Kicking the Hornet's Nest

The Complete Writings, Emails, and Forum Posts of Satoshi Nakamoto, the Founder of Bitcoin and Cryptocurrency

This book is available in print at http://www.lulu.com

It would have been nice to get this attention in any other context. WikiLeaks has kicked the hornet's nest, and the swarm is headed towards us.

- Satoshi Nakamoto, December 11, 2010, 23:39:16 UTC

This statement was in reference to an article by PC World. It can be accessed at https://www.pcworld.com/article/213230/could_wikileaks_scandal_lead_to_new_virtual_currency.html.

Two days later, Satoshi Nakamoto disappeared from making further public postings.

Sources:

- *https://satoshi.nakamotoinstitute.org - This was the main resource for this book. Their work and organization is priceless.*
- *https://BitcoinTalk.org/ - The forum set up by Satoshi.*
- *http://www.metzdowd.com/pipermail/cryptography - The Cryptography Mailing List was used by the group generally known as "cypherpunks."*
- *https://plan99.net/~mike - Personal emails to/from Mike Hearn, publicly shared on the Internet at this site.*
- *https://en.bitcoin.it/wiki/Source:Trammell/Nakamoto_emails - Personal emails to/from Dustin Trammel (aka Druid) from January 2009.*
- *https://online.wsj.com/public/resources/documents/finneynakamotoemails.pdf - Personal emails to/from Hal Finney, publicly shared on this Wall Street Journal site.*

Notes from the Editor

Ten years ago, on January 3, 2009, Bitcoin went live. That day, Satoshi Nakamoto generated the first Bitcoin block, which has since come to be known as the "Genesis block." In the Genesis block, Satoshi encoded the message, "The Times 03/Jan/2009 Chancellor on brink of second bailout for banks." This was likely to both timestamp the block to the outside world (using the title of the article on the front page of the London's daily The Times*), but, more importantly, to offer a comment. The comment gave insight that was both outward toward the financial system and inward toward Satoshi himself. Any article title could have been chosen as a timestamp. This one was clearly meant to convey a message. Satoshi sent the message that he does not favor banks. More likely, he does not like the fractional reserve banking system and the endless creation of fiat currency that coincides with fractional reserve banking. 2008 and 2009, when Bitcoin was born, were the years of rampant "cash injections," "stimulus packages," "quantitative easing," and "too-big-to-fail" bank bailouts. Bitcoin, with its hard-coded 21 million coin limit, would solve the fiat addiction. Infinite paper money would be replaced by finite numbers written in code.*

What's more, Satoshi fired a shot across the bow of the financial powers-that-be. Bankers, politicians, and the manipulators of the money supply have not been happy about Bitcoin and cryptocurrency. Ten years in, the powers seem to be warming to the idea a bit—or, at least, they're beginning to realize the use-cases and the inevitability of crypto. Still, their reluctant "embrace" is very slow and very cautious. I imagine one of the most threatening things to the powerful is to suggest that power be taken from them and then dispersed to the people themselves. Putting power into the hands of the people means saying, "You know what? We the people really don't need you after all. Have a nice day." Bitcoin suggests this very thing financially—it gives the power, freedom, and responsibility to the individual. As a boy, my brother and I would occasionally come upon a hornet's nest while playing in the woods. When we did, being boys, there was really nothing else to do but to throw a rock or stick at it, or kick it. Kicking a hornet's nest isn't rational, but just too tempting and just too much fun not to. And when you do it, you do it fast and then you run like hell!

Since January 2009, some people have placed an almost religious status onto Satoshi and his writings (the term "Genesis block" serves an example). I do not subscribe to this position, and I discourage anyone from doing so. Satoshi is, or was, a man, or a woman, or a group—as fallible and as human as us all. And, I'm sure he holds just as many hang-ups and weaknesses as anyone else. Applying demi-god status to a mortal man is unfair to that person, and sets one's self up for disappointment. And yet, Satoshi was very clever. So, I do think his writings, interactions, and thought processes are important, revolutionary, and worth documenting. I realize that all of these words are fantastically preserved and organized on websites, particularly at the Satoshi Nakamoto Institute (https://nakamotoinstitute.org/). Still, having a hard copy for reference or referral may be appealing to some. And, I realize other such books exist already. However, they include most, but not all, of Satoshi's writings and they include excellent commentary as well. This book is distinct in that it has the entirety of Satoshi's work included, is arranged chronologically rather than topically, and offers almost zero

commentary. The goals here were to be complete, to build a chronological chain of Satoshi's words and thoughts, and to allow Satoshi's words to speak for themselves free from an editor's interjections. Thus, this book was assembled.

Following are all of the public writings of Satoshi Nakamoto, the founder of Bitcoin—at least these are all that I could find. They are arranged in chronological order. Many of the writings are very technical. Some are purely code and will read as jibberish to most of us. I debated whether to include these "writings" or not. But, I wished to have a full account of all of Satoshi's writings, and so, even the code was included. Though unwieldly to read, even they convey a message—Satoshi was focused, businesslike, and pragmatic in his dealings and work. Since many of the writings are in response to others' comments, and for the sake of revealing the context of Satoshi's words, there are writings by other people included here as well. However, any non-Satoshi writings <u>are</u> italicized. Satoshi's writings can be identified by the fact that they are <u>not</u> italicized.

> Satoshi's words <u>are not</u> italized. They look like this.
>
> *Words by others <u>are</u> italicized. They look like this.*

Compiling these writings was educational to the editor. It seemed to offer insight into Satoshi Nakamoto. Lessons were learnt regarding Satoshi combing through his words, or, at the least, following were my interpretations:

Satoshi is polite. He said "Thanks" or "Thank you" several times. Often, an exclamation point was included for emphasis. And, he apologizes when appropriate.

Satoshi is a good teacher. In the earlier phases especially, he patiently and clearly answers questions one-by-one.

Satoshi is a clear communicator. His English, grammar, and syntax are nearly flawless. Although he does, on occasion, dabble in textese—he throws in a WTF and an AFAIK—nearly all of his communications are in clear, declarative, complete and correct sentences.

Satoshi is a fantastic thinker. He is able to think with beautiful logic. He is able to think abstractly in concepts via analogies (such as the Gambler's Ruin problem in the whitepaper). His more formal logic is seen in his code, naturally, but it is also witnessed in his writings. For example, in a response to theymos, Satoshi simply states, "The premise is false," then he explains why. As something of an aside, that statement harkens to Ayn Rand's Atlas Shrugged*, where "check your premises" is an ongoing sub-theme in the novel. For anyone unfamiliar with the book, the phrase does two things. First, it's a reference to Aristotelian logic of non-contradiction—if two things seem to contradict, they actually don't, one of*

them is wrong—check your premises. And secondly, the uber-theme of the novel itself is an indictment of government bailouts very similar to the Chancellor's brink-of-bailout of January 3, 2009. Atlas Shrugged *damns governments and powers which purport to know what's best and act for the people's best interest, rather than freeing the people to simply act for themselves. I don't think Satoshi was thinking* Atlas Shrugged *when he wrote the premise statement to theymos. I believe he was merely thinking clearly. But, the theme of* Atlas Shrugged*, and the "theme" of Bitcoin, certainly do seem to coincide with those words.*

Satoshi likes to double-space after a sentence is complete. This was the standard taught to typing or keyboarding students up until roughly the year 2000. Stylometry, studying a person's literary quirks in writing, has been a ripe field for pondering the identity of Satoshi Nakamoto. It may be a stretch, but with few clues, this double-space idiosyncrasy has often been noted in places like /r/Bitcoin on Reddit. There has also been discussion about Satoshi's tendencies toward British spellings of words, such as cheques for checks, neighbours for neighbors, decentralised or formalised with an s rather than a z, or use of the word "bloody." Some say these British tendencies were for obfuscation—to fake the world. I personally think there is something to the British influence. His British usage reads very organicly and unforced. I interpret that Satoshi indeed had some British-influenced upbringing (e.g., Britain, or Canada or Australia or a British Caribbean island). Like micro-expressions in facial body language, wording, in organic thought or writing, becomes hard-coded. To <u>not</u> release those tendencies would require constant and extreme discipline. Of course, Satoshi just might well have those qualities and fool me right there! Regarding double-spacing, I tend to believe that the double-spacing may well hint at Satoshi's age…he most likely learned to type when double-spacing after a period was standard. Revolutionary ideas have often come in history from people in their 20s or early 30s, but in this case, that seems too young. Typing this specific way, given the revolutionary thoughts for Bitcoin, and the technical skill acquired and necessary to create the code, as well as the polish in writing, Satoshi was likely not young when working on Bitcoin. Purely speculating, I would guess that he was likely around 40 when the whitepaper came out in 2008…meaning he was likely born around 1968, give-or-take a few years.

Satoshi is a heads-down programmer. Many of the writings here are mundane coder-talk. They are likely cryptic jibberish to nearly everyone. Satoshi does not fiddle with small-talk or niceties. He consistently remains focused and practical. When wished a happy Christmas by Mike Hearn if he celebrates Christmas, Satoshi makes no response either way. He merely proceeds to the task-at-hand.

Satoshi values privacy. This is witnessed in his words—naturally for a cypherpunk—but also in his focused neglect of including anything personal about himself (or herself), such as the Christmas non-comment. It's worth noting here that since Satoshi Nakamoto is unknown, Satoshi's sex is unknown. Satoshi may

be a man, woman, or group. However, since サトシ is generally a male's name in Japan, Satoshi is referred to here using singular, male pronouns.

Satoshi can pack a lot into a few words. His writing style is brief and to-the-point, but not impolite or terse. On the day the whitepaper was revealed, when he writes, "I've been working on a new electronic cash system that's fully peer-to-peer, with no trusted third party," *he could have almost simply stopped right there.*

Satoshi has a practical sense of marketing about him. He understands the importance of a good icon or logo. He understands that slow growth is not necessarily a bad thing. And he gets that there is such a thing as bad publicity (e.g., the WikiLeaks, hornet's nest comment).

Despite his focused, logical, business-minded tendencies, there seems to me to be a bit of boyishness about him. This is seldom shown, but it is there, revealed in his writings in rare glints. This leads to a final conclusion…

Satoshi is human. When he writes to Mike Hearn on Wed, March 9, 2011, "That's great news!" *the guarded wall that normally shields Satoshi-the-person seems to quaver. It hints at a real person, with emotions, excitement, and an almost childlike glee in what he's doing, lying somewhere behind the façade of Satoshi Nakamoto. He's kicking the hornet's nest himself, and he knows it. And, when just two days before withdrawing from public posts he writes,* "That means a lot coming from you, Hal. Thanks." *I hear a deep sigh after sending that comment.*

- Editor
January 3, 2019

Satoshi Nakamoto’s PGP Key

```
-----BEGIN PGP PUBLIC KEY BLOCK-----
Version: GnuPG v1.4.7 (MingW32)

mQGiBEkJ+qcRBADKDTcZlYDRtP1Q7/ShuzBJzUh9hoVVowogf2W07U6G9BqKW24r
piOxYmErjMFfvNtozNk+33cd/sq3gi05O1IMmZzg2rbF4ne5t3iplXnNuzNh+j+6
VxxA16GPhBRprvnng8r9GYALLUpo9Xk17KE429YYKFgVvtTPtEGUlpO1EwCg7FmW
dBbRp4mn5GfxQNT1hzp9WgkD/3pZ0cB5m4enzfylOHXmRfJKBMF02ZDnsY1GqeHv
/LjkhCusTp2qz4thLycYOFKGmAddpVnMsE/TYZLgpsxjrJsrEPNSdoXk3IgEStow
mXjTfr9xNOrB20Qk0ZOO1mipOWMgse4PmIu02X24OapWtyhdHsX3oBLcwDdke8aE
gAh8A/sHlK7fL1Bi8rFzx6hb+2yIlD/fazMBVZUe0r2uo7ldqEz5+GeEiBFignd5
HHhqjJw8rUJkfeZBoTKYlDKo7XDrTRxfyzNuZZPxBLTj+keY8WgYhQ5MWsSC2MX7
FZHaJddYa0pzUmFZmQh0ydulVUQnLKzRSunsjGOnmxiWBZwb6bQjU2F0b3NoaSBO
YWthbW90byA8c2F0b3NoaW5AZ214LmNvbT6IYAQTEQIAIAUCSQn6pwIbAwYLCQgH
AwIEFQIIAwQWAgMBAh4BAheAAAoJEBjAnoZeyUihXGMAnjiWJ0fvmSgSM3o6Tu3q
RME9GN7QAKCGrFw9SUD0e9/YDcqhX1aPMrYue7kCDQRJCfqnEAgA9OTCjLa6Sj7t
dZcQxNufsDSCSB+yznIGzFGXXpJk7GgKmX3H9Zl4E6zJTQGXL2GAV4klkSfNtvgs
SGJKqCnebuZVwutyq1vXRNVFPQFvLVVo2jJCBHWjb03fmXmavIUtRCHoc8xgVJMQ
LrwvS943GgsqSbdoKZWdTnfnEq+UaGo+Qfv66NpT3Yl0CXUiNBITZOJcJdjHDTBO
XRqomX2WSguv+btYdhQGGQiaEx73XMftXNCxbOpqwsODQns7xTcl2ENru9BNIQME
I7L9FYBQUiKHm1k6RrBy1as8XElS2jEos7GAmlfF1wShFUX+NF1VOPdbN3ZdFoWq
sUjKk+QbrwADBQgA9DiD4+uuRhwk2B1TmtrXnwwhcdkE7ZbLHjxBfCsLPAZiPh8c
ICfV3S418i4H1YCz2ItcnC8KAPoS6mipyS28AU1B7zJYPODBn8E7aPSPzHJfudMK
MqiCHljVJrE23xsKTC0sIhhSKcr2G+6ARoG5lwuoqJqEyDrblVQQFpVxBNPHSTqu
O5PoLXQc7PKgC5SyQuZbEALEkItl2SL2yBRRGOlVJLnvZ6eaovkAlgsbGdlieOr0
UwWuJCwzZuBDruMYAfyQBvYfXZun3Zm84rW7Jclp18mXITwGCVHg/P5n7QMbBfZQ
A25ymkuj636Nqh+c4zRnSINfyrDcID7AcqEb6IhJBBgRAgAJBQJJCfqnAhsMAAoJ
EBjAnoZeyUihPrcAniVWl5M44RuGctJe+IMNX4eVkC08AJ9v7cXsp5uDdQNo8q3R
8RHwN4Gk8w==
=3FTe
-----END PGP PUBLIC KEY BLOCK-----
```

The Bitcoin Whitepaper

Source: https://bitcoin.org/bitcoin.pdf

Bitcoin:
A Peer-to-Peer Electronic Cash System
Satoshi Nakamoto

October 31, 2008

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 required to prevent double-spending. We propose a solution to the double-spending problem using a peer-to-peer network. The 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. The longest chain not only serves as proof of the sequence of events witnessed, but proof that it came from the largest pool of CPU power. As long as a majority of CPU power is controlled by nodes that are not cooperating to attack the network, they'll generate the longest chain and outpace attackers. The network itself requires minimal structure. Messages are broadcast on a best effort basis, and nodes can leave and rejoin the network at will, accepting the longest proof-of-work chain as proof of what happened while they were gone.

1. Introduction

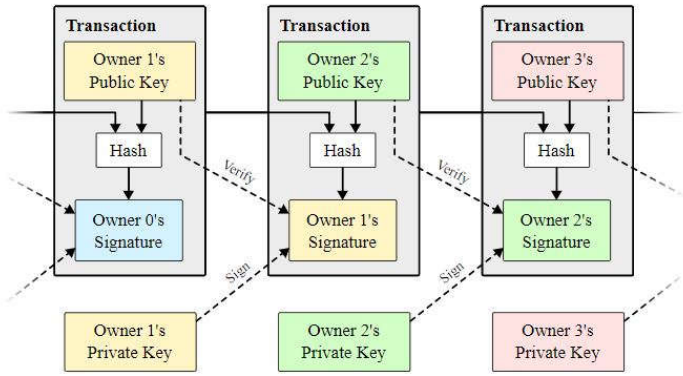
Commerce on the Internet has come to rely almost exclusively on financial institutions serving as trusted third parties to process electronic payments. While the system works well enough for most transactions, it still suffers from the inherent weaknesses of the trust based model. Completely non-reversible transactions are not really possible, since financial institutions cannot avoid mediating disputes. The cost of mediation increases transaction costs, limiting the minimum practical transaction size 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. With the possibility of reversal, the need for trust spreads. Merchants 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. These 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.

What 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. Transactions that are computationally impractical to reverse would protect sellers from fraud, and routine escrow mechanisms could easily be implemented to protect buyers. In 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. The system is secure as long as honest nodes collectively control more CPU power than any cooperating group of attacker nodes.

2. Transactions

We define an electronic coin as a chain of digital signatures. Each 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.



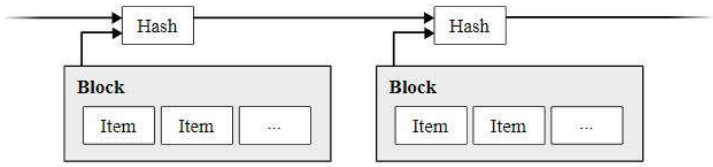
The problem of course is the payee can't verify that one of the owners did not double-spend the coin. A common solution is to introduce a trusted central authority, or mint, that checks every transaction for double spending. After each transaction, the coin must be returned to the mint to issue a new coin, and only coins issued directly from the mint are trusted not to be double-spent. The 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, just like a bank.

We need a way for the payee to know that the previous owners did not sign any earlier transactions. For our purposes, the earliest transaction is the one that counts, so we don't care about later attempts to double-spend. The only way to confirm the absence of a transaction is to be aware of all transactions. In the mint based model, the mint was aware of all transactions and decided which arrived first. To accomplish this without a trusted party, transactions must be publicly announced[1], and we need a system for participants to agree on a single history of the order in which they were received. The payee needs proof that at the time of each transaction, the majority of nodes agreed it was the first received.

3. Timestamp Server

The solution we propose begins with a timestamp server. A timestamp server works by taking a hash of a block of items to be timestamped and widely publishing the hash, such as in a newspaper or Usenet post[2-5]. The timestamp proves that the data must have existed at the time, obviously, in order to get into

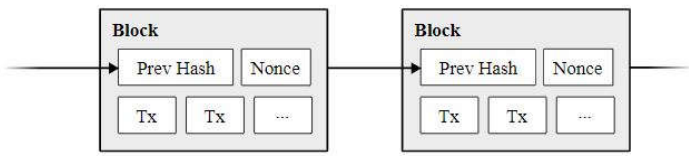
the hash. Each timestamp includes the previous timestamp in its hash, forming a chain, with each additional timestamp reinforcing the ones before it.



4. Proof-of-Work

To implement a distributed timestamp server on a peer-to-peer basis, we will need to use a proof-of-work system similar to Adam Back's Hashcash[6], rather than newspaper or Usenet posts. The proof-of-work involves scanning for a value that when hashed, such as with SHA-256, the hash begins with a number of zero bits. The average work required is exponential in the number of zero bits required and can be verified by executing a single hash.

For 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 required zero bits. Once the CPU 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.



The proof-of-work also solves the problem of determining representation in majority decision making. If the majority were based on one-IP-address-one-vote, it could be subverted by anyone able to allocate many IPs. Proof-of-work is essentially one-CPU-one-vote. The majority decision is represented by the longest chain, which has the greatest proof-of-work effort invested in it. If a majority of CPU power is controlled by honest nodes, the honest chain will grow the fastest and outpace any competing chains. To 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. We will show later that the probability of a slower attacker catching up diminishes exponentially as subsequent blocks are added.

To 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. If they're generated too fast, the difficulty increases.

5. Network

The steps to run the network are as follows:

1. New transactions are broadcast to all nodes.
2. Each node collects new transactions into a block.
3. Each node works on finding a difficult proof-of-work for its block.
4. When a node finds a proof-of-work, it broadcasts the block to all nodes.
5. Nodes accept the block only if all transactions in it are valid and not already spent.
6. 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. If two nodes broadcast different versions of the next block simultaneously, some nodes may receive one or the other first. In that case, they work on the first one they received, but save the other branch in case it becomes longer. The tie will be broken when the next proof-of-work is found and one branch becomes longer; the nodes that were working on the other branch will then switch to the longer one.

New transaction broadcasts do not necessarily need to reach all nodes. As long as they reach many nodes, they will get into a block before long. Block broadcasts are also tolerant of dropped messages. If a node does not receive a block, it will request it when it receives the next block and realizes it missed one.

6. Incentive

By convention, the first transaction in a block is a special transaction that starts a new coin owned by the creator of the block. This adds an incentive for nodes to support the network, and provides a way to initially distribute coins into circulation, since there is no central authority to issue them. The steady addition of a constant of amount of new coins is analogous to gold miners expending resources to add gold to circulation. In our case, it is CPU time and electricity that is expended.

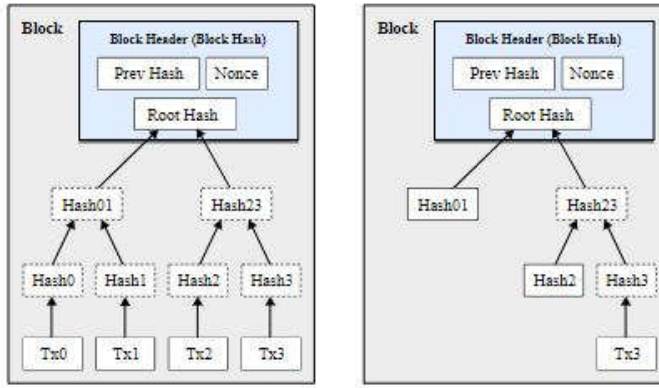
The incentive can also be funded with transaction fees. If the output value of a transaction is less than its input value, the difference is a transaction fee that is added to the incentive value of the block containing the transaction. Once a predetermined number of coins have entered circulation, the incentive can transition entirely to transaction fees and be completely inflation free.

The incentive may help encourage nodes to stay honest. If a greedy attacker is able to assemble more CPU power than all the honest nodes, he would have to choose between using it to defraud people by stealing back his payments, or using it to generate new coins. He ought to find it more profitable to play by the rules, such rules that favour him with more new coins than everyone else combined, than to undermine the system and the validity of his own wealth.

7. Reclaiming Disk Space

Once the latest transaction in a coin is buried under enough blocks, the spent transactions before it can be discarded to save disk space. To facilitate this without breaking the block's hash, transactions are hashed in a Merkle

Tree [7][2][5], with only the root included in the block's hash. Old blocks can then be compacted by stubbing off branches of the tree. The interior hashes do not need to be stored.

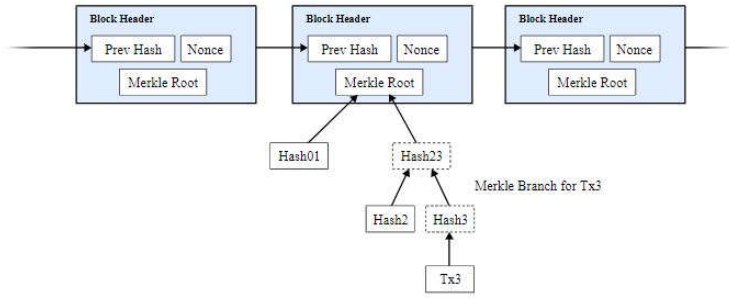


Transactions Hashed in a Merkle Tree | After Pruning Tx0-2 from the Block

A block header with no transactions would be about 80 bytes. If we suppose blocks are generated every 10 minutes, 80 bytes * 6 * 24 * 365 = 4.2MB per year. With computer systems typically selling with 2GB of RAM as of 2008, and Moore's Law predicting current growth of 1.2GB per year, storage should not be a problem even if the block headers must be kept in memory.

8. Simplified Payment Verification

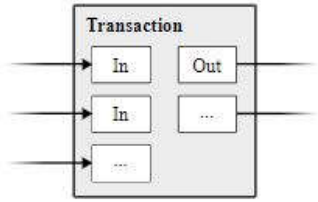
It is possible to verify payments without running a full network node. A user only needs to keep a copy of the block headers of the longest proof-of-work chain, which he can get by querying network nodes until he's convinced he has the longest chain, and obtain the Merkle branch linking the transaction to the block it's timestamped in. He 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. While 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. One 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. Businesses that receive frequent payments will probably still want to run their own nodes for more independent security and quicker verification.

9. Combining and Splitting Value

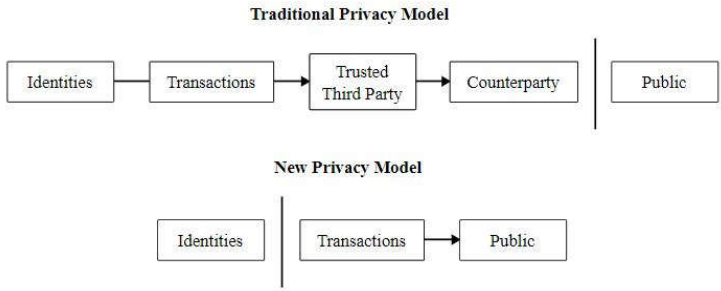
Although it would be possible to handle coins individually, it would be unwieldy to make a separate transaction for every cent in a transfer. To allow value to be split and combined, transactions contain multiple inputs and outputs. Normally there will be either a single input from a larger previous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



It should be noted that fan-out, where a transaction depends on several transactions, and those transactions depend on many more, is not a problem here. There is never the need to extract a complete standalone copy of a transaction's history.

10. Privacy

The traditional banking model achieves a level of privacy by limiting access to information to the parties involved and the trusted third party. The necessity to announce all transactions publicly precludes this method, but privacy can still be maintained by breaking the flow of information in another place: by keeping public keys anonymous. The public can see that someone is sending an amount to someone else, but without information linking the transaction to anyone. This is similar to the level of information released by stock exchanges, where the time and size of individual trades, the "tape", 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. The risk is that if the owner of a key is revealed, linking could reveal other transactions that belonged to the same owner.

11. Calculations

We consider the scenario of an attacker trying to generate an alternate chain faster than the honest chain. Even 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.

The race between the honest chain and an attacker chain can be characterized as a Binomial Random Walk. The success event is the honest chain being extended by one block, increasing its lead by +1, and the failure event is the attacker's chain being extended by one block, reducing the gap by -1.

The probability of an attacker catching up from a given deficit is analogous to a Gambler's Ruin problem. Suppose a gambler with unlimited credit starts at a deficit and plays potentially an infinite number of trials to try to reach breakeven. We can calculate the probability he ever reaches breakeven, or that an attacker ever catches up with the honest chain, as follows[8]:

pqqz=== probability an honest node finds the next block probability the attacker finds the next block probability the attacker will ever catch up from z blocks behindp= probability an honest node finds the next blockq= probability the attacker finds the next blockqz= probability the attacker will ever catch up from z blocks behind

qz={1(q/p)zifp≤qifp>q}qz={1ifp≤q(q/p)zifp>q}

Given our assumption that p>qp>q, the probability drops exponentially as the number of blocks the attacker has to catch up with increases. With 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.

We now consider how long the recipient of a new transaction needs to wait before being sufficiently certain the sender can't change the transaction. We 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. The receiver will be alerted when that happens, but the sender hopes it will be too late.

The receiver generates a new key pair and gives the public key to the sender shortly before signing. This 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. Once the transaction is sent, the dishonest sender starts working in secret on a parallel chain containing an alternate version of his transaction.

The recipient waits until the transaction has been added to a block and zz blocks have been linked after it. He 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 Poisson distribution with expected value:

λ=zqpλ=zqp

To get the probability the attacker could still catch up now, we multiply the Poisson density for each amount of progress he could have made by the probability he could catch up from that point:

∑k=0∞λke λk! {(q/p)(z k)1ifk≤zifk>z}∑k=0∞λke λk! {(q/p)(z k)ifk≤z 1ifk>z}

Rearranging to avoid summing the infinite tail of the distribution...

1 ∑k=0zλke λk!(1 (q/p)(z k))1 ∑k=0zλke λk!(1 (q/p)(z k))

Converting to C code...

```
#include
double AttackerSuccessProbability(double q, int z)
{
    double p = 1.0 - q;
    double lambda = z * (q / p);
    double sum = 1.0;
    int i, k;
    for (k = 0; k <= z; k++)
    {
            double poisson = exp(-lambda);
            for (i = 1; i <= k; i++)
                    poisson *= lambda / i;
            sum -= poisson * (1 - pow(q / p, z - k));
    }
    return sum;
}
```

Running some results, we can see the probability drop off exponentially with zz.

```
q=0.1
z=0    P=1.0000000
```

z=1 P=0.2045873
z=2 P=0.0509779
z=3 P=0.0131722
z=4 P=0.0034552
z=5 P=0.0009137
z=6 P=0.0002428
z=7 P=0.0000647
z=8 P=0.0000173
z=9 P=0.0000046
z=10 P=0.0000012

q=0.3
z=0 P=1.0000000
z=5 P=0.1773523
z=10 P=0.0416605
z=15 P=0.0101008
z=20 P=0.0024804
z=25 P=0.0006132
z=30 P=0.0001522
z=35 P=0.0000379
z=40 P=0.0000095
z=45 P=0.0000024
z=50 P=0.0000006

Solving for P less than 0.1%...
P < 0.001
q=0.10 z=5
q=0.15 z=8
q=0.20 z=11
q=0.25 z=15
q=0.30 z=24
q=0.35 z=41
q=0.40 z=89
q=0.45 z=340

12. Conclusion

We have proposed a system for electronic transactions without relying on trust. We 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. To solve this, we proposed a peer-to-peer network using proof-of-work to record a public history of transactions that quickly becomes computationally impractical for an attacker to change if honest nodes control a majority of CPU power. The network is robust in its unstructured simplicity. Nodes work all at once with little coordination. They 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 rejoin the network at will, accepting the proof-of-work chain as proof of what

happened while they were gone. They vote with their CPU power, expressing their acceptance of valid blocks by working on extending them and rejecting invalid blocks by refusing to work on them. Any needed rules and incentives can be enforced with this consensus mechanism.

References

1. W. Dai, "b-money," http://www.weidai.com/bmoney.txt, 1998.
2. H. Massias, X.S. Avila, and J.-J. Quisquater, "Design of a secure timestamping service with minimal trust requirements," In *20th Symposium on Information Theory in the Benelux*, May 1999.
3. S. Haber, W.S. Stornetta, "How to time-stamp a digital document," In *Journal of Cryptology*, vol 3, no 2, pages 99-111, 1991.
4. D. Bayer, S. Haber, W.S. Stornetta, "Improving the efficiency and reliability of digital time-stamping," In *Sequences II: Methods in Communication, Security and Computer Science*, pages 329-334, 1993.
5. S. Haber, W.S. Stornetta, "Secure names for bit-strings," In *Proceedings of the 4th ACM Conference on Computer and Communications Security*, pages 28-35, April 1997.
6. A. Back, "Hashcash - a denial of service counter-measure,"http://www.hashcash.org/papers/hashcash.pdf, 2002.
7. R.C. Merkle, "Protocols for public key cryptosystems," In *Proc. 1980 Symposium on Security and Privacy*, IEEE Computer Society, pages 122-133, April 1980.
8. W. Feller, "An introduction to probability theory and its applications," 1957.

Emails, mailing list writings, forum posts by Satoshi Nakamoto (arranged in chronological order):

Cryptography Mailing List

Bitcoin P2P e-cash paper

2008-10-31 18:10:00 UTC - -

I've been working on a new electronic cash system that's fully peer-to-peer, with no trusted third party.

The paper is available at:
http://www.bitcoin.org/bitcoin.pdf

The main properties:
Double-spending is prevented with a peer-to-peer network.
No mint or other trusted parties.
Participants can be anonymous.
New coins are made from Hashcash style proof-of-work.
The proof-of-work for new coin generation also powers the network to prevent double-spending.

Bitcoin: A Peer-to-Peer Electronic Cash System

Abstract. A purely peer-to-peer version of electronic cash would allow online payments to be sent directly from one party to another without the burdens of going through a financial institution. Digital signatures provide part of the solution, but the main benefits are lost if a trusted party is still required to prevent double-spending. We propose a solution to the double-spending problem using a peer-to-peer network. The 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. The longest chain not only serves as proof of the sequence of events witnessed, but proof that it came from the largest pool of CPU power. As long as honest nodes control the most CPU power on the network, they can generate the longest chain and outpace any attackers. The network itself requires minimal structure. Messages are broadcasted on a best effort basis, and nodes can leave and rejoin the network at will, accepting the longest proof-of-work chain as proof of what happened while they were gone.

Full paper at:
http://www.bitcoin.org/bitcoin.pdf

Satoshi Nakamoto

Cryptography Mailing List

Bitcoin P2P e-cash paper

2008-11-03 01:37:43 UTC - -

>Satoshi Nakamoto wrote:
>> I've been working on a new electronic cash system that's fully
>> peer-to-peer, with no trusted third party.
>>
>> The paper is available at:
>> http://www.bitcoin.org/bitcoin.pdf
>
>We very, very much need such a system, but the way I understand your
>proposal, it does not seem to scale to the required size.
>
>For transferable proof of work tokens to have value, they must have
>monetary value. To have monetary value, they must be transferred within
>a very large network - for example a file trading network akin to
>bittorrent.
>
>To detect and reject a double spending event in a timely manner, one
>must have most past transactions of the coins in the transaction, which,
> naively implemented, requires each peer to have most past
>transactions, or most past transactions that occurred recently. If
>hundreds of millions of people are doing transactions, that is a lot of
>bandwidth - each must know all, or a substantial part thereof.
>

Long before the network gets anywhere near as large as that, it would be safe for users to use Simplified Payment Verification (section 8) to check for double spending, which only requires having the chain of block headers, or about 12KB per day. Only people trying to create new coins would need to run network nodes. At first, most users would run network nodes, but as the network grows beyond a certain point, it would be left more and more to specialists with server farms of specialized hardware. A server farm would only need to have one node on the network and the rest of the LAN connects with that one node.

The bandwidth might not be as prohibitive as you think. A typical transaction would be about 400 bytes (ECC is nicely compact). Each transaction has to be broadcast twice, so lets say 1KB per transaction. Visa processed 37 billion transactions in FY2008, or an average of 100 million transactions per day. That many transactions would take 100GB of bandwidth, or the size of 12 DVD or 2 HD quality movies, or about $18 worth of bandwidth at current prices.

If the network were to get that big, it would take several years, and by then, sending 2 HD movies over the Internet would probably not seem like a big deal.

Satoshi Nakamoto

Cryptography Mailing List

Bitcoin P2P e-cash paper

2008-11-03 16:23:49 UTC - -

>> As long as honest nodes control the most CPU power on the network,
>> they can generate the longest chain and outpace any attackers.
>
>But they don't. Bad guys routinely control zombie farms of 100,000
>machines or more. People I know who run a blacklist of spam sending
>zombies tell me they often see a million new zombies a day.
>
>This is the same reason that hashcash can't work on today's Internet
>-- the good guys have vastly less computational firepower than the bad
>guys.

Thanks for bringing up that point.

I didn't really make that statement as strong as I could have. The requirement is that the good guys collectively have more CPU power than any single attacker.

There would be many smaller zombie farms that are not big enough to overpower the network, and they could still make money by generating bitcoins. The smaller farms are then the "honest nodes". (I need a better term than "honest") The more smaller farms resort to generating bitcoins, the higher the bar gets to overpower the network, making larger farms also too small to overpower it so that they may as well generate bitcoins too. According to the "long tail" theory, the small, medium and merely large farms put together should add up to a lot more than the biggest zombie farm.

Even if a bad guy does overpower the network, it's not like he's instantly rich. All he can accomplish is to take back money he himself spent, like bouncing a check. To exploit it, he would have to buy something from a merchant, wait till it ships, then overpower the network and try to take his money back. I don't think he could make as much money trying to pull a carding scheme like that as he could by generating bitcoins. With a zombie farm that big, he could generate more bitcoins than everyone else combined.

The Bitcoin network might actually reduce spam by diverting zombie farms to generating bitcoins instead.

Satoshi Nakamoto

Cryptography Mailing List

Bitcoin P2P e-cash paper

2008-11-06 20:15:40 UTC - -

>[Lengthy exposition of vulnerability of a systm to use-of-force
>monopolies ellided.]
>
>You will not find a solution to political problems in cryptography.

