Bitcoin: A Peer-to-Peer Electronic Cash System

Satoshi Nakamoto
satoshin@gmx.com
www.bitcoin.org

Abstract. A purely peer-to-peer version of electronic cash would allow online payments to be sent directly from one party to another without going through a financial institution. Digital signatures provide part of the solution but the main benefits are lost if a trusted third party is still re!uired to prevent double-spending. " e propose a solution to the double-spending problem using a peer-to-peer network. #he network timestamps transactions by hashing them into an ongoing chain of hash-based proof-of-work forming a record that cannot be changed without redoing the proof-of-work. #he longest chain not only serves as proof of the se!uence of events witnessed but proof that it came from the largest pool of $%& power. As long as a ma'ority of $%& power is controlled by nodes that are not cooperating to attack the network they(ll generate the longest chain and outpace attackers. #he network itself re!uires minimal structure.) essages are broadcast on a best effort basis and nodes can leave and re'oin the network at will accepting the longest proof-of-work chain as proof of what happened while they were gone.

1. Introduction

$ommerce on the *nternet has come to rely almost exclusively on financial institutions serving as trusted third parties to process electronic payments. " hile the system works well enough for most transactions it still suffers from the inherent weaknesses of the trust based model. $ompletely non-reversible transactions are not really possible since financial institutions cannot avoid mediating disputes. #he cost of mediation increases transaction costs limiting the minimum practical transaction si+e and cutting off the possibility for small casual transactions and there is a broader cost in the loss of ability to make non-reversible payments for non-reversible services. " ith the possibility of reversal the need for trust spreads.) erchants must be wary of their customers hassling them for more information than they would otherwise need. A certain percentage of fraud is accepted as unavoidable. #hese costs and payment uncertainties can be avoided in person by using physical currency but no mechanism exists to make payments over a communications channel without a trusted party.

" hat is needed is an electronic payment system based on cryptographic proof instead of trust allowing any two willing parties to transact directly with each other without the need for a trusted third party. #ransactions that are computationally impractical to reverse would protect sellers from fraud and routine escrow mechanisms could easily be implemented to protect buyers. *n this paper we propose a solution to the double-spending problem using a peer-to-peer distributed timestamp server to generate computational proof of the chronological order of transactions. #he system is secure as long as honest nodes collectively control more $%& power than any cooperating group of attacker nodes.

2. Transactions

"e define an electronic coin as a chain of digital signatures. -ach owner transfers the coin to the next by digitally signing a hash of the previous transaction and the public key of the next owner and adding these to the end of the coin. A payee can verify the signatures to verify the chain of ownership.

#he problem of course is the payee can(t verify that one of the owners did not double-spend the coin. A common solution is to introduce a trusted central authority or mint that checks every transaction for double spending. After each transaction the coin must be returned to the mint to issue a new coin and only coins issued directly from the mint are trusted not to be double-spent. #he problem with this solution is that the fate of the entire money system depends on the company running the mint with every transaction having to go through them 'ust like a bank.

!. Proo"-o"-#or$

#o implement a distributed timestamp server on a peer-to-peer basis we will need to use a proof-of-work system similar to Adam 3ack(s 4ashcash /50 rather than newspaper or &senet posts. #he proof-of-work involves scanning for a value that when hashed such as with S4A-125 the hash begins with a number of +ero bits. #he average work re!uired is exponential in the number of +ero bits re!uired and can be verified by executing a single hash.

.or our timestamp network we implement the proof-of-work by incrementing a nonce in the block until a value is found that gives the block(s hash the re!uired +ero bits. 6nce the $%& effort has been expended to make it satisfy the proof-of-work the block cannot be changed without redoing the work. As later blocks are chained after it the work to change the block would include redoing all the blocks after it.

#he proof-of-work also solves the problem of determining representation in ma'ority decision making. *f the ma'ority were based on one-*%-address-one-vote it could be subverted by anyone able to allocate many *%s. %roof-of-work is essentially one-$%&-one-vote. #he ma'ority decision is represented by the longest chain which has the greatest proof-of-work effort invested in it. *f a ma'ority of $%& power is controlled by honest nodes the honest chain will grow the fastest and outpace any competing chains. #o modify a past block an attacker would have to redo the proof-of-work of the block and all blocks after it and then catch up with and surpass the work of the honest nodes. " e will show later that the probability of a slower attacker catching up diminishes exponentially as subse!uent blocks are added.

