20% of the coins over the *modifier interval*.

[36]Grinding attacks require significant computing power in addition to large portions of the staked coins. For the following illustration, we assume that the attacker uses 10% of the hash power of Bitcoin's network. Increases above this amount of hash power provide minimal gain.

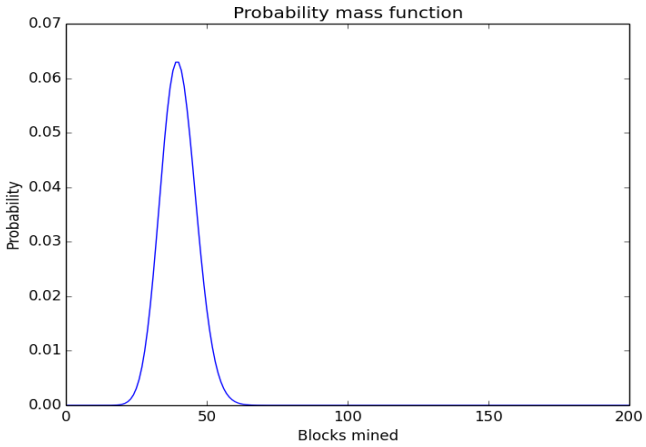

Figure 6: Probability mass function for an attacker with 20% of the mining coins over a 200 minutes time interval.

In this particular case, the probability for a given *stake modifier* to create 160 *kernels* that will mine is $\sim 10^{-46}$.

As will be shown below, given NeuCoin's choice of *modifier interval*, a grinding attack would require an attacker to own approximately 30% of the total supply of staked currency (assuming he had access to thousands of ASICs' worth of hash power).

Mathematical model of grinding attack

At every *modifier interval*, we model the arrival of the blocks on the attacking branch as a Poisson process of intensity p times the intensity of all the mining coins combined. Therefore, the probability to create more blocks than the main chain is[37]:

$$P(success)_{/trial} = \Gamma\left[(1-p)\frac{T_{modifier}}{\tau}, p\frac{T_{modifier}}{\tau}\right]$$

with p the portion of mining coins held by the attacker, Γ the regularized incomplete gamma function, $T_{modifier}$ the *modifier interval* and τ the *block time*.

Therefore, the probability for an attacker to successfully conduct a grinding attack is:

$$P[success] = 1 - \left(1 - \Gamma\left[(1-p)\frac{T_{mod}}{\tau}, p\frac{T_{mod}}{\tau}\right]\right)^{min(\frac{H_{/s}T_{mod}}{N_{stakes}}, 2^{64})}$$

with $H_{/s}$ the hash rate of the attacker, N_{stakes} the number of stakes the attacker controls (we need N much larger than $\frac{T_{mod}}{\tau}$ but we don't take into account the fact that probability to succeed goes down if N is too small), 2^{64} is size of the *stake modifier* space.

The graph below shows the probability of successfully conducting a grinding attack as a function of the portion of coins that he controls for different amounts of hash power. As we can see:

[37] We suppose that the transaction outputs used for the attack are still unspent at present time.

1. Although the grinding strategy allows to lower the cost of a 51% attack, the attacker must still own north of $\sim$ 30% to succeed. For a coin with a significant market, this provides a much greater level of security than proof-of-work, since acquiring $\sim$ 30% of the coins would be orders of magnitude more costly than buying 51% of the hash power of the network.
2. Although more hash power does decrease the percentage of coins necessary to conduct the attack, its influence is actually very limited. To draw a comparison, an attacker with $\sim$ 51% of Bitcoin's computing power would need $\sim$ 30% of the coins whereas an attacker with a single ASIC miner would need $\sim$ 34%. An attacker with a hundred times the hash power of the entire Bitcoin network would need to control $\sim$ 29% of the coins.

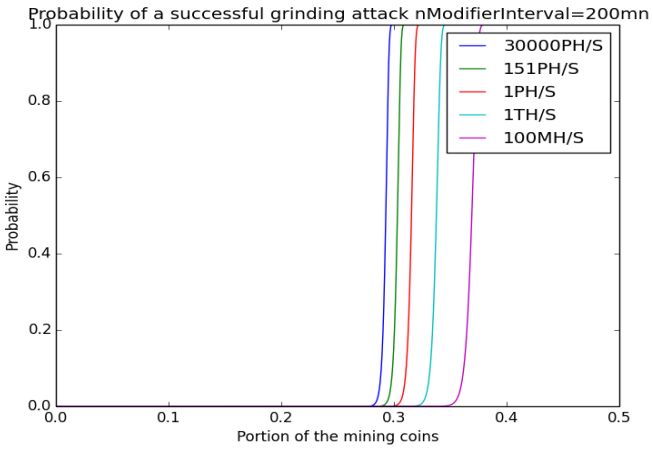

Figure 7

To conclude this discussion on grinding in proof-of-stake, here is a statistic for the critics who say it is "costless" mini to grind through the blockspace: if an attacker owned 10% of NeuCoin's staked currency and had access to all of the hash power of the Bitcoin network, his odds of success in a grinding attack would be one out of $\sim 10^{87}$. By the way, there are only $\sim 10^{80}$ atoms in the observable universe.

3.3.4 Preprogrammed, long-range attack

The final attack vector - the preprogrammed long-range attack - has an ominous name, but is not threatening at all to NeuCoin's design.