Yes, but we can win a major battle in the arms race and gain a new territory of freedom for several years.

Governments are good at cutting off the heads of a centrally controlled networks like Napster, but pure P2P networks like Gnutella and Tor seem to be holding their own.

Satoshi

Cryptography Mailing List

Bitcoin P2P e-cash paper

2008-11-08 18:54:38 UTC - -

Ray Dillinger:
> the "currency" is inflationary at about 35%
> as that's how much faster computers get annually
> ... the inflation rate of 35% is almost guaranteed
> by the technology

Increasing hardware speed is handled: "To 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. If they're generated too fast, the difficulty increases."

As computers get faster and the total computing power applied to creating bitcoins increases, the difficulty increases proportionally to keep the total new production constant. Thus, it is known in advance how many new bitcoins will be created every year in the future.

The fact that new coins are produced means the money supply increases by a planned amount, but this does not necessarily result in inflation. If the supply of money increases at the same rate that the number of people using it increases, prices remain stable. If it does not increase as fast as demand, there will be deflation and early holders of money will see its value increase.

Coins have to get initially distributed somehow, and a constant rate seems like the best formula.

Satoshi Nakamoto

Cryptography Mailing List

Bitcoin P2P e-cash paper

2008-11-09 01:58:48 UTC - -

Hal Finney wrote:
> it is mentioned that if a broadcast transaction does not reach all nodes,
> it is OK, as it will get into the block chain before long. How does this
> happen - what if the node that creates the "next" block (the first node
> to find the hashcash collision) did not hear about the transaction,
> and then a few more blocks get added also by nodes that did not hear
> about that transaction? Do all the nodes that did hear it keep that
> transaction around, hoping to incorporate it into a block once they get
> lucky enough to be the one which finds the next collision?

Right, nodes keep transactions in their working set until they get into a block. If a transaction reaches 90% of nodes, then each time a new block is found, it has a 90% chance of being in it.

> Or for example, what if a node is keeping two or more chains around as
> it waits to see which grows fastest, and a block comes in for chain A
> which would include a double-spend of a coin that is in chain B? Is that
> checked for or not? (This might happen if someone double-spent and two
> different sets of nodes heard about the two different transactions with
> the same coin.)

That does not need to be checked for. The transaction in whichever branch ends up getting ahead becomes the valid one, the other is invalid. If someone tries to double spend like that, one and only one spend will always become valid, the others invalid.

Receivers of transactions will normally need to hold transactions for perhaps an hour or more to allow time for this kind of possibility to be resolved. They can still re-spend the coins immediately, but they should wait before taking an action such as shipping goods.

> I also don't understand exactly how double-spending, or cancelling
> transactions, is accomplished by a superior attacker who is able to muster
> more computing power than all the honest participants. I see that he can
> create new blocks and add them to create the longest chain, but how can
> he erase or add old transactions in the chain? As the attacker sends out

> his new blocks, aren't there consistency checks which honest nodes can
> perform, to make sure that nothing got erased? More explanation of this
> attack would be helpful, in order to judge the gains to an attacker from
> this, versus simply using his computing power to mint new coins honestly.

The attacker isn't adding blocks to the end. He has to go back and redo the block his transaction is in and all the blocks after it, as well as any new blocks the network keeps adding to the end while he's doing that. He's rewriting history. Once his branch is longer, it becomes the new valid one.

This touches on a key point. Even though everyone present may see the shenanigans going on, there's no way to take advantage of that fact.

It is strictly necessary that the longest chain is always considered the valid one. Nodes that were present may remember that one branch was there first and got replaced by another, but there would be no way for them to convince those who were not present of this. We can't have subfactions of nodes that cling to one branch that they think was first, others that saw another branch first, and others that joined later and never saw what happened. The CPU power proof-of-work vote must have the final say. The only way for everyone to stay on the same page is to believe that the longest chain is always the valid one, no matter what.

> As far as the spending transactions, what checks does the recipient of a
> coin have to perform? Does she need to go back through the coin's entire
> history of transfers, and make sure that every transaction on the list is
> indeed linked into the "timestamp" block chain? Or can she just do the
> latest one?

The recipient just needs to verify it back to a depth that is sufficiently far back in the block chain, which will often only require a depth of 2 transactions. All transactions before that can be discarded.

> Do the timestamp nodes check transactions, making sure that
> the previous transaction on a coin is in the chain, thereby enforcing
> the rule that all transactions in the chain represent valid coins?

Right, exactly. When a node receives a block, it checks the signatures of every transaction in it against previous transactions in blocks. Blocks can only contain transactions that depend on valid transactions in previous blocks or the same block. Transaction C could depend on transaction B in the same block and B depends on transaction A in an earlier block.

> Sorry about all the questions, but as I said this does seem to be a
> very promising and original idea, and I am looking forward to seeing

> how the concept is further developed. It would be helpful to see a more
> process oriented description of the idea, with concrete details of the
> data structures for the various objects (coins, blocks, transactions),
> the data which is included in messages, and algorithmic descriptions
> of the procedures for handling the various events which would occur in
> this system. You mentioned that you are working on an implementation,
> but I think a more formal, text description of the system would be a
> helpful next step.

I appreciate your questions. I actually did this kind of backwards. I had to write all the code before I could convince myself that I could solve every problem, then I wrote the paper. I think I will be able to release the code sooner than I could write a detailed spec. You're already right about most of your assumptions where you filled in the blanks.

Satoshi Nakamoto

Cryptography Mailing List

Bitcoin P2P e-cash paper

2008-11-09 03:09:49 UTC - -

James A. Donald wrote:
> The core concept is that lots of entities keep complete and consistent
> information as to who owns which bitcoins.
>
> But maintaining consistency is tricky. It is not clear to me what
> happens when someone reports one transaction to one maintainer, and
> someone else transports another transaction to another maintainer. The
> transaction cannot be known to be valid until it has been incorporated
> into a globally shared view of all past transactions, and no one can
> know that a globally shared view of all past transactions is globally
> shared until after some time has passed, and after many new
> transactions have arrived.
>
> Did you explain how to do this, and it just passed over my head, or
> were you confident it could be done, and a bit vague as to the details?

The proof-of-work chain is the solution to the synchronisation problem, and to knowing what the globally shared view is without having to trust anyone.

A transaction will quickly propagate throughout the network, so if two versions of the same transaction were reported at close to the same time, the one with the head start would have a big advantage in reaching many more nodes first. Nodes will only accept the first one they see, refusing the second one to arrive, so the earlier transaction would have many more nodes working on incorporating it into the next proof-of-work. In effect, each node votes for its viewpoint of which transaction it saw first by including it in its

proof-of-work effort.

If the transactions did come at exactly the same time and there was an even split, it's a toss up based on which gets into a proof-of-work first, and that decides which is valid.

When a node finds a proof-of-work, the new block is propagated throughout the network and everyone adds it to the chain and starts working on the next block after it. Any nodes that had the other transaction will stop trying to include it in a block, since it's now invalid according to the accepted chain.

The proof-of-work chain is itself self-evident proof that it came from the globally shared view. Only the majority of the network together has enough CPU power to generate such a difficult chain of proof-of-work. Any user, upon receiving the proof-of-work chain, can see what the majority of the network has approved. Once a transaction is hashed into a link that's a few links back in the chain, it is firmly etched into the global history.

Satoshi Nakamoto

Cryptography Mailing List

Bitcoin P2P e-cash paper

2008-11-09 16:31:26 UTC - -

James A. Donald wrote:
>OK, suppose one node incorporates a bunch of
>transactions in its proof of work, all of them honest
>legitimate single spends and another node incorporates a
>different bunch of transactions in its proof of
>work, all of them equally honest legitimate single
>spends, and both proofs are generated at about the same
>time.
>
>What happens then?

They both broadcast their blocks. All nodes receive them and keep both, but only work on the one they received first. We'll suppose exactly half received one first, half the other.

In a short time, all the transactions will finish propagating so that everyone has the full set. The nodes working on each side will be trying to add the transactions that are missing from their side. When the next proof-of-work is found, whichever previous block that node was working on, that branch becomes longer and the tie is broken. Whichever side it is, the new block will contain the other half of the transactions, so in either case, the branch will contain all transactions. Even in the unlikely event that a split happened twice in a row, both sides of the second split would contain the full set of transactions anyway.

It's not a problem if transactions have to wait one or a few extra cycles to get into a block.

Satoshi Nakamoto

Cryptography Mailing List

Bitcoin P2P e-cash paper

2008-11-10 02:14:30 UTC - -

James A. Donald wrote:
> Furthermore, it cannot be made to work, as in the
> proposed system the work of tracking who owns what coins
> is paid for by seigniorage, which requires inflation.

If you're having trouble with the inflation issue, it's easy to tweak it for transaction fees instead. It's as simple as this: let the output value from any transaction be 1 cent less than the input value. Either the client software automatically writes transactions for 1 cent more than the intended payment value, or it could come out of the payee's side. The incentive value when a node finds a proof-of-work for a block could be the total of the fees in the block.

Satoshi Nakamoto

Cryptography Mailing List

Bitcoin P2P e-cash paper

2008-11-10 22:18:20 UTC - -

James A. Donald wrote:
> So what happened to the coin that lost the race?
>
> ... it is a bit harsh if the guy who came second
> is likely to lose his coin.

When there are multiple double-spent versions of the same transaction, one and only one will become valid.

The receiver of a payment must wait an hour or so before believing that it's valid. The network will resolve any possible double-spend races by then.

The guy who received the double-spend that became invalid never thought he had it in the first place. His software would have shown the transaction go from "unconfirmed" to

"invalid". If necessary, the UI can be made to hide transactions until they're sufficiently deep in the block chain.

> *Further, your description of events implies restrictions*
> *on timing and coin generation - that the entire network*
> *generates coins slowly compared to the time required for*
> *news of a new coin to flood the network*

Sorry if I didn't make that clear. The target time between blocks will probably be 10 minutes.

Every block includes its creation time. If the time is off by more than 36 hours, other nodes won't work on it. If the timespan over the last 6*24*30 blocks is less than 15 days, blocks are being generated too fast and the proof-of-work difficulty doubles. Everyone does the same calculation with the same chain data, so they all get the same result at the same link in the chain.

> *We want spenders to have certainty that their*
> *transaction is valid at the time it takes a spend to*
> *flood the network, not at the time it takes for branch*
> *races to be resolved.*

Instantant non-repudiability is not a feature, but it's still much faster than existing systems. Paper cheques can bounce up to a week or two later. Credit card transactions can be contested up to 60 to 180 days later. Bitcoin transactions can be sufficiently irreversible in an hour or two.

> *If one node is ignoring all spends that it does not*
> *care about, it suffers no adverse consequences.*

With the transaction fee based incentive system I recently posted, nodes would have an incentive to include all the paying transactions they receive.

Satoshi Nakamoto

Cryptography Mailing List

Bitcoin P2P e-cash paper

2008-11-13 22:56:55 UTC - -

James A. Donald wrote:
> *It is not sufficient that everyone knows X. We also*
> *need everyone to know that everyone knows X, and that*

> everyone knows that everyone knows that everyone knows X
> - which, as in the Byzantine Generals problem, is the
> classic hard problem of distributed data processing.

The proof-of-work chain is a solution to the Byzantine Generals' Problem. I'll try to rephrase it in that context.

A number of Byzantine Generals each have a computer and want to attack the King's wi-fi by brute forcing the password, which they've learned is a certain number of characters in length. Once they stimulate the network to generate a packet, they must crack the password within a limited time to break in and erase the logs, otherwise they will be discovered and get in trouble. They only have enough CPU power to crack it fast enough if a majority of them attack at the same time.

They don't particularly care when the attack will be, just that they all agree. It has been decided that anyone who feels like it will announce a time, and whatever time is heard first will be the official attack time. The problem is that the network is not instantaneous, and if two generals announce different attack times at close to the same time, some may hear one first and others hear the other first.

They use a proof-of-work chain to solve the problem. Once each general receives whatever attack time he hears first, he sets his computer to solve an extremely difficult proof-of-work problem that includes the attack time in its hash. The proof-of-work is so difficult, it's expected to take 10 minutes of them all working at once before one of them finds a solution. Once one of the generals finds a proof-of-work, he broadcasts it to the network, and everyone changes their current proof-of-work computation to include that proof-of-work in the hash they're working on. If anyone was working on a different attack time, they switch to this one, because its proof-of-work chain is now longer.

After two hours, one attack time should be hashed by a chain of 12 proofs-of-work. Every general, just by verifying the difficulty of the proof-of-work chain, can estimate how much parallel CPU power per hour was expended on it and see that it must have required the majority of the computers to produce that much proof-of-work in the allotted time. They had to all have seen it because the proof-of-work is proof that they worked on it. If the CPU power exhibited by the proof-of-work chain is sufficient to crack the password, they can safely attack at the agreed time.

The proof-of-work chain is how all the synchronisation, distributed database and global view problems you've asked about are solved.

Cryptography Mailing List

Bitcoin P2P e-cash paper

2008-11-14 18:55:35 UTC - -

Hal Finney wrote:
> I think it is necessary that nodes keep a separate
> pending-transaction list associated with each candidate chain.
> ... One might also ask ... how many candidate chains must
> a given node keep track of at one time, on average?

Fortunately, it's only necessary to keep a pending-transaction pool for the current best branch. When a new block arrives for the best branch, ConnectBlock removes the block's transactions from the pending-tx pool. If a different branch becomes longer, it calls DisconnectBlock on the main branch down to the fork, returning the block transactions to the pending-tx pool, and calls ConnectBlock on the new branch, sopping back up any transactions that were in both branches. It's expected that reorgs like this would be rare and shallow.

With this optimisation, candidate branches are not really any burden. They just sit on the disk and don't require attention unless they ever become the main chain.

> Or as James raised earlier, if the network broadcast
> is reliable but depends on a potentially slow flooding
> algorithm, how does that impact performance?

Broadcasts will probably be almost completely reliable. TCP transmissions are rarely ever dropped these days, and the broadcast protocol has a retry mechanism to get the data from other nodes after a while. If broadcasts turn out to be slower in practice than expected, the target time between blocks may have to be increased to avoid wasting resources. We want blocks to usually propagate in much less time than it takes to generate them, otherwise nodes would spend too much time working on obsolete blocks.

I'm planning to run an automated test with computers randomly sending payments to each other and randomly dropping packets.

> 3. The bitcoin system turns out to be socially useful and valuable, so
> that node operators feel that they are making a beneficial contribution
> to the world by their efforts (similar to the various "@Home" compute
> projects where people volunteer their compute resources for good causes).
>
> In this case it seems to me that simple altruism can suffice to keep the
> network running properly.

It's very attractive to the libertarian viewpoint if we can explain it properly. I'm better with code than with words though.

Satoshi Nakamoto

Bitcoin P2P e-cash paper

2008-11-15 04:43:00 UTC - -

I'll try and hurry up and release the sourcecode as soon as possible to serve as a reference to help clear up all these implementation questions.

Ray Dillinger (Bear) wrote:
> When a coin is spent, the buyer and seller digitally sign a (blinded)
> transaction record.

Only the buyer signs, and there's no blinding.

> If someone double spends, then the transaction record
> can be unblinded revealing the identity of the cheater.

Identities are not used, and there's no reliance on recourse. It's all prevention.

> This is done via a fairly standard cut-and-choose
> algorithm where the buyer responds to several challenges
> with secret shares

No challenges or secret shares. A basic transaction is just what you see in the figure in section 2. A signature (of the buyer) satisfying the public key of the previous transaction, and a new public key (of the seller) that must be satisfied to spend it the next time.

> They may also receive chains as long as the one they're trying to
> extend while they work, in which the last few "links" are links
*> that are *not* in common with the chain on which they're working.*
> These they ignore.

Right, if it's equal in length, ties are broken by keeping the earliest one received.

> If it contains a double spend, then they create a "transaction"
> which is a proof of double spending, add it to their pool A,
> broadcast it, and continue work.

There's no need for reporting of "proof of double spending" like that. If the same chain contains both spends, then the block is invalid and rejected.

Same if a block didn't have enough proof-of-work. That block is invalid and rejected.

There's no need to circulate a report about it. Every node could see that and reject it before relaying it.

If there are two competing chains, each containing a different version of the same transaction, with one trying to give money to one person and the other trying to give the same money to someone else, resolving which of the spends is valid is what the whole proof-of-work chain is about.

We're not "on the lookout" for double spends to sound the alarm and catch the cheater. We merely adjudicate which one of the spends is valid. Receivers of transactions must wait a few blocks to make sure that resolution has had time to complete. Would be cheaters can try and simultaneously double-spend all they want, and all they accomplish is that within a few blocks, one of the spends becomes valid and the others become invalid. Any later double-spends are immediately rejected once there's already a spend in the main chain.

Even if an earlier spend wasn't in the chain yet, if it was already in all the nodes' pools, then the second spend would be turned away by all those nodes that already have the first spend.

> If the new chain is accepted, then they give up on adding their
> current link, dump all the transactions from pool L back into pool
> A (along with transactions they've received or created since
> starting work), eliminate from pool A those transaction records
> which are already part of a link in the new chain, and start work
> again trying to extend the new chain.

Right. They also refresh whenever a new transaction comes in, so L pretty much contains everything in A all the time.

> CPU-intensive digital signature algorithm to
> sign the chain including the new block L.

It's a Hashcash style SHA-256 proof-of-work (partial pre-image of zero), not a signature.

> Is there a mechanism to make sure that the "chain" does not consist
> solely of links added by just the 3 or 4 fastest nodes? 'Cause a
> broadcast transaction record could easily miss those 3 or 4 nodes
> and if it does, and those nodes continue to dominate the chain, the
> transaction might never get added.

If you're thinking of it as a CPU-intensive digital signing, then you may be thinking of a race to finish a long operation first and the fastest always winning.

The proof-of-work is a Hashcash style SHA-256 collision finding. It's a memoryless process where you do millions of hashes a second, with a small chance of finding one each time. The 3 or 4 fastest nodes' dominance would only be proportional to their share of the total CPU power. Anyone's chance of finding a solution at any time is proportional to their CPU power.

There will be transaction fees, so nodes will have an incentive to receive and include all the transactions they can. Nodes will eventually be compensated by transaction fees alone when the total coins created hits the pre-determined ceiling.

> Also, the work requirement for adding a link to the chain should
> vary (again exponentially) with the number of links added to that
> chain in the previous week, causing the rate of coin generation
> (and therefore inflation) to be strictly controlled.

Right.

> You need coin aggregation for this to scale. There needs to be
> a "provable" transaction where someone retires ten single coins
> and creates a new coin with denomination ten, etc.

Every transaction is one of these. Section 9, Combining and Splitting Value.

Satoshi Nakamoto

Cryptography Mailing List

Bitcoin P2P e-cash paper

2008-11-15 18:02:00 UTC - -

Ray Dillinger wrote:
> One way to do this would be
> to have the person recieving the coin generate an asymmetric
> key pair, and then have half of it published with the
> transaction. In order to spend the coin later, s/he must
> demonstrate posession of the other half of the asymmetric
> key pair, probably by using it to sign the key provided by
> the new seller.

Right, it's ECC digital signatures. A new key pair is used for every transaction.

It's not pseudonymous in the sense of nyms identifying people, but it

is at least a little pseudonymous in that the next action on a coin can be identified as being from the owner of that coin.

> Mmmm. I don't know if I'm comfortable with that. You're saying
> there's no effort to identify and exclude nodes that don't
> cooperate? I suspect this will lead to trouble and possible DOS
> attacks.

There is no reliance on identifying anyone. As you've said, it's futile and can be trivially defeated with sock puppets.

The credential that establishes someone as real is the ability to supply CPU power.

> Until.... until what? How does anybody know when a transaction
> has become irrevocable? Is "a few" blocks three? Thirty? A
> hundred? Does it depend on the number of nodes? Is it logarithmic
> or linear in number of nodes?

Section 11 calculates the worst case under attack. Typically, 5 or 10 blocks is enough for that. If you're selling something that doesn't merit a network-scale attack to steal it, in practice you could cut it closer.

> But in the absence of identity, there's no downside to them
> if spends become invalid, if they've already received the
> goods they double-spent for (access to website, download,
> whatever). The merchants are left holding the bag with
> "invalid" coins, unless they wait that magical "few blocks"
> (and how can they know how many?) before treating the spender
> as having paid.
>
> The consumers won't do this if they spend their coin and it takes
> an hour to clear before they can do what they spent their coin on.
> The merchants won't do it if there's no way to charge back a
> customer when they find the that their coin is invalid because
> the customer has doublespent.

This is a version 2 problem that I believe can be solved fairly satisfactorily for most applications.

The race is to spread your transaction on the network first. Think 6 degrees of freedom -- it spreads exponentially. It would only take something like 2 minutes for a transaction to spread widely enough

that a competitor starting late would have little chance of grabbing very many nodes before the first one is overtaking the whole network. During those 2 minutes, the merchant's nodes can be watching for a double-spent transaction. The double-spender would not be able to blast his alternate transaction out to the world without the merchant getting it, so he has to wait before starting.

If the real transaction reaches 90% and the double-spent tx reaches 10%, the double-spender only gets a 10% chance of not paying, and 90% chance his money gets spent. For almost any type of goods, that's not going to be worth it for the scammer.

Information based goods like access to website or downloads are non-fencible. Nobody is going to be able to make a living off stealing access to websites or downloads. They can go to the file sharing networks to steal that. Most instant-access products aren't going to have a huge incentive to steal.

If a merchant actually has a problem with theft, they can make the customer wait 2 minutes, or wait for something in e-mail, which many already do. If they really want to optimize, and it's a large download, they could cancel the download in the middle if the transaction comes back double-spent. If it's website access, typically it wouldn't be a big deal to let the customer have access for 5 minutes and then cut off access if it's rejected. Many such sites have a free trial anyway.

Satoshi Nakamoto

Cryptography Mailing List

Bitcoin P2P e-cash paper

2008-11-17 17:24:43 UTC - -

James A. Donald wrote:
> > Fortunately, it's only necessary to keep a
> > pending-transaction pool for the current best branch.
>
> This requires that we know, that is to say an honest
> well behaved peer whose communications and data storage
> is working well knows, what the current best branch is -

I mean a node only needs the pending-tx pool for the best branch it has. The branch that it currently thinks is the best branch. That's the branch it'll be trying to make a block out of, which is all it needs the pool for.

> > Broadcasts will probably be almost completely
> > reliable.
>
> Rather than assuming that each message arrives at least
> once, we have to make a mechanism such that the
> information arrives even though conveyed by messages
> that frequently fail to arrive.

I think I've got the peer networking broadcast mechanism covered.

Each node sends its neighbours an inventory list of hashes of the new blocks and transactions it has. The neighbours request the items they don't have yet. If the item never comes through after a timeout, they request it from another neighbour that had it. Since all or most of the neighbours should eventually have each item, even if the coms get fumbled up with one, they can get it from any of the others, trying one at a time.

The inventory-request-data scheme introduces a little latency, but it ultimately helps speed more by keeping extra data blocks off the transmit queues and conserving bandwidth.

> You have an outline
> and proposal for such a design, which is a big step
> forward, but the devil is in the little details.

I believe I've worked through all those little details over the last year and a half while coding it, and there were a lot of them. The functional details are not covered in the paper, but the sourcecode is coming soon. I sent you the main files.
(available by request at the moment, full release soon)

Satoshi Nakamoto

bitcoin-list

[**bitcoin-list**] Welcome

2008-12-10 17:00:23 UTC - -

Welcome to the Bitcoin mailing list!

Genesis Block

Bitcoin v0.1 released

2009-01-03 18:15:05 UTC - -

The Times 03/Jan/2009 Chancellor on brink of second bailout for banks

```
00000000   01 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ................
00000010   00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ................
00000020   00 00 00 00 3B A3 ED FD  7A 7B 12 B2 7A C7 2C 3E   ....;£íýz{.²zÇ,>
00000030   67 76 8F 61 7F C8 1B C3  88 8A 51 32 3A 9F B8 AA   gv.a.È.Ã^ŠQ2:Ÿ¸ª
00000040   4B 1E 5E 4A 29 AB 5F 49  FF FF 00 1D 1D AC 2B 7C   K.^J)«_Iÿÿ...¬+|
00000050   01 01 00 00 00 01 00 00  00 00 00 00 00 00 00 00   ................
00000060   00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ................
00000070   00 00 00 00 00 00 FF FF  FF FF 4D 04 FF FF 00 1D   ......ÿÿÿÿM.ÿÿ..
00000080   01 04 45 54 68 65 20 54  69 6D 65 73 20 30 33 2F   ..EThe Times 03/
00000090   4A 61 6E 2F 32 30 30 39  20 43 68 61 6E 63 65 6C   Jan/2009 Chancel
000000A0   6C 6F 72 20 6F 6E 20 62  72 69 6E 6B 20 6F 66 20   lor on brink of
000000B0   73 65 63 6F 6E 64 20 62  61 69 6C 6F 75 74 20 66   second bailout f
000000C0   6F 72 20 62 61 6E 6B 73  FF FF FF FF 01 00 F2 05   or banksÿÿÿÿ..ò.
000000D0   2A 01 00 00 00 43 41 04  67 8A FD B0 FE 55 48 27   *....CA.gŠý°þUH'
000000E0   19 67 F1 A6 71 30 B7 10  5C D6 A8 28 E0 39 09 A6   .gñ¦q0·.\Ö¨(à9.¦
000000F0   79 62 E0 EA 1F 61 DE B6  49 F6 BC 3F 4C EF 38 C4   ybàê.aÞ¶Iö¼?Lï8Ä
00000100   F3 55 04 E5 1E C1 12 DE  5C 38 4D F7 BA 0B 8D 57   óU.å.Á.Þ\8M÷º..W
00000110   8A 4C 70 2B 6B F1 1D 5F  AC 00 00 00 00           ŠLp+kñ._¬....
```

View the Genesis block on blockchain.com at https://www.blockchain.com/btc/block/0000000000019d6689c085ae165831e934ff763ae46a2a6c172b3f1b60a8ce26f (shortcut: goo.gl/hm5ZzY) and at the encrypted note by Satoshi at https://www.blockchain.com/btc/tx/4a5e1e4baab89f3a32518a88c31bc87f618f76673e2cc77ab2127b7afdeda33b (shortcut: goo.gl/2csRje). Image credit: Bitcoin Wiki at https://en.bitcoin.it/wiki/Genesis_block

Cryptography Mailing List

Bitcoin v0.1 released

2009-01-08 19:27:40 UTC - -

Announcing the first release of Bitcoin, a new electronic cash system that uses a peer-to-peer network to prevent double-spending. It's completely decentralized with no server or central authority.

See bitcoin.org for screenshots.

Download link:
http://downloads.sourceforge.net/bitcoin/bitcoin-0.1.0.rar

Windows only for now. Open source C++ code is included.

- Unpack the files into a directory
- Run BITCOIN.EXE
- It automatically connects to other nodes

If you can keep a node running that accepts incoming connections, you'll really be helping the network a lot. Port 8333 on your firewall needs to be open to receive incoming connections.

The software is still alpha and experimental. There's no guarantee the system's state won't have to be restarted at some point if it becomes necessary, although I've done everything I can to build in extensibility and versioning.

You can get coins by getting someone to send you some, or turn on Options->Generate Coins to run a node and generate blocks. I made the proof-of-work difficulty ridiculously easy to start with, so for a little while in the beginning a typical PC will be able to generate coins in just a few hours. It'll get a lot harder when competition makes the automatic adjustment drive up the difficulty. Generated coins must wait 120 blocks to mature before they can be spent.

There are two ways to send money. If the recipient is online, you can enter their IP address and it will connect, get a new public key and send the transaction with comments. If the recipient is not online, it is possible to send to their Bitcoin address, which is a hash of their public key that they give you. They'll receive the transaction the next time they connect and get the block it's in. This method has the disadvantage that no comment information is sent, and a bit of privacy may be lost if the address is used multiple times, but it is a useful alternative if both users can't be online at the same time or the recipient can't receive incoming connections.

Total circulation will be 21,000,000 coins. It'll be distributed to network nodes when they make blocks, with the amount cut in half every 4 years.

first 4 years: 10,500,000 coins
next 4 years: 5,250,000 coins
next 4 years: 2,625,000 coins
next 4 years: 1,312,500 coins
etc...

When that runs out, the system can support transaction fees if needed. It's based on open market competition, and there will

probably always be nodes willing to process transactions for free.

Satoshi Nakamoto

---------- Forwarded message ----------
From: **Satoshi Nakamoto** <satoshi@vistomail.com>
Date: Sat, Jan 10, 2009 at 11:52 AM
Subject: RE:Crash in bitcoin 0.1.0
To: hal.finney@gmail.com

Normally I would keep the symbols in, but they increased the size of the EXE from 6.5MB to 50MB so I
just couldn't justify not stripping them. I guess I made the wrong decision, at least for this early
version. I'm kind of surprised there was a crash, I've tested heavily and haven't had an outright
exception for a while. Come to think of it, there isn't even an exception print at the end of debug.log. I've been testing on XP SP2, maybe SP3 is something.

I've attached bitcoin.exe with symbols. (gcc symbols for gdb, if you're using MSVC I can send you an
MSVC build with symbols)

Thanks for your help!

>Hi Satoshi - I tried running bitcoin.exe from the 0.1.0 package, and
>it crashed. I am running on an up to date version of XP, SP3. The
>debug.log output is attached. There was also a file db.log but it was
>empty.
>
>The crash allowed me to start up a debugger, but there were no
>symbols. The exception was at address 00930AF7. The displayed call
>stack was 942316 called by 508936.
>
>When I have a chance, I'll try building it, although it looks like it
>would take me a while to acquire all the dependencies.
>
>Hal

From: **Satoshi Nakamoto** <satoshi@vistomail.com>
Date: Sat, Jan 10, 2009 at 2:59 PM
Subject: Re: Crash in bitcoin 0.1.0
To: hal.finney@gmail.com

I was temporarily able to reproduce the bug and narrowed it down to the "mapAddresses.count" in the
following code. It was absolutely the last piece of code to go in and mainly only got tested with the
MSVC build. It's not essential and I'm inclined to turn off optimization and delete the section of code
until I figure out what's going on.

I'm attaching a dbg exe you can try that deletes the line of code and turns off optimization. I'm not able
to reproduce it anymore at the moment.

irc.cpp:

```
if (pszName[0] == 'u')
{
CAddress addr;
if (DecodeAddress(pszName, addr))
{
CAddrDB addrdb;
if (AddAddress(addrdb, addr))
printf("new ");
else
{
// make it try connecting sooner
CRITICAL_BLOCK(cs_mapAddresses)
if (mapAddresses.count(addr.GetKey()))
mapAddresses[addr.GetKey()].nLastFailed = 0;
}
addr.print();
}
else
{
printf("decode failed\n");
}
}
```

>Yes, actually the version with MSVC symbols would be better, that is
>the one I am using.
>
>I found that if I launched this one from a cygwin shell, it does not
>crash. But if I launch it from Windows, double-clicking on the file,
>it does crash similarly to the previous version. However, I am pretty
>sure that the previous version did crash even when I launched it from
>cygwin.
>
>I have to go out but I'll leave this version running for a while.
>
>Hal

---------- Forwarded message ----------
From: **Satoshi Nakamoto** <satoshi@vistomail.com>
Date: Sat, Jan 10, 2009 at 6:55 PM
Subject: Re: Crash in bitcoin 0.1.0
To: hal.finney@gmail.com

I isolated the problem. If I spawn a thread and do
mapAddresses.count, even as the very first thing in the program,
it segfaults. The workaround is to needlessly call
mapAddresses.count in the main thread once and it's fine from then
on. I hate to blame the compiler, and I've never had a GCC
compiler bug before, but this feels like one. Maybe some bit of
init code it tries to optimize out if it's not called at least once
in the same thread, or some STL optimization that's not thread
friendly. I'm really dismayed to have this botch up the release
after all that stress testing.

The attached file: bitcoin-0.1.1.rar (filesize 2,132,686) is the
version where I deleted the mapAddresses.count line, and that
should be the safest version. (that was the only use of
mapAddresses.count) If you could try this version and confirm
that the crash is fixed, I'd appreciate it.

Thanks,
Satoshi

---------- Forwarded message ----------
From: **Satoshi Nakamoto** <satoshi@vistomail.com>
Date: Sat, Jan 10, 2009 at 7:11 PM
Subject: Re: Crash in bitcoin 0.1.0
To: hal.finney@gmail.com

OK, thanks. The one in bitcoin-0.1.1-exe-dbg.rar is the same build as in bitcoin-0.1.1.rar.

I forgot, when you build debug on MSVC, it uses the debug versions of the runtime DLLs, which aren't
included with Windows distributions. Actually, MSVC 6.0's runtime (MSVC60.DLL) is the last version that
shipped preinstalled on Windows, which is why the continued interest in that ancient version of the
compiler. Later Visual C versions can't create a standalone EXE that doesn't require additional runtime
packages installed.

I can't use MSVC 6.0 for the release because its optimization of the SHA-256 routines is too slow.

I've attached a copy of the debug runtime DLLs. (They're redistributable)

>Hi Satoshi - The version with the .pdb file did not run for me, I got
>an error about MSVCP60D.DLL not being found. I imagine this is due to
>the version incompatibility you were worried about.
>
>The next version, that deleted the questionable line of code and
>turned off optimization, seems to run fine for me. So the problem may
>be related to that bit.
>
>Hal

---------- Forwarded message ----------
From: **Satoshi Nakamoto** <satoshi@vistomail.com>
Date: Sun, Jan 11, 2009 at 4:36 PM

Subject: How's v0.1.2 going?
To: hal.finney@gmail.com

Well this doesn't look good. After you upgraded to 0.1.2, your node responded to one or two messages
and then stopped replying to messages. It's still accepting connections and seems to be alive on
IRC. That could happen if ThreadSocketHandler or ThreadMessageHandler is hung or crashed or
blocked. Usually when there's an exception or other problem, it only stops the affected thread and
everything else keeps running.

I'm attaching the msvc debug version in case you need it.

Satoshi

---------- Forwarded message ----------
From: **Satoshi Nakamoto** <satoshi@vistomail.com>
Date: Sun, Jan 11, 2009 at 4:49 PM
Subject: v0.1.2 gcc debug build attached
To: hal.finney@gmail.com

Could you send me your debug.log?

The gcc debug version is attached.

gdb is easier to use than you'd think. gdb.exe is the only file. You run
gdb bitcoin.exe
then type "run"
then if it crashes, type "backtrace" for a stack dump, or it may do it automatically. (The stack trace
doesn't always go far enough back unfortunately)

---------- Forwarded message ----------
From: **Satoshi Nakamoto** <satoshi@vistomail.com>

Date: Sun, Jan 11, 2009 at 5:25 PM
Subject: Re: v0.1.2 debug.log
To: hal.finney@gmail.com

OK, so no crash or exception window or anything. debug.log is all I need then.

It looks like there's a "select failed: 10038" error (the sockets select function failed) and then network
communication goes quiet after that (except for IRC which is still working). I've never had select fail
before. It looks like sockets is somehow partially hosed. At least now I know what's wrong now.

You should restart it. It's not doing anything right now. I don't know if it'll just get the "select failed"
error again, or be fine for a while.

If I can't think of anything else, I can always shut down and restart sockets if it gets hosed like that. I'm
sure everyone who's written an internet app like a browser or p2p app had to slog through all the ways
the Internet can trash you. The Internet is a brutal, rough and tumble place.

The issue of bitcoin.exe still running after you close it is a known issue. It does a careful shutdown of
everything to be extra safe, in case some important transaction is in progress, but it's completely fine
and totally safe to just kill it if it doesn't exit on its own. I'll have to work on figuring out what's getting
hung up. I may just have it kill itself after a timeout.

Thanks!

