Transaction Locking and Masternode Consensus:

A Mechanism for Mitigating Double Spending Attacks

Document version: 2, published September 22nd, 2014

Evan Duffield - evan@darkcoin.io
Holger Schinzel - holger@darkcoin.qa
Fernando Gutierrez - gutierrezf@gmail.com

Abstract. *Bitcoin and other cryptographic currencies use a distributed system called the blockchain to gain consensus across the network [Nielsen13]. To protect against double spending attacks, vendors and other merchants usually wait for confirmation from a block that the transaction being sent was valid. A double spend is where an attacker sends two competing transactions, one of them promised to a merchant (or any other party), the other back to himself.*

In Bitcoin a standard confirmation event on average takes 10 minutes. Depending on the needed level of security required, a merchant can require multiple of these events, taking between 30 and 60 minutes in total.

In this paper, we explore a solution to a long standing issue with Bitcoin and other cryptographic currencies, the ability to enable instant validation of payments without having to wait for blockchain confirmation.

1 Introduction

Invented in 2009, Bitcoin [Nakamoto09] is a decentralized peer-to-peer payment system created by Satoshi Nakamoto. Bitcoin has been steadily gaining popularity since it's introduction in 2009 and has been adopted successfully by many merchants [Reuters14]. While Bitcoin has been greatly successful, it has one main significant disadvantage to it's largest competitor, credit cards. In point-of-sale transactions, credit cards can have nearly instantaneous authorization of payment, whereas to get finality in a Bitcoin transaction one must wait for blockchain confirmation. With Credit Cards, an authority is contacted to when making a purchase that results in a buyers money being held until it clears later. In contrast, Bitcoin clients blindly propagate messages they believe are correct while getting no feedback from the network.

Darkcoin is a privacy-centric crypto-currency based on the work of Satoshi Nakamoto and includes various improvements to the technology first implemented in the Bitcoin client. These improvements include enhanced privacy features and a network that is incentivized to provide services [Duffield14].

In this paper we will introduce the masternode network as observer network, utilising a distributed consensus and locking algorithm “TX locking” to secure unconfirmed transactions. The observer network reports on transactions granting they a finalized status immediately after their original propagation. Further we will discuss attack vectors and how the masternode network will mitigate these.

2 Masternode Network

Masternodes were originally introduced to Darkcoin as engineering effort to support the mixing process used in Darkcoins’ DarkSend implementation. The original requirements were described in April 2014 by Evan Duffield:

“These nodes are the foundation of DarkSend, all transactions will be routed through these nodes. Each masternode requires that 1000DRK be kept on the node and each time that node is selected the network will dedicate 10% [As of this writing, the reward has been changed to 20%] of that block to these nodes. If you are running a masternode you need to be fairly familiar with network administration and securing your host.” [Masternodes]

When running a Masternode, users store the Darkcoin as something akin to collateral, although unlike traditional collateral, the Darkcoin never leaves the user’s possession and has no chance of being forfeited. It can be moved or spent at any time by the user - doing so simply removes the Masternode from service and makes it ineligible to receive rewards.

An addition to the core protocol is made to support a second P2P network, which propagates messages synchronizing a list of all known Masternodes across the network. In result, all clients on the network know about all Masternodes and can utilize their services at any time.

Unlike Gnutella [Gnutella] which uses a hierarchical network of client-node and super-nodes, where client-nodes only make a connection to one super-node (figure 1), Masternodes and normal peers are equal in their connection behaviour, forming a classical P2P network.