#o compensate for increasing hardware speed and varying interest in running nodes over time the proof-of-work difficulty is determined by a moving average targeting an average number of blocks per hour. *f they(re generated too fast the difficulty increases.

%. &et ' or$

#he steps to run the network are as follows7

,8 New transactions are broadcast to all nodes.
18 -ach node collects new transactions into a block.
98 -ach node works on finding a difficult proof-of-work for its block.
:8 " hen a node finds a proof-of-work it broadcasts the block to all nodes.
28 Nodes accept the block only if all transactions in it are valid and not already spent.
58 Nodes express their acceptance of the block by working on creating the next block in the chain using the hash of the accepted block as the previous hash.

Nodes always consider the longest chain to be the correct one and will keep working on extending it. *f two nodes broadcast different versions of the next block simultaneously some nodes may receive one or the other first. *n that case they work on the first one they received but save the other branch in case it becomes longer. #he tie will be broken when the next proof-

-. Simpli"ied Payment .eri"ication

*t is possible to verify payments without running a full network node. A user only needs to keep a copy of the block headers of the longest proof-of-work chain which he can get by !uerying network nodes until he(s convinced he has the longest chain and obtain the)erkle branch linking the transaction to the block it(s timestamped in. 4e can(t check the transaction for himself but by linking it to a place in the chain he can see that a network node has accepted it and blocks added after it further confirm the network has accepted it.

As such the verification is reliable as long as honest nodes control the network but is more vulnerable if the network is overpowered by an attacker. "hile network nodes can verify transactions for themselves the simplified method can be fooled by an attacker(s fabricated transactions for as long as the attacker can continue to overpower the network. 6ne strategy to protect against this would be to accept alerts from network nodes when they detect an invalid block prompting the user(s software to download the full block and alerted transactions to confirm the inconsistency. 3usinesses that receive fre!uent payments will probably still want to run their own nodes for more independent security and !uicker verification.

/. Com0inin+ and Splittin+ .alue

Although it would be possible to handle coins individually it would be unwieldy to make a separate transaction for every cent in a transfer. #o allow value to be split and combined transactions contain multiple inputs and outputs. Normally there will be either a single input from a larger previous transaction or multiple inputs combining smaller amounts and at most two outputs7 one for the payment and one returning the change if any back to the sender.

*t should be noted that fan-out where a transaction depends on several transactions and those transactions depend nenPnd aluEpbu0o

11. Pri acy

#he traditional banking model achieves a level of privacy by limiting access to information to the parties involved and the trusted third party. #he necessity to announce all transactions publicly precludes this method but privacy can still be maintained by breaking the flow of information in another place7 by keeping public keys anonymous. #he public can see that someone is sending an amount to someone else but without information linking the transaction to anyone. #his is similar to the level of information released by stock exchanges where the time and si+e of individual trades the DtapeD is made public but without telling who the parties were.

As an additional firewall a new key pair should be used for each transaction to keep them from being linked to a common owner. Some linking is still unavoidable with multi-input transactions which necessarily reveal that their inputs were owned by the same owner. #he risk is that if the owner of a key is revealed linking could reveal other transactions that belonged to the same owner.

11. Calculations

"e consider the scenario of an attacker trying to generate an alternate chain faster than the honest chain. -ven if this is accomplished it does not throw the system open to arbitrary changes such as creating value out of thin air or taking money that never belonged to the attacker. Nodes are not going to accept an invalid transaction as payment and honest nodes will never accept a block containing them. An attacker can only try to change one of his own transactions to take back money he recently spent.

#he race between the honest chain and an attacker chain can be characteri+ed as a 3inomial Bandom "alk. #he success event is the honest chain being extended by one block increasing its lead by E, and the failure event is the attacker(s chain being extended by one block reducing the gap by -,.

#he probability of an attacker catching up from a given deficit is analogous to a Aambler(s Buin problem. Suppose a gambler with unlimited credit starts at a deficit and plays potentially an infinite number of trials to try to reach breakeven. "e can calculate the probability he ever reaches breakeven or that an attacker ever catches up with the honest chain as follows /=07

p @ probability an honest node finds the next block
q

Aiven our assumption that p > q the probability drops exponentially as the number of blocks the attacker has to catch up with increases. "ith the odds against him if he doesn(t make a lucky lunge forward early on his chances become vanishingly small as he falls further behind.

