v0.1 — 12/2015

DTB001: Decred Technical Brief

Christina Jepson[1]*

Abstract
Distributed, token-incentivized timestamping systems ledgers such as Bitcoin have become a mainstay of modern finance. We have implemented a platform combining elements of proof-of-work (PoW) and proof-of-stake (PoS) blockchains in an attempt to garner the benefits of both systems. Less major change to our consensus network include additional elliptic curve signature suites, refinement and modification of the existing scripting language, added extensibility in multiple areas to support future changes to the protocol, and distributed stake pooling.

Keywords
hybrid PoW/PoS — PoA

[1] *Company 0, Chicago, Illinois*
***Correspondence**: cjepson@decred.org

Contents

Introduction

Distributed timestamping protocols were first applied to decentralizing a financial network in the ground-breaking paper on Bitcoin by Nakamoto [1]. The field has seen explosive research follow-up from both amateurs and professionals, competing to offer extensions, adjustments, improvements, and refinements of the existing protocol. Notable implementations of new ideas include Ethereum [2], which extended scripting, CryptoNote [3], which refined privacy, and Sidechains [4], which investigated two-way pegs with 1:1 Bitcoin tokens. These protocols all utilize proof-of-work (PoW) as originally described in the Bitcoin whitepaper.

A common extension to the Bitcoin protocol modifies the consensus mechanism to either completely or partially use proof-of-stake (PoS), or the use of one's stake (tokens) rather than one's computational power to participate in the timestamping process. The first proof-of-stake blockchain based on the Bitcoin protocol was implemented in 2012 by King and Nadal [5], and includes both PoW and PoS that gradually skew towards complete PoS over time. Criticisms of pure PoS consensus systems have themselves been abundant [6][7], with the most vehement opposition coming from those working with purely PoW blockchains. The most common argument against PoS for distributed timestamping is "nothing-at-stake" or "costless simulation", describing the systematic instability resulting from stakeholders being able to generate alternatively timestamped histories with no cost to themselves.

Despite the controversy, it is apparent that systems who have a PoS overlay dependent on a PoW timestamping system may be able to independently achieve consensus. This is mathematically explored by Bentov and colleagues [8] in a paper on their scheme, proof-of-activity (PoA), and appears to be a viable extension to the PoW protocols that may enable some interesting new properties. A similar design called MC2 was earlier proposed by Mackenzie in 2013 [9]. Here we describe the construction and implementation of a similar consensus system that we have named "Decred".

1. Hybrid PoW/PoS Design

The major contrast to the follow-the-satoshi scheme previously described [8] is a new lottery system in which tickets must be purchased and then wait on a maturity period before they can be selected and spent. Selection of tickets for a block is performed lexicographically from a mature ticket pool based on pseudorandomness contained in the block header. Because manipulation of this pseudorandomness is difficult in a PoW system, manipulation of ticket selection is associated with a fundamental cost to the PoW miner. The selection of tickets over a time period can be described by a probability density function similar to the probability of obtaining a block in PoW at a constant hash rate over time at a constant difficulty [1], yielding a probability distribution with a mode approximately equal to half the ticket pool size. The price to purchase a ticket is regulated by a new stake difficulty that is determined by the exponentially weight average number of tickets purchased and the size of the mature ticket pool in prior blocks.

The validation of PoW blocks is explained by the following steps:

i. A block is mined by a PoW miner, who selects the transactions to put inside it. Stake system related transactions are inserted into the UTXO set.

ii. PoS miners vote on the block by producing a vote transaction from their ticket. The vote both enables a block to be built on top of the previous block and selects whether or not the previous regular transaction tree (containing the coinbase and non-stake related transactions) is valid.

iii. Another PoW miner begins building a block, inserting the PoS miners' votes. A majority of the votes cast must be included in the following block for that block to be accepted by the network. Of the vote transactions in this new block, the PoW miner checks a flag to see if the PoS miner indicated if the block's regular transaction tree was valid. These voting flags are tallied and, based on majority vote, a bit flag is set in this block to indicate if the previous block's regular transaction tree is valid.

iv. A nonce is found that satisfies the network difficulty, and the block is inserted into the blockchain. If the previous block's regular transaction tree was validated, insert these transactions into the UTXO set. Go to (i.).

To discourage manipulation of the included votes, a linear subsidy penalty is applied to the current block if they fail to include all the voting transactions into their block. The 'soft' penalty of invalidating the previous transaction tree helps prevent the discarding of work, which is necessary to secure the system, and makes the assumption that the next block will be obtained by a miner who is disinterested in preserving the subsidy of the former block in favour of their own. Even in the case that this is not true, a malicious miner with a high hash rate will still need at least $(\text{number for majority}/2)+1$ votes in favour of their previous block's transaction tree in order to produce a block that gives them any subsidy from the previous block.