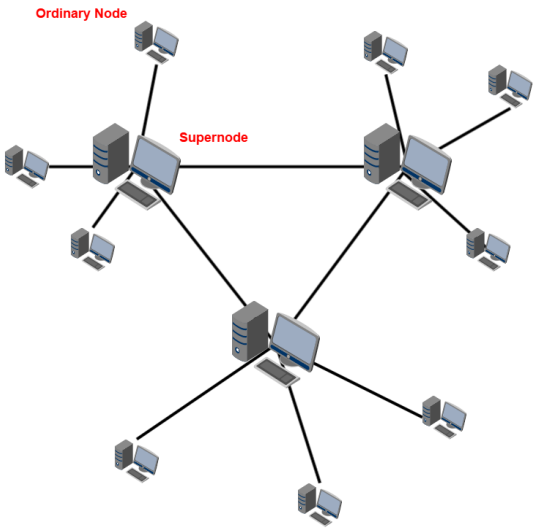

Figure 1: A P2P supernode network

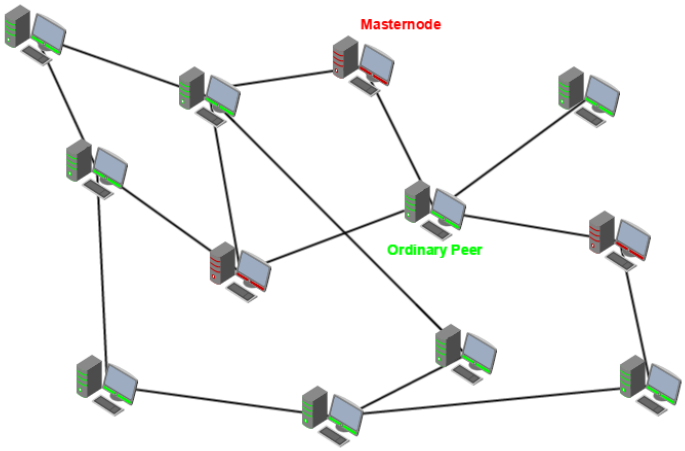

Figure 2: Darkcoin Masternode Network

Running the appropriate peer software (“wallet”) and matching the requirements (static IP, 1000 DRK vin) actually each node/peer can turn into a masternode.

Albeit the original intention of Masternodes was to facilitate the mixing of coins, having a network of incentivized peers opens the possibility for further applications.

3 Transaction Locking

In large scale distributed systems (like Darkcoin) it is a common problem how to ensure that only one peer across a large number of peers acts on a resource (coins). Solutions to that challenge involve different kinds of consensus algorithms like e.g. Paxos [Chandra07].

Bitcoin uses proof of work to maintain consensus throughout its network of peers. Due to its technical parameters this limits the speed at which a transaction can be considered confirmed and safe against double spend attacks.

To decrease the time a transaction needs to be confirmed it's possible to lower the block generation time. which has the drawback of blockchain bloat and has a lower boundary of ~30 seconds due to network latency.

We are proposing to combine the proof of work algorithm with an implementation of a distributed lock manager (DLM) which will utilise the masternode network: Transaction Locking.

In contrast to Chubby [Burrows06] which is providing locks on file resources, we will be implementing a framework for locking Darkcoin inputs.

Locally when using Darkcoin, the client can lock inputs in the wallet from being used elsewhere. In most cases this is done in specific implementations that use the RPC API of the client to make manual transactions.

The concept of transaction locking can be further extended to lock inputs across the entire network, rather than just locally like most crypto-currency implementations. Such an implementation must overcome consensus issues and race conditions to successfully stop double spending attacks.

3.1 Solution To Double Spending via Transaction Locking

In most implementations it is recommended that merchants have some form of double spending protection. This can be accomplished by having clients acting as an observer on the network and reporting back to the merchant when they see double spending attacks [Karame12]. In our solution we propose using the masternode network as observers and extending the protocol to give a set of masternodes the ability to be the authority on transactions.

Transaction locking is a concept where a client sends the network an intention to lock funds from a specific input to a specific output (or multiple of each). This is done by relaying an object consisting of a full transaction and the locking command. The user will sign a message using the input(s), and relay the message throughout the network.

Transaction Lock: ("txlock", CTransaction, nBlockHeight, Signed Message)

Locking messages will propagate across the whole Darkcoin network and reach all clients. Once the lock has reached everyone, a set of deterministically selected masternodes will form a consensus. Next, upon a successful consensus, a message will be broadcasted across the network and at this point all clients will respect the lock on the funds.

3.1 Masternode Locking Authority and Consensus

By utilizing the masternode network, we can gain a degree of certainty that the transaction in question is valid and will be accepted into the blockchain after that. Immediately after the propagation of a lock, the selected masternodes will begin to vote on the validity of the transaction lock.

If consensus is reached on a lock by the Masternode network, all conflicting transactions would be rejected thereafter, unless they matched the exact transaction ID of the lock in place. Clients would be tasked with clearing out conflicting locks and possibly reversing attacker transactions. This would only happen in a case where an attacker submitted multiple locks to the network at once and the network formed consensus on one but not the other.

If no consensus is reached, standard confirmation will be required to assure that a transaction is valid.