>Hi Satoshi - debug.log attached. When I started 0.1.2 this afternoon,
>I first quit the previous version which was running. However, 0.1.2
>would not start up. Looking at the debug log, it said "Existing
>instance found". I ran task manager, and found two processes called
>bitcoin.exe running. I killed them both and started up the new one,
>and it seemed to run OK. It says at the bottom "3 connections". I

>haven't tried the debug version, I'm not sure what I would look for.
>
>Hal

---------- Forwarded message ----------
From: **Satoshi Nakamoto** <satoshi@vistomail.com>
Date: Sun, Jan 11, 2009 at 9:31 PM
Subject: select failed 10038 fix
To: hal.finney@gmail.com

I believe I've fixed the bug related to "select failed: 10038"
(error WSAENOTSOCK). The select error is not a big deal, but it
led the communications thread to get blocked on a socket that
should have been in non-blocking mode but wasn't. It never came
up until now because as long as select never failed, receive would
never be called unless there was data.

Without this fix, your node's communication sometimes goes dead.
Connections are still made, but no data is passed. Any generated
blocks would probably not be accepted since you can't broadcast
them and other nodes will leave your branch behind. That's why
Generate doesn't run when you're not connected.

This could also have caused bitcoin.exe to fail to exit. There's
no reason for shutdown to wait for the com thread, so I made it
only wait for the message processing thread. I'll do a more
thorough forced shutdown later.

Looks like your node's com thread just now got blocked on this
bug again. It went for a few hours this time before it did.

Version 0.1.3 exe attached.

bitcoin-list

[**bitcoin-list**] Bitcoin v0.1.2 now available

2009-01-11 22:32:18 UTC - -

Bitcoin v0.1.2 is now available for download.

See http://www.bitcoin.org for the download link.

All the problems I've been finding are in the code that automatically finds and connects to other nodes, since I wasn't able to test it in the wild until now. There are many more ways for connections to get screwed up on the real Internet.

Bugs fixed:
- Fixed various problems that were making it hard for new nodes to see other nodes to connect to.
- If you're behind a firewall, it could only receive one connection, and the second connection would constantly disconnect and reconnect.

These problems are kind of screwing up the network and will get worse as more users arrive, so please make sure to upgrade.

Satoshi Nakamoto

From dtrammell@dustintrammell.com Sun Jan 11 23:14:04 2009

On Fri, 2009-01-09 at 03:27 +0800, Satoshi Nakamoto wrote:
> Announcing the first release of Bitcoin, a new electronic cash
> system that uses a peer-to-peer network to prevent double-spending.
> It's completely decentralized with no server or central authority.

I'm currently reading through your paper. At the timestamp server section you mention newspapers and usenet, so I thought you might be interested in this if you have not seen it already:

http://www.publictimestamp.org/

By the way, I'm also currently running the alpha code on one of my workstations. So far it has two "Generated" messages, however the "Credit" field for those is 0.00 and the balance hasn't changed. Is this due to the age/maturity requirement for a coin to be valid?

Cheers,

Dustin D. Trammell
dtrammell@dustintrammell.com
http://www.dustintrammell.com

---------- Forwarded message ----------
From: **Satoshi Nakamoto** <satoshi@vistomail.com>
Date: Mon, Jan 12, 2009 at 8:41 AM
Subject: Re: select failed 10038 fix
To: hal.finney@gmail.com

It definitely looks like 0.1.3 solved it. It was getting so there
were so many zombie nodes, I was having a hard time getting a
reply to any of my messages. Now, four inventory messages go out,
four getdata messages come back.

Did you get any "not accepted" blocks? The connectivity bug could
have caused a generated block not to be accepted if the node
wasn't able to broadcast at the time. Once the status is above 5
or so it's safely accepted.

Unfortunately, I can't receive incoming connections from where I
am, which has made things more difficult. Your node receiving
incoming connections was the main thing keeping the network going
the first day or two.

You can send to my Bitcoin address if you want to, but you won't
get to see the full transfer sequence:
1NSwywA5Dvuyw89sfs3oLPvLiDNGf48cPD

You could always findstr /c:"version message" debug.log and send a
test to some random person you're connected to near the end of the
list. The ones ending in port 8333 can receive connections.

I just thought of something. Eventually there'll be some interest
in brute force scanning bitcoin addresses to find one with the
first few characters customized to your name, kind of like getting
a phone number that spells out something. Just by chance I have
my initials.

Satoshi

>Thanks, Satoshi, this new version seems to be running much better.
>I've got 8 connections, and watching debug.log there seems to be quite
>a bit of activity. I see you sent me a payment, thanks! Let me know
>your address and I will try sending one to you. I managed to generate
>a block yesterday and the coins are about to mature, if I understand
>it correctly.
>
>Hal

---------- Forwarded message ----------
From: **Satoshi Nakamoto** <satoshi@vistomail.com>
Date: Mon, Jan 12, 2009 at 10:50 AM
Subject: Re: select failed 10038 fix
To: hal.finney@gmail.com

Could you send me the debug.log from the 0.1.3 crash?
I can usually get a lot just from that.

I'll send you the debug builds shortly.

>Looks like 0.1.3 crashed during the night, unfortunately. Next time I
>will try running the debug version. Today I am working and will need
>to take this computer up and down quite a bit, so I won't be able to
>run it for most of the day. Tonight I will try to look at it a little
>bit.
>
>Hal

---------- Forwarded message ----------
From: **Satoshi Nakamoto** <satoshi@vistomail.com>
Date: Mon, Jan 12, 2009 at 11:26 AM
Subject: Re: v0.1.3 msvc debug build
To: hal.finney@gmail.com

Here's the 0.1.3 MSVC debug build

>Looks like 0.1.3 crashed during the night, unfortunately. Next time I
>will try running the debug version. Today I am working and will need
>to take this computer up and down quite a bit, so I won't be able to
>run it for most of the day. Tonight I will try to look at it a little
>bit.
>
>Hal
>
>On Mon, Jan 12, 2009 at 8:41 AM, Satoshi Nakamoto <satoshi@vistomail.com> wrote:
>> It definitely looks like 0.1.3 solved it. It was getting so there
>> were so many zombie nodes, I was having a hard time getting a
>> reply to any of my messages. Now, four inventory messages go out,
>> four getdata messages come back.
>>
>> Did you get any "not accepted" blocks? The connectivity bug could
>> have caused a generated block not to be accepted if the node
>> wasn't able to broadcast at the time. Once the status is above 5
>> or so it's safely accepted.

>>
>> Unfortunately, I can't receive incoming connections from where I
>> am, which has made things more difficult. Your node receiving
>> incoming connections was the main thing keeping the network going
>> the first day or two.
>>
>> You can send to my Bitcoin address if you want to, but you won't
>> get to see the full transfer sequence:
>> 1NSwywA5Dvuyw89sfs3oLPvLiDNGf48cPD
>>
>> You could always findstr /c:"version message" debug.log and send a
>> test to some random person you're connected to near the end of the
>> list. The ones ending in port 8333 can receive connections.
>>
>> I just thought of something. Eventually there'll be some interest
>> in brute force scanning bitcoin addresses to find one with the
>> first few characters customized to your name, kind of like getting
>> a phone number that spells out something. Just by chance I have
>> my initials.
>>
>> Satoshi
>>
>>>Thanks, Satoshi, this new version seems to be running much better.
>>>I've got 8 connections, and watching debug.log there seems to be quite
>>>a bit of activity. I see you sent me a payment, thanks! Let me know
>>>your address and I will try sending one to you. I managed to generate
>>>a block yesterday and the coins are about to mature, if I understand
>>>it correctly.
>>>
>>>Hal
>>>
>>>On Sun, Jan 11, 2009 at 9:31 PM, Satoshi Nakamoto <satoshi@vistomail.com>
wrote:
>>>> I believe I've fixed the bug related to "select failed: 10038"
>>>> (error WSAENOTSOCK). The select error is not a big deal, but it
>>>> led the communications thread to get blocked on a socket that
>>>> should have been in non-blocking mode but wasn't. It never came
>>>> up until now because as long as select never failed, receive would
>>>> never be called unless there was data.
>>>>
>>>> Without this fix, your node's communication sometimes goes dead.
>>>> Connections are still made, but no data is passed. Any generated
>>>> blocks would probably not be accepted since you can't broadcast
>>>> them and other nodes will leave your branch behind. That's why
>>>> Generate doesn't run when you're not connected.
>>>>
>>>> This could also have caused bitcoin.exe to fail to exit. There's

>>>> no reason for shutdown to wait for the com thread, so I made it
>>>> only wait for the message processing thread. I'll do a more
>>>> thorough forced shutdown later.
>>>>
>>>> Looks like your node's com thread just now got blocked on this
>>>> bug again. It went for a few hours this time before it did.
>>>>
>>>> Version 0.1.3 exe attached.

---------- Forwarded message ----------
From: **Satoshi Nakamoto** <satoshi@vistomail.com>
Date: Mon, Jan 12, 2009 at 11:39 AM
Subject: Re: v0.1.3 gcc debug build
To: hal.finney@gmail.com

and the gcc debug build w/gdb.exe

>Looks like 0.1.3 crashed during the night, unfortunately. Next time I
>will try running the debug version. Today I am working and will need
>to take this computer up and down quite a bit, so I won't be able to
>run it for most of the day. Tonight I will try to look at it a little
>bit.
>
>Hal

From satoshi@vistomail.com Mon Jan 12 18:52:45 2009

> I'm currently reading through your paper. At the timestamp server
> section you mention newspapers and usenet, so I thought you might be
> interested in this if you have not seen it already:
>
> http://www.publictimestamp.org/

Thanks, I hadn't seen that yet. It looks very well presented. There was an older one that's been running for a long time that publishes its hashes to Usenet. I'm surprised this one isn't using Usenet, although it is kind of difficult to get access to post to Usenet in an automated way these days. If they can get a magazine or newspaper to publish their hashes, it would work a lot easier in court for their purposes. Bitcoin and all timestamp servers share the basic functionality of periodically collecting things into blocks and hashing them into a chain.

> By the way, I'm also currently running the alpha code on one of my
> workstations. So far it has two "Generated" messages, however the

> "Credit" field for those is 0.00 and the balance hasn't changed. Is
> this due to the age/maturity requirement for a coin to be valid?

Right, the credit field stays 0.00 until it matures, then it'll be 50.00. Do you think it would be clearer if I left the credit field blank until it matures? I should put some text in the transaction details (when you double click on it) explaining how it works. (was it obvious you can doubleclick on a line for details?)

Be sure to upgrade to v0.1.3 if you haven't already. This version has really stabilized things.

Satoshi

bitcoin-list

[**bitcoin-list**] Bitcoin v0.1 Alpha release notes

2009-01-12 20:20:47 UTC - -

Release notes for Bitcoin v0.1 Alpha

Bitcoin is a new electronic cash system that uses a peer-to-peer network to prevent double-spending. It's completely decentralized with no server or central authority.

You can find screenshots and the download link at:
http://www.bitcoin.org

Windows only for now. Open source C++ code is included.

- Unpack the files into a directory
- Run BITCOIN.EXE
- It automatically connects to other nodes

If you can keep a node running that accepts incoming connections, you'll really be helping the network a lot. Port 8333 on your firewall needs to be open to receive incoming connections.

The software is still alpha and experimental. There's no guarantee the system's state won't have to be restarted at some point if it becomes necessary, although I've done everything I can to build in extensibility and versioning.

You can get coins by getting someone to send you some, or turn on Options->Generate Coins to run a node and generate blocks. I made the proof-of-work difficulty ridiculously easy to start with, so

for a little while in the beginning a typical PC will be able to generate coins in just a few hours. It'll get a lot harder when competition makes the automatic adjustment drive up the difficulty. Generated coins must wait 120 blocks to mature before they can be spent.

There are two ways to send money. If the recipient is online, you can enter their IP address and it will connect, get a new public key and send the transaction with comments. If the recipient is not online, it is possible to send to their Bitcoin address, which is a hash of their public key that they give you. They'll receive the transaction the next time they connect and get the block it's in. This method has the disadvantage that no comment information is sent, and a bit of privacy may be lost if the address is used multiple times, but it is a useful alternative if both users can't be online at the same time or the recipient can't receive incoming connections.

Total circulation will be 21,000,000 coins. It'll be distributed to network nodes when they make blocks, with the amount cut in half every 4 years.

first 4 years: 10,500,000 coins
next 4 years: 5,250,000 coins
next 4 years: 2,625,000 coins
next 4 years: 1,312,500 coins
etc...

When that runs out, the system can support transaction fees if needed. It's based on open market competition, and there will probably always be nodes willing to process transactions for free.

Satoshi Nakamoto

bitcoin-list

[**bitcoin-list**] Bitcoin v0.1.3

2009-01-12 22:48:23 UTC - -

It looks like we're through with the worst of the Internet connection issues. 0.1.3 fixed a problem where your node's communications could go dead after a while. The network is running much more smoothly now with this version.

If you've successfully generated a block, you've seen it has a

maturation countdown before you can spend it. Once it matures, the Credit column will change from 0.00 to 50.00. For a block to be valid, it has to be broadcasted to the network and get into the block chain, which is why Generate does not run if you're not connected. If you generated a block without being connected, the network wouldn't know about it and would continue building the chain without it, leaving it behind, and the maturation countdown would change to "(not accepted)" when your node sees that it wasn't used. If you subtract 1 from the status column, that's how many blocks have been chained after yours.

Satoshi Nakamoto

---------- Forwarded message ----------
From: **Satoshi Nakamoto** <satoshi@vistomail.com>
Date: Mon, Jan 12, 2009 at 11:59 PM
Subject: Re: select failed 10038 fix
To: hal.finney@gmail.com

Definitely the disk full. I completely put off disk full handling until a later version. Probably about time I did it now.

Well, that's a relief.

Satoshi

>Hi Satoshi - Sorry I have not been able to do more today, this looks
>like a busy week for me. I started 0.1.3 again under the MSVC debugger
>this time so if it crashes tonight I may be able to get some more
>information.
>
>I remember now that last night, my disk filled up. I had downloaded a
>bunch of the dependencies (boost, etc) with an eye towards trying to
>build it myself, and my disk was already pretty full. I'm pretty sure
>this is what caused 0.1.3 to crash. I've attached the debug.log, which
>also includes some other runs. The error is about 1/3 of the way down
>and says,
>
>EXCEPTION: NSt8ios_base7failureE
>CAutoFile::read : end of file
>
>Normally this should be a rare occurrence with the large disk sizes
>people have today.
>
>Hal
>

>On 1/12/09, Satoshi Nakamoto <satoshi@vistomail.com> wrote:
>> Could you send me the debug.log from the 0.1.3 crash?
>> I can usually get a lot just from that.
>>
>> I'll send you the debug builds shortly.
>>
>>
>>>Looks like 0.1.3 crashed during the night, unfortunately. Next time I
>>>will try running the debug version. Today I am working and will need
>>>to take this computer up and down quite a bit, so I won't be able to
>>>run it for most of the day. Tonight I will try to look at it a little
>>>bit.
>>>
>>>Hal

---------- Forwarded message ----------
From: **Satoshi Nakamoto** <satoshi@vistomail.com>
Date: Tue, Jan 13, 2009 at 2:42 PM
Subject: Re: disk full
To: hal.finney@gmail.com

If you build the dependencies, let me know how that goes.
Everything is always harder to build on Windows than Linux. I've
always hated projects with a lot of big dependencies, but there's
no avoiding it, each one is essential.

I still haven't figured out how you managed to get a read
exception rather than a write exception when your disk filled up.
It's unlikely but maybe possible that the incident could have
messed up your block data file. In that case, it might manifest
as a similar exception again, or if your block count in the status
bar stopped going up, that would also indicate a problem. As of
this moment it's at 375 blocks.

If there is a problem, it could easily be solved by deleting your
block files, as follows:

(exit Bitcoin and make sure it's stopped)
cd /d "%appdata%\bitcoin"
(backup this directory first)
del blk0001.dat
del blkindex.dat

It'll then re-download the block chain. Your transactions and
generated blocks show as 0/unconfirmed until it's done downloading.

The crucial file to backup is wallet.dat. If bitcoin is running then you have to backup the whole %appdata%\bitcoin directory including the database subdirectory, but even if it's not running it certainly feels safer to always backup the whole directory.

The database unfortunately names its files "log.0000000001". To the rest of the world, "log" means delete-at-will, but to database people it means delete-and-lose-everything-in-your-other-files. I tried to put them out of harm's way by putting them in the database subdirectory. Later I'll write code to flush the logs after every wallet change so wallet.dat will be standalone safe almost all the time.

Satoshi

>Hi Satoshi - Sorry I have not been able to do more today, this looks
>like a busy week for me. I started 0.1.3 again under the MSVC debugger
>this time so if it crashes tonight I may be able to get some more
>information.
>
>I remember now that last night, my disk filled up. I had downloaded a
>bunch of the dependencies (boost, etc) with an eye towards trying to
>build it myself, and my disk was already pretty full. I'm pretty sure
>this is what caused 0.1.3 to crash. I've attached the debug.log, which
>also includes some other runs. The error is about 1/3 of the way down
>and says,
>
>EXCEPTION: NSt8ios_base7failureE
>CAutoFile::read : end of file
>
>Normally this should be a rare occurrence with the large disk sizes
>people have today.
>
>Hal

Cryptography Mailing List

Bitcoin v0.1 released

2009-01-16 16:03:14 UTC - -

> Dustin D. Trammell wrote:
> > Satoshi Nakamoto wrote:
> > You know, I think there were a lot more people interested in the 90's,
> > but after more than a decade of failed Trusted Third Party based systems
> > (Digicash, etc), they see it as a lost cause. I hope they can make the

> > distinction that this is the first time I know of that we're trying a
> > non-trust-based system.
>
> Yea, that was the primary feature that caught my eye. The real trick
> will be to get people to actually value the BitCoins so that they become
> currency.

I would be surprised if 10 years from now we're not using electronic currency in some way, now that we know a way to do it that won't inevitably get dumbed down when the trusted third party gets cold feet.

It could get started in a narrow niche like reward points, donation tokens, currency for a game or micropayments for adult sites. Initially it can be used in proof-of-work applications for services that could almost be free but not quite.

It can already be used for pay-to-send e-mail. The send dialog is resizeable and you can enter as long of a message as you like. It's sent directly when it connects. The recipient doubleclicks on the transaction to see the full message. If someone famous is getting more e-mail than they can read, but would still like to have a way for fans to contact them, they could set up Bitcoin and give out the IP address on their website. "Send X bitcoins to my priority hotline at this IP and I'll read the message personally."

Subscription sites that need some extra proof-of-work for their free trial so it doesn't cannibalize subscriptions could charge bitcoins for the trial.

It might make sense just to get some in case it catches on. If enough people think the same way, that becomes a self fulfilling prophecy. Once it gets bootstrapped, there are so many applications if you could effortlessly pay a few cents to a website as easily as dropping coins in a vending machine.

Satoshi Nakamoto
http://www.bitcoin.org

bitcoin-list

Re: [**bitcoin-list**] Bitcoin v0.1 released

2009-01-16 18:35:32 UTC - -

> Dustin D. Trammell wrote:
> > Satoshi Nakamoto wrote:
> > You know, I think there were a lot more people interested in the 90's,
> > but after more than a decade of failed Trusted Third Party based systems
> > (Digicash, etc), they see it as a lost cause. I hope they can make the
> > distinction that this is the first time I know of that we're trying a
> > non-trust-based system.
>
> Yea, that was the primary feature that caught my eye. The real trick
> will be to get people to actually value the BitCoins so that they become
> currency.

I would be surprised if 10 years from now we're not using
electronic currency in some way, now that we know a way to do it
that won't inevitably get dumbed down when the trusted third party
gets cold feet.

It could get started in a narrow niche like reward points,
donation tokens, currency for a game or micropayments for adult
sites. Initially it can be used in proof-of-work applications
for services that could almost be free but not quite.

It can already be used for pay-to-send e-mail. The send dialog is
resizeable and you can enter as long of a message as you like.
It's sent directly when it connects. The recipient doubleclicks
on the transaction to see the full message. If someone famous is
getting more e-mail than they can read, but would still like to
have a way for fans to contact them, they could set up Bitcoin and
give out the IP address on their website. "Send X bitcoins to my
priority hotline at this IP and I'll read the message personally."

Subscription sites that need some extra proof-of-work for their
free trial so it doesn't cannibalize subscriptions could charge
bitcoins for the trial.

It might make sense just to get some in case it catches on. If
enough people think the same way, that becomes a self fulfilling
prophecy. Once it gets bootstrapped, there are so many
applications if you could effortlessly pay a few cents to a
website as easily as dropping coins in a vending machine.

Satoshi Nakamoto
http://www.bitcoin.org

---------- Forwarded message ----------
From: **Satoshi Nakamoto** <satoshi@vistomail.com>
Date: Sat, Jan 24, 2009 at 4:47 PM
Subject: Re: disk full
To: hal.finney@gmail.com

I hate duplicating code, but the compiler forces us. Copy the body
of the function above it, like this:

```
    void insert(iterator it, const_iterator first, const_iterator last)
{
if (it == vch.begin() + nReadPos && last - first <= nReadPos)
{
// special case for inserting at the front when there's room
nReadPos -= (last - first);
memcpy(&vch[nReadPos], &first[0], last - first);
}
else
vch.insert(it, first, last);
}
#if !defined(_MSC_VER) || _MSC_VER >= 1300
void insert(iterator it, const char* first, const char* last)
{
if (it == vch.begin() + nReadPos && last - first <= nReadPos)
{
// special case for inserting at the front when there's room
nReadPos -= (last - first);
memcpy(&vch[nReadPos], &first[0], last - first);
}
else
vch.insert(it, first, last);
}
#endif
```

The modified version of serialize.h is attached.

BTW, in my tests, VC8 produced an EXE that would only run on
systems that had VC8 installed on them. The error it gives
is extremely vague. I think they expect you to install a
package during setup, but bitcoin doesn't have a setup.

My testing has been with MSVC 6.0 SP6 and GCC 3.4.5.
GCC is the release build. There's nothing wrong with the
MSVC 6.0 build other than its optimization of the SHA routines
for generating blocks is slow.

Satoshi

Cryptography Mailing List

Bitcoin v0.1 released

2009-01-25 15:47:10 UTC - -

Hal Finney wrote:
> > * Spammer botnets could burn through pay-per-send email filters
> > trivially
> If POW tokens do become useful, and especially if they become money,
> machines will no longer sit idle. Users will expect their computers to
> be earning them money (assuming the reward is greater than the cost to
> operate). A computer whose earnings are being stolen by a botnet will
> be more noticeable to its owner than is the case today, hence we might
> expect that in that world, users will work harder to maintain their
> computers and clean them of botnet infestations.

Another factor that would mitigate spam if POW tokens have value:
there would be a profit motive for people to set up massive
quantities of fake e-mail accounts to harvest POW tokens from
spam. They'd essentially be reverse-spamming the spammers with
automated mailboxes that collect their POW and don't read the
message. The ratio of fake mailboxes to real people could become
too high for spam to be cost effective.

The process has the potential to establish the POW token's value
in the first place, since spammers that don't have a botnet could
buy tokens from harvesters. While the buying back would
temporarily let more spam through, it would only hasten the
self-defeating cycle leading to too many harvesters exploiting the
spammers.

Interestingly, one of the e-gold systems already has a form of
spam called "dusting". Spammers send a tiny amount of gold dust
in order to put a spam message in the transaction's comment field.
If the system let users configure the minimum payment they're
willing to receive, or at least the minimum that can have a
message with it, users could set how much they're willing to get
paid to receive spam.

Satoshi Nakamoto

Re: [**bitcoin-list**] Problems

2009-01-25 16:45:25 UTC - -

From: Nicholas Bohm 2009-01-25 10:17
> I have had a couple of problems running bitcoin: is this an appropriate
> list for reporting them (with about 70kb of attachments)?

What's the problem you're having?

If you send me your debug.log file directly (best not to send attachments to the list), I can take a look at what's happening.

Satoshi Nakamoto
bitcoin-help at vistomail dot com

bitcoin-list

[**bitcoin-list**] Bitcoin v0.1.5 released

2009-02-04 19:46:04 UTC - -

Version 0.1.5 is now available. It includes the fix for the problem Nicholas had, checking for disk full and changes to try to improve things that were confusing.

Special thanks to Nicholas and Dustin for all their help and feedback!

Download link:
http://sourceforge.net/project/showfiles.php?group_id=244765&package_id=298441

Changes:
- disk full warning
- fixed a bug that could occur if dns lookup failed
- prevent entering your own address in the address book,
which confusingly changed the label for your own address
- moved change address button to menu under options
- tweaks to make it get connected faster
- close sockets on exit
- created minimum fee for transactions less than 1 cent
- hid the transaction-type selection box that only had one choice
- cleaned up ParseMoney a little
- slightly cleaner reformatting of message text
- changed the font in transaction details dialog
- added some explanation text to transaction details for generated coins
- reworded the description for transactions received with bitcoin address

Satoshi Nakamoto
http://www.bitcoin.org

P2P Foundation

Bitcoin open source implementation of P2P currency

2009-02-11 22:27:00 UTC - -

I've developed a new open source P2P e-cash system called Bitcoin. It's completely decentralized, with no central server or trusted parties, because everything is based on crypto proof instead of trust. Give it a try, or take a look at the screenshots and design paper:

Download Bitcoin v0.1 at http://www.bitcoin.org

The root problem with conventional currency is all the trust that's required to make it work. The central bank must be trusted not to debase the currency, but the history of fiat currencies is full of breaches of that trust. Banks must be trusted to hold our money and transfer it electronically, but they lend it out in waves of credit bubbles with barely a fraction in reserve. We have to trust them with our privacy, trust them not to let identity thieves drain our accounts. Their massive overhead costs make micropayments impossible.

A generation ago, multi-user time-sharing computer systems had a similar problem. Before strong encryption, users had to rely on password protection to secure their files, placing trust in the system administrator to keep their information private. Privacy could always be overridden by the admin based on his judgment call weighing the principle of privacy against other concerns, or at the behest of his superiors. Then strong encryption became available to the masses, and trust was no longer required. Data could be secured in a way that was physically impossible for others to access, no matter for what reason, no matter how good the excuse, no matter what.

It's time we had the same thing for money. With e-currency based on cryptographic proof, without the need to trust a third party middleman, money can be secure and transactions effortless.

One of the fundamental building blocks for such a system is digital signatures. A digital coin contains the public key of its owner. To transfer it, the owner signs the coin together with the public key of the next owner. Anyone can check the signatures to verify the chain of ownership. It works well to secure ownership, but leaves one big problem unsolved: double-spending. Any owner could try to re-spend an already spent coin by signing it again to another owner. The usual solution is for a trusted company with a central database to check for double-spending, but that just gets back to the trust model. In its central position, the company can override the users, and the fees needed to support the company make micropayments impractical.

Bitcoin's solution is to use a peer-to-peer network to check for double-spending. In a nutshell, the network works like a distributed timestamp server, stamping the first transaction to spend a coin. It takes advantage of the nature of information being easy to spread but hard to stifle. For details on how it works, see the design paper at http://www.bitcoin.org/bitcoin.pdf

The result is a distributed system with no single point of failure. Users hold the crypto keys to their own money and transact directly with each other, with the help of the P2P network to check for double-spending.

Satoshi Nakamoto
http://www.bitcoin.org

P2P Foundation

Bitcoin open source implementation of P2P currency

2009-02-15 16:42:00 UTC - -

Could be. They're talking about the old Chaumian central mint stuff, but maybe only because that was the only thing available. Maybe they would be interested in going in a new direction.

A lot of people automatically dismiss e-currency as a lost cause because of all the companies that failed since the 1990's. I hope it's obvious it was only the centrally controlled nature of those systems that doomed them. I think this is the first time we're trying a decentralized, non-trust-based system.

P2P Foundation

Bitcoin open source implementation of P2P currency

2009-02-18 20:50:00 UTC - -

It is a global distributed database, with additions to the database by consent of the majority, based on a set of rules they follow:

- Whenever someone finds proof-of-work to generate a block, they get some new coins
- The proof-of-work difficulty is adjusted every two weeks to target an average of 6 blocks per hour (for the whole network)
- The coins given per block is cut in half every 4 years

You could say coins are issued by the majority. They are issued in a limited, predetermined amount.

As an example, if there are 1000 nodes, and 6 get coins each hour, it would likely take a week before you get anything.

To Sepp's question, indeed there is nobody to act as central bank or federal reserve to adjust the money supply as the population of users grows. That would have required a trusted party to determine the value, because I don't know a way for software to know the real world value of things. If there was some clever way, or if we wanted to trust someone to actively manage the money supply to peg it to something, the rules could have been programmed for that.

In this sense, it's more typical of a precious metal. Instead of the supply changing to keep the value the same, the supply is predetermined and the value changes. As the number of users grows, the value per coin increases. It has the potential for a positive feedback loop; as users increase, the value goes up, which could attract more users to take advantage of the increasing value.

bitcoin-list

Re: [**bitcoin-list**] Bitcoin v0.1.5 released

2009-02-22 17:47:52 UTC - -

> What's next?

The next thing for v0.1.6 is to take advantage of multiple processors to generate blocks. Currently it only starts one thread. If you have a multi-core processor like a Core Duo or Quad this will double or quadruple your production.

Later I want to add interfaces to make it really easy to integrate into websites from any server side language.

Satoshi

http://www.bitcoin.org

bitcoin-list

Re: [**bitcoin-list**] Bitcoin v0.1.5 released

2009-03-04 16:59:12 UTC - -

Hal Finney wrote:
> That sounds good. I'd also like to be able to run multiple coin/block
> generators on multiple machines, all behind a single NAT address. I

> haven't tried this yet so I don't know if it works on the current
> software.

The current version will work fine. They'll each connect over the
Internet, while incoming connections only come to the host that
port 8333 is routed to.

As an optimisation, I'll make a switch "-connect=1.2.3.4" to make
it only connect to a specific address. You could make your extra
nodes connect to your primary, and only the primary connects over
the Internet. It doesn't really matter for now, since the network
would have to get huge before the bandwidth is anything more than
trivial.

> BTW I don't remember if we talked about this, but the other day some
> people were mentioning secure timestamping. You want to be able to
> prove that a certain document existed at a certain time in the past.
> Seems to me that bitcoin's stack of blocks would be perfect for this.

Indeed, Bitcoin is a distributed secure timestamp server for
transactions. A few lines of code could create a transaction with
an extra hash in it of anything that needs to be timestamped.
I should add a command to timestamp a file that way.

> > > Later I want to add interfaces to make it really easy to integrate
> > > into websites from any server side language.
>
> Right, and I'd like to see more of a library interface that could be
> called from programming or scripting languages, on the client side as
> well.

Exactly.

Satoshi Nakamoto

http://www.bitcoin.org

Mike Hearn <mike@plan99.net> Sun, Apr 12, 2009 at 12:46 PM
To: satoshin@gmx.com

Hi Satoshi,

I read your paper on BitCoin with great interest. I found it a bit confusing though - I believe it may be easier to follow if you provide some examples.

Specifically, it's not quite clear to me what blocks contain. If I understand correctly, there is only one (or maybe a few) global chain[s] into which all transactions are hashed. If there is only one chain recording "the story of the economy" so to speak, how does this scale? In an imaginary planet-wide deployment there would be millions of even billions of transactions per hour being hashed into the chain. I realize that each PoW can wrap many transactions in one block, nonetheless, that's a large amount of data to hash. If there are many chains, how are transactions assigned to each chain such that it is still difficult to overpower the network? Eg, if there are 10 global chains, the amount of cpu power you need to beat the system is only 10% of what it was previously.

I also wonder if the assumption of 1 core = 1 vote is sound. If the majority of nodes are on standard computers, it seems likely that an attacker could use FPGA or custom ASICs to get significantly better performance. What are your thoughts on using custom hardware to beat the chain?

I found the section on incentives hard to follow. In particular, I'm not clear on what triggers the transition from minting new coins as a reason to run a node, to charging transaction fees (isn't the point of BitCoin largely to zero transaction costs anyway?). Presumably there's some human in charge of the system - eg, you decided somehow that 24 million coins was a good number to have, and would distribute some kind of rules file saying "coins minted after this timestamp must have an N+1 zero bits prefix", which honest nodes enforce.

How did you decide on the inflation schedule for v1? Where did 24 million coins come from? What denominations are these coins? You mention a way to combine and split value but I'm not clear on how this works. For instance are bitcoins always denominated by an integer or can you have fractional bitcoins?

So many questions :) But it's rare that I encounter truly revolutionary ideas. The last time I was this excited about a new monetary scheme was when I discovered Ripple. If you have any thoughts on Ripple, I'd also love to hear them.

thanks -mike

Satoshi Nakamoto <satoshin@gmx.com> Sun, Apr 12, 2009 at 10:44

PM

To: Mike Hearn <mike@plan99.net>

Hi Mike,

I'm glad to answer any questions you have. If I get time, I ought to write a FAQ to supplement the paper.

There is only one global chain.

The existing Visa credit card network processes about 15 million Internet purchases per day worldwide. Bitcoin can already scale much larger than that with existing hardware for a fraction of the cost. It never really hits a scale ceiling. If you're interested, I can go over the ways it would cope with extreme size.

By Moore's Law, we can expect hardware speed to be 10 times faster in 5 years and 100 times faster in 10. Even if Bitcoin grows at crazy adoption rates, I think computer speeds will stay ahead of the number of transactions.

I don't anticipate that fees will be needed anytime soon, but if it becomes too burdensome to run a node, it is possible to run a node that only processes transactions that include a transaction fee. The owner of the node would decide the minimum fee they'll accept. Right now, such a node would get nothing, because nobody includes a fee, but if enough nodes did that, then users would get faster acceptance if they include a fee, or slower if they don't. The fee the market would settle on should be minimal. If a node requires a higher fee, that node would be passing up all transactions with lower fees. It could do more volume and probably make more money by processing as many paying transactions as it can. The transition is not controlled by some human in charge of the system though, just individuals reacting on their own to market forces.

Eventually, most nodes may be run by specialists with multiple GPU cards. For now, it's nice that anyone with a PC can play without worrying about what video card they have, and hopefully it'll stay that way for a while. More computers are shipping with fairly decent GPUs these days, so maybe later we'll transition to that.

A key aspect of Bitcoin is that the security of the network grows as the size of the network and the amount of value that needs to be protected grows. The down side is that it's vulnerable at the beginning when it's small, although the value that could be stolen should always be smaller than the amount of effort required to steal it. If someone has other motives to prove a point, they'll just be proving a point I already concede.

My choice for the number of coins and distribution schedule was an educated guess. It was a difficult choice, because once the network is going it's locked in and we're stuck with it. I wanted to pick something that would make prices similar to existing currencies, but without knowing the future, that's very hard. I ended up

picking something in the middle. If Bitcoin remains a small niche, it'll be worth less per unit than existing currencies. If you imagine it being used for some fraction of world commerce, then there's only going to be 21 million coins for the whole world, so it would be worth much more per unit. Values are 64-bit integers with 8 decimal places, so 1 coin is represented internally as 100000000. There's plenty of granularity if typical prices become small. For example, if 0.001 is worth 1 Euro, then it might be easier to change where the decimal point is displayed, so if you had 1 Bitcoin it's now displayed as 1000, and 0.001 is displayed as 1.

Ripple is interesting in that it's the only other system that does something with trust besides concentrate it into a central server.

Satoshi
[Quoted text hidden]

Mike Hearn <mike@plan99.net> Mon, Apr 13, 2009 at 1:39 PM
To: Satoshi Nakamoto <satoshin@gmx.com>

Thanks Satoshi,

I tried the app yesterday. It seems to work pretty well running on Wine (I tried it on MacOS but it should run on Linux too, and will try that next week when I am back at work).

In the lower right hand corner it has a block count which increases rapidly and then stops. Is this the length of the global chain? It seems to advance far too fast for that. Or is this the number of genesis blocks that have been tried but did not result in a partial collision? I'm not sure if the way it stops and starts is expected, or some glitch caused by it running under emulation. My best guess - it is the length of the global chain, and the rapid advance at the start is as the software downloads and verifies the preceding blocks in the chain as being valid.

With regards to the buyer/seller experience, I understand that the global chain advances at about 6-7 blocks per hour under the current settings. If we assume that 0.1% is a good risk rate, then z=5 thus any transaction must wait a bit less than an hour before being solidified in the chain. As micropayments for things like web content or virtual goods are by definition something that requires low overhead, waiting an hour seems like quite a significant hurdle.