Bit flags are explicitly added to both the block header and votes so that either miner can easily vote in new hard or soft forks.

2. Decentralized Stake Pooling

One issue arising from previous PoS designs is how to perform pooling in PoS mining analogous to PoW mining pooling. This is advantageous to PoW miner pooling as PoS miner pooling does not require dedicated hardware to mine beyond simply running a node, and, unlike PoW mining, it's unlikely that the centralization-promoting scenario will arise in which capital costs to mine increase as profit decreases. Decred solves this problem by allowing multiple inputs into a ticket purchase transaction and committing to the UTXO subsidy amount for each input proportionally, while also committing to a new output public key or script for these proportional rewards. The subsidy is then given to those generating the ticket in a trustless manner, and the ticket can be signed round robin before submission to the network. Importantly, control over the production of the vote itself is given to another public key or script which can not manipulate the subsidy given to the recipients. Production of the vote in a distributed manner can be achieved by using a script in the ticket that allows for multiple signers.

3. Minor Design Elements

3.1 Elliptic curve signature algorithms

Although secp256k1 is widely considered to have a secure choice of elliptic curve parameters, some questions about the origin of the curve remain. For example, the selection of the Koblitz curve $(y^2+xy=x^3+ax^2+b$ and $a=a^2$, $b=b^2$; $a=1$ or 2, $b!=0)$ is typically done by enumerating the binary extension Galois fields $GF(2^m)$ where m is a prime in the range $\{0$, ..., higher limit$\}$ and $x,y \in GF(2^m)$ [10]. For 128-bit security, m is required to be $\geqslant 257$ and typically the smallest prime possible in this range to facilitate fast calculation. In this case, the obvious choice for m is 277, $a=0$; despite the existence of this appropriate m value being known by the curators of the curve parameters [11] and the fact that it was the most computationally efficient solution, the parameters $m=283$ and $a=0$ were selected out of three possible options $(m=277,\ a=0;\ m=283,\ a=0;\ m=283,\ a=1)$. For all other Koblitz curve specifications, the most obvious m value is selected. Although this is curious, there are no known attacks that can be applied by using a slightly larger m value for the Galois field. Other objections to the parameters used by secp256k1 have also been raised [12].

Another extremely popular digital signature algorithm (DSA) with 128-bits of security is Ed25519 [13]. This uses the EdDSA signing algorithm over a curve birationally equivalent to Curve25519 and is widely employed today. Unlike secp256k1's ECDSA, Ed25519 uses simpler Schnorr signatures that are provably secure in a random oracle model [Appendix A].

Schnorr signatures have also been proposed for Bitcoin [14]. However, instead of using an OP code exclusive to Schnorr signatures utilizing the curve parameters for secp256k1, Decred instead uses a new OP code OP_CHECKSIGALT to verify an unlimited number of new signature schemes. In the current implementation, both secp256k1 Schnorr signatures and Ed25519 signatures are available to supplement secp256k1 ECDSA signatures. In the future, it is trivial to add new signature schemes in a soft fork, such as those that are quantum secure. Having these two Schnorr suites available also allows for the generation of simple group signatures occupying the same space of a normal signature [15], which for both is implemented. In the future, threshold signatures using dealerless secret sharing will also enable t-of-n threshold signatures occupying the same amount of space [16].

3.2 Hash function

SHA256, used in Bitcoin, has a number of technical shortcomings due to its Merkle–Damgård construction. These vulnerabilities led to the SHA3 competition for a new hash function based on a different fundamental construction. Decred has chosen BLAKE256 as its hash function, a finalist for the competition [17][18]. The hash function is based around a HAIFA construction that incorporates a variation of the ChaCha stream cipher by Bernstein. The hash function is notable for its high performance on x86-64 microarchitecture, being faster for short messages than SHA256 [19] despite being considered to have a much higher security margin at 14-rounds.

3.3 Script extensions

Aside from the previously mentioned OP_CHECKSIGALT and OP_CHECKSIGALTVERIFY, other modifications to Bitcoin scripting have been made. A version byte has been added to output scripts to enable simple soft forking to new scripting languages, as first suggested by Wuille [20]. All math and logic related OP codes have been re-enabled and now operate on int32 registers. Various byte string manipulation OP codes have also been implemented and re-enabled. The remaining unused Bitcoin OP codes have been repurposed for future soft forks. Some longstanding bugs in the Bitcoin scripting language have been also been fixed [21][22].