3.2 Election Algorithm and Voting

A special deterministic algorithm is used to determine a pseudo-random ordering of the masternodes. By using the hash from the proof-of-work for each block, security of this functionality will be provided by the mining network.

Pseudo Code, for selecting a masternode:

```
For(mastenode in masternodes){
    n = masternode.CalculateScore();

    if(n > best_score){
        best_score = n;
        winning_node = masternode;
    }
}

CMasterNode::CalculateScore(){
   n1 = GetProofOfWorkHash(nBlockHeight); // get the hash of this block
   n2 = x11(n1); //hash the POW hash to increase the entropy
   n3 = abs(n2 - masternode_vin);

   return n3;
}
```

In each round of voting, a winning Masternode is chosen to carry out Darksend transactions. This process is carried out by the individual nodes across the network independently using the Masternode election algorithm. This algorithm chooses a winning node for Darksend, but there is also a runner up, third, fourth, fifth place, etc.

Utilizing this code, we can make a deterministic list of the Masternodes that will act as the authority for the transaction lock. These will be the same nodes across the network and they will vote on the validity of the transaction lock in question. For each block, a completely different list of 10 nodes will be chosen to be the authority.

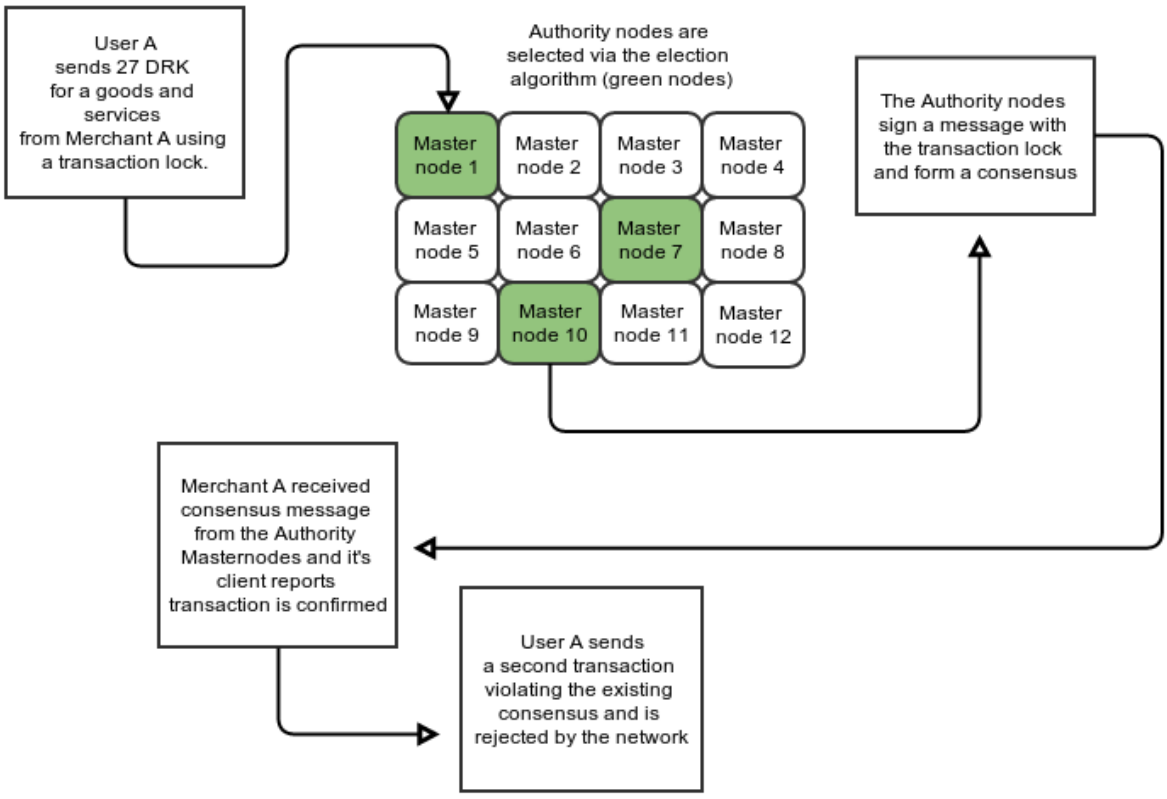

Figure 3: Gaining Consensus on Transaction Locks via the Masternode Network

3.3 An Example Transaction

1. User A sends a transaction for 27 DRK for a widget from merchant B using a “locked transaction” message.
2. The transaction is propagated to the whole network and eventually reach a set of elected authority nodes.
3. The authority nodes collectively send messages to the network , forming a consensus about the validating the transaction and each sign a "consensus transaction" message, which is propagated to the network.
4. When a node sees all consensus messages, they can consider the transaction confirmed.