I understand that nodes attempt to find a POW to advance the global chain in an uncoordinated fashion. This sentence however:

"If a majority of CPU power is controlled by honest nodes, the honest chain will grow the fastest and outpace any competing chains."

is confusing for me, because it appears the only way the honest chain can grow faster than a chain worked on by 1 attacking cpu is if the keyspace to scan looking for a partial collision is sharded evenly amongst the participating honest nodes. That way the speed at which collisions are found would be proportional to the number of nodes. Yet I don't see any discussion of such work sharding, which obviously adds complexity. Likewise:

"To 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. If they're generated too fast, the difficulty increases."

How is the required difficulty of each block communicated through the network and agreed upon?

Thanks once again. I have yet more questions but this is enough for one email :) I will be happy to summarize these discussions into an FAQ-like document at some point. Apologies if the questions seem trivial.

-mike
[Quoted text hidden]

Mike Hearn <mike@plan99.net> Mon, Apr 13, 2009 at 10:51 PM
To: Satoshi Nakamoto <satoshin@gmx.com>

Something else that isn't clear to me - does the global chain only get extended when there is actual work to do? Currently it seems to grow all the time, although there are only a few people in the network. So presumably it gets extended with null blocks. Is this actually required? The timestamping doesn't have to be actually in parallel with real time does it ... it's merely establishing an ordering of events.
[Quoted text hidden]

Satoshi Nakamoto <satoshin@gmx.com> Mon, Apr 13, 2009 at 11:00 PM
To: Mike Hearn <mike@plan99.net>

Mike Hearn wrote:
My best guess - it
is the length of the global chain, and the rapid advance at the start
is as the software downloads and verifies the preceding blocks in the
chain as being valid.

Right. I'm trying to think of more clear wording for that, maybe "%d network blocks" or "%d block chain".

If we assume that 0.1% is a good risk rate, then z=5 thus
any transaction must wait a bit less than an hour before being
solidified in the chain. As micropayments for things like web content
or virtual goods are by definition something that requires low
overhead, waiting an hour seems like quite a significant hurdle.

For the actual risk, multiply the 0.1% by the probability that the buyer is an attacker with a huge network of computers.

For micropayments, you can safely accept the payment immediately. The size of the payment is too small for the effort to steal it. Micropayments are almost always for intellectual property, where there's no physical loss to the merchant. Anyone trying to steal a micropayment would probably not be a paying customer anyway, and if they want to steal intellectual property they can use the file sharing networks.

Currently, businesses accept a certain chargeoff rate. I believe the risk with 1 or even 0 confirming blocks will be much less than the rate of chargebacks on verified credit card transactions.

The usual scam against a merchant that doesn't wait for confirming blocks would be to send a payment to a merchant, then quickly try to propagate a double-spend to the network before the merchant's copy. What the merchant can do is broadcast his transaction and then monitor the network for any double-spend copies. The thief would not be able to broadcast during the monitoring period or else the merchant's node would receive a copy. The merchant would only have to monitor for a minute or two until most of the network nodes have his version and it's too late for the thief's version to catch up and reach many nodes. With just a minute or two delay, the chance of getting away without paying could be made much too low to scam. A thief usually needs a high probability of getting an item for free to make it worthwhile. Using a lot of CPU power to do the brute force attack discussed in the paper in addition to the above scam would not increase the thief's chances very much.

Anything that grants access to something, like something that takes a while to download, access to a website, web hosting, a subscription or service, can be cancelled a few minutes later if the transaction is rejected.

is confusing for me, because it appears the only way the honest chain can grow faster than a chain worked on by 1 attacking cpu is if the keyspace to scan looking for a partial collision is sharded evenly amongst the participating honest nodes. That way the speed at which collisions are found would be proportional to the number of nodes. Yet I don't see any discussion of such work sharding, which obviously adds complexity.

The keyspace is huge, 2^256. The thing being hashed includes the node's public key and a random nonce, so the chance of any two nodes duplicating work on the same space is negligible.

How is the required difficulty of each block communicated through the network and agreed upon?

It's not communicated. The formula is hardcoded in the program and every node does the same calculation to know what difficulty is required for the next block. If someone diverged from the formula, their block would not be accepted by the majority.

Thanks once again. I have yet more questions but this is enough for one email :) I will be happy to summarize these discussions into an FAQ-like document at some point. Apologies if the questions seem trivial.

No problem, thanks for testing it on Mac Wine.

Satoshi
[Quoted text hidden]

Satoshi Nakamoto <satoshin@gmx.com> Mon, Apr 13, 2009 at 11:11 PM
To: Mike Hearn <mike@plan99.net>

It keeps getting extended all the time. If it stopped, an attacker would have time to catch up. Don't worry, empty blocks aren't very big.

As you say, it's the order of events that matters.
[Quoted text hidden]

Mike Hearn <mike@plan99.net> Mon, Apr 13, 2009 at 11:18 PM
To: Satoshi Nakamoto <satoshin@gmx.com>

Oh yes, of course, that's fundamental. Silly me. Thanks for your answers. I'd recommend being over-explicit for early versions of the software, something like "Global chain is currently %d blocks long".

I guess the key problem right now is that once you generate coins, there's nobody to test it with, even for dummy transactions. Is there a plan for a mailing list or some kind of trivial marketplace to give people something to do with their newly minted bitcoins?

Satoshi Nakamoto <satoshin@gmx.com> Tue, Apr 14, 2009 at 7:41 PM

To: Mike Hearn <mike@plan99.net>

I started implementing a marketplace feature earlier that facilitates offering things for sale and taking orders, it's only half done though. A bit like e-bay but without auctions, just "buy now". Among other things, it would make it easy for anyone to offer currency exchange.

If you send to 1PhUXucRd8FzQved2KGK3g1eKfTHPGjgFu and e-mail me your bitcoin address, or IP if you can accept incoming connections, I'll send back the same amount +50.
[Quoted text hidden]

Mike Hearn <mike@plan99.net> Sat, Apr 18, 2009 at 3:08 PM

To: Satoshi Nakamoto <satoshin@gmx.com>

Hi Satoshi,

I sent you 32.51 coins, my bitcoin address is 1JuEjh9znXwqsy5RrnKqgzqY4Ldg7rnj5n

My IP is currently 84.73.233.199, however, it's a laptop so may or may not be online at the time you act on this mail. I suggest using the bitcoin address instead. It'd be convenient if the same comment functionality was available via indirect transfer. Can the comment be encrypted using the public key of the receiver and placed into a block?
[Quoted text hidden]

Satoshi Nakamoto <satoshin@gmx.com> Sat, Apr 18, 2009 at 6:16 PM

To: Mike Hearn <mike@plan99.net>

I sent back 32.51 and 50.00.

I badly wanted to find some way to include a comment with indirect transfers, but there just wasn't a way to do it. Bitcoin uses EC-DSA, which was essential for making the block chain compact enough to be practical with today's technology because its signatures are an order of magnitude smaller than RSA. But EC-DSA can't encrypt messages like RSA, it can only be used to verify signatures.
[Quoted text hidden]

Mike Hearn <mike@plan99.net> Sat, Apr 18, 2009 at 9:25 PM
To: Satoshi Nakamoto <satoshin@gmx.com>

Thanks. I sent you back 50, so now we're even.

For some reason your transfer to me shows up as "From: unknown" even though I added you to my address book.

I have a "Generated (not accepted)" line in my transaction list, it seems like an attempt to generate a coin went wrong somehow. Not sure what happened here - presumably my node successfully solved a block but then I went offline before it was sent to the network?

I suppose for sending metadata with a transaction some other mechanism will be needed, for instance, broadcast of encrypted messages associated with a transaction that persist for (say) a month, with some kind of budget on how much storage a node can use for messages. Alternatively, a payee could generate some reference number which is of some significance to themselves but otherwise opaque, and give it to the payer, thus it does not need to be encrypted and can be put into the block directly.
[Quoted text hidden]

Satoshi Nakamoto <satoshin@gmx.com> Sat, Apr 18, 2009 at 10:52 PM
To: Mike Hearn <mike@plan99.net>

Got the 50.

Transactions sent to a bitcoin address will always say "from: unknown". The transaction only tells who it's to. Sending by bitcoin address has a number of problems, but it's so nice having the fallback option to be able to send to anyone whether they're online or not. There are a number of ideas to try to improve things later. For now, if things work out like the real world where the vast majority of transactions are with merchants, they'll pretty much always make sure to set up to

receive by IP. The P2P file sharing networks seem fairly successful at getting a large percentage of their users to set up their firewalls to forward a port.

The "Generated (not accepted)" normally happens if two nodes find a block at close to the same time, one of them will not be accepted. It's normal and unavoidable. I plan in v0.1.6 to hide those, since they're just confusing and annoying and there's no reason for users to have to see them. While the network is still small like it is now, if you can't receive incoming connections you're at more of a disadvantage because you can't receive block announcements as directly.
[Quoted text hidden]

Mike Hearn <mike@plan99.net> Sat, Apr 18, 2009 at 11:23 PM
To: Satoshi Nakamoto <satoshin@gmx.com>

Yes, I believe most P2P clients use the UPnP protocol to get routers to open up the port automatically. That would probably improve the listen rate significantly. I just discovered DMZ wasn't enabled on my router, though I thought it was. That's now fixed.

Is there a way to be told of new versions? Does the app auto update itself? Again, some kind of mailing list would be excellent.

I was thinking through how a practical micropayment implementation for the web might work in the last few days. One key issue is ensuring micropayments are fully automatic, yet can't be easily abused to drain the users account. I think the right approach would be to allow any website that presents an EV SSL cert to automatically request a micropayment, by default the browser always accepts as long as the charge is "low" and displays a small notification of what has occurred. Sites can then show that content requires payment in any way that suits their site design. Abusive sites that don't meet some simple guidelines (eg, showing unambiguously that clicking a link will trigger payment, or taking payment from direct search engine links) would simply have their SSL cert blacklisted, much like anti-phishing filters work today.

The protocol could be very straightforward and implemented by a Firefox extension or an IE BHO. Some static file (eg, a protocol buffer) is hosted on the site. It specifies the charge, a transaction description, the target IP and a URL for the browser to load after the transaction was accepted by the target node, to which the user identifier is sent in a URL parameter. The site can then give back a cookie and the paywalled content. The entire process is automatic and simply results in, say, a little coin animation in the URL bar. Thus it's as convenient as regular web browsing. The users software would

have some limit on what payments are automatically accepted.

The main problem with this approach is that somebody has to decide what the user interface guidelines are, then enforce them via blacklisting, as well as decide what payment requirements are low enough to be automatic vs requiring a user prompt. This introduces a trusted authority back into the system. However, it's one that the user can choose in an open market.

By the way, if you're not already using protocol buffers for the node-to-node traffic, I recommend them. We use them here at Google for everything, they solve a lot of versioning problems simply and efficiently.

Satoshi Nakamoto <satoshin@gmx.com> Sun, Apr 19, 2009 at 2:14 AM

To: Mike Hearn <mike@plan99.net>

The list is:
bitcoin-list@lists.sourceforge.net
Subscribe/unsubscribe page:
http://lists.sourceforge.net/mailman/listinfo/**bitcoin-list**
Archives:
http://sourceforge.net/mailarchive/forum.php?forum_name=**bitcoin-list**

I'll always announce new versions there. Automatic update, or at least notification of new versions, is definitely on the list. There could potentially be necessary changes in the future where nobody will want to talk to you until you upgrade, and there needs to be code in the older version to convey that to the user. This is all the harder in the context of not trusting anyone.

Your approach to micropayments sounds right. At first, it might be a good idea to default to asking permission until the user gets comfortable and is ready to set it to automatic. The end goal though should get to something like you describe, where it's similar to using your cell phone without really having to think about the per minute charges.

I looked at Google protocol buffers when they were released last year, but I had already written everything by then. What I did was something similar to Boost Serialisation. For this application, where I was parsing messages from strangers who might have extreme incentive to hack the protocol, it was necessary to make it as basic as possible so I could crawl over every line of code to convince myself it was airtight. It became clear that any unnecessary degrees of freedom in the binary format multiplied the potential angles of attack. You guys are so right though to

standardize across the company on protocol buffers. I think you've got the optimal solution in the general case.

Lack of chargeback support
4 messages

Mike Hearn <mike@plan99.net> Sat, Apr 25, 2009 at 9:30 PM
To: Satoshi Nakamoto <satoshin@gmx.com>

Hi Satoshi,

I just read the following wiki page:

http://en.wikipedia.org/wiki/Chargeback

which claims that "U.S. debit card holders are guaranteed reversal rights by Federal Reserve Regulation E under the Electronic Funds Transfer Act. Similar rights extend globally pursuant to the rules established by the corresponding card association or bank network."

The "Electronic Funds Transfer Act" sounds awfully generic, do you think it'd apply to BitCoin? If so, would the inability to do chargebacks risk making it illegal?

Satoshi Nakamoto <satoshin@gmx.com> Mon, Apr 27, 2009 at 12:11 AM
To: Mike Hearn <mike@plan99.net>

I am not a lawyer and I can't possibly answer that. I suppose if the law applies to a bank or financial institution or other intermediary, then it would not apply since there is no bank involved, only two parties trading directly with each other, as they would in person with cash or barter with physical commodities.

Bitcoin is fundamentally designed to be able to do non-reversible transactions, and there certainly are applications that need that.

If someone wants the possibility of chargeback, they can use an escrow transaction, which isn't implemented yet but will be one of the next things. For instance, a transaction can be written to designate a third party to decide whether it is returned if the payer does not release it, with auto-release after a number of days. I'll implement a more basic form of escrow first, but the network infrastructure includes a predicate language that can express any number of options.

bitcoin-list

Re: [**bitcoin-list**] Does Bitcoin Crash in Windows?

2009-10-23 23:57:51 UTC - -

Liberty Standard wrote:
> Do you Windows users experience occasional Bitcoin crashes?
> Lately Bitcoin running in wine-1.0.1 has been crashing frequently. I was
> just wondering whether this is a Wine issue or a Bitcoin issue.

I haven't had any reports of crashes in v0.1.5. It's been rock solid
for me on Windows. I think it must be Wine related. If you get another
crash in Wine and it prints anything on the terminal, e-mail me and I
may be able to figure out what happened, maybe something I can work
around. Martti and I have been working on a new version to release soon
and it would be nice to get any Wine fixes in there.

> The following four lines print from the terminal when I start Bitcoin.
> fixme:toolhelp:CreateToolhelp32Snapshot Unimplemented: heap list snapshot
> fixme:toolhelp:Heap32ListFirst : stub
> fixme:toolhelp:CreateToolhelp32Snapshot Unimplemented: heap list snapshot
> fixme:toolhelp:Heap32ListFirst : stub

Those don't look like anything to worry about. Probably functions
unimplemented by Wine that are harmlessly stubbed out.

> I previously wasn't starting Bitcoin from the terminal, so I don't know what
> gets printed out when it crashes, but I'll reply with the results the next
> time it crashes.
>
> While Bitcoin first downloads previously completed blocks, the file
> debug.log grows grows to 17.4 MB and then stops growing. I imagine it will
> continue to grow as more bitcoins are completed.

You can delete debug.log occasionally if you don't want to take the disk
space. It's just status messages that help with debugging.

bitcoin.sourceforge.net looks fine now. Maybe sourceforge was doing
some maintenance.

Satoshi

BitcoinTalk

Welcome to the new Bitcoin forum!

2009-11-22 18:04:28 UTC - -

Welcome to the new Bitcoin forum!

The old forum can still be reached here:
http://bitcoin.sourceforge.net/boards/index.php

I'll repost some selected threads here and add updated answers to questions where I can.

FAQ
http://bitcoin.sourceforge.net/wiki/index.php?page=FAQ

Download
http://sourceforge.net/projects/bitcoin/files/

BitcoinTalk

Repost: Bitcoin Maturation

2009-11-22 18:31:44 UTC - -

---
bitcoinbitcoin:
Bitcoin Maturation
Posted:Thu 01 of Oct, 2009 (14:12 UTC)

From the user's perspective the bitcoin maturation process can be broken down into 8 stages.

1. The initial network transaction that occurs when you first click Generate Coins.
2. The time between that initial network transaction and when the bitcoin entry is ready to appear in the All Transactions list.
3. The change of the bitcoin entry from outside the All Transaction field to inside it.
4. The time between when the bitcoin appears in the All Transfers list and when the Description is ready to change to Generated (50.00 matures in x more blocks).
5. The change of the Description to Generated (50.00 matures in x more blocks).
6. The time between when the Description says Generated (50.00 matures in x more blocks) to when it is ready to change to Generated.
7 The change of the Description to Generated.
8. The time after the Description has changed to Generated.

Which stages require network connectivity, significant local CPU usage and or significant remote CPU usage? Do any of these stages have names?

---
sirius-m:
Re: Bitcoin Maturation
Posted:Thu 22 of Oct, 2009 (02:36 UTC)

As far as I know, there's no network transaction when you click Generate Coins - your computer just starts calculating the next proof-of-work. The CPU usage is 100% when you're generating coins.

In this example, the network connection is used when you broadcast the information about the proof-of-work block you've created (that which entitles you to the new coin). Generating coins successfully requires constant connectivity, so that you can start working on the next block when someone gets the current block before you.

BitcoinTalk

Repost: Request: Make this anonymous?

2009-11-22 18:32:00 UTC - -

---
anonguy54:
Request: Make this anonymous?
Posted:Thu 15 of Oct, 2009 (19:58 UTC)

Are there any plans to make this service anonymous?

e.g; Being able to route BitCoin through Tor.

BitcoinTalk

Repost: How anonymous are bitcoins?

2009-11-25 18:15:57 UTC - -

---
bitcoinbitcoin:
How anonymous are bitcoins?

Can nodes on the network tell from which and or to which bitcoin address coins are being sent? Do blocks contain a history of where bitcoins have been transfered to and from? Can nodes tell which bitcoin addresses belong to which IP addresses? Is there a command line option to enable the sock proxy the first time that bitcoin starts? What

happens if you send bitcoins to an IP address that has multiple clients connected through network address translation (NAT)?

BitcoinTalk

Re: Repost: How anonymous are bitcoins?

2009-11-25 18:17:23 UTC - -

> Can nodes on the network tell from which and or to which bitcoin
> address coins are being sent? Do blocks contain a history of where
> bitcoins have been transfered to and from?

Bitcoins are sent to and from bitcoin addresses, which are essentially random numbers with no identifying information.

When you send to an IP address, the transaction is still written to a bitcoin address. The IP address is only used to connect to the recipient's computer to request a fresh bitcoin address, give the transaction directly to the recipient and get a confirmation.

Blocks contain a history of the bitcoin addresses that a coin has been transferred to. If the identities of the people using the bitcoin addresses are not known and each address is used only once, then this information only reveals that some unknown person transferred some amount to someone else.

The possibility to be anonymous or pseudonymous relies on you not revealing any identifying information about yourself in connection with the bitcoin addresses you use. If you post your bitcoin address on the web, then you're associating that address and any transactions with it with the name you posted under. If you posted under a handle that you haven't associated with your real identity, then you're still pseudonymous.

For greater privacy, it's best to use bitcoin addresses only once. You can change addresses as often as you want using Options->Change Your Address. Transfers by IP address automatically use a new bitcoin address each time.

> Can nodes tell which bitcoin addresses belong to which IP addresses?

No.

> Is there a command line option to enable the sock proxy the first
> time that bitcoin starts?

In the next release (version 0.2), the command line to run it through a proxy from the first time is:
bitcoin -proxy=127.0.0.1:9050

The problem for TOR is that the IRC server which Bitcoin uses to initially discover other

nodes bans the TOR exit nodes, as all IRC servers do. If you've already connected once before then you're already seeded, but for the first time, you'd need to provide the address of a node as such:
bitcoin -proxy=127.0.0.1:9050 -addnode=<someipaddress>

If someone running a node with a static IP address that can accept incoming connections could post their IP to use for -addnode, that would be great.

> What happens if you send bitcoins to an IP address that has multiple
> clients connected through network address translation (NAT)?

Whichever one you've set your NAT to forward port 8333 to will receive it. If your router can change the port number when it forwards, you could allow more than one client to receive. For instance, if port 8334 forwards to a computer's port 8333, then senders could send to "x.x.x.x:8334"

If your NAT can't translate port numbers, there currently isn't a command line option to change the incoming port that bitcoin binds to, but I'll look into it.

BitcoinTalk

Repost: Linux/UNIX compile

2009-11-27 17:17:22 UTC - -

---

scott:
Linux/UNIX compile
Posted:Thu 08 of Oct, 2009 (05:49 UTC)

Can we get instructions or modifications to compile and install BitCoin on Linux? A command line version would be great.

BitcoinTalk

Re: Repost: Linux/UNIX compile

2009-11-27 17:27:09 UTC - -

The Linux version is on its way. Martti's Linux port was merged into the main code branch and New Liberty Standard has been testing it. It'll be in the next release, version 0.2.

Command line is on the to-do list after 0.2.

BitcoinTalk

[OLD THREAD] Bitcoin version 0.2 development status

2009-11-27 22:48:39 UTC - -

We've been working hard on improvements for the next version release. Martti (sirius-m) added some nice features to make it more user friendly and easier to run in the background:
- Minimize to system tray option
- Autostart on boot option so you can keep it running in the background automatically
- New options dialog layout
- Setup EXE for Windows, in addition to the archive download

I've been working on a number of refinements to the networking code and laying the groundwork for future functionality. Also coming in version 0.2:
- Multi-processor support for coin generation
- Proxy support

BitcoinTalk

Re: A few suggestions

2009-12-09 18:45:10 UTC - -

Helpful suggestions, thanks.

Quote from: madhatter on December 09, 2009, 05:34:46 AM

- When the bitcoin software establishes a connection with a peer (client TCP socket) have the client send the handshake string. Right now you have the server (server TCP socket) send the handshake. My reasons for this are anonymity of course. It is far too easy for ISPs to portscan clients and detect they are running this program.

That's a good idea. The side accepting the connection just needs to withhold from sending anything until it receives a valid handshake. Any portscan would only get a dead connection that doesn't volunteer to identify itself.

Quote

- Use some sort of encryption during the handshake (sort of goes with the statement/request above) to obfuscate what the software is during DPI (deep packet inspection). I am really thinking about people in non-free (as in freedom) countries such as China/Iran.

I have thought about eventually SSLing all the connections. I assume anything short of SSL would be pointless against DPI. Maybe a better more immediate solution is to connect through TOR, which will be possible with 0.2.

Quote

- Some sort of an API is needed so that this system can be integrated with websites to provide instant-on services. A simple https receipt mechanism would do wonders. Have the client post each incoming payment to an https url with all of the relevant information and provide status updates. Also an outbound payment mechanism would be nice. So one could automate payments (and batch payments) outbound. Status could be returned via the https receipt interface.

That's one of the main things on the agenda after 0.2.

Quote

- Static port/Random port. Have a setting to randomly assign the port that it runs on. (also be able to set it statically for very restrictive firewalls).

Yeah, the other stealth stuff would be kinda pointless if it's always the same port number.

Quote

- UPnP support. Have the client automatically create the port forward on upstream routers. Enabled by default. Can be turned off in the options menu.

I'm looking forward to trying UPnP. Do most P2P clients typically have UPnP enabled by default?

Quote

*- Ability to compile a headless (console only) install for *NIX systems. Also have the ability to just run as a network service. Perhaps with a telnet-able port for control (or even a unix socket would be ok).*

I'm still thinking about how best to structure the management interface. Maybe command line commands to communicate with the background daemon to query transactions received and initiate sending transfers. That would be more automation friendly. Or what about an http interface on some port other than 80 to manage it with a browser?

BitcoinTalk

Re: A few suggestions

2009-12-10 19:31:49 UTC - -

Quote from: madhatter2 on December 10, 2009, 02:00:17 PM

Front ends can also be ran on clients with very low cpu power such as mobile phones.

That's a good approach for mobile. Programmatic API used by PHP (any language) to present a web UI covers remote admin, mobile and any other client that can't be online

all the time with a static IP. It would be like webmail. It would be easier for new users to get started if they only need to create an account on a website, not install software.

Quote

The app could be pre-seeded before downloading. Pre-seeding would also cure the TOR+IRC problem. I know that people will want to run this system over I2P+TOR.

Yeah, we can phase out IRC when there are enough static nodes to preprogram a seed list. Once you get seeded, you don't need IRC.

Quote

Also you could pre-seed the blocks so they won't have to be downloaded upon initial run. (Downloading 28,000 blocks on a slower ADSL takes forever I couldn't imagine how long it would take when there are millions of blocks -- a lifetime).

There were some issues in 0.1.5 where the initial block download could get bogged down. 0.2 has code to make sure it goes smoothly. It ought to take less than an hour, I think. I need to hurry up and get 0.2 out the door.

The blocks increase linearly, it'll be decades before it's millions. In theory, the block download time should top out 8 months from now when Moore's Law will be growing faster than the block chain.

Quote

Can you give me CVS access or something? (If not, can I send you patches?) I'd like to help out.

It's SVN on sourceforge. PM or e-mail me your sourceforge account and I'll give you access.

Quote

I am mostly a Linux/BSD guy and I would like to lend my expertise in those areas.

That's great because that's where I have less expertise. For instance, I haven't researched the best way to do the "Start Bitcoin on system startup" feature on Linux. On Windows, the option adds/removes an icon in the Startup folder.

BitcoinTalk

Re: Questions about Bitcoin

2009-12-10 20:49:02 UTC - -

1-3:

For that level of anonymity you need to connect through TOR, which will be possible with version 0.2, which is only a few weeks away. I'll post TOR instructions at that time.

4:
Version 0.1.5: backup the whole %appdata%\Bitcoin directory.
Version 0.2: you can backup just wallet.dat.

5:
Nope. The whole design is all about preventing that from working.

6:
Those coins can never be recovered, and the total circulation is less. Since the effective circulation is reduced, all the remaining coins are worth slightly more. It's the opposite of when a government prints money and the value of existing money goes down.

7:
It's currently 29,296 blocks. The circulation is the number of blocks times 50, so the current circulation is 1,464,800 bc.

If you only have 24k blocks, it must not have finished the initial block download. Exit bitcoin and start it again. Version 0.2 is better/faster at the initial block download.

8:
Typically a few hundred right now. It's easy now but it'll get harder as the network grows.

9:
Good question, it's TCP. The website needs to be updated to say TCP port 8333.

The port forwarding is so other nodes can connect to you, so it helps you stay connected because you are able to be connected with more nodes. You also need it to receive payments by IP address.

10:
No, the other nodes won't accept that.

Being open source means anyone can independently review the code. If it was closed source, nobody could verify the security. I think it's essential for a program of this nature to be open source.

11:
Slower machines produce fewer coins. It's proportional to CPU speed.

12:
There are more coming.

13:
It uses a transactional database called Berkeley DB. It will not lose data in a system crash. Transactions are written to the database immediately when they're received.

14:
For now, you can just multiply the total blocks by 50. The Bitcoin network has been running for almost a year now. The design and coding started in 2007.

BitcoinTalk

Re: Questions about Bitcoin

2009-12-11 17:58:57 UTC - -

That's true, with the send-to-IP option, you are sending to whoever answers that IP. Sending to a bitcoin address doesn't have that problem.

The plan is to implement an IP + bitcoin address option that would have the benefits of both. It would still use a different address for each transaction, but the receiver would sign the one-time-use address with the given bitcoin address to prove it belongs to the intended receiver.

BitcoinTalk

Re: A few suggestions

2009-12-11 19:27:55 UTC - -

Right, the SVN has the almost-release-candidate 0.2 source, which can also be built and run on Linux. It hasn't been tested on FreeBSD.

Quote from: madhatter2 on December 11, 2009, 04:59:19 AM
If we can get to the point where we have a working backend process that will run on FreeBSD I can run always-on seeds.

That would be a big help. TOR users wouldn't have to worry about how to get seeded, and we wouldn't depend on IRC.

It can be run in a few simple modes without access to the UI if you don't mind a minimized window on the desktop. (0.1.5 doesn't have -min so it would be an open window)

To only run a seed:
bitcoin -min -gen=0

You could sort of monitor it by looking at debug.log. To stop it, kill the process, the database won't mind.

To generate:
bitcoin -min -gen

To get the generated bitcoins, you'd have to copy wallet.dat (with version 0.2) to a machine with a UI, swap in the wallet.dat, run bitcoin and transfer the coins to your main account. (With version 0.1.5 you'd have to copy the whole "%appdata%/Bitcoin" directory.) There is one caveat about copying wallet.dat: if you happened to kill the program at the exact moment that it generated a coin or received a payment, wallet.dat might not work by itself and you'd have to copy the whole directory.

Quote

I really think that having the download package contain a daily seed snapshot will improve the bootstrapping. I have seen instances on new test installs here where the application will sit with 0 connections / 1 block. Upon inspecting the debug.log I find that the IRC server (freenode, I believe) claims I am already connected and refuses to let me seed the application. (Just an example).

I see, that would happen with multiple nodes using the same NAT or VPN or some ISP that funnels everyone through a few proxy servers. I just committed a fix to SVN for this. If it gets "433" name already in use (it was error 433, right?), it'll retry with a non-address random username.

Quote

In any event, I would like to help. I have a lot of time and a project like this one is very exciting.

That's great, any help is really appreciated!

BitcoinTalk

Re: A few suggestions

2009-12-12 17:52:44 UTC - -

The average total coins generated across the network per day stays the same. Faster machines just get a larger share than slower machines. If everyone bought faster machines, they wouldn't get more coins than before.

We should have a gentleman's agreement to postpone the GPU arms race as long as we can for the good of the network. It's much easer to get new users up to speed if they don't have to worry about GPU drivers and compatibility. It's nice how anyone with just a CPU can compete fairly equally right now.

BitcoinTalk

Re: A few suggestions

2009-12-12 18:17:10 UTC - -

Quote from: madhatter2 on December 12, 2009, 06:34:21 AM

I almost have the svn 0.2 compiling on Mac OS X 10.4.11/Intel (I also have a PPC970 machine here as well so a PPC build would be possible as well). The windowing is native carbon too via wxwidgets! It is FAST!  *I had to create a new makefile (makefile.osx; based on makefile.unix of course.. given any thought to using autoconf?) and put some ifdef's into header.h. I have patches. I will keep toying around. I might try it on FreeBSD next.*

Mac support would be nice. wxWidgets really pays off for cross platform.

Please don't try PPC. PPC is big-endian and Bitcoin is little-endian, there would be endless endian bugs making it harder for me to debug the network if there's a potentially byte-swapping node out there. PPC is on its way out anyway.

Considered autoconf. Autoconf is a necessity for large projects with a quagmire makefile, but I think we're small enough that it's more optimal without it. I'd rather keep the makefile simple as long as possible.

Quote

I think that breaking bitcoin into two apps is ideal. A wxwidgets front end (since it is mostly all there) and a backend that binds to a control TCP socket. I have been reading over the source to see how hard it would be to break it apart and I think it should be fairly simple. Of course an API would have to be developed.

My head hurts just thinking about that. Funnelling all the UI backend through a TCP connection would make everything twice as hard. There's too much bandwidth between the UI and the internal data structures in order to keep the listview control updated, because of the way the listview control works.

I'd rather have command line control, that would get us remote admin and batch automation.

BitcoinTalk

Re: A few suggestions

2009-12-13 16:51:25 UTC - -

There would be a command line switch at runtime to tell it to run without UI. All it needs to do is not create the main window. A simplistic way would be to disable "pframeMain->Show" and "ptaskbaricon->Show" in ui.cpp. The network threads don't care that the UI isn't there. The only other UI is a message box in CheckDiskSpace if it runs out of disk space.

Then a separate command line utility to communicate with it to do things. Not sure what

it should be named.

"natural deflation"... I like that name for it. Yes, there will be natural deflation due to payment mistakes and lost data. Coin creation will eventually get slow enough that it is exceeded by natural deflation and we'll have net deflation.

BitcoinTalk

Re: A few suggestions

2009-12-14 17:15:56 UTC - -

Quote from: madhatter2 on December 14, 2009, 03:01:39 PM
Can anyone shed some light here?

*g++ -c -O0 -Wno-invalid-offsetof -Wformat -g -D__**WXMAC**__ -DNOPCH -DBUILD_MACOSX -I"/usr/include" -I"/usr/local/include/wx-2.8" -I"/usr/local/include" -I"/usr/local/boost_1_41_0" -I"/sw/include/db4" -I"/usr/local/ssl/include" -I"/usr/local/lib/wx/include/mac-ansi-release-2.8" -o headers.h.gch headers.h*
...
*ui.h:430: error: no matching function for call to 'wxTextCtrl::SetValue(const **std::basic_string**<char, std::char_traits<char>, std::allocator<char> >&)'*
*/usr/local/include/wx-2.8/wx/textctrl.h:303: note: candidates are: virtual void wxTextCtrlBase::SetValue(**const wxString&**)*

It looks like the implicit conversion from std::string to wxString isn't working. \u00a0That's used everywhere, the conversion needs to work.

wxString is complicated by supporting win32's 16-bit wchar and 8-bit ansi dual-compile. \u00a0You can get that problem on Windows if the "unicode" (meaning wchar) build is used, so that wxString is wchar and std::string is char.

It's probably some wxWidgets compile defines or build configuration. \u00a0What "configure" options did you use?

I'm not sure __WXMAC__ is the right define. \u00a0It may be the Mac Classic support that's complicating wxString, and we only want OSX. \u00a0Try __WXOSX__ (or see below)

http://docs.wxwidgets.org/stable/wx_cppconst.html
"There are two wxWidgets ports to Mac OS. One of them, wxMac, exists in two versions: Classic and Carbon. The Classic version is the only one to work on Mac OS version 8. The Carbon version may be built either as CFM or Mach-O (binary format, like ELF) and the former may run under OS 9 while the latter only runs under OS X. Finally, there is a new Cocoa port which can only be used under OS X. To summarize:

\u00a0\u00a0 \u00a0* If you want to test for all Mac platforms, classic and OS X, you should test both __WXMAC__ and __WXCOCOA__.
\u00a0\u00a0 \u00a0* If you want to test for any GUI Mac port under OS X, use __WXOSX__.
\u00a0\u00a0 \u00a0* If you want to test for any port under Mac OS X, including, for example, wxGTK and also wxBase, use __DARWIN__"

BitcoinTalk

Re: A few suggestions

2009-12-15 20:37:32 UTC - -

Quote from: madhatter2 on December 15, 2009, 05:21:09 AM
It is also throwing the same std::string issue on the latest version of Ubuntu Linux.

Then it must be something you're doing differently with building or configuring wxWidgets.

What options did you use on the wxWidgets "configure" script?\u00a0 The options I used are in build-unix.txt.

Quote
One question: how do I enable the debug.log? I have tried stopping bitcoin and touching ~/.bitcoin/debug.log and starting bitcoin again. It never seems to write to the file. Am I missing something?

Never heard of that happening.\u00a0 Is there anything in debug.log?\u00a0 If you touched the file, that sounds like something is there.\u00a0 Does the program have write access to the file?

BitcoinTalk

Bitcoin 0.2 released!

2009-12-16 22:45:36 UTC - -

Bitcoin version 0.2 is here!

Download links:
http://sourceforge.net/projects/bitcoin/files/Bitcoin/bitcoin-0.2.0-win32-setup.exe/download
http://sourceforge.net/projects/bitcoin/files/Bitcoin/bitcoin-0.2.0-win32.zip/download
http://sourceforge.net/projects/bitcoin/files/Bitcoin/bitcoin-0.2.0-linux.tar.gz/download

New Features

Martti Malmi
 - Minimize to system tray option
 - Autostart on boot option so you can keep it running in the background automatically
 - New options dialog layout for future expansion
 - Setup program for Windows
 - Linux version (tested on Ubuntu)
Satoshi Nakamoto
 - Multi-processor support for coin generation
 - Proxy support for use with TOR
 - Fixed some slowdowns in the initial block download

Major thanks to Martti Malmi (sirius-m) for all his coding work and for hosting the new site and this forum, and New Liberty Standard for his help with testing the Linux version.

bitcoin-list

[**bitcoin-list**] Bitcoin 0.2 released

2009-12-17 06:52:09 UTC - -

Bitcoin 0.2 is here!