3.4 Signature script isolation and fraud proofs

To prevent transaction malleability, the ability to generate a transaction with the same input references and outputs and yet a different transaction ID, input scripts have been removed from the calculation of the transaction hash. The origins of this modification have been controversial, although it appears to have been implemented in both CryptoNote coins and sidechains in the past [3][23]. It is now being proposed for Bitcoin as a soft fork referred to as "Segregated Witness" [20]. As in the Elements sidechains implementation, commitments to the witness data are included in the merkle tree of the block [23]. In addition, fraud proofs, as suggested for Bitcoin's soft fork [20], are set by miners and also committed to as part of the data in the merkle tree.

3.5 Transaction extensions

Transaction expiry has been added, which allows one to prune transactions from the memory pool if the blockchain has reached a certain height [24]. Previously the only way to remove a transaction from the mempool was to double spend it.

3.6 Miscellaneous improvements

As in Bitcoin, subsidy decays exponentially with block height. However, Decred's algorithm, though also extremely simple, better interpolates this decay over time so as not to produce market shock with steep subsidy drops similar to CryptoNote [2]. Like PeerCoin [5], the PoW difficulty is calculated from the exponentially weighted average of differences in previous block times. However, this calculation is also interpolated into Bitcoin-like difficulty window periods. The "timewarp" bug in Bitcoin is corrected [25], by ensuring that every difference in block time in incorporated into the difficulty calculation.

It should also be noted that many well known mining attacks, such as selfish mining [26] and stubborn mining [27], will no longer function advantageously in a system where there is effective decentralization of stake mining and no PoW-PoS miner collusion. This is simply because it is impossible to generate secret extensions to blockchains without the assistance of stake miners.

Resilience to previously described mining attacks, and newly conceived mining attacks specific to our system, will be a fruitful area for future research.

References

[1] Nakamoto N. 2008. Bitcoin: A Peer-to-Peer Electronic Cash System. Self-published.
https://decred.org/research/nakamoto2008.pdf

[2] Buterin V. 2014. A Next-Generation Smart Contract and Decentralized Application Platform. Self-published.
https://decred.org/research/buterin2014.pdf

[3] von Saberhagen N. 2013. CryptoNote v 2.0. Self-published.
https://decred.org/research/saberhagen2013.pdf

[4] Back A., Corallo M., Dashjr L., Friedenbach M., Maxwell G., Miller A., Poelstra A., Timon A., Wuille P. 2014. Enabling Bitcoin Innovations with Pegged Sidechains. BlockStream.
https://decred.org/research/back2014.pdf

[5] King S. and Nadal S. 2012. PPCoin: Peer-to-Peer Crypto-Currency with Proof-of-Stake. Self-published.
https://decred.org/research/king2012.pdf

[6] Bentov I., Gabizon A., Mizrahi A. 2015. Cryptocurrencies without Proof of Work. arXiv Cryptography and Security.
https://decred.org/research/bentov2015.pdf

[7] Poelstra A. 2015. On Stake and Consensus. Self-published.
https://decred.org/research/poelstra2015.pdf

[8] Bentov I., Lee C., Mizrahi A., Rosenfeld M. 2014. Proof-of-Activity: Extending Bitcoin's Proof of Work via Proof of Stake. Proceedings of the ACM SIGMETRICS 2014 Workshop on Economics of Networked Systems, NetEcon.
https://decred.org/research/bentov2014.pdf

[9] Mackenzie A. 2013. MEMCOIN2: A Hybrid Proof-of-Work, Proof-of-Stake Crypto-currency. Self-published.
https://decred.org/research/mackenzie2013.pdf

[10] Pornin T. 2013. StackExchange Cryptography: Should we trust the NIST-recommended ECC parameters?
https://decred.org/research/pornin2013.pdf

[11] Solinas J. 2000. Efficient Arithmetic on Koblitz Curves. Designs, Codes and Cryptography. 19(2):195-249.
https://decred.org/research/solinas2000.pdf

[12] Bernstein D. and Lange T. 2014. SafeCurves: choosing safe curves for elliptic-curve cryptography.
http://safecurves.cr.yp.to

[13] Bernstein D., Duif N., Lange T., Schwabe P., Yang B. 2012. High-speed high-security signatures. Journal of Cryptographic Engineering. 2:77-89.
https://decred.org/research/bernstein2012.pdf

[14] Osuntokun O. 2015. OP_SCHNORRCHECKSIG: Exploring Schnorr Signatures as an Alternative to ECDSA for Bitcoin. Self-published.
https://decred.org/research/osuntokun2015.pdf

[15] Petersen T. 1992. Distributed Provers and Verifiable Secret Sharing Based on the Discrete Logarithm Problem. Aarhus University Ph.D. Thesis. 55-57.
https://decred.org/research/petersen1992.pdf

[16] Stinson D. and Strobl R. 2001. Provably Secure Distributed Schnorr Signatures and a (t,n) Threshold Scheme for Implicit Certificates. Certicom Corporation.