4 Security

In order to secure the network from attack, we must mitigate attacks such as:

- Sybil attack
- Finney attack
- Transaction lock race attack
- Multiple consensus messages

4.1 Attacking The Consensus System via Sybil attack

The probability of winning the election will be 1 in N Masternodes. Currently the network is supported by 895 Masternodes. Each Masternode has a probability of 1 in N of winning the election. Therefore to attack the network, it will require the election process to select all of the attackers Masternodes.

We will consider attack on the transaction locking system by purchasing masternodes in order to rig the voting system. For simplicity we will use a network consisting of 1000 masternodes. Currently the Darkcoin network has 895 active Masternodes.

Probabilities of attack can be calculated by the chance of a masternode being selected as the winning node for a given block (1/1000). To subvert the system an attacker would require operating all ten masternodes that won a given election.

At a cost of 1000DRK per masternode, it's expensive to attempt to attack the transaction locking system. To gain a probability of 1.72% of being selected for a specific block, one has to control ⅔ of the Masternode network (see Table 1 for more information). To gain control of ⅔ of the network, an attacker would need to purchase 2000 masternodes (requiring the purchase of 2 million Darkcoin).

Attacker Controlled Masternodes / Total Masternodes	Probability of success $\prod_{i=1}^{n}((r-(i-1))/(t-(i-1)))$	Darkcoin Required
10/1010	3.44e-24	10,000DRK
100/1100	2.52e-11	100,000DRK
1000/2000	9.55e-03	1,000,000DRK
2000/3000	1.72e-02	2,000,000DRK

Table 1. The probability of a successful attack given the attacker controls N Nodes.

Where:
n is the length of the chain of masternodes
t is the total number of masternodes in the network
r is the number of rogue masternodes controlled by the attacker and it is $\geq n$
The selection of masternodes is random

Considering the limited supply of Darkcoin (4.6 million at the time of writing) and the low liquidity available on the market, it becomes an impossibility to attain a large enough to supply to succeed at such an attack.

In the case of an attacker attempting to rig the voting system in favor of the wrong transaction lock (i.e. the lock that isn't propagated across the rest of the network), the network will form an irreversible lock causing the transaction to the merchant to be invalidated. The merchants client in question will permanently show an unconfirmed transaction due to a double spend and will never show the transaction was instantly validated.

4.2 Finney Attacks

In a Finney Attack[9], an attacker mines a blocks normally, in the block he is trying to mine he includes a transaction where he sends coins back to himself. When he successfully finds a block, he does not broadcast the block, but instead he sends coins to a merchant for goods or services. Immediately after the goods or services have been produced and before the network has produced the next block the attacker broadcasts his block overriding the payment he just made.

To stop a Finney Attack from succeeding, the network must be capable of rejecting blocks that violate existing transaction locks. They must also be able to differentiate between a transaction lock on a given transaction and a successfully locked transaction via the Masternode network locking consensus system. Only when the elected masternodes have relayed the lock for the given transaction is it to be considered successfully locked and a block with a conflicting transaction rejected.

4.3 Transaction Locking Race Attack

In a transaction locking race attack a client would submit two competing locks to the network. One promising money to the merchant and the other to himself. To improve the probability of a successful attack, the attacker would submit a transaction locking command directly to the elected masternodes making sure they propagate that the merchant will receive the money while at the same time propagating a competing lock to send the money back to their own wallet.

In an attack like this one, the network would be split between two valid transactions until the winning masternodes propagated their votes for the correct lock. All clients on the network would then remove the invalid transaction and take the valid one into their memory pool. This would happen very quickly, in the matter of a few seconds in most cases.

4.4 Incomplete Locks

An incomplete lock happens when the Masternode network lacks consensus about a specific lock. A lack of consensus could happen in rare cases such as a rogue masternode that refuses to vote when it has a consensus task or loss of network traffic. In cases like these, no finalized lock will be formed and the network will gain consensus via standard confirmation.