Download (Windows, and now Linux version available)
http://sourceforge.net/projects/bitcoin/files/

New Features

Martti Malmi
- Minimize to system tray option
- Autostart on boot option so you can keep it running in the
background automatically
- New options dialog layout for future expansion
- Setup program for Windows
- Linux version (tested on Ubuntu)
Satoshi Nakamoto
- Multi-processor support for coin generation
- Proxy support for use with TOR
- Fixed some slowdowns in the initial block download

We also have a new forum at http://www.bitcoin.org/smf/

Many thanks to Martti (sirius-m) for all his development work, and to
New Liberty Standard for his help with testing the Linux version.

Satoshi Nakamoto

BitcoinTalk

Re: A few suggestions

2009-12-17 18:38:06 UTC - -

That's good, is it running fine on FreeBSD?

I committed the changes to headers.h.\u00a0 For consistency, I used __BSD__.\u00a0 The complete list of defines is at http://docs.wxwidgets.org/stable/wx_cppconst.html
#ifdef __BSD__
#include <netinet/in.h>
#endif

malloc.h is only needed on windows, I'll move that into the __WXMSW__ section before it causes any more trouble.

BitcoinTalk

Re: A few suggestions

2009-12-18 17:37:48 UTC - -

What you can currently do is set "Minimize to the tray" in options, then run it as "bitcoin -min" so it starts minimized. \u00a0The only visible part will be a small (20x20) icon on the tray, which can be doubleclicked if you want to access the UI. \u00a0Note: there's a bug with tray icons sometimes disappearing on 64-bit Karmic Koala, not sure if it's from 64-bit or Karmic, it was fine on 32-bit Jaunty.

We didn't have time to implement the "Start Bitcoin on system startup" feature on Linux in time for 0.2 so it's greyed out. \u00a0I figured Linux people wouldn't mind doing that manually anyway. \u00a0I guess they need to know about the -min switch to do it right.

You can locate the data directory where you want with the "-datadir=<directory>" switch. \u00a0I know someone is already doing that to put it on a TrueCrypt USB drive.

BitcoinTalk

Re: Is my second Transaction working correctly? +Transfer Question

2010-01-05 20:00:46 UTC - -

The transfer is immediate if you send by IP address. If you send by bitcoin address and the recipient isn't online at the time, it might take 30 minutes or more to see it.

Also, the recipient needs to be synced up with the block chain before it'll see the received transaction. That means the status bar at the bottom needs to say at least 33000 blocks, like "x connections 33200 blocks x transactions".

> ***Quote from: sirius-m on January 05, 2010, 01:20:06 AM***
> ***Quote***
> *However, once that transaction was complete, a new transaction hasn't started. Or maybe it has. There's only one transaction in the list but I'm up to 131 Blocks under "Status". Is this the way it's supposed to happen? Does it keep processing on the same transaction and generating coins every 120 blocks or so? Or is it supposed to start a new transaction?*

The number of blocks of a transaction is the amount of new blocks that have been generated by the whole network after the transaction. Each new block in the chain means new coins to its creator. One "generated" -transaction in your transaction list means that you have generated one block. You're not the first one to find the concept of a "block" a bit confusing on the first sight.

Would it be clearer if the status said "x confirmations", like:
2/unconfirmed
3/unconfirmed
4/unconfirmed
5/unconfirmed
6 confirmations
7 confirmations
8 confirmations

Each block essentially means another node has confirmed that it agrees with all transactions up to that point.

BitcoinTalk

Re: 64bit support

2010-01-14 20:17:20 UTC - -

I haven't tried compiling 64-bit yet. 64-bit wouldn't make it any faster, since it uses 64-bit numbers in only a few places and SHA-256 is a 32-bit algorithm, but it may be convenient for those running a 64-bit OS. If I get a chance I'll try -m64 and see what the problem is.

You can run the 32-bit version on 64-bit Linux by installing ia32-libs. (sudo apt-get install ia32-libs) If we made a Debian package, it could automatically pull that in as a dependency.

BitcoinTalk

Re: Number of connections?

2010-01-20 20:07:15 UTC - -

Coins generate at the same speed with any number of connections >= 1.

More connections just add redundancy. If you only had one connection, what if that node is slow or busy, or only connected to you? Having several connections increases the certainty that you're well connected to the network. That hasn't been a problem in practice, the network is very thoroughly connected. If you have 2 or 3 connections, you're fine.

BitcoinTalk

Re: TOR and I2P

2010-01-20 22:05:28 UTC - -

I've been thinking about that for a while. I want to add the backend support for .onion addresses and connecting to them, then go from there.

There aren't many .onion addresses in use for anything because the user has to go through a number of steps to create one. Configure TOR to generate a .onion address, restart TOR, configure it with the generated address. Perhaps this is intentional to keep TOR so it can't be integrated into file sharing programs in any sufficiently automated way.

BitcoinTalk

Re: Bitcoin crash when sending coins

2010-01-27 21:52:27 UTC - -

That is what happens if you copy wallet files around. If you copy your wallet file to a second computer, then they both think the money in the wallet is theirs. If one spends any of it, the other doesn't know those coins are already spent and would try to spend them again, and that's the error you would hit.

Now that it's clear this is a key error message, it ought to be something more like "the money appears to be already spent... this could happen if you used a copy of your wallet file on another computer."

You can move or backup your wallet file, but it needs to have only one "lineage" and only used in one place at a time. Any time you transfer money out of it, then you must no longer use any previous copies.

This brings up a good point. In the case of restoring a backup that may be from before you spent some coins, we need to add functionality to resync it to discover which coins have already been spent. This would not be hard to do, it just hasn't been implemented yet. I'll add it to the list. This would make it mostly repair the situation instead of giving that error message.

BitcoinTalk

Re: A newb's test - anyone want to buy a picture for $1?

2010-01-28 01:01:48 UTC - -

Yes, it's a technical limitation. Sending by bitcoin address enters the transaction into the network and the recipient discovers it from the network. You don't connect directly with them and they don't have to be online at the time.

I very much wanted to find some way to include a short message, but the problem is, the whole world would be able to see the message. As much as you may keep reminding people that the message is completely non-private, it would be an accident waiting to happen.

Unfortunately, ECDSA can only sign signatures, it can't encrypt messages, and we need the small size of ECDSA. RSA can encrypt messages, but it's many times bigger than ECDSA.

BitcoinTalk

Re: 64bit support

2010-01-29 00:42:49 UTC - -

I committed a fix for 64-bit compile and some fixes to support wxWidgets 2.9.0.

There was one compile error in serialize.h with min(sizeof()) that I fixed for 64-bit. The rest of the 64-bit compile errors I was getting were in wxWidgets 2.8.9, so I started working on supporting wxWidgets 2.9.0.

wxWidgets 2.9.0 is UTF-8. We've been using the ANSI version of wxWidgets 2.8.9 in anticipation of wxWidgets UTF-8 support.

I compiled and ran on 64-bit Ubuntu 9.10 Karmic.

I think the only bug left is where the status number is mashed up. I'm not sure why, I have to suspect it's a UTF-8 thing, but no idea how that could happen. Haven't looked into it.

build-unix.txt is updated and two makefiles on SVN:
makefile.unix.wx2.8
makefile.unix.wx2.9

Unfortunately there's still no debian package for either version of wxWidgets we use. They only have the wchar ("unicode") version of wxWidgets 2.8, which is a disaster because wchar wxString doesn't convert to std::string. We use either ANSI wxWidgets 2.8, or wxWidgets 2.9. So you still have to get it and build it yourself.

BitcoinTalk

Re: Bitcoin crash when sending coins

2010-02-03 23:29:57 UTC - -

I uploaded this fix to the SVN. It watches for spent coins and updates your wallet on load and also continuously as blocks come in. I also put a better error message, but it should never hit it because it always finds spent coins ahead of time, unless you spent the same money at the same time on two computers at once.

If you want to try it, PM or e-mail me your e-mail address where I can send it as an attachment and also what OS (win, linux 32-bit, linux 64-bit).

BitcoinTalk

Re: Win32 CPU Cycles vs 'Live Protection' Engines ?

2010-02-03 23:36:54 UTC - -

Thanks for that. Which version of Windows?

BitcoinTalk

Re: Questions about Addresses

2010-02-04 00:07:07 UTC - -

Port forwarding forwards a port to one computer. It tells the router which computer handles connections to that port. So that's the computer receiving.

If you didn't set up port forwarding, then incoming connections won't go to any computer, and attempts to send to that IP would just say it couldn't connect to the recipient and nothing is sent. When sending by IP, you still send to a bitcoin address, but your computer connects to that IP, gets a new bitcoin address from it, gives the

transaction directly to the them and confirms that it was received and accepted.

Someone should post their static IP so people can try out sending by IP and also give that user free money.

There's a 32-bit checksum in bitcoin addresses so you can't accidentally type an invalid address.

If 4) you send to a recipient who has abandoned or lost their wallet.dat, then the money is lost. A subtle point can be made that since there is then less total money in circulation, everyone's remaining money is worth slightly more, aka "natural deflation".

BitcoinTalk

Re: TOR and I2P

2010-02-04 00:30:50 UTC - -

When using proxy port 9050, it will only make one attempt to connect to IRC, then give up, since it knows it will probably always fail because IRC servers ban all the TOR exit nodes. If you're using another port, it would assume it might be a regular old normal proxy and would keep retrying IRC at longer and longer intervals. You should not use Polipo or Privoxy as those are http filters and caches that would corrupt Bitcoin's messages if they make any changes. Bitcoin might be trying to overcome it by reconnecting. You should use port 9050.

As riX says, the "is giving Tor only an IP address. Apps that do DNS..." warnings are nothing to worry about. Bitcoin doesn't use DNS at all in proxy mode.

Since Bitcoin can't get through to IRC through Tor, it doesn't know which nodes are currently online, so it has to try all the recently seen nodes. It tries to conserve connection attempts as much as possible, but also people want it to connect quickly when they start it up and reconnect quickly if disconnected. It uses an algorithm where it tries an IP less and less frequently the longer ago it was successful connected. For example, for a node it saw 24 hours ago, it would wait 5 hours between connection attempts. Once it has at least 2 connections, it won't try anything over a week old, and 5 connections it won't try anything over 24 hours old.

BitcoinTalk

Proof-of-work difficulty increasing

2010-02-05 19:19:12 UTC - -

We had our first automatic adjustment of the proof-of-work difficulty on 30 Dec 2009.

The minimum difficulty is 32 zero bits, so even if only one person was running a node, the difficulty doesn't get any easier than that. For most of last year, we were hovering below the minimum. On 30 Dec we broke above it and the algorithm adjusted to more difficulty. It's been getting more difficult at each adjustment since then.

The adjustment on 04 Feb took it up from 1.34 times last year's difficulty to 1.82 times more difficult than last year. That means you generate only 55% as many coins for the same amount of work.

The difficulty adjusts proportionally to the total effort across the network. If the number of nodes doubles, the difficulty will also double, returning the total generated to the target rate.

For those technically inclined, the proof-of-work difficulty can be seen by searching on "target:" in debug.log. It's a 256-bit unsigned hex number, which the SHA-256 value has to be less than to successfully generate a block. It gets adjusted every 2016 blocks, typically two weeks. That's when it prints "GetNextWorkRequired RETARGET" in debug.log.

```
minimum    00000000ffff0000000000000000000000000000000000000000000000000000
30/12/2009 00000000d86a000000000000000000000000000000000000000000000000000000
11/01/2010 00000000c428000000000000000000000000000000000000000000000000000000
25/01/2010 00000000be710000000000000000000000000000000000000000000000000000000
04/02/2010 000000008cc300000000000000000000000000000000000000000000000000000000
14/02/2010 0000000065465700000000000000000000000000000000000000000000000000000
24/02/2010 0000000043b3e50000000000000000000000000000000000000000000000000000000
08/03/2010 00000000387f6f000000000000000000000000000000000000000000000000000000
21/03/2010 0000000038137500000000000000000000000000000000000000000000000000000
01/04/2010 000000002a11150000000000000000000000000000000000000000000000000000000
12/04/2010 0000000020bca70000000000000000000000000000000000000000000000000000000
21/04/2010 0000000016546f0000000000000000000000000000000000000000000000000000000
04/05/2010 0000000013ec530000000000000000000000000000000000000000000000000000000
19/05/2010 00000000159c2400000000000000000000000000000000000000000000000000000000
29/05/2010 000000000f675c000000000000000000000000000000000000000000000000000000
11/06/2010 0000000000eba640000000000000000000000000000000000000000000000000000000
24/06/2010 000000000d31420000000000000000000000000000000000000000000000000000000
06/07/2010 000000000ae493000000000000000000000000000000000000000000000000000000
13/07/2010 0000000005a3f4000000000000000000000000000000000000000000000000000000
16/07/2010 000000000168fd0000000000000000000000000000000000000000000000000000000
27/07/2010 00000000010c5a000000000000000000000000000000000000000000000000000000
05/08/2010 0000000000ba18000000000000000000000000000000000000000000000000000000
15/08/2010 0000000000800e000000000000000000000000000000000000000000000000000000
26/08/2010 000000000069200000000000000000000000000000000000000000000000000000000
```

date, difficulty factor,	% change
2009	1.00	
30/12/2009	1.18	+18%
11/01/2010	1.31	+11%
25/01/2010	1.34	+2%
04/02/2010	1.82	+36%
14/02/2010	2.53	+39%
24/02/2010	3.78	+49%
08/03/2010	4.53	+20%

21/03/2010	4.57	+9%
01/04/2010	6.09	+33%
12/04/2010	7.82	+28%
21/04/2010	11.46	+47%
04/05/2010	12.85	+12%
19/05/2010	11.85	-8%
29/05/2010	16.62	+40%
11/06/2010	17.38	+5%
24/06/2010	19.41	+12%
06/07/2010	23.50	+21%
13/07/2010	45.38	+93%
16/07/2010	181.54	+300%
27/07/2010	244.21	+35%
05/08/2010	352.17	+44%
15/08/2010	511.77	+45%
26/08/2010	623.39	+22%

BitcoinTalk

Re: Questions about Addresses

2010-02-05 19:44:46 UTC - -

Quote from: Sabunir on February 05, 2010, 05:31:30 PM

Perhaps there should be a feature against this? For instance, if a transaction isn't accepted by the recipient for a long period of time (a month?), the transaction will be canceled and the coins returned to the one who sent them?

That's not possible. You've handed control of the money over to the recipient's keypair. Only that key can control it.

It's similar to if you encrypt a file with AES and a strong password, and you lose the password. The data is lost.

BitcoinTalk

Re: Repost: Request: Make this anonymous?

2010-02-06 21:06:32 UTC - -

When you send to a bitcoin address, you don't connect to the recipient. You send the transaction to the network the same way you relay transactions. There's no distinction between a transaction you originated and one you received from another node that you're relaying in a broadcast. With a very small network though, someone might still figure it out by process of elimination. It'll be better when the network is larger.

If you send by IP, the recipient sees you because you connect to their IP. You could use TOR to mask that.

You could use TOR if you don't want anyone to know you're even using Bitcoin.

Bitcoin is still very new and has not been independently analysed. If you're serious about privacy, TOR is an advisable precaution.

BitcoinTalk

Re: How divisible are bitcoins and other market/economic questions

2010-02-06 23:25:53 UTC - -

Eventually at most only 21 million coins for 6.8 billion people in the world if it really gets huge.

But don't worry, there are another 6 decimal places that aren't shown, for a total of 8 decimal places internally. It shows 1.00 but internally it's 1.00000000. If there's massive deflation in the future, the software could show more decimal places.

If it gets tiresome working with small numbers, we could change where the display shows the decimal point. Same amount of money, just different convention for where the ","'s and "."'s go. e.g. moving the decimal place 3 places would mean if you had 1.00000 before, now it shows it as 1,000.00.

BitcoinTalk

Re: Make your "we accept Bitcoin" logo

2010-02-08 01:22:29 UTC - -

No, sorry. I've been meaning to redo it. The largest icon that still looks good is the 20x20 one which is used for the tray icon in GNOME. Any larger than that looks bad. The 16x16 and 20x20 ones have quite a bit of hand tweaking to get the pixels to work out right. If you just scale down a larger image, the pixels end up blurred and awkward in places where the lines in "BC" don't land square on a pixel.

The best 16x16 with full alpha channel is in src/rc/bitcoin.ico. I don't like the 32x32 version.

I'm attaching bitcoin20x20.png, the 20x20 version with full transparency.

BitcoinTalk

Bitcoin client and website translation

2010-02-08 01:27:02 UTC - -

Thank you for the offer to help translate. That is probably the best way you could help.

I will need to prepare the code for translation first. wxWidgets has locale support, and most strings are in generated code that is already wrapped, so it shouldn't be too hard. We also must finish upgrading to wxWidgets-2.9.0 to get UTF-8 support. I've done test builds with 2.9.0 and there is one bug left to fix.

What operating system are you using? Windows, Linux 32-bit or 64 bit?

Split from another thread.
sirius-m

BitcoinTalk

Bitcoin client and website translation

2010-02-08 16:10:37 UTC - -

It's much easier to have a single binary and multiple .mo files. It's too much maintenance work to have lots of build variations. Once the software support is implemented, anyone could contribute translations.

wxWidgets uses the gettext standard. You use the gettext tools or something like poedit to create a .po file by scanning the sourcefiles for strings and editing the translations into the .po file, then compile it into a .mo file. The program loads the .mo file at runtime and reskins all the strings. Additional languages can be added to an existing program by adding .mo files without recompiling the program.

On Windows, the .mo files would go in a lang subdirectory in the directory where the EXE is located.

Right now I'm working on JSON-RPC and command line support, but when I'm finished with that I hope to do this next.

BitcoinTalk

Re: Simple to implement feature requests

2010-02-08 16:37:24 UTC - -

There are command line options:

bitcoin -addnode=1.2.3.4 to tell bitcoin about a node to connect to

bitcoin -connect=1.2.3.4 connect only to the specified node(s)

You can use more than one of these, for instance
bitcoin -connect=(first to try) -connect=(next to try) ...

You can specify non-routable IPs with -connect like 192.168.x.x, so if you had a server farm and you wanted one server to connect to the world and the rest to connect to the one server, you could do that.

In particular, -addnode is needed if you're always going to connect through TOR, since the IRC server blocks all the TOR exit nodes. To connect through TOR, you could use:

bitcoin -proxy=127.0.0.1:9050 -addnode=212.159.72.216

BitcoinTalk

Re: DEB Package?

2010-02-12 02:33:02 UTC - -

Are you just trying to run the program or do you really need to compile it? There's a 32-bit linux binary that can be run on 64-bit ubuntu if you "sudo apt-get ia32-libs".
http://sourceforge.net/projects/bitcoin/files/Bitcoin/bitcoin-0.2.0-linux.tar.gz/download

I recently updated the SVN for building on 64-bit Karmic with wxWidgets 2.9.0. This was after the 0.2.0 release. The 0.2.0 release did not build on 64-bit yet.

Unfortunately there currently isn't a -dev deb package of either of the versions of wxWidgets that we can use. On Karmic they only have the UTF-16 version. We need either the ANSI (libwxgtk2.8-ansi-dev) version or the UTF-8 (wxWidgets 2.9.0) version. We're moving towards 2.9.0.

I know you said you didn't want VM, but as a last resort, last I checked the Windows version runs fine in Wine.

BitcoinTalk

Re: Repost: Request: Make this anonymous?

2010-02-12 17:28:32 UTC - -

True, sending by IP through Tor trades one problem for another. The Tor exit node can see the text of your message and potentially MITM you.

Best to only send to bitcoin addresses then. Payments by bitcoin address are broadcast

over the network as part of the normal network traffic. All communications with the network are broadcasts of public information.

BitcoinTalk

Re: DEB Package?

2010-02-13 01:38:37 UTC - -

I couldn't get wxWidgets 2.8.9 to compile on Karmic 64-bit either.

I have been compiling the latest SVN on Karmic 64-bit with wxWidgets 2.9.0, which compiles fine on 64-bit. Read build-unix.txt and use the given ../configure parameters on wxWidgets so you can use the makefile.unix.wx2.9 as supplied. (--enable-debug --disable-shared --enable-monolithic)

~~There's one cosmetic bug with 2.9.0 I still need to fix where the status number display is bunched up for some reason.~~ -- fixed

The download link on the homepage is to the sourceforge tar.gz archive which contains the 32-bit binary and the 0.2.0 sources, which were not yet buildable on 64-bit at the time.

The SVN was first buildable on 64-bit with wx2.9.0 on 28 January 2010.

Hopefully they'll have a wxWidgets 2.9.0 debian package someday.

BitcoinTalk

Re: What's with this odd generation?

2010-02-14 06:28:03 UTC - -

Quote from: theymos on February 12, 2010, 08:31:52 AM

Does the sending client send more BitCoins to account for the fee (so the recipient gets what he's expecting)?

Yes.

Quote from: SmokeTooMuch on February 12, 2010, 01:11:09 PM

why do we even need fees ? i thougt the no-fees-feature was one of the advantages of bitcoin ?!

Almost all transactions are free. A transaction is over the maximum size limit if it has to add up more than 500 of the largest payments you've received to make up the amount. A transaction over the size limit can still be sent if a small fee is added.

The average transaction, and anything up to 500 times bigger than average, is free.

It's only when you're sending a really huge transaction that the transaction fee ever comes into play, and even then it only works out to something like 0.002% of the amount. It's not money sucked out of the system, it just goes to other nodes. If you're sad about paying the fee, you could always turn the tables and run a node yourself and maybe someday rake in a 0.44 fee yourself.

BitcoinTalk

Re: What's with this odd generation?

2010-02-14 15:52:23 UTC - -

Right. Otherwise we couldn't have a finite limit of 21 million coins, because there would always need to be some minimum reward for generating. In a few decades when the reward gets too small, the transaction fee will become the main compensation for nodes. I'm sure that in 20 years there will either be very large transaction volume or no volume.

BitcoinTalk

Re: Proof-of-work difficulty increasing

2010-02-15 06:28:38 UTC - -

14/02/2010
00000000654657000000000000000000000000000000000000000000000000000000

2009 1.00
30/12/2009 1.18 +18%
11/01/2010 1.31 +11%
25/01/2010 1.34 +2%
04/02/2010 1.82 +36%
14/02/2010 2.53 +39%

Another big jump in difficulty yesterday from 1.82 times to 2.53 times, a 39% increase since 10 days ago. It was 10 days apart not 14 because more nodes joined and generated the 2016 blocks in less time.

BitcoinTalk

Re: Setting up multiple bitcoin machines behind NAT

2010-02-16 01:34:56 UTC - -

Right now there isn't a port number setting to do that. It's a feature yet to be implemented. You can only set up your NAT to port-forward to one of the computers. (I said something earlier about NAT port translation, but that wouldn't work, other nodes wouldn't know to connect to that port)

If you want, as a small optimization, you could run the rest of your computers as:
bitcoin -connect=<the IP of the first computer>

so they get all their network communication from the first computer and don't all connect over the net individually for the same information. This saves bandwidth, although it doesn't use much bandwidth to begin with, so it wouldn't really matter unless you had tons of computers.

For redundancy in case the first computer goes down, you could have two that connect out and the rest connect to both of them. The first two are run normally, the rest are run like:
bitcoin -connect=<IP1> -connect=<IP2>

BitcoinTalk

Re: Proof-of-work difficulty increasing

2010-02-17 17:58:03 UTC - -

Quote from: Sabunir on February 16, 2010, 08:51:51 AM
. Perhaps it has to do with my connection's very high latency (2000ms or more on average)

2 seconds of latency in both directions should reduce your generation success by less than 1%.

Quote from: Sabunir on February 16, 2010, 08:51:51 AM
and/or my high packet loss (sometimes up to 10% loss)?

Probably OK, but I'm not sure. The protocol is designed to resync to the next message, and messages get re-requested from all the other nodes you're connected to until received. If you miss a block, it'll also keep requesting it every time another blocks comes in and it sees there's a gap. Before the original release I did a test dropping 1 out of 4 random messages under heavy load until I could run it overnight without any nodes getting stuck.

BitcoinTalk

Re: Bitcoin client and website translation

2010-02-17 19:19:43 UTC - -

I updated the SVN with changes to support translation. Translatable strings are all enclosed in _(""), and we're using UTF-8 on all platforms.

When the program runs, it looks in the directory of the EXE for the file:
locale\<langcode>\LC_MESSAGESitcoin.mo

<langcode> is the two letter code of the language your OS is set to, like "de" or "nl".

On Linux, it also looks for:
/usr/share/locale/<langcode>/LC_MESSAGES/bitcoin.mo
/usr/local/share/locale/<langcode>/LC_MESSAGES/bitcoin.mo
(are there other standard places it should look on linux?)

Here's a quick walkthrough using poedit to make a .po and .mo file:

- Download the bitcoin sourcecode from SVN
- In the trunk directory, mkdir locale\<lang>\LC_MESSAGES
- In poedit, File->New catalog->Paths tab
- Click the "New item" dotted rectangle button
- Put "../../.." and MAKE SURE TO PRESS ENTER to add the path
- Click OK
- Save the file as "bitcoin.po" in the LC_MESSAGES directory you made
- It should then scan the sourcecode and find about 170 strings
- If it didn't find anything, check Catalog->Settings->Path tab, make sure the "../../.." was added

When you're done translating, commit both bitcoin.po (the editable catalog file) and bitcoin.mo (compiled data used by the program).

BitcoinTalk

Re: Number of connections

2010-02-21 03:43:48 UTC - -

Nodes stop trying to initiate connections once they have 15. If you can accept incoming connections, then you can get well above that from nodes connecting to you, otherwise you max out at 15.

I don't know if there's any reason to have 15 connections. Maybe it should be 10.

Since nodes that can only connect out are probably at or near 15 most of the time now, you should level off to an equilibrium. 45 suggests a ratio of 3 out-only nodes to every 1

in-accepting node.

The number of connections won't be a good gauge of the size of the network any more. Someone should periodically IRC to the bitcoin channel on chat.freenode.net and count the number of users. That gives you the total count of network nodes (except TOR nodes).

Block generation is again running ahead of pace. We're in for another big step up in difficulty at the next adjustment in about 5 days.

BitcoinTalk

Post your static IP

2010-02-21 04:19:53 UTC - -

It would be nice to have a list of static IPs for new users to send test donations to so they can see how the software works. If you can accept incoming connections and you have a static IP address, post it here!

Anything sent to these IPs should be considered a donation.

If you do request a round-trip, be sure to include your return bitcoin address or IP in the comment, but please assume it'll be one-way. They won't necessarily be watching for incoming transactions to send back.

BitcoinTalk

Re: Current Bitcoin economic model is unsustainable

2010-02-21 05:44:24 UTC - -

Excellent analysis, xc.

A rational market price for something that is expected to increase in value will already reflect the present value of the expected future increases. In your head, you do a probability estimate balancing the odds that it keeps increasing.

In the absence of a market to establish the price, NewLibertyStandard's estimate based on production cost is a good guess and a helpful service (thanks). The price of any commodity tends to gravitate toward the production cost. If the price is below cost, then production slows down. If the price is above cost, profit can be made by generating and selling more. At the same time, the increased production would increase the difficulty, pushing the cost of generating towards the price.

In later years, when new coin generation is a small percentage of the existing supply,

market price will dictate the cost of production more than the other way around.

At the moment, generation effort is rapidly increasing, suggesting people are estimating the present value to be higher than the current cost of production.

BitcoinTalk

UI improvements

2010-02-21 21:48:01 UTC - -

Uploaded some UI changes to SVN as version 0.2.5.

Instead of View->Show Generated, we now have tabs:
- All Transactions
- Sent/Received
- Sent
- Received

Makes it a lot easier to flip to received and check for payments.

Moved the "Your Addresses" book inside the main address book. It was confusing having two address books.

I found the "To:" in "From: unknown, To: (one of your bitcoin addresses)" still confusing, so I changed it to "From: unknown, Received with:". The bitcoin address is abbreviated so you can see the label that you set in the Receiving tab of the address book.

Fixed a few UI glitches from the upgrade to wxWidgets 2.9.0.

I haven't forgotten about you people who want non-UI, but I had to do some fun stuff before more build bashing.

BitcoinTalk

Re: generation slowed down dramatically

2010-02-23 00:49:56 UTC - -

Just a random streak of bad luck. It looks steady to me.

Competition doesn't have an effect until the next automatic retarget adjustment, and we haven't reached the next one yet.

The adjustments are every 2016 blocks. To calculate our progress towards the next one, divide the block total by 2016. The fractional part is how far we are to the next one.

My back-of-the-envelope projection: 42032 blocks/2016 = 20.85 = 85% of the way. About 1.5 days to go until the next one. That'll only be about 10 days since the last one, the target is 14 days, so 14/10 = 1.4 = around 40% difficulty increase.

BitcoinTalk

Re: UI improvements

2010-02-23 01:16:28 UTC - -

There are now "Sending" and "Receiving" tabs in the Address Book. Your addresses are referred to as "receiving addresses".

madhatter was working on building it on Mac. He had errors probably caused by UTF-16 wxWidgets 2.8. Should have better luck now with 2.9.0. wxWidgets 2.9.0 is UTF-8 and wouldn't have that problem.

I think he had it working on FreeBSD, but he wanted a non-UI version.

I have the command line and JSON-RPC daemon version working now. Will SVN it in a day or two.

I disabled gdm on my Ubuntu system so it boots into command line. I hope I will be able to get it enabled again with rcconf.

BitcoinTalk

Re: Bitcoin Address Collisions

2010-02-23 16:26:09 UTC - -

There's a separate public/private keypair for every bitcoin address. You don't have a single private key that unlocks everything. Bitcoin addresses are a 160-bit hash of the public key, everything else in the system is 256-bit.

If there was a collision, the collider could spend any money sent to that address. Just money sent to that address, not the whole wallet.

If you were to intentionally try to make a collision, it would currently take 2^126 times longer to generate a colliding bitcoin address than to generate a block. You could have got a lot more money by generating blocks.

The random seed is very thorough. On Windows, it uses all the performance monitor data that measures every bit of disk performance, network card metrics, cpu time, paging etc. since your computer started. Linux has a built-in entropy collector. Adding to that,

every time you move your mouse inside the Bitcoin window you're generating entropy, and entropy is captured from the timing of disk ops.

BitcoinTalk

Re: UI improvements

2010-02-23 16:53:27 UTC - -

Quote from: Xunie on February 23, 2010, 12:28:27 PM
/etc/init.d/gdm start and it will start gdm!

Ah yes, there we go, back to normal again.

The ctrl+alt+F[1-8] thing never worked on this computer. The screen just goes haywire.

BitcoinTalk

Command Line and JSON-RPC

2010-02-23 22:15:41 UTC - -

Version 0.2.6 on SVN can now run as a daemon and be controlled by command line or JSON-RPC.

On Linux it needs libgtk2.0-0 installed, but does not need a GUI running. Hopefully gtk can be installed without having a windowing system installed.

The command to start as a daemon is:
bitcoin -daemon [switches...]

Or, to run the UI normally and also be able to control it from command line or JSON-RPC, use the "-server" switch.
bitcoin -server [switches...]

With either switch, it runs an HTTP JSON-RPC server that accepts local socket connections on 127.0.0.1:8332. The port is bound to loopback and can only be accessed from the local machine, but from any account, not just the user it's running under.

To control it from the command line, the interface is a command name without any switches, followed by parameters if any.
bitcoin <command> [params...]

For example:
bitcoin getinfo
bitcoin getdifficulty

bitcoin setgenerate true
bitcoin stop

It's a simple JSON-RPC client and prints the JSON result. Look at rpc.cpp for the list of commands.

Web apps or anything automated will normally use JSON-RPC directly, not command line. There are JSON-RPC libraries for all the major languages. In script languages like PHP and Python the syntax is as natural as calling a local function.

BitcoinTalk

Re: Bitcoin Address Collisions

2010-02-23 22:24:00 UTC - -

Quote from: NewLibertyStandard on February 23, 2010, 07:04:47 PM
Are generated bitcoins encrypted with whichever address is currently displayed in the main Bitcoin window?

No, each generated transaction uses a new, single-use address.

Nothing uses the address in the main window, it's just there for convenience for you to copy. 0.2.5 has a "New..." button next to it to make it easy to change each time you use it.

BitcoinTalk

Re: URI-scheme for bitcoin

2010-02-24 05:57:43 UTC - -

That would be nice at point-of-sale. The cash register displays a QR-code encoding a bitcoin address and amount on a screen and you photo it with your mobile.

BitcoinTalk

Re: Command Line and JSON-RPC

2010-02-24 06:17:23 UTC - -

Quote from: theymos on February 24, 2010, 03:07:37 AM
Quote from: satoshi on February 23, 2010, 10:15:41 PM
On Linux it needs libgtk2.0-0 installed
Will this requirement be removed sometime? I'd rather not have to deal with GTK.

How much "dealing with" does GTK actually require? Is it just a matter of "sudo apt-get install libgtk2.0-0" and having some extra libraries sitting around? GTK doesn't have to do anything, just be there for bitcoin to link to when it loads up, have the gtk-init-check call fail because no GUI present, then it's done.

It saves us butchering everything with ifdefs and a separate compile and binary to use wxBase just to try to avoid linking GTK.

BitcoinTalk

New icon/logo

2010-02-24 21:24:23 UTC - -

New icons, what do you think? Better than the old one?

bitcointalk.org image proxy: Invalid image bitcointalk.org image proxy: Invalid image
bitcointalk.org image proxy: Invalid image bitcointalk.org image proxy: Invalid image

Full size 530x529 image for scaling down to custom sizes:
http://www.bitcoin.org/download/bitcoin530.png

The perspective shadow was too thick on the larger sizes. I updated 32, 48 and the full size.

I release these images into the public domain (copyright-free). I request that derivative works be made public domain.

BitcoinTalk

Re: Make your "we accept Bitcoin" logo

2010-02-24 21:53:52 UTC - -

If you GPL stuff, I have to avoid using it. Nothing against GPL per-se, but Bitcoin is an MIT license project. Anything GPL please clearly mark it as such.

BitcoinTalk

Re: Command Line and JSON-RPC

2010-02-24 22:08:55 UTC - -

When and how fast did memory usage increase? Right away, slowly over a long time, or starting at some later event?

I have -daemon running on ubuntu 9.10 64-bit and memory usage is steady.

It has to be something about the difference on the server besides 64-bit. Maybe some malfunction from the lack of GUI. A memory leak debug tool could give a clue.

BitcoinTalk

Re: Proof-of-work difficulty increasing

2010-02-24 22:42:24 UTC - -

The automatic adjustment happened earlier today.

24/02/2010
0000000043b3e50000000000000000000000000000000000000000000000000000

24/02/2010 3.78 +49%

I updated the first post.

BitcoinTalk

Re: New icon/logo

2010-02-25 01:56:24 UTC - -

Quote from: Sabunir on February 25, 2010, 01:47:56 AM
I like them. Do they come in higher resolutions?

Yes, the original is 546x531 pixels.

It looks good at larger size too, but since the small icons are what you mostly always see, I wanted to judge it on those first. I'll post larger sizes and full size a little later.

BitcoinTalk

Re: Command Line and JSON-RPC

2010-02-25 22:54:17 UTC - -

OK, I made a build target bitcoind that only links wxBase and does not link GTK. Version 0.2.7 on SVN.

I split out the init and shutdown stuff from ui.cpp into init.cpp, so now ui.cpp is pure UI. ui.h provides inline stubs if wxUSE_GUI=0. We only have four functions that interface from the node to the UI. In the bitcoind build, we don't link ui.o or uibase.o.

> ***Quote from: sirius-m on February 25, 2010, 04:32:17 PM***
> *It started increasing right away. I'll see if valgrind can help me.*

Sure feels like it could be something in wxWidgets retrying endlessly because some UI thing failed or something wasn't inited correctly. Our hack to ignore the initialize failure and run anyway means we're in uncharted territory. We're relying on the fact that we hardly use wx in this mode. We do still use a few things like wxGetTranslation and wxMutex.

Another way to debug would be to run in gdb, wait until everything is quiet and all threads should be idle, and break it and see which thread is busily doing something and what it's doing.

I suspect bitcoind will probably work fine, but I hope you can still debug the problem.