[17] Aumasson J., Henzen L., Meier W., Phan R. 2010. SHA-3 Proposal BLAKE. Self-published.
https://decred.org/research/aumasson2010.pdf

[18] Aumasson J., Henzen L., Meier W., Phan R. 2014. The Hash Function BLAKE. Springer-Verlag Berlin Heidelberg.

[19] Bernstein D. and Lange T. eBACS: ECRYPT Benchmarking of Cryptographic Systems.
http://bench.cr.yp.to

[20] Wuille P. 2015. Segregated Witness for Bitcoin. Scaling Bitcoin Hong Kong.
https://prezi.com/lyghixkrguao/segregated-witness-and-deploying-it-for-bitcoin/

[21] Todd P. The difficulty of writing consensus critical code: the SIGHASH_SINGLE bug. Bitcoin-development mailing list.
https://decred.org/research/todd2014.pdf

[22] Franco P. Understanding Bitoin, 6.3: Multisignature (M-of-N) Transactions. John Wiley Sons Inc. p. 84.

[23] Maxwell G. 2015. Bringing New Elements to Bitcoin with Sidechains. SF Bitcoin Devs Meetup.
https://decred.org/research/maxwell2015.pdf

[24] ByteCoin. 2010. Need OP_BLOCKNUMBER to allow "time" limited transactions.
https://decred.org/research/bytecoin2010.pdf

[25] ArtForz. 2011. Re: Possible way to make a very profitable 50 plus ish attack for pools? Bitcointalk Bitcoin Forums.
https://decred.org/research/artforz2011.pdf

[26] Eyal I. 2015. The Miner‘s Dilemma. In IEEE Symposium on Security and Privacy, 2015.
https://decred.org/research/eyal2015.pdf

[27] Nayak K., Kumar S., Miller A., Shi E. 2015. Stubborn Mining: Generalizing Selfish Mining and Combining withan Eclipse Attack. Cryptology 2015/796.
https://decred.org/research/nayak2015.pdf

[28] Wuille P. 2015. Tree Signatures: Multisig on steroids using tree signatures.
https://decred.org/research/wuille2015.pdf

Appendix A: Schnorr Multisig

Schnorr signatures have been proposed for Bitcoin. They have also been used extensively in other cryptocurrencies, such as Nxt and CryptoNote coins. In the simplest case, a Schnorr signature ECDSA cryptosystem can be described as follows:

- $y = xG$ where y is the public key point on the curve, x is the private scalar, G is the curve generator.
- $r = kG$ where r is the point on the curve resulting from the multiplication of k, the nonce scalar, by the generator.
- $h = H(M||r)$ where H is a secure hash function, M is the message (usually a 32 byte hash), and r is the encoded point previously described. $||$ denotes concatenation.
- $s = k - hx$ where s is the scalar denoted from $k - hx$.
- The signature is (r, s), and verification is simply $H(M||r) == hQ + sG$.

In the above, multiplications by a capital letter (e.g., kG) are point multiplications by a scalar, and so always result in a point on the curve. Addition of these points results in another point. Additions and multiplications of scalars amongst themselves is the same as regular multiplication you would do with any integer. It's important to note that multiplying a point by a scalar is considered an irreversible step, because the calculation of the scalar from the new point defaults to the discrete logarithm problem.

From the above it is clear that r is a point on the curve, while s is a scalar. Consider the group of signers represented by $x\_sum = x_1 + ... + x_n$ with nonces $k\_sum = k\_1 + ... + k\_n$. The public key for the private scalar sum would be: $y = x\_sumG$. The signature for these sums (from all group participants) would be: $r' = k\_sumGs' = k\_sum - hx\_sum$. To generate this signature all participants would have to share their private key and nonces beforehand. We want to obviously avoid this, so instead let us have each participant create a partial signature. $r\_n = k\_1G + ... + k\_nG = r'$ (the sum of the public nonce points, which the participants may freely individually publish) $s\_n = k\_n - hx\_n$. Substituting this into the general formulas for signatures and using point or scalar addition: $r = r\_n = r'(thesameasabove)s = s\_1 + ... + s\_n = s'$ (simple scalar addition; it must be true that $(k\_1 - hx\_1) + ... + (k\_n - hx\_n) = s_1 + ... + s\_n = s'$). Doing an m-of-n signature is non-trivial. It has been suggested that a merkle tree containing all possible public key sums for m participants be used for these cases, generating a $log(n)$ sized signature [28].