4.5 Multiple Consensus Messages

If attackers gain control of the 10 Masternodes for a given block and propagate multiple conflicting messages, the network must appropriately handle the conflict. For example, an attacker that controls a large portion of masternodes might propagate a message to Merchant B and nowhere else ,while propagating a messages to many other nodes spending the inputs back to himself.

In this case it is suggested that conflicting messages will cancel each other out and clients wait for normal block confirmation.

5 Further work

Many impressive features become possible after implementing the transaction locking system and consensus system into the Darkcoin network. These include a completely backwards compatible architecture and instantaneous transactions from client-to-client without waiting for a confirmation.

5.1 Transaction Lock Compatibility Mode

To enable backwards compatibility with all existing software (exchanges, pools, etc), clients will default to showing 24 hours of confirmations of transactions that have been successfully locked. This will provide all services using Darkcoin to benefit from instant transactions without having to implement anything specific.

If a client needs the daemon to function in the old way, there will be a flag to disable this mode.

5.2 No Wait Client-To-Client Transactions

In a normal situation, after a client receives new funds to a wallet, he will have to wait for one block confirmation in order to spend any of the newly available funds. When instant validation is implemented the client will react as though it has full confirmation of a transaction and allow the sending of funds with no risk to the user. This will allow a series of transactions to happen before a block event on the network using the same inputs.

6 Conclusion

Bitcoin and crypto-currencies rely heavily on confirmation through mining to stop double spending attacks. Although, a huge accomplishment in technology, it fails to compete with the near instant transaction speed of credit cards due to their use of a centralized authority.

Fast validation of payments via transaction locking and Masternode consensus could be used to avoid having to wait for confirmation via a new block and reach speeds nearly as fast as credit cards. In most cases a transaction should be validated by the network within a few seconds of originally being broadcasted.

Clients will respect the authority of the masternode network and as a result the network can come into consensus without a block event happening.

By using the Masternode network as an authority and selecting Masternodes via a deterministic algorithm powered based on the proof-of-work, we gain a system that gives us comparable transaction time to a credit card transactions while also being tamper resistant, backwards compatible and secure.

Revision History

Version 2

- Removed the section “Blockchain Size Considerations” due to some feedback leading from users. A better method of reducing the blockchain size in the future would be Blockchain pruning.
- Added some information about credit card authorizations and the analogy we’re trying to making between authorizations and the feedback from the consensus network.

Version 1

- Initial release

References

[Nakamoto09] Satoshi Nakamoto (2009), Bitcoin: A Peer-to-Peer Electronic Cash System
https://bitcoin.org/bitcoin.pdf

[Reuters14] Reuters (2014), Analysis - Bitcoin shows staying power as online merchants chase digital sparkle
http://uk.reuters.com/article/2014/08/28/uk-usa-bitcoin-retailers-analysis-idUKKBN0GS0AQ20140828

[Karame12] Ghassan O. Karame, Elli Androulaki, Srdjan Capkun (2012): Two Bitcoins at the Price of One? Double-Spending Attacks on Fast Payments in Bitcoin
https://eprint.iacr.org/2012/248.pdf

[Duffield14] Evan Duffield (2014): Darkcoin: Peer-to-Peer Crypto-Currency with Anonymous Blockchain Transactions and an Improved Proof-of-Work System
https://www.darkcoin.io/downloads/DarkcoinWhitepaper.pdf

[Masternodes14] Evan Duffield (2014):
https://darkcointalk.org/threads/darkcoin-update-masternode-requirements-masternode-payments.225/

[Lo14] Stephanie Lo, J. Christina Wang (2014) Bitcoin as Money?
http://www.bostonfed.org/economic/current-policy-perspectives/2014/cpp1404.pdf

[Nielsen13] Michael Nielsen, How the Bitcoin protocol actually works
http://www.michaelnielsen.org/ddi/how-the-bitcoin-protocol-actually-works/

[Gnutella03] Chawathe et. al. (2003), Making Gnutella-like P2P Systems Scalable
http://www.cs.cornell.edu/people/egs/cornellonly/syslunch/fall03/gnutella.pdf

[Chandra07] Chandra et. al. (2007), Paxos Made Live - An Engineering Perspective
http://static.googleusercontent.com/media/research.google.com/en//archive/paxos_made_live.pdf

[Burrows06] Mike Burrows (2006), The Chubby lock service for loosely-coupled distributed systems
http://static.googleusercontent.com/media/research.google.com/en//archive/chubby-osdi06.pdf