BitcoinTalk

Re: Proof-of-work difficulty increasing

2010-02-25 23:06:29 UTC - -

The formula is based on the time it takes to generate 2016 blocks. The difficulty is multiplied by 14/(actual days taken). For instance, this time it took 9.4 days, so the calculation was 14/9.4 = 1.49. Previous difficulty 2.53 * 1.49 = 3.78, a 49% increase.

I don't know what you're talking about accepting easier difficulties.

BitcoinTalk

Re: Command Line and JSON-RPC

2010-02-26 16:29:21 UTC - -

wx/clipbrd.h isn't used, move it inside the #if wxUSE_GUI.

Updated headers.h on SVN.

Sorry, I linked to wxbase but I had full wxWidgets on my computer.

The db.h:140 class Db no member named "exisits" is stranger. pdb->get, pdb->put, pdb->del compiled before that. Do you have version 4.7.25 of Berkeley DB?

Db::exists()
http://www.oracle.com/technology/documentation/berkeley-db/db/api_reference/CXX/frame_main.html
http://www.oracle.com/technology/documentation/berkeley-db/db/api_reference/CXX/dbexists.html

I suppose they might have added exists recently, using get before that.

BitcoinTalk

Re: New icon/logo

2010-02-26 23:17:19 UTC - -

Good suggestion. I made the B slightly lighter and the background slightly darker. Very slightly. The foreground is now exactly the same colour as the BC in the old one.

It's kind of OK if you can't easily read the B in the 16x16. At that size, you just need to see that it's a coin. It doesn't matter so much what's embossed on it, just that there be some detail there because it wouldn't look like a coin if it was a blank smooth circle.

It's slightly wider than tall because the dark perspective under it goes more to the right than down.

I finished and posted the 32x31 and 48x47 versions in the first message. I like the 48 a lot.

How does everyone feel about the B symbol with the two lines through the outside? Can we live with that as our logo?

BitcoinTalk

Re: Command Line and JSON-RPC

2010-02-26 23:48:44 UTC - -

Are you using wxWidgets 2.9.0? I don't recommend using anything other than 2.9.0.

It looks like they've got a reference in the wx headers (arrstr.h) to something outside of wxBase.

Removing -D__WXDEBUG__ from bitcoin's makefile would probably solve it.

If that doesn't work and you just want to get it working, you could edit wxWidgets include/wx/arrstr.h, line 167 and comment out the wxASSERT_MSG.

BitcoinTalk

Re: New icon/logo

2010-02-27 04:28:29 UTC - -

Quote from: Cdecker on February 27, 2010, 03:24:07 AM
How about an SVG version? That way we could automatically generate smaller and larger versions as needed.

I don't know how to do SVG, but I did the original very large, over 500 pixels across, so it can be scaled down. I'll give the original when I'm finished.

I had to custom tweak each icon size so the vertical lines land square on their pixels, otherwise they're ugly blurry and inconsistent. Such is the challenge of making icons. The original will be good for scaling to custom sizes between 48 and 500 but not smaller.

BitcoinTalk

Re: wxWidgets 2.9.0

2010-02-27 21:22:53 UTC - -

Quote from: Cdecker on February 27, 2010, 05:09:59 PM
Looking through the source of 2.8.10 it appears that unicode is possible with that version too.

In the Windows world, "unicode" means UTF-16 (wchar).

2.8 has two build variations, ANSI and UTF-16 (unicode). The UTF-16 version is the "unicode" version provided in the Debian package. I believe 2.8 and its UTF-16 build labelled simply "unicode" has been the source of build problems described in the forum. We were previously using 2.8 ANSI in anticipation of getting to UTF-8 without going through UTF-16 hell. We cannot compile with UTF-16.

2.9 has only one version, UTF-8. On Windows, we set the codepage to UTF-8, so on all platforms our code is UTF-8 and wxWidgets interfaces with us in UTF-8. On Linux I assume the codepage is already UTF-8. By standardizing on 2.9 we avoid the multi-build confusion of 2.8, and we need 2.9 for UTF-8 internationalization.

Make sure you read build-unix.txt and configure wxWidgets using the configure parameters given.

Curious, why is it incredibly hard to provide wxWidgets 2.9.0? If you mean for users, that's why we static link it.

It's unfortunate that we require so many big dependencies, but we need them all. At least on Debian/Ubuntu, all but wxWidgets are available as packages. Eventually they'll provide a 2.9 package.

BitcoinTalk

Re: New icon/logo

2010-03-02 02:33:05 UTC - -

We have the standard icon sizes, and the full size scales nicely to anything else.

I added the full size to the first post.

BitcoinTalk

Re: Money Transfer Regulations

2010-03-03 04:28:56 UTC - -

When there's enough scale, maybe there can be an exchange site that doesn't do transfers, just matches up buyers and sellers to exchange with each other directly, similar to how e-bay works.

To make it safer, the exchange site could act as an escrow for the bitcoin side of the payment. The seller puts the bitcoin payment in escrow, and the buyer sends the conventional payment directly to the seller. The exchange service doesn't handle any real world money.

This would be a step better than e-bay. E-bay manages to work fine even though shipped goods can't be recovered if payment falls through.

BitcoinTalk

Re: Command Line and JSON-RPC

2010-03-05 01:46:25 UTC - -

Quote from: sirius-m on February 24, 2010, 06:17:35 PM

This is strange... When I start Bitcoin as a daemon on my 64 bit Linux server, it eats up all the 250MB of remaining RAM, 700MB of swap and eventually crashes. On my 32 bit Ubuntu desktop, it works fine and stays at 15MB of memory usage. The server is running a 64 bit build of Bitcoin. Maybe there's something wrong with the build or something.

sirius-m debugged this, it was 64-bit related.

The fix is now available on SVN, file util.cpp.

BitcoinTalk

Re: bitcoin auto-renice-ing

2010-03-15 18:44:12 UTC - -

It sets different priorities for each thread. The generate threads run at PRIO_MIN. The other threads rarely take any CPU and run at normal.

```
#define THREAD_PRIORITY_LOWEST          PRIO_MIN
#define THREAD_PRIORITY_BELOW_NORMAL    2
#define THREAD_PRIORITY_NORMAL          0
```

The priorities converted from Windows priorities were probably from a table like this:

"The following table shows the mapping between nice values and Win32 priorities. Refer to the Win32 documentation for SetThreadPriority() for more information on Win32 priority issues.

```
nice value    Win32 Priority
-20 to -16    THREAD_PRIORITY_HIGHEST
-15 to -6     THREAD_PRIORITY_ABOVE_NORMAL
-5 to +4      THREAD_PRIORITY_NORMAL
+5 to +14     THREAD_PRIORITY_BELOW_NORMAL
+15 to +19    THREAD_PRIORITY_LOWEST"
```

If you have better values, suggestions welcome.

Also, there was some advice on the web that PRIO_PROCESS is used on Linux because threads are processes. If that's not true, maybe it accounts for unexpectedly setting the priority of the whole app.

```
// threads are processes on linux, so PRIO_PROCESS affects just the one thread
setpriority(PRIO_PROCESS, getpid(), nPriority);
```

BitcoinTalk

Idea for file hosting and proxy services

2010-03-15 19:16:56 UTC - -

When you want to upload an image to embed in a forum post, there are services like imageshack, but because they're free, they limit the number of views. It's a minuscule amount of bandwidth cost, but they can't just give it away for free, there has to be something in it for them. It would be nice to be able to pay for the bandwidth and avoid the limits, but conventional payments are too inconvenient for such a minor thing.

It's worse if you want to upload a file for others to download. There are services like rapidshare, but they require the downloaders to go through extra steps and delays to make them look at advertising or encourage upgrading to a paid subscription, and they limit it to 10 or so downloads.

It would be nice if we made some free PHP code for an image and file hosting service that charges Bitcoins. Anyone with some extra bandwidth quota could throw it on their webserver and run it. Users could finally pay the minor fee to cover bandwidth cost and avoid the limits and hassles. Ideally, it should be MIT license or public domain.

Services like this would be great for anonymous users, who have trouble paying for things.

BitcoinTalk

Re: On IRC bootstrapping

2010-03-16 19:48:47 UTC - -

Thanks soultcer for talking with the Freenode staffer. Good to know it's OK at the current size, and now they know who we are. They're supportive of projects like TOR so I hope they would probably be friendly to us. We don't want to overstay our welcome. If we get too big, then by the same token, we're big enough that we don't need IRC anymore and we'll get off.

We only needed IRC because nobody had a static IP. In the early days there were some steady supporters, but they all had pool-allocated IPs that change every few days. IRC was only intended as a temporary solution. Bitcoin's built-in addr system is the main solution.

Bitcoin can get the list of IPs from any bitcoin node. In that sense, every node serves as a directory server.

When there are enough static IP nodes to have a good chance that at least one will still be running by the time the current version goes out of use, we can preprogram a seed list.

How do you think we should compile the seed list? Would it be OK to create it from the currently connected IPs that have been static for a while?

BTW, if we want to supplement by deploying separate directory server software, may I suggest IRC? IRC is a good directory server (I've heard it has other uses too), and there are mature IRC server implementations available that anyone can run. Bitcoin's IRC client implementation is already thoroughly tested.

BitcoinTalk

Re: Idea for file hosting service

2010-03-16 20:17:34 UTC - -

That's a great idea. There's a thriving business in those services, but I've always thought the standard payment methods are at odds with privacy minded customers.

Would you consider making your software freely available so anyone could easily set one up? I know for competitive reasons the inclination is to keep it to yourself, but it could get an order of magnitude more use if anyone could give proxy access to their country just by putting the software on a server.

I wonder if there are other kinds of web application servers where we would only have to tack on the payment mechanism to an already existing system?

BitcoinTalk

Re: who is bitcoin.com

2010-03-23 15:22:41 UTC - -

It's unrelated. There wasn't anything there when I started.

The price of .com registrations is lower than it should be, therefore any good name you might think of is always already taken by some domain name speculator. Fortunately, it's standard for open source projects to be .org.

BitcoinTalk

Re: Exchange Methods

2010-03-23 17:35:34 UTC - -

LR and Pecunix have many established exchanges to paper currencies by various payment methods, and a number of vendors accept them as payment, so an exchange link between Bitcoin and LR/Pecunix would give us 2nd-hop access to all that. The possibility to cash out through them would help support the value of bitcoins.

Bitcoin has unique properties that would be complementary. LR/Pecunix are easy to spend anonymously, but hard to buy anonymously and not worth the trouble to buy in small amounts. Bitcoin, on the other hand, is easy to get in small amounts anonymously. It would be convenient to buy LR/Pecunix with bitcoins rather than through conventional payment methods.

Most customers who convert to LR to buy something would probably ask the seller first if they accept Bitcoin, encouraging them to start accepting it.

BitcoinTalk

Re: Idea for file hosting and proxy services

2010-03-24 18:01:57 UTC - -

Title changed.

It helps that we have someone with actual experience running a proxy service. Do you think Psiphon is the best one currently? (sometimes the one you run was the best when you started but you found better ones later)

BitcoinTalk

Re: Idea for file hosting and proxy services

2010-03-24 18:02:55 UTC - -

Mihalism Multi Host is a popular open source PHP file hosting server.

It's geared toward image hosting, but I think by increasing the file size limit and liberalising the allowed file extensions, it could just as easily be used for general file upload hosting. They need the limits to keep it reasonable as a free service, but if we bolt on a Bitcoin payment mechanism, the limits could be relaxed.

It doesn't have a bunch of client side scripting or anti-embedding junk to rip out. It generates standard links that work normally.

There's a turnover churn in these free hosting sites. Small sites can give free image hosting, but once one starts getting popular, it gets too swamped with moochers using them for free bandwidth. Any site that gets well known has to become more aggressively pay-naggy to cover bandwidth costs. It's a perfect example of a service where the needed

price point is in the no-man's-land between just a little too expensive to be free, but too cheap for most users to take the trouble of a conventional payment. It's in the gap between 0 and 19.95. The best they can do is try to maybe get 1 out of 1000 users to pay 9.95, but that has 999/1000 users treated like freeloaders. It can't really be advertising supported because the images are embedded in other sites and downloaded without going to the hosting site.

An example of a site running the software:
http://www.imagez.ws/

Forum:
http://www.mihalism.net/

Download:
http://code.google.com/p/mihalismmh/

What do you think? If I made a Bitcoin payment integration for this, would anyone be interested in running it? It might be the first fully automated service available to buy with Bitcoins. The advantage it could offer over the free services is general file upload hosting of large files without making downloading users go to the upload site and jump through hoops. It would give a normal link directly to the file.

BitcoinTalk

Re: Could the bitcoin network be destroyed by someone generating endless bitcoin add

2010-05-16 21:01:44 UTC - -

When you generate a new bitcoin address, it only takes disk space on your own computer (like 500 bytes). It's like generating a new PGP private key, but less CPU intensive because it's ECC. The address space is effectively unlimited. It doesn't hurt anyone, so generate all you want.

BitcoinTalk

Re: For a website taking payments with bitcoins, better: IP or bitcoin addresses?

2010-05-16 21:37:36 UTC - -

Quote from: Xunie on May 14, 2010, 09:52:53 PM
I suggest we disable IP transactions while the user uses a Proxy!
Just to be on the safe side.

That's a good idea. At the very least a warning dialog explaining that it'll connect to the IP and send the information cleartext, giving the chance to cancel.

BitcoinTalk

Re: Exception: 9key_error error

2010-05-16 22:53:59 UTC - -

Does it happen every time you run it, or just happened once at some random time?

I've never seen that fail before. It's a call to OpenSSL that I assumed would never fail, but I put an error check there just in case. I can't imagine how it would fail. Out of memory maybe.

The code is:

key.h:

```
EC_KEY* pkey;

pkey = EC_KEY_new_by_curve_name(NID_secp256k1);
if (pkey == NULL)
    throw key_error("CKey::CKey() : EC_KEY_new_by_curve_name failed");
```

NID_secp256k1 is a constant.

BitcoinTalk

Re: removing bitcoin addresses

2010-05-16 23:34:40 UTC - -

SheriffWoody:
Bitcoin addresses you generate are kept forever. A bitcoin address must be kept to show ownership of anything sent to it. If you were able to delete a bitcoin address and someone sent to it, the money would be lost. They're only about 500 bytes.

sirius-m:
Thousands of own addresses should not be any problem at all. If you've generated 50000 BTC, then you already have 1000 own addresses, one for each 50 generated. Those are hidden, they're not shown in the UI.

It would be a good idea to add a little code that keeps giving the same address to the same IP. Here's what I did in C++ to keep giving the same key (aka bitcoin address) until they use it:

```
// Keep giving the same key to the same ip until they use it
if (!mapReuseKey.count(pfrom->addr.ip))
```

```
        mapReuseKey[pfrom->addr.ip] = GenerateNewKey();

    ...sends the key mapReuseKey[pfrom->addr.ip]

...later...

    // Received something with this key
    mapReuseKey.erase(pfrom->addr.ip);
```

If it's not convenient to know when you've received, just clear the cached keys every 20 minutes.

I want to add a parameter to getnewaddress for number of days to expire if nothing is received with the address.

BitcoinTalk

Re: Setting up multiple bitcoin machines behind NAT

2010-05-16 23:56:03 UTC - -

At the moment, it always assumes the incoming port is 8333, so it would tell other bitcoin nodes to connect to router:8333 even if you're redirecting from another port number.

I'm not in a big hurry to fix this because I can't think of any benefit to having more than one incoming connection port. If you're providing one incoming port, then you've done your bit to help the network. Having two incoming ports to the same person doesn't help redundancy.

If you have many computers, then using the -connect switch on most of them to connect locally makes more sense.

BitcoinTalk

Re: Is there a way to automate bitcoin payments for a website?

2010-05-18 02:58:11 UTC - -

A little late, but in case anyone else has the same issue. The compile dump had 2 warnings (that were 20 lines long) and 2 link errors. The errors were:

Quote

obj/nogui/init.o(.gnu.linkonce.t._ZNK13wxArrayString4ItemEm+0x13): In function `wxArrayString::Item(unsigned long) const':
/usr/local/include/wx-2.9/wx/buffer.h:42: undefined reference to `wxTheAssertHandler'

obj/nogui/init.o(.gnu.linkonce.t._ZNK13wxArrayString4ItemEm+0x45): In function `wxArrayString::Item(unsigned long) const':
/usr/src/bitcoin/trunk/uint256.h:526: undefined reference to `wxOnAssert(char const, int, char const*, char const*, wchar_t const*)'*

Those are probably due to switching to the release build of wxWidgets instead of debug. They're moving towards only debug build and ditching the release build, so they probably don't care that their release build is broken by referring to non-existent assert stuff. There's nothing to fear about the debug build. It's fully suitable for releases.

bitcoind runs as a daemon and can either be controlled by command line or JSON-RPC.

Thanks madhatter and generica for detailing the instructions for building on freebsd.

BitcoinTalk

Re: Ummmm... where did my bitcoins go?

2010-05-18 20:06:46 UTC - -

It's not the download so much as verifying all the signatures in all the blocks as it downloads that takes a long time.

How long is the initial block download typically taking? Does it slow down half way through or is about the same speed the whole way?

I've thought about ways to do a more cursory check of most of the chain up to the last few thousand blocks. It is possible, but it's a lot of work, and there are a lot of other higher priority things to work on.

Simplified Payment Verification is for lightweight client-only users who only do transactions and don't generate and don't participate in the node network. They wouldn't need to download blocks, just the hash chain, which is currently about 2MB and very quick to verify (less than a second to verify the whole chain). If the network becomes very large, like over 100,000 nodes, this is what we'll use to allow common users to do transactions without being full blown nodes. At that stage, most users should start running client-only software and only the specialist server farms keep running full network nodes, kind of like how the usenet network has consolidated.

SPV is not implemented yet, and won't be implemented until far in the future, but all the current implementation is designed around supporting it.

In the meantime, sites like vekja.net and www.mybitcoin.com have been experimenting with account-based sites. You create an account on a website and hold your bitcoins on account there and transfer in and out. Creating an account on a website is a lot easier than installing and learning to use software, and a more familiar way of doing it for most

people. The only disadvantage is that you have to trust the site, but that's fine for pocket change amounts for micropayments and misc expenses. It's an easy way to get started and if you get larger amounts then you can upgrade to the actual bitcoin software.

BitcoinTalk

Re: We accept Bitcoins

2010-05-20 21:43:42 UTC - -

Quote from: DataWraith on May 19, 2010, 07:52:42 PM

Can I just butt in with a question on why that is? To me it seems that if Bitcoin uses public-key cryptography to transfer ownership of the coins, it should be a trivial matter to include a short message that is only readable by the recipient.

Almost but not quite. Bitcoin uses EC-DSA, which can only do digital signing, not encryption. RSA can do both, but I didn't use it because it's an order of magnitude bigger and would have been impractical.

BitcoinTalk

JSON-RPC programming tips using labels

2010-05-26 18:27:25 UTC - -

I added label related functions to help with managing multiple addresses per user. New or renamed functions are:

getreceivedbyaddress -- amount received on a single address
getreceivedbylabel -- amount received by all addresses with this label
listreceivedbyaddress -- list addresses and amounts they've received
listreceivedbylabel -- list labels and amounts they've received
setlabel -- misc label functions for completeness
getlabel
getaddressesbylabel

For consistency I renamed getamountreceived->getreceivedbyaddress and getallreceived->listreceivedbyaddress. The old names are still there so as not to break existing code, but they're deprecated.

The idea is that if you give the username whenever you call getnewaddress, you can get the user's total received across all their addresses using the "bylabel" functions. You can freely change their address without worrying about tracking all their old addresses.

A good way to automate changing the user's receiving address: just before displaying their current address, check if it has been used to receive anything, if it has then replace it with a new one:

```
// Get a new address whenever the current one has received anything
if (strAddr == "" || getreceivedbyaddress(strAddr) > 0)
   strAddr = getnewaddress(strUsername); // Label the address with username
Display(strAddr); // Display their current receiving address

// Get total received by all the user's addresses
getreceivedbylabel(strUsername, 0) // unconfirmed
getreceivedbylabel(strUsername, 1) // available balance
```

If you're just getting one particular user's balance, such as in response to a page request by that user, use getreceivedbylabel, but if you're scanning over all users, it's better to use listreceivedbylabel to get the complete list and scan against the result. Scanning users with getreceivedbylabel would be n-squared, using listreceivedbylabel is n-log-n (or n linear).

You should only really need to scan all users if you're polling in order to spontaneously take action in response to money received, rather than the user going to a webpage, seeing their balance and telling you what to do with it. It's not necessary to poll very frequently. If you require 1 confirmation, that'll take an average of 10 minutes anyway, so there's no point in polling more often than every few minutes.

If you're selling digital goods and services, where you don't lose much if someone gets a free access, and it can't be resold for profit, I think you're fine to accept 0 confirmations.

It's mostly only if you were selling gold or currency that you'd need multiple confirmations.

BitcoinTalk

Re: Tracing a coin's lineage

2010-05-26 18:51:04 UTC - -

Quote from: Xunie on May 26, 2010, 12:50:04 AM

Can't we force a user to use a new address for receiving payments?
Every time a payment is received display another Bitcoin address in the address bar.
(only transactions via Bitcoin addresses, NOT IPs of course, since that'd be useless, right?)
The actual key would still be kept to ensure that the user would still receive payments of people sending to the same address.

This is on my list. I will soon make the "Your Bitcoin Address:" window automatically change whenever you receive anything to the address displayed.

I'm also recommending this approach for the implementation of web apps. I just posted some sample code showing a suggested way of implementing this.

Versions on SVN since 0.2.4 already have a "New..." button next to the address bar to encourage changing it manually too.

@theymos: If nothing else, we can fall back on that solution in the future.

BitcoinTalk

Re: CLI bitcoin generation

2010-05-26 20:09:34 UTC - -

Quote from: molybdenum on May 22, 2010, 06:44:20 PM

An optional parameter to specify the minimum number of blocks after that transaction (getallreceived 1 for current behavior, or just getallreceived, getallreceived 5 for the paranoid, getallreceived 0 for instant confirms)?

Yeah, that actually is what it is. getallreceived 0 should do what you want. (now it's renamed to listreceivedbyaddress 0) The default is 1 confirmation, but I think in reality most digital goods and services can be 0 confirmations. Like you say, if you need more than 0 confirmations, you could show two numbers, unconfirmed and available balance, so they immediately see their transaction went through.

listreceivedbyaddress [minconf=1] [includeempty=false]
[minconf] is the minimum number of confirmations before payments are included.
[includeempty] whether to include addresses that haven't received any payments.
Returns an array of objects containing:
"address" : receiving address
"label" : the label of the receiving address
"amount" : total amount received by the address
"confirmations" : number of confirmations of the most recent transaction included

or listreceivedbylabel if you're labelling addresses with their username.

So far I've concentrated on functions for web merchants, not so much on stuff for remote management of headless coin generators yet.

BitcoinTalk

Re: Share database blocks ?

2010-05-26 20:34:34 UTC - -

It does in fact download 500 blocks at a time, then the counter counts one at a time as it verifies the blocks.

The advantage of letting bitcoin download and verify the blocks is that you do not have to trust the person you're downloading them from. If you downloaded the blk*.dat files from some site, you would have to trust that site, since you would be accepting the data without verifying it yourself. If you're copying blk*.dat from another computer of yours, that should be fine.

How long is the initial block download taking for you?

BitcoinTalk

Re: Website translations

2010-05-26 21:16:34 UTC - -

Does anyone want to translate the Bitcoin client itself? It would be great to have at least one other language in the 0.3 release.

All you have to do is get poedit and translate the po file I'm attaching to this post. It's less than 750 words.

Updated bitcoin.po attachment for 0.3.1

BitcoinTalk

Re: Odd amount of generated coins

2010-05-26 21:34:32 UTC - -

In the SVN version, if a transaction requires a transaction fee, it says
"This transaction is over the size limit. You can still send it for a fee of #,
which goes to the nodes that process your transaction and helps to support the network.
Do you want to pay the fee?"

If you don't have enough money with the fee added, it says
"Total exceeds your balance when the # transaction fee is included "

BitcoinTalk

Re: Website translations

2010-05-27 14:18:22 UTC - -

Hurray! We have our first language. I uploaded it to SVN to go in with the 0.3 release.

BitcoinTalk

Re: Hostnames instead of IP Addresses

2010-06-02 18:18:15 UTC - -

The current sending by IP is not very useful: it connects to the IP, so you'd like to use TOR for anonymity, but then it can totally be eavesdropped and man-in-the-middled.

The future plan for sending to an IP is to make it a bitcoin address plus IP, like:

1auaDZCFYqaGx4FKS5WenNfurk2SkoDu4h<someseparatorcharacter>1.2.3.4
or
1auaDZCFYqaGx4FKS5WenNfurk2SkoDu4h<someseparatorcharacter>domain.com

I need suggestions for the separator character. ":" is a candidate, but IPv6 has : in it and that might get confusing. Something that's allowed in url parameters would be nice.

I want to use SSL for the connection, using the bitcoin address' public key as the cert. You would be certain you're connected to who you thought, and safely encrypted. The bitcoin address would not be used for the transaction, only for authentication. A new generated bitcoin address would be sent through the SSL connection.

Since it's authenticated, it would then be safe to allow the IP address to be a domain name. Some care taken that if a proxy is used, it uses socks4a instead of DNS lookup.

BitcoinTalk

Re: Proof-of-work difficulty increasing

2010-06-02 18:45:38 UTC - -

That's a good idea. I'm not sure where exactly to fit that in, but it could certainly calculate the expected average time between blocks generated, and then people would know what to expect.

Every node and each processor has a different public key in its block, so they're guaranteed to be scanning different territory.

Whenever the 32-bit nonce starts over at 1, bnExtraNonce gets incremented, which is an arbitrary precision integer.

BitcoinTalk

Re: Website translations

2010-06-02 22:18:09 UTC - -

I uploaded the 93% complete Dutch translation to SVN. Thanks!

BitcoinTalk

Re: On IRC bootstrapping

2010-06-14 18:13:21 UTC - -

Bitcoin has its own distributed address directory using the "addr" message. It's about time we coded in a list of the current long running static nodes to seed from. I can add code so new nodes do not preferentially stay connected to the seed nodes, just connect and get the list, so it won't be a burden on them.

What do you think, should I go ahead with adding the seeds?

It'll still try IRC first. The IRC has the advantage that it lists nodes that are currently online, since they have to stay connected to stay on the list, but the disadvantage that it's a single point of failure. The "addr" system has no single point of failure, but can only tell you what nodes have recently been seen, so it takes a little longer to get connected since some of the nodes you try have gone offline. The combination of the two gets us the best of both worlds and more total robustness.

Is there anyone who wants to volunteer to run an IRC server in case freenode gets tired of us?

BitcoinTalk

Re: Hostnames instead of IP Addresses

2010-06-14 19:53:44 UTC - -

SirArthur has a good point about the normal online merchant case, which is what the send-by-IP option is more suited to. This is the case where the merchant will have a server on a static IP and their own domain name and SSL cert.

Instead of connecting by IP, we can connect to a domain name by SSL, using the existing CA infrastructure to authenticate that you're connected to the owner of that domain.

The user would send to domain.com (or www.domain.com is ok too). That would be very natural and users could see and verify that what they entered is who they intend to pay.

The SSL also makes it safe for TOR users.

Problem is, I think merchants would still prefer to use bitcoin addresses to be certain they know what the payment is for. You simply cannot count on users to enter the right thing in the comment fields to identify the transaction. It would only approach practical if we had a mailto style link that prepopulates the comment field with the order number, but then the link could just as well be a bitcoin address.

Just having an open bitcoin server at domain.com that users could send unidentified payments to would be too much of a liability. Regular users aren't used to the idea of having to identify the payment. Merchants would get too many blank payments followed by "I paid you, where's my stuff?!" a week later.

The payment sequence does have a step where the receiver verifies the order before accepting it. It can reject the payment and return an error message if it doesn't contain a valid order number. That would require a difficult level of integration of custom code with the bitcoin server though.

BitcoinTalk

Re: Dealing with SHA-256 Collisions

2010-06-14 20:39:50 UTC - -

SHA-256 is very strong. It's not like the incremental step from MD5 to SHA1. It can last several decades unless there's some massive breakthrough attack.

If SHA-256 became completely broken, I think we could come to some agreement about what the honest block chain was before the trouble started, lock that in and continue from there with a new hash function.

If the hash breakdown came gradually, we could transition to a new hash in an orderly way. The software would be programmed to start using a new hash after a certain block number. Everyone would have to upgrade by that time. The software could save the new hash of all the old blocks to make sure a different block with the same old hash can't be used.

BitcoinTalk

Re: Technical clarifications

2010-06-14 22:21:55 UTC - -

3) Nothing, if sending by bitcoin address
5) It is decentralised. After you have connected to the network the first time, you no longer need IRC.

BitcoinTalk

Re: Can't Build r80 from SVN

2010-06-14 22:40:14 UTC - -

Sorry, I didn't test compile on linux the last few revisions.

Reverted makefile.unix.

BitcoinTalk

Re: What is the incentive to collect transactions?

2010-06-15 23:41:29 UTC - -

Quote from: theymos on June 05, 2010, 04:26:09 PM
Adding transactions to the block you're working on will slow down your generation rate

The premise is false. Adding more transactions to the block you're working on does NOT slow down your generation rate. When generate is scanning hashes, it only hashes the header of the block, which is constant size. The header contains a hash of the transactions (the Merkle root) and is only updated occasionally.

If necessary I can write code to make nodes prefer not to use a block if it doesn't contain enough of the transactions they know about. A discouraged block would almost always fail to be included in the main chain, but would be accepted if it did get in. I doubt this will be necessary, since there's no real advantage for nodes not to include all transactions.

BitcoinTalk

Re: URI-scheme for bitcoin

2010-06-16 00:15:47 UTC - -

http://127.0.0.1:8330/?to=domain.com&amount=200.00&comment=order_12345
or
http://127.0.0.1:8330/?to=<bitcoinaddress><separatorchar>1.2.3.4&amount=200.00

But as long as the link is already doing the typing for you, I don't see much benefit in using a domain address instead of bitcoin address. With a bitcoin address, the user can't send an unidentified payment. They can't send payment until they've been given a correct bitcoin address to send to.

What would be nice about sending by domain is you could visually verify who it's going to.

A more crucial issue is what if the browser isn't allowed to connect to 127.0.0.1: http://BitcoinTalk.org/index.php?topic=63.msg1589#msg1589

and if that's true, then what about that example freenet link that had 127.0.0.1 in it?

BitcoinTalk

Re: Website translations

2010-06-16 16:53:34 UTC - -

Thanks DataWraith! The German translation is uploaded to SVN.

This is great, we've already got 3 major languages.

BitcoinTalk

Re: new binary release?

2010-06-17 17:07:56 UTC - -

I'm working on getting version 0.3 released as soon as I can. Just a last few things left to do. It's been a long time since 0.2 and we need to get a prebuilt bitcoind with command line and JSON-RPC available. This time we'll have both 32-bit and 64-bit linux binaries, and Laszlo is going to build a Mac OSX release. Plus, we'll include the German, Dutch and Italian translations by DataWraith, Xunie and Joozero (thanks you guys!).

BitcoinTalk

Re: Transactions and Scripts: DUP HASH160 ... EQUALVERIFY CHECKSIG

2010-06-17 18:46:08 UTC - -

The nature of Bitcoin is such that once version 0.1 was released, the core design was set in stone for the rest of its lifetime. Because of that, I wanted to design it to support every possible transaction type I could think of. The problem was, each thing required special support code and data fields whether it was used or not, and only covered one special case at a time. It would have been an explosion of special cases. The solution was script, which generalizes the problem so transacting parties can

describe their transaction as a predicate that the node network evaluates. The nodes only need to understand the transaction to the extent of evaluating whether the sender's conditions are met.

The script is actually a predicate. It's just an equation that evaluates to true or false. Predicate is a long and unfamiliar word so I called it script.

The receiver of a payment does a template match on the script. Currently, receivers only accept two templates: direct payment and bitcoin address. Future versions can add templates for more transaction types and nodes running that version or higher will be able to receive them. All versions of nodes in the network can verify and process any new transactions into blocks, even though they may not know how to read them.

The design supports a tremendous variety of possible transaction types that I designed years ago. Escrow transactions, bonded contracts, third party arbitration, multi-party signature, etc. If Bitcoin catches on in a big way, these are things we'll want to explore in the future, but they all had to be designed at the beginning to make sure they would be possible later.

I don't believe a second, compatible implementation of Bitcoin will ever be a good idea. So much of the design depends on all nodes getting exactly identical results in lockstep that a second implementation would be a menace to the network. The MIT license is compatible with all other licenses and commercial uses, so there is no need to rewrite it from a licensing standpoint.

BitcoinTalk

Re: Transactions and Scripts: DUP HASH160 ... EQUALVERIFY CHECKSIG

2010-06-18 16:17:14 UTC - -

A second version would be a massive development and maintenance hassle for me. It's hard enough maintaining backward compatibility while upgrading the network without a second version locking things in. If the second version screwed up, the user experience would reflect badly on both, although it would at least reinforce to users the importance of staying with the official version. If someone was getting ready to fork a second version, I would have to air a lot of disclaimers about the risks of using a minority version. This is a design where the majority version wins if there's any disagreement, and that can be pretty ugly for the minority version and I'd rather not go into it, and I don't have to as long as there's only one version.

I know, most developers don't like their software forked, but I have real technical reasons in this case.

Quote from: gavinandresen on June 17, 2010, 07:58:14 PM

I admire the flexibility of the scripts-in-a-transaction scheme, but my evil little mind immediately starts to think of ways I might abuse it. I could encode all sorts of

interesting information in the TxOut script, and if non-hacked clients validated-and-then-ignored those transactions it would be a useful covert broadcast communication channel.

That's a cool feature until it gets popular and somebody decides it would be fun to flood the payment network with millions of transactions to transfer the latest Lady Gaga video to all their friends...

That's one of the reasons for transaction fees. There are other things we can do if necessary.

Quote from: laszlo on June 17, 2010, 06:50:31 PM

How long have you been working on this design Satoshi? It seems very well thought out, not the kind of thing you just sit down and code up without doing a lot of brainstorming and discussion on it first. Everyone has the obvious questions looking for holes in it but it is holding up well 

Since 2007. At some point I became convinced there was a way to do this without any trust required at all and couldn't resist to keep thinking about it. Much more of the work was designing than coding.

Fortunately, so far all the issues raised have been things I previously considered and planned for.

BitcoinTalk

Re: On IRC bootstrapping

2010-06-18 17:28:18 UTC - -

The SVN version now uses IRC first and if that fails it falls back to a hardcoded list of seed nodes. There are enough seed nodes now that many of them should still be up by the time of the next release. It only briefly connects to a seed node to get the address list and then disconnects, so your connections drop back to zero for while. At that point, be patient. It's only slow to get connected the first time.

This means TOR users won't need to -addnode anymore, it'll get connected automatically.

BitcoinTalk

Re: Get 5 free bitcoins from freebitcoins.appspot.com

2010-06-18 23:08:34 UTC - -

Excellent choice of a first project, nice work. I had planned to do this exact thing if someone else didn't do it, so when it gets too hard for mortals to generate 50BTC, new users could get some coins to play with right away. Donations should be able to keep it filled. The display showing the balance in the dispenser encourages people to top it up.

You should put a donation bitcoin address on the page for those who want to add funds to it, which ideally should update to a new address whenever it receives something.

BitcoinTalk

Re: Bitcoin in Ubuntu 10.04

2010-06-21 17:20:21 UTC - -

Quote from: NewLibertyStandard on May 23, 2010, 04:28:12 PM

Bitcoin looks ugly in Ubuntu's new default theme. It seems that some, but not all of the theme settings are being picked up. The unselected file menu should have light text with a dark background, but it incorrectly has light text with a light background. They're similar enough that it's unreadable on my display. It should be fixed before the next stable release.

This is now fixed in the SVN version.
1) Menu bar default color.
2) Balance bar not a different color.
3) Background behind bitcoin address and balance now the same color as toolbar.

I checked all the standard themes and it seems reasonable with all of them.

Ubuntu minimize,maximize,close buttons to the right:
gconf-editor
apps->metacity->general
button_layout=menu:minimize,maximize,close

They've got it awfully buried considering 9 out of 10 users are used to having it on the right.