This attack involves putting together a collection of stakes that will perform very well in a specific time window in the future (potentially a year or more). This is a very potent attack vector against Peercoin due to Peercoin's use of coin age in the mining equation (which allows "supercharging" stakes) and because Peercoin's *stake modifier* yields a static value after the *selection interval*.

NeuCoin chose to use a dynamic rather than static *stake modifier* for the principal reason of neutralizing this attack vector. With a *stake modifier* that changes at every *modifier interval* (200 minutes for NeuCoin), attackers cannot pre-compute future proof-of-stakes. Without

any way to precompute future stakes, there is no way to put together a collection of stakes that perform well in a future time window, which is the essence of this attack.

The paragraphs below describe how this attack works against a proof-of-stake design with a static *stake modifier*.

In Peercoin, the *stake modifier* of a particular stake is computable after the *selection interval* and remains the same until it mines (or is used as an input in a transaction). Therefore, after the *selection interval*, all the components of the *kernel* are determined and the miner is able to predict at what time stamp the stake is likely able to mine. Although the target cannot be known precisely in advance, a miner can guess its future value with an acceptable margin of error.

In Peercoin, this allows an attacker to precompute future proof-of-stakes in order to carry out a long-range attack. The steps to conduct such an attack are as follows:

1. The attacker splits his coins into a large number of stakes and chooses a distant time window (e.g. 1 year or more in the future) during which he wants to conduct the attack.
2. After T_{SM}, the *stake modifier* is generated and the attacker can compute the hashes of all the *kernels* for time stamps included in the considered time frame.
3. The attacker keeps all the stakes that have a high probability of mining within that time frame and then resends all the remaining stakes back to himself (in order to modify their *kernels*)
4. He repeats the two previous steps repeatedly, in each cycle retaining the stakes that will perform well during the attack window, until the time of the targeted attack window.
5. Once the attack window is reached, the attacker will be able to create proofs and generate blocks with the stakes he kept. If he has a high enough number of stakes that can generate blocks (say more than 60 to fool a merchant waiting for 60 confirmations), he may be able to perpetrate an attack.

4 Conclusion

This paper has shown that NeuCoin's carefully constructed proof-of-stake design, derived from Sunny King's Peercoin, which itself was derived from Satoshi Nakamoto's Bitcoin, is secure, cost-efficient and decentralized in the long run.

The paper has also demonstrated several drawbacks of Bitcoin's proof-of-work design - including higher transaction fees in the long term, increasing centralization of mining and a divergence of interests between miners and Bitcoin holders - and shown how proof-of-stake technology completely addresses these drawbacks by introducing two crucial differences:

1. Rewarding miners based on number of coins owned rather than amount of electricity and computing resources spent - effectively replacing the operating costs of proof-of-work mining with the capital costs of holding coins;
2. Making mining rewards proportionate to number of coins held and time passed since they last generated a reward, making them akin to interest payments on the miner's coin holdings.

Since there are virtually no operating costs in proof-of-stake, transaction fees can be far lower than for proof-of-work in the long run, regardless of transaction volumes. And because all proof-of-stake miners earn the same rate of return regardless of computing hardware or electricity costs, there is no gradual centralization.

The paper also rebutted the proof-of-work community's various "nothing at stake" objections to proof-of-stake and mathematically demonstrated that all commonly cited attack vectors would fail against NeuCoin's design, which increases security relative to Peercoin and other existing proof-of-stake currencies in numerous ways, including:

1. High mining rewards, lower *minimum stake age* and omission of coin age from the mining equation all incentivize nodes to continuously stake coins over time
2. Decreased *block time*, improving user experience and enhancing security against some attack vectors
3. Causing the *stake modifier* parameter to change over time for each stake, to substantially increase security against precomputation attacks
4. Utilizing a client that punishes nodes that attempt to mine on multiple branches with duplicate stakes

As a result, NeuCoin's design solves both the mounting cost and centralization problems of proof-of-work, and the security and centralization problems with earlier proof-of-stake coins. As such, NeuCoin is the first peer-to-peer cryptocurrency, regardless of technology, that is secure, cost-efficient and decentralized in the long run.

References

[1] G. Andresen. Neutralizing a 51% attack. `http://gavintech.blogspot.fr/2012/05/neutralizing-51-attack.html`, 2012. [Online].

[2] G. Andresen. Back-of-the-envelope calculations for marginal cost of transactions. `https://gist.github.com/gavinandresen/5044482`, 2013. [Online].

[3] Bank of England. One bank research agenda, discussion paper. `http://www.bankofengland.co.uk/research/Documents/onebank/discussion.pdf/`, 2014. [Online].

[4] V. Buterin. Proof of stake: How i learned to love weak subjectivity. `https://blog.ethereum.org/2014/11/25/proof-stake-learned-love-weak-subjectivity/`, 2015. [Online].

[5] C. Decker and R. Wattenhofer. Information propagation in the bitcoin network. In *Peer-to-Peer Computing (P2P), 2013 IEEE Thirteenth International Conference on*, pages 1–10. IEEE, 2013.

[6] M. J. Fischer, N. A. Lynch, and M. Paterson. Impossibility of distributed consensus with one faulty process. *J. ACM*, 32(2):374–382, April 1985.

[7] S. King. Ppcoin: Peer-to-peer crypto-currency with proof-of-stake. June 2012. [Online].

[8] S. Nakamoto. Bitcoin: A peer-to-peer electronic cash system. *Consulted*, 1(2012):28, 2008.

[9] A. Poelstra. Distributed consensus from proof of stake is impossible. 2014. [Online].