"e now consider how long the recipient of a new transaction needs to wait before being sufficiently certain the sender can(t change the transaction. "e assume the sender is an attacker who wants to make the recipient believe he paid him for a while then switch it to pay back to himself after some time has passed. #he receiver will be alerted when that happens but the sender hopes it will be too late.

#he receiver generates a new key pair and gives the public key to the sender shortly before signing. #his prevents the sender from preparing a chain of blocks ahead of time by working on it continuously until he is lucky enough to get far enough ahead then executing the transaction at that moment. 6nce the transaction is sent the dishonest sender starts working in secret on a parallel chain containing an alternate version of his transaction.

#he recipient waits until the transaction has been added to a block and z blocks have been linked after it. 4e doesn(t know the exact amount of progress the attacker has made but assuming the honest blocks took the average expected time per block the attacker(s potential progress will be a %oisson distribution with expected value7

$$z \frac{q}{}$$

Bunning some results we can see the probability drop off exponentially with +.

```
q!#."
z!#    P!".#######
z!"    P!#.-#./012
z!-    P!#.#/#3113
z!2    P!#.#"2"1--
z!.    P!#.##2.//-
z!/    P!#.###3"21
z!4    P!#.###-.-0
z!1    P!#.####4.1
z!0    P!#.####"12
z!3    P!#.#####.4
z!"#   P!#.#####"-

q!#.2
z!#    P!".#######
z!/    P!#."112/-2
z!"#   P!#.#."44#/
z!"/   P!#.#"#"##0
z!-#   P!#.##-.0#.
z!-/   P!#.###4"2-
z!2#   P!#.###"/--
z!2/   P!#.####213
z!.#   P!#.#####3/
z!./   P!#.#####-.
z!/#   P!#.######4
```

Solving for % less than >., F ...

```
P < #.##"
q!#."#   z!/
q!#."/   z!0
q!#.-#   z!""
q!#.-/   z!"/
q!#.2#   z!-.
q!#.2/   z!."
q!#..#   z!03
q!#../   z!2.#
```

12. Conclusion

"e have proposed a system for electronic transactions without relying on trust. "e started with the usual framework of coins made from digital signatures which provides strong control of ownership but is incomplete without a way to prevent double-spending. #o solve this we proposed a peer-to-peer network using proof-of-work to record a public history of transactions that !uickly becomes computationally impractical for an attacker to change if honest nodes control a ma'ority of $%& power. #he network is robust in its unstructured simplicity. Nodes work all at once with little coordination. #hey do not need to be identified since messages are not routed to any particular place and only need to be delivered on a best effort basis. Nodes can leave and re'oin the network at will accepting the proof-of-work chain as proof of what happened while they were gone. #hey vote with their $%& power expressing their acceptance of valid blocks by working on extending them and re'ecting invalid blocks by refusing to work on them. Any needed rules and incentives can be enforced with this consensus mechanism.

*e"erences

/,0 ". Dai Db-money D http7GGwww.weidai.comGbmoney.txt ,HH=.

/10 4.) assias I.S. Avila and J.-J. Kuis!uater DDesign of a secure timestamping service with minimal trust re!uirements D*n 20th Symposium on Information Theory in the Benelux) ay ,HHH.

/90 S. 4aber ".S. Stornetta D4ow to time-stamp a digital document D*n Journal of Cryptology vol 9 no 1 pages HH-,,, ,HH,.

/:0 D. 3ayer S. 4aber ".S. Stornetta D*mproving the efficiency and reliability of digital time-stamping D*n Sequences II *! etho"s in Communication# Security an" Computer Science pages 91H-99: ,HH9.

/20 S. 4aber ".S. Stornetta DSecure names for bit-strings D*n $rocee"ings of the %th &C *! Conference on Computer an" Communications Security pages 1=-92 April ,HH<.

/50 A. 3ack D4ashcash - a denial of service counter-measure D http7GGwww.hashcash.orgGpapersGhashcash.pdf 1>>1.

/<0 B.$.) erkle D%rotocols for public key cryptosystems D*n $roc' ()*0 Symposium on Security an" $ri+acy *- - - $omputer Society pages ,11-,99 April ,H=>.

/=0 ". .eller DAn introduction to probability theory and its applications D ,H2<.