BitcoinTalk

Re: Dying bitcoins

2010-06-21 17:48:26 UTC - -

Lost coins only make everyone else's coins worth slightly more. Think of it as a donation to everyone.

Quote from: laszlo on June 21, 2010, 01:54:29 PM

I wonder though, is there a point where the difficulty of generating a new coinbase is so high that it would make more sense to try to recover keys for lost coins or steal other people's coins instead? The difficulty of that is really high so for now it makes a lot more sense to generate but I just wonder what the real figures are.. would that ever become more productive? Maybe Satoshi can address this..

Computers have to get about 2^200 times faster before that starts to be a problem. Someone with lots of compute power could make more money by generating than by trying to steal.

BitcoinTalk

Re: Proof-of-work difficulty increasing

2010-06-21 18:09:17 UTC - -

I integrated the hashmeter idea into the SVN version. It displays khash/s in the left section of the status bar.

Two new log messages:
21/06/2010 01:23 hashmeter 2 CPUs 799 khash/s
21/06/2010 01:23 generated 50.00

grep your debug.log for "generated" to see what you've generated, and grep for "hashmeter" to see the performance. On windows, use:
findstr "hashmeter generated" "%appdata%itcoin\debug.log"

I have the hashmeter messages once an hour. How often do you think it should be?

BitcoinTalk

Re: Bitcoin in Ubuntu 10.04

2010-06-22 03:45:56 UTC - -

On Ubuntu 10.04 it wouldn't remove the taskbar button cleanly, so I made it leave it there.

But now that you mention it, it's probably better to have the feature, even if it's messy, than not to have it, though it may confuse a few people when the taskbar button temporarily stays around but disappears if you click on it.

Updated SVN.

Thanks for testing.

BitcoinTalk

0.3 almost ready -- please test the Mac version!

2010-06-22 04:01:53 UTC - -

I finished everything on my list to do for version 0.3. The code on SVN is about ready to release.

Testing at this point is much appreciated.

BitcoinTalk

Re: How fast do the fastest computers generate bitcoins?

2010-06-22 04:35:26 UTC - -

I've noticed that hashing performance doesn't vary as much between CPUs as you'd expect. Compared to an old CPU, a newer CPU doesn't show as much of a speedup at hashing as it does on general benchmarks.

I guess recent CPU optimizations must have concentrated on things like I/O and branch prediction. Most programs are a bunch of memory access, comparisons and branching, they rarely get down to cranking away at maths for very long.

The latest SVN version has a khash/s display. Around 400 khash/s per processor is typical.

BitcoinTalk

Re: Bitcoin in Ubuntu 10.04

2010-06-22 16:39:43 UTC - -

It's too late now for feature changes to 0.3, but I'll add that to the post-0.3 to do list. I never would have noticed that if you hadn't pointed it out.

BitcoinTalk

Re: Proof-of-work difficulty increasing

2010-06-22 16:51:14 UTC - -

Agree. Certainly too trivial to clutter the user's attention with.

I changed it to every 30 minutes.

If I increased it to every 10 minutes, it would still be a small enough presence in the log file. Question is whether that would be more output than the user wants when they grep.

BitcoinTalk

Re: 0.3 almost ready

2010-06-22 17:02:07 UTC - -

Quote from: lachesis on June 22, 2010, 06:20:02 AM

It would be nice if the listtransactions RPC method were finished before the next release, though.

My fear is too many programmers would latch onto that for checking for received payments. It can never be reliable that way. The list/getreceivedbyaddress/label functions are the only way to do it reliably.

We shouldn't delay forever until every possible feature is done. There's always going to be one more thing to do.

BitcoinTalk

Re: 0.3 almost ready

2010-06-22 17:37:08 UTC - -

Here's RC1 for windows for testing:
(removed, see RC2 below)

Please only download this if you're going to test and report back whether everything seems fine or not. Make sure to look through the files in "c:\program filesitcoin"

BitcoinTalk

Re: 0.3 almost ready

2010-06-22 19:11:41 UTC - -

Quote from: davidonpda on June 22, 2010, 06:23:26 PM

EXCEPTION: 22DbRunRecoveryException
DBEnv::open: DB RUNRECOVERY: Fatal error, run database recovery
C:\Program Files\Bitcoinitcoin.exe in OnInit()

What operating system?

Normally when it does that it's because the directory where the data directory should go doesn't exist. See if the "%appdata%" directory exists.

Do you get that error with 0.2 also? It's hard to see how you could get that with 0.3 and not with 0.2 since there's nothing different in that regard.

BitcoinTalk

Re: 0.3 almost ready

2010-06-22 19:25:13 UTC - -

davidonpda, were you also running laszlo's build previously?

Check if the "%appdata%" directory exists, and "%appdata%itcoin"

Try:
rename "%appdata%itcoin" bitcoin2

does it work then?

BitcoinTalk

Re: 0.3 almost ready

2010-06-22 19:46:23 UTC - -

You figured it out faster than I could post a reply. 

It looks like laszlo's build of Berkeley DB has database/log.* files that are not compatible with ours. The .dat files are fine, their format shouldn't ever change. All data is stored in the .dat files. All your own data is stored in wallet.dat. If you had waited for it to redownload the block chain, your missing transactions and generateds would have appeared as the block chain reached the point where those transactions were recorded.

When you copied the directory except log.0000000002, that's the best solution. You should be good now.

The database/log.* files only contain temporary database data. If you exited bitcoin

normally the last time, not exited by forced terminating it or crashing, then the database/log.* files can normally be deleted safely. They're only used so that if the database is in the middle of a transaction when the computer crashes or the program is killed or crashes, then it could recover without losing data.

Please keep running v0.3 if at all possible, don't go back to v0.2.10.

Anyone else who hits this problem, move the database\log.000000000* files somewhere else. (if it works fine after that, you can delete them later)

I'm reluctant to make the installer delete or move those files. If the previous run was stopped by crashing or killed, that would be the wrong thing to do.

BitcoinTalk

Re: 0.3 almost ready

2010-06-22 22:23:39 UTC - -

Laszlo figured out that enabling some more optimisation increased performance about 20%, so 0.3 hashes 20% faster than 0.2.0, but I assume he used that in his own build.

30khash increase to what total rate? (to figure the % increase)

BitcoinTalk

Re: 0.3 almost ready

2010-06-24 17:40:05 UTC - -

Here's RC1 for linux for testing:
(link removed, see below)

It contains both 32-bit and 64-bit binaries.

Recent changes:

build-unix.txt:
- Added instructions for building wxBase, which is needed to compile bitcoind.
- The package libboost-dev doesn't install anything anymore, you need to get libboost-all-dev.
- Updated version numbers.

makefile.unix:
- The libboost libraries have removed the "-mt" from their filenames in 1.40. If you're

compiling with Boost 1.38 or lower, like on Ubuntu Karmic, you would need to change it back to boost_system-mt and boost_filesystem-mt.

BitcoinTalk

Re: 0.3 almost ready

2010-06-25 02:17:41 UTC - -

I don't know. Maybe someone with more Linux experience knows how to install the library it needs.

I built it on Ubuntu 10.04. I hope that wasn't a mistake. Maybe it should have been built on an older version for more backward compatibility. Is this a problem on Linux, that if you build on the latest version, then it has trouble working on older versions? Is there any way I can downgrade to an older version of GCC on 10.04?

The 64-bit version shouldn't be any faster than the 32-bit version, but it would be great if someone could do a side-by-side comparison of the two linux versions and check. SHA-256 is a 32-bit algorithm and nothing in BitcoinMiner uses 64-bit at all.

We don't need to bother with a 64-bit version for Windows. 32-bit programs work on all versions of Windows. It's not like Linux where the 64-bit OS wants 64-bit programs.

I'm also curious if it's a little faster on linux than windows.

Do you think I should make the directories:
/bin32/
/bin64/
instead of
/bin/32/
/bin/64/

BitcoinTalk

Re: 0.3 almost ready

2010-06-25 14:10:06 UTC - -

Thanks virtualcoin, that's a perfect comparison.

The 8% speedup from 32-bit Windows (2310k) to 32-bit Linux (2500k) is probably from the newer version of GCC on Linux (4.4.3 vs 3.4.5).

The 15% speedup from 32-bit to 64-bit Linux is more of a mystery. The code is completely 32-bit.

Hmm, I think the 8 extra registers added by x86-64 must be what's helping. That would make a significant difference to SHA if it could hold most of the 16 state variables in registers.

BitcoinTalk

Re: Bitcoin clients getting k-lined from the IRC bootstrapping channel

2010-06-25 21:15:15 UTC - -

We need more details about what happened MadHatter.

Both 0.2 and 0.3 have a backup way of getting connected without IRC, it's just slower to get connected.

0.2 can find other nodes without IRC if it's ever been connected before, but a new install can't discover the network for the first time without IRC.

0.3 can also seed without IRC. It can operate entirely without IRC if it needs to, but it's better having IRC for redundancy.

BitcoinTalk

Re: On IRC bootstrapping

2010-06-25 22:40:47 UTC - -

Quote from: laszlo on June 14, 2010, 06:30:58 PM

I run an IRC server you can use, it's fairly stable but it's not on redundant connections or anything. It is only two servers right now but we don't mess with it or anything, it just runs.

My box is a dedicated irc server:
2:28PM up 838 days, 20:54, 1 user, load averages: 0.06, 0.08, 0.08

You can use irc.lfnet.org to connect.

This seems like a good idea.

What does everyone think, should we make the switch for 0.3?

BitcoinTalk

Re: 0.3 almost ready

2010-06-26 00:32:09 UTC - -

Lets try using Laszlo's irc.lfnet.org instead of freenode. Here's RC2, that's the only change in it:

(see below for download links)

BitcoinTalk

Re: Bitcoin clients getting k-lined from the IRC bootstrapping channel

2010-06-26 14:28:06 UTC - -

Freenode is too visible, right in the middle of where all those users and moderators are hanging out. Laszlo's option is a much better fit for us.

I made 0.3.0.RC2 available that uses irc.lfnet.org instead of freenode if you want to start switching over:
http://BitcoinTalk.org/index.php?topic=199.msg1787#msg1787

BitcoinTalk

Re: 0.3 almost ready

2010-06-26 15:10:10 UTC - -

The first panel of the status bar is shared with the help description of menu items as you hover over them. Since all our menu item descriptions are blank, it replaces it with blank when you're hovering in a menu.

BitcoinTalk

Beta?

2010-06-26 17:02:43 UTC - -

Is it about time we lose the Beta? I would make this release version 1.3.

BitcoinTalk

Re: 1.3 almost ready

2010-06-26 19:21:05 UTC - -

Changed the version number to 1.3 and removed "Beta".

(links removed, see below)

Uses irc.lfnet.org.

BitcoinTalk

Re: Bitcoin mobile.

2010-06-26 20:58:26 UTC - -

Quote from: sirius-m on June 10, 2010, 01:51:16 PM
You can of course use services like vekja.net or mybitcoin.com on a mobile browser, depositing money there to the extent you trust them.

I think that's the best option right now. Like cash, you don't keep your entire net worth in your pocket, just walking around money for incidental expenses.

They could make a smaller version of the site optimized for mobile. If there was an app, it could be a front end to one of those, with the main feature being QR-code reader, or maybe there's already a universal QR-code reading app that web sites can be designed to accept scans from.

If there was an iPhone app that was just a front end for vekja or mybitcoin, not a big involved P2P, would apple approve it and if not, on what basis? It could always be an Android app instead. An app is not really necessary though, just a mobile sized website.

A web interface to your own Bitcoin server at home wouldn't be a solution for everyone. Most users don't have a static IP, and it's too much trouble to set up port forwarding.

BitcoinTalk

Re: Building BitCoin Client completely Headless

2010-06-26 21:06:06 UTC - -

The linux release candidate in the "1.3 almost ready" thread contains prebuilt bitcoind.

BitcoinTalk

Re: Bitcoin Faucet changes

2010-06-26 21:39:52 UTC - -

Many big ISPs give you a new IP every time you connect, usually in the same class B (a.b.?.?). Maybe you should have a minimum time between payments per class-B.

If you can't solve the problem, you can always keep lowering the amount of bitcoins given until it's manageable, and always require captcha.

BitcoinTalk

Re: Beta?

2010-06-27 12:43:50 UTC - -

But 1.0 sounds like the first release. For some things newness is a virtue but for this type of software, maturity and stability are important. I don't want to put my money in something that's 1.0. 1.0 might be more interesting for a moment, but after that we're still 1.0 and everyone who comes along thinks we just started. This is the third major release and 1.3 reflects that development history. (0.1, 0.2, 1.3)

BitcoinTalk

Re: IPv6, headless client, and more

2010-06-27 13:02:38 UTC - -

Welcome, Harry.

I hadn't thought about starting out using bitcoind without using bitcoin first. I guess for now, this thread serves as the tutorial.

The focus for bitcoind so far has been more on backend support for websites. There's demand for things that would be nice for adminning headless generators like listgenerated. For the moment, you can grep the debug.log file for "generated" and "hashmeter" for some feedback. Generated blocks take about 24 hours before they're credited to your balance.

BitcoinTalk

Re: 1.3 almost ready

2010-06-27 15:30:13 UTC - -

MinGW still only has good old stable 3.4.5. There's not much reason for them to update it.

When I looked at the 3.4.5 compiled SHA disassembly, I couldn't see any room for improvement at all. I can't imagine how 8% more could be squeezed out of it. Is it possible Windows could have 8% more overhead? Not making system calls or anything, just plain busy computational code, could task switching and other housekeeping operations take away that much?

BitcoinTalk

Re: Major Meltdown

2010-06-27 19:06:09 UTC - -

Here's an answer to a similar question about how to recover from a major meltdown.
https://www.bitcoin.org/smf/index.php?topic=191.msg1585#msg1585

Quote from: satoshi on June 14, 2010, 08:39:50 PM

If SHA-256 became completely broken, I think we could come to some agreement about what the honest block chain was before the trouble started, lock that in and continue from there with a new hash function.

If the hash breakdown came gradually, we could transition to a new hash in an orderly way. The software would be programmed to start using a new hash after a certain block number. Everyone would have to upgrade by that time. The software could save the new hash of all the old blocks to make sure a different block with the same old hash can't be used.

BitcoinTalk

Re: Feature Request: Limiting Connections

2010-07-02 19:21:36 UTC - -

Thanks for the feedback on this.

One thing we could do is lower the outbound connections from 15 to 10 or maybe even 5. The choice of 15 was arbitrary. It just needs to be enough for redundancy and fast exponential propagation of messages. 10 would still be plenty. 5 should be fine. 10 is good as a nice round number so users can see that it stopped intentionally.

It would help to implement UPnP so there would be more inbound accepting nodes. Your number of connections is the ratio of inbound accepting nodes to out-only times 15. We need to encourage more people to accept inbound connections.

I will implement a feature to stop accepting inbound connections once you hit a certain number.

Which version are you running?

Anyone know how many connections typical P2P software like BitTorrent can get up to?

BitcoinTalk

Re: 1.3 almost ready

2010-07-02 20:37:17 UTC - -

Quote from: dkaparis on June 27, 2010, 10:02:25 PM
On a related note, is the thing compilable by Visual C++? I'm inclined to give it a try when I get around to it.

It is, but generating is more than twice as slow.

BitcoinTalk

Re: 0.3 almost ready

2010-07-02 21:57:45 UTC - -

(reverted to rc2)

Links removed, 0.3 is now released, so go to http://www.bitcoin.org to download it.

BitcoinTalk

Re: Beta?

2010-07-02 22:03:41 UTC - -

OK, back to 0.3 then.

Please download RC4 and check it over as soon as possible. I'd like to release it soon.

http://BitcoinTalk.org/index.php?topic=199.msg1927#msg1927

Other than the version number change, which included changes in readme.txt and setup.nsi, I reduced the maximum number of outbound connections from 15 to 8 so nodes that accept inbound don't get too many connections. 15 was a lot more than needed. 8 is still plenty for redundancy.

BitcoinTalk

Re: Feature Request: Limiting Connections

2010-07-02 22:20:20 UTC - -

I reduced max outbound connections from 15 to 8 in RC4.

15 was way more than we needed for redundancy. 8 is still plenty of redundancy.

As the nodes upgrade to this version, this will cut in half the number of connections that inbound accepting nodes get.

If anyone wants more than 8 connections, they can open port 8333 on their firewall.

BitcoinTalk

Re: 0.3 almost ready -- please test the Mac version!

2010-07-04 21:52:28 UTC - -

Laszlo's build is going to be our first Mac release so please test it!

BitcoinTalk

Re: Slashdot Submission for 1.0

2010-07-05 21:31:14 UTC - -

BTW, I did come to my senses after that brief bout with 1.3, this release is still going to be 0.3 beta not 1.0.

I really appreciate the effort, but there are a lot of problems.

We don't want to lead with "anonymous". (I've been meaning to edit the homepage)

"The developers expect that this will result in a stable-with-respect-to-energy currency outside the reach of any government." -- I am definitely not making an such taunt or assertion.

It's not stable-with-respect-to-energy. There was a discussion on this. It's not tied to the cost of energy. NLS's estimate based on energy was a good estimated starting point, but market forces will increasingly dominate.

Sorry to be a wet blanket. Writing a description for this thing for general audiences is bloody hard. There's nothing to relate it to.

BitcoinTalk

Bitcoin 0.3 released!

2010-07-06 18:32:35 UTC - -

Announcing version 0.3 of Bitcoin, the P2P cryptocurrency! Bitcoin is a digital currency using cryptography and a distributed network to replace the need for a trusted central server. Escape the arbitrary inflation risk of centrally managed currencies! Bitcoin's total circulation is limited to 21 million coins. The coins are gradually released to the network's nodes based on the CPU power they contribute, so you can get a share of them by contributing your idle CPU time.

What's new:
- Command line and JSON-RPC control
- Includes a daemon version without GUI
- Transaction filter tabs
- 20% faster hashing
- Hashmeter performance display
- Mac OS X version (thanks to Laszlo)
- German, Dutch and Italian translations (thanks to DataWraith, Xunie and Joozero)

Get it at http://www.bitcoin.org or read the forum to find out more.

BitcoinTalk

Re: 0.3 almost ready -- please test the Mac version!

2010-07-06 19:43:18 UTC - -

0.3 released
http://BitcoinTalk.org/index.php?topic=238.msg2004#msg2004

BitcoinTalk

Re: 0.3 almost ready -- please test the Mac version!

2010-07-06 19:43:18 UTC - -

0.3 released
http://BitcoinTalk.org/index.php?topic=238.msg2004#msg2004

bitcoin-list

[**bitcoin-list**] Bitcoin 0.3 released!

2010-07-06 21:53:53 UTC - -

Announcing version 0.3 of Bitcoin, the P2P cryptocurrency! Bitcoin is a digital currency using cryptography and a distributed network to replace the need for a trusted central server. Escape the arbitrary inflation risk of centrally managed currencies! Bitcoin's total circulation is limited to 21 million coins. The coins are gradually released to the network's nodes based on the CPU power they contribute, so you can get a share of them by contributing your idle CPU time.

What's new:
- Command line and JSON-RPC control
- Includes a daemon version without GUI
- Transaction filter tabs
- 20% faster hashing
- Hashmeter performance display
- Mac OS X version (thanks to Laszlo)
- German, Dutch and Italian translations (thanks to DataWraith, Xunie and Joozero)

Get it at http://www.bitcoin.org, and read the forum to find out more.

BitcoinTalk

Re: On IRC bootstrapping

2010-07-07 01:31:07 UTC - -

Everybody needs to connect to the same IRC server and channel so they can find each other.

Quote from: Vasiliev on June 25, 2010, 11:50:15 PM
You may want to leave Freenode in as a fallback server -- if his server doesn't work, use Freenode's.

It might not be good if we suddenly rushed freenode with a ton of users all at once.

The fallback is our own seed system.

irc.lfnet.org is pretty old and has impressive uptime. I think it's going to be fine.

We could take IRC out at some point if we want, but I'd rather ease into it and just test our own seed system as a backup for now, and I really like the complementary redundant attributes of the two different systems.

BitcoinTalk

Re: bitcoin 0.3 win64 - broken access to APPDATA if non-latin characters in username

2010-07-08 18:24:19 UTC - -

Thanks for finding that. We switched from ANSI in 0.2 to UTF-8 in version 0.3, so it must be related to that.

Just to confirm, if you log in with the non-latin character username, not having an appdata/Bitcoin directory yet, and run Bitcoin and let it create the database from scratch, does it work or not?

BitcoinTalk

Re: Anonymity

2010-07-08 19:12:00 UTC - -

It's hard to imagine the Internet getting segmented airtight. It would have to be a country deliberately and totally cutting itself off from the rest of the world.

Any node with access to both sides would automatically flow the block chain over, such as someone getting around the blockade with a dial-up modem or sat-phone. It would only take one node to do it. Anyone who wants to keep doing business would be motivated.

If the network is segmented and then recombines, any transactions in the shorter fork that were not also in the longer fork are released into the transaction pool again and are eligible to get into future blocks. Their number of confirmations would start over.

If anyone took advantage of the segmentation to double-spend, such that there are different spends of the same money on each side, then the double-spends in the shorter fork lose out and go to 0/unconfirmed and stay that way.

It wouldn't be easy to take advantage of the segmentation to double-spend. If it's impossible to communicate from one side to the other, how are you going to put a spend on each side? If there is a way, then probably someone else is also using it to flow the block chain over.

You would usually know whether you're in the smaller segment. For example, if your country cuts itself off from the rest of the world, the rest of the world is the larger segment. If you're in the smaller segment, you should assume nothing is confirmed.

BitcoinTalk

Re: bitcoin 0.3 win64 - broken access to APPDATA if non-latin characters in username

2010-07-09 03:01:35 UTC - -

I think I see where the problem is. Coincidentally, I recently coded a replacement for the function in question which should fix it. It's not enabled yet, but in the SVN version it prints a debug message in debug.log showing the new directory value and old value for comparison.

BitcoinTalk

Re: BTC Vulnerability? (Massive Attack against BTC system. Is it really?)

2010-07-09 03:28:46 UTC - -

What the OP described is called "cornering the market". When someone tries to buy all the world's supply of a scarce asset, the more they buy the higher the price goes. At some point, it gets too expensive for them to buy any more. It's great for the people who owned it beforehand because they get to sell it to the corner at crazy high prices. As the price keeps going up and up, some people keep holding out for yet higher prices and refuse to sell.

The Hunt brothers famously bankrupted themselves trying to corner the silver market in 1979:
"Brothers Nelson Bunker Hunt and Herbert Hunt attempted to corner the world silver markets in the late 1970s and early 1980s, at one stage holding the rights to more than half of the world's deliverable silver.[1] During Hunt's accumulation of the precious metal silver prices rose from $11 an ounce in September 1979 to nearly $50 an ounce in January 1980.[2] Silver prices ultimately collapsed to below $11 an ounce two months later,[2] much of the fall on a single day now known as Silver Thursday, due to changes made to exchange rules regarding the purchase of commodities on margin.[3]"

http://en.wikipedia.org/wiki/Cornering_the_market

BitcoinTalk

Re: bitcoin 0.3 win64 - broken access to APPDATA if non-latin characters in username

2010-07-09 15:37:05 UTC - -

I tested this with a non-lower-ASCII account name on XP and confirmed the bug, then tested that the new GetDefaultDataDir fixed it. This change is revision 102 of the SVN.

BitcoinTalk

Re: Security

2010-07-10 12:58:02 UTC - -

I'll start thinking about how to do this.

At the moment, you can kind of use -connect. You can use -connect to make it connect to local computers on your LAN, like -connect=192.168.0.100. If you start it out blank and don't let it connect to the main network, the difficulty is still at the original low difficulty. If you've port-forwarded though, then outside nodes might still connect inward to you.

With -connect it still uses IRC, do you think it shouldn't get on IRC when you're telling it to only connect to specific nodes with -connect? The main scenario for -connect is where you have a server farm, with two connected to the network and the rest connected to the first two. In that case, you wouldn't want the -connect computers on IRC.

```
void ThreadIRCSeed(void* parg)
{
   if (mapArgs.count("-connect"))
      return;
```

BitcoinTalk

Re: Major Meltdown

2010-07-10 13:36:17 UTC - -

Quote from: llama on July 01, 2010, 10:21:47 PM

However, if something happened and the signatures were compromised (perhaps integer factorization is solved, quantum computers?), then even agreeing upon the last valid block would be worthless.

True, if it happened suddenly. If it happens gradually, we can still transition to something stronger. When you run the upgraded software for the first time, it would re-sign all your money with the new stronger signature algorithm. (by creating a transaction sending the money to yourself with the stronger sig)

BitcoinTalk

Re: No blocks downloaded... why?

2010-07-14 16:22:03 UTC - -

So that was responsible for keeping blocks from downloading?

The link: "Win32 CPU Cycles vs 'Live Protection' Engines"

For BitcoinFX, Live Protection was keeping it from getting CPU for generating coins. You said your friend was getting 1400-1600 khash/s, so it was getting CPU. I guess Live Protection must have been blocking some other part of the program then?

BitcoinTalk

Re: resource hog

2010-07-14 16:29:39 UTC - -

In Windows, you select the process in the task manager, right click, Set Priority. Set it to BelowNormal or Low. That shouldn't make a difference though.

If you turn off Generate Coins, does the CPU usage go flat? That would confirm that all the CPU time it's taking is generate, which is idle priority already.

It could be it's slow just because you have too many things running at once and you're out of memory. When you switch from one thing to another, it has to page it in from disk.

BitcoinTalk

Re: stopped prodicing coins

2010-07-14 17:04:02 UTC - -

Thanks for making that calculator.

The difficulty doubled a day or two ago, plus it's just random and you can have surprisingly long dry spells.

BitcoinTalk

Re: Building Bitcoin 0.3

2010-07-14 17:34:50 UTC - -

It doesn't work with wxWidgets 2.8, it needs wxWidgets 2.9. Unfortunately, there isn't a Debian package of wxWidgets 2.9 yet.

BitcoinTalk

Re: bitcoin auto-renice-ing

2010-07-14 17:38:31 UTC - -

Laszlo corrected this, but unfortunately it was too late to make it into 0.3.0. There will probably be a 0.3.1 soon though.

The problem is I used PRIO_MIN, I should have used PRIO_MAX for the lowest priority. The OS isn't supposed to let you increase priority, so the PRIO_MIN ought to leave it at priority 0.

BitcoinTalk

Re: Stuck on 513 blocks

2010-07-14 18:02:28 UTC - -

This is the second time I've seen this "Live Protection" problem reported.

It must be blocking the program's network communication. It sounds like it's allowing connections to be made, hence the 10 connections shown, but not allowing any data to be sent or received on them.

We need to understand this problem better.

Can someone write some instructions on the wiki explaining how to turn off or add an exclusion to Live Protection or whatever its full proper name is.

BitcoinTalk

Re: Error on Ubuntu 10.04

2010-07-14 18:25:41 UTC - -

What language is your computer set to? Is it set to German, Dutch or Italian? Is it one of those sub-languages like "nl-??"?

It's trying to load a translation and failing. You could delete the locale directory that came with bitcoin so it doesn't try to use it.

Can someone test each language on Ubuntu and see if there's a problem with just one of them or maybe all three?

BitcoinTalk

Re: Runaway CPU usage for 64bit BitCoin (Linux Client)

2010-07-14 18:45:53 UTC - -

After it initially tries incorrectly to set itself to the lowest priority, the generate thread only changes its priority again temporarily when it finds a block. When you've found a block, you should want it to hurry up and broadcast it as soon a possible before someone else finds one and makes yours invalid. The generate thread only changes to higher priority for less than a second every few days.

There should be a 0.3.1 release for this soon. There are a few other issues we need to look at fixing in 0.3.1 before making a release.

> ***Quote from: knightmb on July 12, 2010, 10:39:13 PM***
> *On a side note, I've tracked down the other GUI issue.*

The "minimize to tray instead of taskbar" is what was eating up all the CPU on my system. After I turned this off, the issue was resolved with Runaway CPU.

This only seems to affect the 64 bit Client, as the 32 bit Clients I have don't seem to be affected by this.

I did notice on the 64 bit Client, what happens is, it spawns multiple "tray" icons until X server finally kills over, so I guess I should submit that as a bug to somewhere? 

That's interesting. I know the minimize to tray on Ubuntu is very clunky, but I didn't know it had a CPU peg problem too. Anyone else able to reproduce this problem? We had this feature disabled on Linux before, but then it seemed better to have the imperfect UI than to lose the feature entirely. I'm thinking we should disable it again on Linux.

BitcoinTalk

Re: Warning this block was not received by any other nodes

2010-07-14 18:56:29 UTC - -

Microsoft Security Essentials Live Protection is blocking your communication with the network. You have connections, which tricks Bitcoin into thinking it's connected, but they are silent because the data is being blocked.

You need to make bitcoin.exe an excluded process in Live Protection.

This is becoming a common problem. Someone should write this up in a pegged thread.

The message "Warning: This block was not received by any other nodes" occurs when Bitcoin broadcasts a block, but nobody confirms they received it. The warning is there just for this kind of situation, where for some reason you have connections, but they have gone dead and nobody can hear you. Your block will never become valid because nobody received it.

BitcoinTalk

Re: Hash/sec Throttling for Democracy

2010-07-14 20:25:06 UTC - -

Quote from: knightmb on July 14, 2010, 07:17:43 PM
So if your computer was only 1% towards solving block 68000

This is a common point of confusion. There's no such thing as being 1% towards solving a block. You don't make progress towards solving it. After working on it for 24 hours, your chances of solving it are equal to what your chances were at the start or at any moment.

It's like trying to flip 37 coins at once and have them all come up heads. Each time you try, your chances of success are the same.

The RNG is the OpenSSL secure random number generator. On Windows it's seeded with the complete set of all hardware performance counters since your computer started, on Linux it's dev/random.

BitcoinTalk

Re: Scalability

2010-07-14 21:10:52 UTC - -

The design outlines a lightweight client that does not need the full block chain. In the design PDF it's called Simplified Payment Verification. The lightweight client can send and receive transactions, it just can't generate blocks. It does not need to trust a node to verify payments, it can still verify them itself.

The lightweight client is not implemented yet, but the plan is to implement it when it's needed. For now, everyone just runs a full network node.

I anticipate there will never be more than 100K nodes, probably less. It will reach an equilibrium where it's not worth it for more nodes to join in. The rest will be lightweight clients, which could be millions.

At equilibrium size, many nodes will be server farms with one or two network nodes that feed the rest of the farm over a LAN.

BitcoinTalk

Re: Runaway CPU usage for 64bit BitCoin (Linux Client)

2010-07-15 00:18:23 UTC - -

OK, the undocumented switch "-minimizetotray" which re-enables the option.

I uploaded the change to SVN.

BitcoinTalk

Re: [Bitcoin 0.3.0] Runtime error

2010-07-15 14:05:20 UTC - -

More directly, this:
http://BitcoinTalk.org/index.php?topic=246.0

I will be posting release candidate of 0.3.1 with this fix shortly. Please try that and let me know if it fixes the problem.

BitcoinTalk

Re: Static Linux x86_64 bins for those having libcrypto troubles

2010-07-15 14:33:04 UTC - -

We don't even specify linking glibcxx_3.4.11, so gcc must automatically link it behind the scenes. There's probably a compiler switch that would tell it to static link

it. I'm not sure what the licensing issues would be. Typically, compiler stuff is fully redistributable.

BitcoinTalk

Re: resource hog

2010-07-15 14:59:00 UTC - -

Then all the CPU time is the generate thread, which definitely runs at the lowest possible priority, idle priority. It's normal that your CPU meter is 100%. Since it's idle priority, it won't actually slow anything else down, even though the CPU meter is 100%.

BitcoinTalk

Bitcoin 0.3.1 released

2010-07-15 17:05:54 UTC - -

This is a bugfix maintenance release. It is now uploaded to SourceForge. Mac OS X didn't need any fixes so we don't really need to update it, 0.3.0 is still good.

The download links are on bitcoin.org

Changes:
- Added Portuguese translation by Tiago Faria
Windows
- Fix for 22DbRunRecoveryException if your username has non-ascii characters in it
Linux
- Laszlo's fix for lowering generate thread to lowest priority
- Fix for if you're having trouble with libcrypto linkage
- Gavin Andresen's implementation of "start on windowing system startup" option

BitcoinTalk

Re: 0.3.1 release candidate, please test

2010-07-15 17:23:48 UTC - -

Well, it can't hurt to do a backup and it's a good idea to backup regularly, but no, a backup is not required before installing this.

BitcoinTalk

Re: 0.3.1 release candidate, please test

2010-07-15 17:56:43 UTC - -

I don't think you have a particular problem, I think your system is laggy because you're running a lot of things at once and hitting the pagefile because memory is full. You confirmed when you shut off generation that your CPU drops to 0%, so the CPU usage is definitely all idle priority. There's nothing in the 0.3.1 that would affect these things.

BitcoinTalk

Re: Website and software translations

2010-07-15 18:30:22 UTC - -

Quote from: aidos on July 15, 2010, 12:49:11 AM
Ok here is the .po file for French. While I'm at it, I noted a couple of issues:

1. The "About" box didn't take the translation into account, it still displays the english version to me, even though the rest of the software is using the translated strings, and the .po file contains the translation string of the "About" box message. Same problem with the "Apply" button in the Settings window.

I need to give an updated .po file.

Quote from: aidos on July 15, 2010, 12:49:11 AM
2. If an transaction's description in the list of transaction in the main window contains a diacritical character (such as "\u00e9\u00e0\u00e8\u00e7"), it's not displayed. I suppose the string is not being properly handled as UTF8 somewhere.

OK, this must be a problem somewhere, I'll have to take a look at it or one of the other devs can.

Quote from: aidos on July 15, 2010, 12:49:11 AM
4. About the .po file :
- There are a few strings in the .po file that don't needs translation (ie: "Bitcoin"). Maybe those shouldn't be inside ("...") ?
- Others shouldn't be split. I can remember one message about transaction fee where the string is split in two to insert the fee value, where you could simply have put a %s. It makes the message harder to translate as I had to go in the source to find exactly what was going on.
- Some strings have whitespace at the end or start, which necessity is very debatable, and it's easy to miss in PoEdit.

Many of the strings are in code automatically generated from uiproject.fbp where nothing can be done about these things. I have a program I use to find all the spacing

inconsistencies at the beginning and ending of strings in your .po file and manually fix them up before I upload them to SVN.

BitcoinTalk

Re: Website and software translations

2010-07-15 18:37:13 UTC - -

I uploaded an updated bitcoin.po for 0.3.1 attached to this message:
http://BitcoinTalk.org/index.php?topic=151.msg1259#msg1259

please use it if you're starting a new translation.

If you already have a po file, poedit can update it.
- Get the src directory from the 0.3.1 release candidate posted in the development forum, any version will do:
http://BitcoinTalk.org/index.php?topic=383.0
- Make a subdirectory under src: locale/??/LC_MESSAGES
(?? could be anything really, "en" or your language 2-letter code)
- Put your .po file there
- Open it with poedit
- In poedit, Catalog->Update from sources

The key is that the src directory with the sourcefiles needs to be 3 directories up from the .po file.

BitcoinTalk

Re: Website and software translations

2010-07-15 18:43:54 UTC - -

Quote from: SmokeTooMuch on July 13, 2010, 06:55:55 PM
I recommend to remove the download links at the bottom of the main page.
As you can see the links on the English page points to the new 0.3 release, but the other languages only contain links for the old 0.2 version.
There's a download box with the current releases on the right anyway, so why not remove the links from the translated pages.
I updated them to 0.3.0.

I am tempted to remove the download links from the other languages and only keep it on English.

They will need to be updated for 0.3.1 soon. Perhaps there's a way for someone to manage the updating of the translated drupal pages.

BitcoinTalk

Re: Website and software translations

2010-07-15 19:12:14 UTC - -

Thanks for the Spanish and French translations! The edited and updated .po files are attached.

I uploaded these to the SVN.

BitcoinTalk

Re: 0.3.1 release candidate, please test

2010-07-15 21:40:34 UTC - -

Quote from: knightmb on July 15, 2010, 07:37:10 PM
On Windows, the priority of the Coin Generation is still net for normal. If you run BitCoin in Generate Coin mode, then load up something to eat up all the CPU (like CPU hog for example: http://www.microtask.ca/cpuhog.html) you'll see that both BitCoin and CPU hog share the CPU 50/50 instead of CPU Hog taking all the CPU and BitCoin running only on idle/low process. The khash/s is also reduced in half, so further evidence that the threads are not running in a lower than normal prioirty.

I was not able to reproduce this. I have dual-proc, so I ran two memory hogs. Bitcoin got 0% of CPU according to the task manager. The khash/sec meter stayed stuck because it couldn't get any CPU to update it.

Do you have dual-proc? Are you sure you weren't running a single processor hog?

BitcoinTalk

Re: 0.3.1 release candidate, please test

2010-07-15 22:07:35 UTC - -

Quote from: knightmb on July 15, 2010, 08:15:46 PM
On the Linux client (64 bit), the "minimize on close" will still minimize to tray (causing X server hang after a short while by spawning multiple tray icons).

I updated the first post with a link to rc2 for linux with the fix for this. Please check that this is fixed for you. Thanks!

http://www.bitcoin.org/download/bitcoin-0.3.1.rc2-linux.tar.gz

BitcoinTalk

Re: 0.3.1 release candidate, please test

2010-07-15 22:10:19 UTC - -

Quote from: db on July 15, 2010, 08:39:08 PM
The listreceivedbyaddress and getreceivedbyaddress commands are duplicated in bincoind help. (Same in 0.3.0.)

Yes a bug. It'll have to be fixed in the next version.

BitcoinTalk

Re: "SetIcons(): icon bundle doesn't contain any suitable icon"

2010-07-16 02:43:29 UTC - -

That must be it then.

It must be looking for a larger icon like 20x20 but we don't have one.

BitcoinTalk

Re: Proof-of-work difficulty increasing

2010-07-16 14:46:12 UTC - -

The proof-of-work difficulty is currently 45.38. (see http://www.alloscomp.com/bitcoin/calculator.php)

It's about to increase again in a few hours. It's only been 3-4 days since the last increase, so I expect it will increase by the max of 4 times, or very nearly the max. That would put it at 181.54.

The target time between adjustments is 14 days, 14/3.5 days = 4.0 times increase.

BitcoinTalk

Re: Assertion Failure - Ubuntu Lucid

2010-07-16 14:52:04 UTC - -

That's the first time I've seen this error.

How many blocks do you have? (in the status bar)

You should move your blk*.dat files (in ~/.bitcoin) to another directory and let it start over downloading the block chain again. If you don't mind, could you keep the old blk*.dat files for a little while in case I need to look at them?

BitcoinTalk

Re: Fedora 13 libcrypto

2010-07-16 14:55:23 UTC - -

Please try the 0.3.1 release candidate, it should at least resolve the libcrypto dependency:

http://BitcoinTalk.org/index.php?topic=383.0

Let me know if that works.

BitcoinTalk

Re: Resending transaction

2010-07-16 15:01:33 UTC - -

Bitcoin automatically rebroadcasts your transactions if it receives new blocks that don't contain them. It may take about an hour to get rebroadcasted. It is relentless though. It will keep nagging the network forever until your transaction gets into a block.

BitcoinTalk

Re: 0.3.1 release candidate, please test

2010-07-16 15:09:59 UTC - -

Because of all the dependencies that different systems don't have. It's easier to just static link what we can. It doesn't increase the size by very much.

BitcoinTalk

Re: Source code documentation

2010-07-16 15:37:00 UTC - -

I like that in libraries for the external API's, but you can probably tell from the code that I'm not a fan of it for interior functions. Big obligatory comment headers for each function space out the code and make you hesitate about creating a small little function where the comment header would be bigger than the function. They're some trouble for maintenance, as changes to the function then require duplicate changes in the comment header. I like to keep code compact so you can see more code on the screen at once.

To add them now at this point, what would be written would just be what's obvious from looking at the function.

The external API we have, in rpc.cpp, the usage documentation is in the help string.

Sorry to be a wet blanket.

BitcoinTalk

Re: Hash() function not secure

2010-07-16 16:13:53 UTC - -

SHA256 is not like the step from 128 bit to 160 bit.

To use an analogy, it's more like the step from 32-bit to 64-bit address space. We quickly ran out of address space with 16-bit computers, we ran out of address space with 32-bit computers at 4GB, that doesn't mean we're going to run out again with 64-bit anytime soon.

SHA256 is not going to be broken by Moore's law computational improvements in our lifetimes. If it's going to get broken, it'll be by some breakthrough cracking method. An attack that could so thoroughly vanquish SHA256 to bring it within computationally tractable range has a good chance of clobbering SHA512 too.

If we see a weakness in SHA256 coming gradually, we can transition to a new hash function after a certain block number. Everyone would have to upgrade their software by that block number. The new software would keep a new hash of all the old blocks to make sure they're not replaced with another block with the same old hash.

BitcoinTalk

Re: Request: expected bitcoins per day display

2010-07-16 16:47:14 UTC - -

Many businesses are like that. For a car salesman, when will the next customer walk in the door?

On the OP's question, it's a good feature, but the question is, how would we word it so people don't expect to get something after that specific amount of time? "it said 7 days and I waited more than a week and didn't get anything!" Approx, average, but still they're going to think that way. It can't be a whole sentence, unless we think of somewhere else to put it, but where would that be? Suggestions?

The difficulty quadrupled a few minutes ago to 181.54. It's going to take typically about a week to generate now.

BitcoinTalk

Re: Proof-of-work difficulty increasing

2010-07-16 16:56:54 UTC - -

It adjusted to 181.54 a few minutes ago. Typical time to get a block is about a week now.

The difficulty can adjust down as well as up.

The network should be generating close to 6 blocks per hour now.

BitcoinTalk

Re: Source code documentation

2010-07-16 17:15:47 UTC - -

It's in init.cpp.

It's a wxWidgets app, so it doesn't have a main() function. It may in a little while, since I'm pretty close to making bitcoind build w/o wxBase. (it'll be in init.cpp)

Sorry about my choice of the filename "main.cpp", another possible name would have been "core.cpp". It's much too late to change. I still prefer main.cpp.

We're still in great need of sample code showing the recommended way to use the JSON-RPC functions, like for a basic account system on a typical storefront website. Using getreceivedbylabel using the username as the label, changing to a new bitcoin address

once the stored one for that account gets used. I posted a sample code fragment on the forum somewhere. (search on getreceivedbylabel or getnewaddress) The sample code could be a plain vanilla bank site where you can deposit and send payments.

BitcoinTalk

Re: 0.3.1 release candidate, please test

2010-07-16 17:26:17 UTC - -

Good point. If you're going to have more than 8 LAN nodes connect to one gateway node, then you'd better have the gateway node set up so it can receive incoming connections. Otherwise, while the gateway node has 8 or more connections, it will not try to add any more outbound connections. As the outside nodes you're connected to come and go, it doesn't make new outbound connections to replace them. You'll be fine if you can accept incoming connections, then there will be plenty of others connecting to you.

BitcoinTalk

Re: Proof-of-work difficulty increasing

2010-07-16 17:29:28 UTC - -

Yes, about 20 hours. (120 conf / 6 blocks per hour = 20 hours) That's the normal length of time before you can spend it. You know long before that that you won one.

BitcoinTalk

Re: bitcoin trademark?

2010-07-16 17:47:05 UTC - -

No, not related at all.

BitcoinTalk

Re: The dollar cost of bitmining energy

2010-07-16 17:58:44 UTC - -

Neat chart.

Difficulty just increased by 4 times, so now your cost is US$0.02/BTC.

BitcoinTalk

Re: Website integration for bitcoin

2010-07-16 18:23:04 UTC - -

I've been trying to encourage someone to write and release some sample Python code showing the recommended way to do the typical accounting stuff, but to no avail. It would be nice if you didn't have to re-invent the wheel like you're doing here. Search on getnewaddress and you should find a thread where I gave a small fragment of sample pseudocode.

BitcoinTalk

Re: Proof-of-work difficulty increasing

2010-07-16 18:43:51 UTC - -

Right, the difficulty adjustment is trying to keep it so the network as a whole generates an average of 6 blocks per hour. The time for your block to mature will always be around 20 hours.

The recent adjustment put us back to close to 6 blocks per hour again.

There's a site where you can see the time between blocks, and since block 68545, it's been more like 10 minutes per block:
http://nullvoid.org/bitcoin/statistix.php

BitcoinTalk

Sample account system using JSON-RPC needed

2010-07-16 19:45:10 UTC - -

We need someone to write sample code, preferably Python or Java, showing the recommended way to use the JSON-RPC interface to create an account system. Most sites that sell things will need something like this. Someone who's kept up on the JSON-RPC threads here should have some idea how it should work.

When a user is logged in to their account, you show the bitcoin address they can send to to add funds. Before showing it, you check if it's been used, if it has then you replace it with a new one (getnewaddress <username>). You only need to keep the latest bitcoin address for the account in your database. (I posted a sample code fragment for this in an earlier thread somewhere, search on getnewaddress)

You use getreceivedbylabel <username> with the username as the label to get the "credit" amount of the account. You need to keep a "debit" amount in your database. The current balance of the account is (credit - debit). When the user spends money, you increase debit.

If you're requiring more than 0 confirmations, it's nice if you show the current balance (0 confirmations) and the available balance (1 or more confirmations), so they can immediately see that their payment is acknowledged. Not all sites need to wait for confirmations, so the dual current & available should be optional. Most sites selling digital goods are fine to accept 0 confirmations.

A nice sample app for this would be a simple bank site, which would have the above, plus the option to send a payment to a bitcoin address. The sample code should be the simplest possible with the minimum extra stuff to make it a working site.

vekja.net is an example of a site like this.

BitcoinTalk

Re: Bitcoin 0.3.1 released

2010-07-16 21:06:57 UTC - -

I uploaded windows 0.3.1 rc1 and linux 0.3.1 rc2 to SourceForge and updated the links on the homepage.

You don't need to update to 0.3.1 unless you had one of the problems listed in the first post. If you've got it working already, stay with 0.3.0.

BitcoinTalk

Re: A New Currency System for the World

2010-07-16 22:20:09 UTC - -

Quote from: hugolp on May 08, 2010, 10:38:51 AM

When I run bitcoin it becomes very sluggish, almost unusable. When I stop bitcoin everything goes ok again. Its running Ubuntu desktop 10.04 amd64 using ia32libs and the binary in bitcoin 0.20 tarball.

0.3.1 fixes that, sets the generate threads to the lowest priority. Download links are on the homepage now.

BitcoinTalk

Re: BUG Report: Rounding glitch

2010-07-17 16:06:12 UTC - -

It must be a rounding error when getinfo converts to floating point to return the JSON-RPC result. The only place where it uses floating point to represent money is returning a value in JSON-RPC.

1.139999999999 is longer than bitcoin can internally represent.

internally, it could only be:
1.13999999 or
1.14000000

1.139999999999 is much much closer to 1.14000000 than 1.13999999, so it must be 1.14000000.

The code is this:
(double)GetBalance() / (double)COIN.

(I can't think of an easy way to fix it at the moment)

BitcoinTalk

Re: Privacy versus Safety: handling change

2010-07-17 16:27:39 UTC - -

We should queue up a supply of pre-made addresses in the wallet to use when a new address is needed. They aren't very big, so it wouldn't hurt to have a lot of them. This would more generally cover the case also where someone backs up, then requests a new address and receives a big payment with it. Maybe there should be separate queues so one type of demand on addresses doesn't deplete it for the others.

The addresses would be created and stored in the normal place, but also listed on a separate list of created-but-never-used addresses. When an address is requested, the address at the front of the never-used queue is handed out, and a new address is created and added to the back.

There's some kind of rescan in the block loading code that was made to repair the case where someone copied their wallet.dat. I would need to check that the rescan handles the case of rediscovering received payments in blocks that were already received, but are forgotten because the wallet was restored.

BitcoinTalk

Re: Nenolod, the guy that wants to prove Bitcoin doesn't work.

2010-07-17 16:56:06 UTC - -

0.3.2 has some security safeguards to lock in the block chain up to this point and limit the damage a little if someone gets 50%.

But if someone has 50%+ of the CPU power and malicious intent, they can prove what it already says in the design document.

BitcoinTalk

Bitcoin 0.3.2 released

2010-07-17 21:35:51 UTC - -

Download links available now on bitcoin.org. Everyone should upgrade to this version.

- Added a simple security safeguard that locks-in the block chain up to this point.
- Reduced addr messages to save bandwidth now that there are plenty of nodes to connect to.
- Spanish translation by milkiway.
- French translation by aidos.

The security safeguard makes it so even if someone does have more than 50% of the network's CPU power, they can't try to go back and redo the block chain before yesterday. (if you have this update)

I'll probably put a checkpoint in each version from now on. Once the software has settled what the widely accepted block chain is, there's no point in leaving open the unwanted non-zero possibility of revision months later.

BitcoinTalk

Re: Bitcoin snack machine (fast transaction problem)

2010-07-17 22:29:13 UTC - -

I believe it'll be possible for a payment processing company to provide as a service the rapid distribution of transactions with good-enough checking in something like 10 seconds or less.

The network nodes only accept the first version of a transaction they receive to

incorporate into the block they're trying to generate. When you broadcast a transaction, if someone else broadcasts a double-spend at the same time, it's a race to propagate to the most nodes first. If one has a slight head start, it'll geometrically spread through the network faster and get most of the nodes.

A rough back-of-the-envelope example:
1 0
4 1
16 4
64 16
80% 20%

So if a double-spend has to wait even a second, it has a huge disadvantage.

The payment processor has connections with many nodes. When it gets a transaction, it blasts it out, and at the same time monitors the network for double-spends. If it receives a double-spend on any of its many listening nodes, then it alerts that the transaction is bad. A double-spent transaction wouldn't get very far without one of the listeners hearing it. The double-spender would have to wait until the listening phase is over, but by then, the payment processor's broadcast has reached most nodes, or is so far ahead in propagating that the double-spender has no hope of grabbing a significant percentage of the remaining nodes.

BitcoinTalk

Re: Assertion Failure - Ubuntu Lucid

2010-07-17 22:37:06 UTC - -

Quote from: singpolyma on July 17, 2010, 10:19:48 PM
My coins disappeared, but I assume they'll come back when it's up to current?

Right, they'll re-appear when it's finished downloading all the blocks.

BitcoinTalk

Re: Bitcoin 0.3.2 released

2010-07-17 22:54:24 UTC - -

Quote from: llama on July 17, 2010, 09:56:25 PM
However, it's important that you don't lock all the way up the very latest block. Otherwise, the attacker could generate a fake block (or a few) right before you happen to lock it, and then his attack would be far easier than it would have been without the block lock.

I went about 200 blocks back. The block chain was a clean straight line without branches, and there was only one known version of the locked block.

Quote from: llama on July 17, 2010, 09:56:25 PM
Also, I'm assuming that the block lock means that the blocks will also come prepackaged with the client. Is this so?

Sorry, not yet, but I do want to make the initial block download faster.

BitcoinTalk

Re: Source code documentation

2010-07-17 23:18:30 UTC - -

I didn't realize you were going to document all the intentionally undocumented commands. They're unsupported and not intended to be used by users.

All the user-facing commands are listed in the -? help.

BitcoinTalk

Re: Network Size

2010-07-17 23:25:16 UTC - -

Quote from: NewLibertyStandard on July 17, 2010, 10:22:09 PM
Version 0.3 was supposed to reduce the number of outgoing connections on non-port forwarded clients from 15 to 8, but I don't think it really happened. I'm not positive if this is the case. Correct me if I'm wrong.

In 0.3.0, the change to 8 only ended up in the Windows version, the other versions still had 15.

Please upgrade to 0.3.2, it's available now.

BitcoinTalk

Re: Bitcoin snack machine (fast transaction problem)

2010-07-18 01:59:15 UTC - -

Quote from: llama on July 18, 2010, 12:03:29 AM
This is a good start, but still not impermeable.

I didn't say impermeable, I said good-enough. The loss in practice would be far lower than with credit cards.

Quote
(for example, by refusing to propogate word of the transaction at the vending machine)

No, the vending machine talks to a big service provider (aka payment processor) that provides this service to many merchants. Think something like a credit card processor with a new job. They would have many well connected network nodes.

BitcoinTalk

Re: Source code documentation

2010-07-18 15:12:54 UTC - -

They're only intended for intrepid programmers who read the sourcecode.

BitcoinTalk

Re: URI-scheme for bitcoin

2010-07-18 16:06:16 UTC - -

Quote from: lachesis on June 16, 2010, 06:14:05 AM
I think you're misunderstanding the issue. My browser will always be able to go to 127.0.0.1 (barring some strange IE settings or a virus). If I type the address into the URL bar or click a link, it will work fine. However, it isn't possible to use Javascript to complete POST requests between domains (or ports on the same domain).

That's what I thought too.

Quote from: sirius-m on June 16, 2010, 08:26:14 AM
Yeah, I meant to say that cross-domain javascript calls are forbidden, so you can't call 127.0.0.1 from a javascript that doesn't reside in 127.0.0.1. Come to think of it, it would be quite funny if browsers allowed malicious cross-domain javascript to change people's Facebook pages etc.

Now I'm hearing a report that it IS possible for javascript to do a cross-domain POST request to 127.0.0.1. Not other domains, but just specifically to that one. Great...

If this is the case, then do not use the -server switch or bitcoind on a system where you do web browsing.

I'll get started on adding the password field.

BitcoinTalk

Re: Bitcoin 0.3.2 released

2010-07-18 18:58:21 UTC - -

The change list is basically encompassed by what's listed in the first message. Everyone should upgrade to get the important security improvements.

Minimizing to tray had at least 3 different glitches and bugs on Linux, including a crash one, so I disabled it again. You can still re-enable the option with "-minimizetotray" if you want to use it anyway. The bugs/glitches are somewhere in wxWidgets or GTK or Gnome and I don't know how to fix them. Sorry, I just don't know what else to do, it's just too glitchy and buggy to have as a mainline feature.

BitcoinTalk

JSON-RPC password

2010-07-18 20:49:22 UTC - -

I uploaded to SVN my changes to add a password to JSON-RPC. If you're set up to build, please test it.

The -server switch is replaced with -rpcpw=<password>, which is also used with bitcoind.
bitcoin -rpcpw=<password> -- runs with JSON-RPC port open
bitcoind -rpcpw=<password> -- daemon with password

If you have a better idea for the switch name, let me know, but keep in mind there will eventually be a password for encrypting the database too. I'm not sure but I think they may want to use different passwords for the two.

It gives a warning if you don't set a password.

All commands now require the password as the first parameter. It'll tell you that if you run "bitcoind help".

The central code:

```
  // Check password
  if (params.size() < 1 || params[0].type() != str_type)
      throw runtime_error("First parameter must be the password.");
  if (params[0].get_str() != strRPCPassword)
  {
```

```
    if (strRPCPassword.size() < 15)
        Sleep(50);
    begin = strRequest.end();
    printf("ThreadRPCServer incorrect password attempt ");
    throw runtime_error("Incorrect password.");
}
```

Any comments on these decisions?

1) if (strRPCPassword.size() < 15) Sleep(50); -- this means if it's a short password, it'll wait 50ms after each attempt. This might be used as a DoS attack, but I figured if it's a short password, it's more important to protect against brute force password scan. This may tell outsiders whether the password is less than 15 characters, but less than 15 isn't all that noteworthy, most passwords are less than 15. If you want to close the DoS possibility, just use a password 15 characters or longer.

2) begin = strRequest.end(); -- if it's a single request with multiple invocations, I throw away the rest if one has a bad password. This is so you can't stuff it with millions of password attempts in one packet. What do you think, is this the right thing to do? (multiple invocation is probably almost never used anyway)

I also fixed the two duplicated commands listed in the help:

```
getaddressesbylabel <pw> <label>
getbalance <pw>
getblockcount <pw>
getblocknumber <pw>
getconnectioncount <pw>
getdifficulty <pw>
getgenerate <pw>
getinfo <pw>
getlabel <pw> <bitcoinaddress>
getnewaddress <pw> [label]
getreceivedbyaddress <pw> <bitcoinaddress> [minconf=1]
getreceivedbylabel <pw> <label> [minconf=1]
help <pw>
listreceivedbyaddress <pw> [minconf=1] [includeempty=false]
listreceivedbylabel <pw> [minconf=1] [includeempty=false]
sendtoaddress <pw> <bitcoinaddress> <amount> [comment] [comment-to]
setgenerate <pw> <generate> [genproclimit]
setlabel <pw> <bitcoinaddress> <label>
stop <pw>
```

Re: MSVC build & SHA-256

2010-07-18 21:24:09 UTC - -

OpenSSL doesn't have any interface for doing just the low level raw block hash part of SHA256. SHA256 begins by wrapping your data in a specially formatted buffer. Setting up the buffer takes an order of magnitude longer than the actual hashing if you're only hashing one or two blocks like we do. It's intended that the time is amortised if you were hashing many KB or MB of data. In BitcoinMiner, we format the buffer once and keep reusing it.

If you can find SHA256 code that's faster (with MinGW/GCC) than what we've got, that would be really great! (although, keep licensing in mind) The one we have is the only one I tried, so there's significant chance for improvement.

When I wrote it more than 2 years ago, there were screaming hot SHA1 implementations but minimal attention to SHA256. That's a lot of time for them to come up with better stuff. SHA256 was a lot slower than the fastest SHA1 at the time than I thought it should be. Obviously SHA256 should be slower than SHA1 by a certain amount, but not by as much as I saw.

(hope you don't mind I renamed your thread, SHA-256 optimisation is something important that I keep forgetting about)

BitcoinTalk

Re: Nenolod, the guy that wants to prove Bitcoin doesn't work.

2010-07-18 21:56:18 UTC - -

Typically, over 25,000 BTC.

BitcoinTalk

Re: Did block generation crawl to a halt?

2010-07-18 23:35:27 UTC - -

Nice graph! A moving average to smooth it out would be nice.

http://nullvoid.org/bitcoin/statistix.php says 212 blocks in the last 24 hours, or 8.8 per hour.

BitcoinTalk

Re: JSON-RPC password

2010-07-19 04:43:13 UTC - -

Right, that is quite a bit better.

Can you give me any examples of other stuff that does it that way? (and what the command line looks like)

The main change you're talking about here is instead of -rpcpw= when you start bitcoind, you'd use a switch that specifies a text file to go and read it from, right? (any ideas what I should name the switch?)

BitcoinTalk

Warning: don't use -server or bitcoind where you web browse (v0.3.2 and lower)

2010-07-19 16:01:38 UTC - -

Don't use the -server or -daemon switch or run bitcoind on a machine where you use a web browser. It opens port 8332 on 127.0.0.1, the local loopback address, and you wouldn't think that web browsers could cross-site access it, but it is possible.

We're working on a release soon that puts a password on the JSON-RPC interface, but until then, avoid using the -server switch, and don't web browse on the same machine where bitcoind is running.

Update:
The JSON-RPC HTTP authentication feature in 0.3.3 solves this problem.

BitcoinTalk

Re: JSON-RPC password

2010-07-19 16:20:50 UTC - -

So you drop a settings file in the ~/.bitcoin directory, that sounds better. In the "no password is set" warning, it could tell you where the file is and what to do.

What is the most popular and common settings file format?

HTTP basic authentication should be considered. In actual practice though, it's more work for web developers to figure out how to specify the password through some extra parameter in the HTTP or JSON-RPC wrapper than to just stick an extra parameter at the beginning of the parameter list. What do you think? Does HTTP basic authentication get us any additional benefits? Moving it off the parameter list but then you still have to

specific it in a more esoteric place I'm not sure is a net win.

> ***Quote from: gavinandresen on July 19, 2010, 12:02:39 PM***
> *I was confused for a bit because the password is given LAST on the command line, but FIRST in the JSON-RPC params list. I agree that reading the command-line password from a file would be more convenient and more secure.*

You're also confusing me, what do you mean? Did I do something unintended?

BitcoinTalk

Re: They want to delete the Wikipedia article

2010-07-20 18:38:28 UTC - -

Bitcoin is an implementation of Wei Dai's b-money proposal http://weidai.com/bmoney.txt on Cypherpunks http://en.wikipedia.org/wiki/Cypherpunks in 1998 and Nick Szabo's Bitgold proposal http://unenumerated.blogspot.com/2005/12/bit-gold.html

The timing is strange, just as we are getting a rapid increase in 3rd party coverage after getting slashdotted. I hope there's not a big hurry to wrap the discussion and decide. How long does Wikipedia typically leave a question like that open for comment?

It would help to condense the article and make it less promotional sounding as soon as possible. Just letting people know what it is, where it fits into the electronic money space, not trying to convince them that it's good. They probably want something that just generally identifies what it is, not tries to explain all about how it works.

If you post in http://en.wikipedia.org/wiki/Wikipedia:Articles_for_deletion/Bitcoinplease don't say "yeah, but bitcoin is really important and special so the rules shouldn't apply" or argue that the rule is dumb or unfair. That only makes it worse. Try to address how the rule is satisfied.

Search "bitcoin" on google and see if you can find more big references in addition to the infoworld and slashdot ones. There may be very recent stuff being written by reporters who heard about it from the slashdot article.

I hope it doesn't get deleted. If it does, it'll be hard to overcome the presumption. Institutional momentum is to stick with the last decision. (edit: or at least I assume so, that's how the world usually works, but maybe Wiki is different)

BitcoinTalk

Re: JSON-RPC password

2010-07-21 00:05:20 UTC - -

Still need to know what's the most typical settings file format on Linux. Is there a standard file extension? I've never seen a settings file using JSON, and it doesn't look very human friendly with everything required to be in quotes. I think what I usually see is like:
# comment
setting=value

Is there a settings file thing in Boost?

When you're using bitcoind to issue commands from the command line as a client, can we have it get the password from the settings file then too?

Gavin pointed out I forgot to increment the column of numbers in CommandLineRPC, so the current -rpcpw= implementation doesn't work right from the command line with non-string parameters. (JSON-RPC is fine) Still under construction.

BitcoinTalk

Re: JSON-RPC password

2010-07-21 05:51:34 UTC - -

I was researching config file formats, here's a comparison.

YAML is massive. I'm not sure there's a lightweight easy to build library we can integrate into our project. Seems overkill.

JSON is tempting and I'm inclined to like it, but two main sticking points:
1) No comments! How can you have a config file where you can't comment out a line to disable it?
2) Not very user friendly to have to "quote" all the strings, including the keys, and also have to remember the comma at the end of lines.

```
{
   "key" : "value",
}
```

I suppose we could easily preprocess JSON reading the config file one line at a time, truncate the lines at any # character (and/or "//"?), concatenate them into a string and pass it to JSON, so you could go:

```
# comment
"key" : "value",   # still have to remember the comma
"key2" : "value",   // comment like this or both
```

Boost has boost::program_options.

We could read lines ourselves and feed them into a map<string, string> mapConfig.

```
while (!eof)
  read line
  if '#' found, truncate line
  split line at first ':' -> key, value
  mapConfig.insert(key, value)
```

If we use the syntax:
comment
key : value

...and don't allow whitespace indenting before the keys, I guess we would be a subset of YAML and could switch to YAML someday if we need more complexity.

If we go with self parsed, that doesn't mean we can't use JSON on particular parameter values as needed. If an option needs a list or more structured data, it could always parse its value as json:
key : ["item1", "item2", "item3"]

Although it has to be all on one line then.

I guess I'm leaning towards self parsed mapConfig:
comment
key : value

BitcoinTalk

Re: JSON-RPC password

2010-07-21 16:07:57 UTC - -

Quote from: gavinandresen on July 21, 2010, 12:11:10 PM

I just did a quick survey of 20 .conf files in /etc on my debian system, and found:
1 file used "key value"
5 used "key=value"

Thanks for that survey!

I find "key value" a little unnatural. There ought to be a more definite separator between key and value that suggests assignment. The space people may just be getting lazy using their language's split function.
key=some full sentence with spaces in it. # seems more clear
key some full sentence with spaces in it. # than this

Allright then, lets go with self-parsed mapConfig, syntax:
# comment
key=value

file extension .conf. What's the filename, is it ~/.bitcoin/settings.conf or ~/.bitcoin/bitcoin.conf or what?

I think we better strip whitespace at the beginning and end of the key and the value.
# user who likes column formatted
k = value
key = value
longerkey = this sentence would be this # "this sentence would be this"
key = value # guess this is ok too
nextkey = value
right = justified

The normal syntax should be "key=value", but you can't blame people for the occasional "key = value".

BitcoinTalk

Re: JSON-RPC password

2010-07-21 17:31:09 UTC - -

boost::program_options has the same "key=value" format. Gavin pointed out we can use it in a simple way as a parser without getting into all the esoteric c++ syntax like typed value extraction. We can use more features if we want later.

Lets go ahead with HTTP basic authentication instead of password as a parameter.

BitcoinTalk

Re: JSON-RPC password

2010-07-22 02:34:23 UTC - -

Quote from: gavinandresen on July 22, 2010, 01:11:26 AM
TODO: dialog box or debug.log warning if no rpc.user/rpc.password is set, explaining how to set.

In many of the contexts of this RPC stuff, you can print to the console with fprintf(stdout, like this:
#if defined(__WXMSW__) && wxUSE_GUI
MyMessageBox("Warning: rpc password is blank, use -rpcpw=<password> ", "Bitcoin", wxOK | wxICON_EXCLAMATION);

```
#else
    fprintf(stdout, "Warning: rpc password is blank, use -rpcpw=<password> ");
#endif
```

BitcoinTalk

Re: JSON-RPC password

2010-07-23 17:07:40 UTC - -

Quote from: gavinandresen on July 23, 2010, 03:11:45 PM

Question for everybody: should I add a section to the wiki page describing, in detail, how to do HTTP Basic authentication? PHP and Python make is really easy--just use the http://user:pass@host:port/ URL syntax.

Yes, I think that would be really good so each dev doesn't have to figure it out themselves. We need a simple example for each of Python, PHP and Java importing the json-rpc library and using it to do a getinfo or something, including doing the http authentication part.

BitcoinTalk

Re: JSON-RPC password

2010-07-23 17:14:31 UTC - -

Gavin's changes look good. I think everything is complete. Here's a test build, please test it!

http://www.bitcoin.org/download/bitcoin-0.3.2.5-win32.zip
http://www.bitcoin.org/download/bitcoin-0.3.2.5-linux.tar.gz

BitcoinTalk

Re: bitcoind not responding to RPC

2010-07-23 17:23:47 UTC - -

If I recall correctly, 500 is the prescribed status code for JSON-RPC error responses. There is still a JSON response in the body of the reply telling the explanation of the error, which could be something like {"result":"","error":"bitcoin address not found","id":"1"}.

BitcoinTalk

Faster initial block download (5x faster)

2010-07-23 18:24:56 UTC - -

By making some adjustments to the database settings, I was able to make the initial block download about 5 times faster. It downloads in about 30 minutes.

The database default had it writing each block to disk synchronously, which is not necessary. I changed the settings to let it cache the changes in memory and write them out in a batch. Blocks are still written transactionally, so either the complete change occurs or none of it does, in either case the data is left in a valid state.

I only enabled this change during the initial block download. When you come within 2000 blocks of the latest block, these changes turn off and it slows down to the old way.

I built a test build if you'd like to start using it:

http://www.bitcoin.org/download/bitcoin-0.3.2.5-win32.zip
http://www.bitcoin.org/download/bitcoin-0.3.2.5-linux.tar.gz

These binaries also include Gavin Andresen's JSON-RPC HTTP authentication feature and the other important security improvements from 0.3.2.

I've been running a test over the last 24 hours that kills and restarts it randomly every 2-60 seconds (poor thing) while it's trying to do an initial block download and it's been fine.

There are no changes to the way it handles wallet.dat. This change is only for blk*.dat and the non-critical addr.dat. You can always delete blk*.dat if it gets screwed up and let it re-download.

BitcoinTalk

Re: Faster initial block download

2010-07-23 20:13:27 UTC - -

Quote from: knightmb on July 23, 2010, 07:32:58 PM
Is there a safety reason to stop within the last 2000 blocks or can it be tweaked to stop at remaining 500 blocks for example?

Not really. I'll change it to 1000 next time.

BitcoinTalk

Re: JSON-RPC password

2010-07-23 20:39:03 UTC - -

I don't think authentication should be disabled by default if there's no conf file or the config file doesn't contain "rpcpassword", but what if it contains "rpcpassword="?

I can see both points.

What if the programmer can't figure out how to do HTTP authentication in their language (Fortran or whatever) or it's not even supported by their JSON-RPC library? Should they be able to explicitly disable the password requirement?

OTOH, what if there's a template conf file, with
rpcpassword= # fill in a password here

There are many systems that don't allow you to log in without a password. This forum, for instance. Gavin's point seems stronger.

BTW, I haven't tested it, but I hope having rpcpassword= in the conf file is valid. It's only if you use -server or -daemon or bitcoind that it should fail with a warning. If it doesn't need the password, it should be fine. Is that right?

BitcoinTalk

Re: JSON-RPC Multiple Invocations

2010-07-24 00:59:08 UTC - -

Obviously it's a bug that it repeats the header.

I was trying to follow the 1.0 spec: http://json-rpc.org/wiki/specification It called for multiple invocation.

I think they mean it's like this, but I'm not sure:

Post:
{"method": "postMessage", "params": ["Hello all!"], "id": 99}
{"method": "postMessage", "params": ["I have a question:"], "id": 101}

Reply:
{"result": 1, "error": null, "id": 99}
{"result": 1, "error": null, "id": 101}

I can't remember where I think I saw that it's supposed to send back HTTP status 500 for

an error reply. If it contains multiple responses and one is an error, I wonder if that makes the status 500 for the whole thing, I guess so. Maybe it should always return 200. I think someone sounded like the 500 might be causing a problem.

This probably gets fixed after 0.3.3. Until then, just use single invocation. I wonder if any JSON-RPC package even supports multiple invocation, probably not.

It would be nice if we could pin down better how multiple-invocation is supposed to work, if at all, before trying to fix it, and whether returning HTTP status 500 for error response is right.

BitcoinTalk

Re: bitcoind not responding to RPC

2010-07-24 01:15:58 UTC - -

Can anyone confirm if JSON-RPC over HTTP is supposed to use status 500 if the reply is an error reply? I can't remember where I picked that up, maybe it's wrong. It seems like 200 would make more sense unless there's something wrong with the mechanics of the HTTP request itself. (and maybe that's what it said and I forgot and spread 500 to all error responses)

BitcoinTalk

Re: Warning: don't use -server or bitcoind on a machine where you web browse

2010-07-24 02:29:09 UTC - -

The JSON-RPC HTTP authentication feature in 0.3.3 solves this problem.

BitcoinTalk

Version 0.3.2.5 -- please test!

2010-07-24 03:32:52 UTC - -

Please test 0.3.2.5 in preparation for the 0.3.3 release! This build is looking good and should be the one that goes into 0.3.3. I encourage you to go ahead and upgrade now if you're on Windows or Linux.

New features:
- Gavin Andresen's HTTP authentication to secure JSON-RPC
- 5x faster initial block download, under 30 minutes

Download here:
http://www.bitcoin.org/download/bitcoin-0.3.2.5-win32.zip
http://www.bitcoin.org/download/bitcoin-0.3.2.5-linux.tar.gz

Thanks!

BitcoinTalk

Re: Reading/Writing Blocks and FLATDATA

2010-07-24 04:04:20 UTC - -

FLATDATA was a workaround to serialize a fixed field length array. There was a cleaner way to make it understand how to serialize arrays directly, but MSVC6 couldn't do it and I wanted to keep compatibility with MSVC6 at that time. We don't support MSVC6 anymore because we use something in Boost that doesn't. We lost support for it after 0.2.0. Maybe someday I'll swap in the clean way that just knows how to serialize fixed length arrays without wrapping them in FLATDATA.

BitcoinTalk

Re: a simple traffic load test run

2010-07-25 14:46:33 UTC - -

Was that on the test network?
http://BitcoinTalk.org/index.php?topic=363.0

BitcoinTalk

Re: a simple traffic load test run

2010-07-25 15:29:52 UTC - -

Please do these tests on the test network. That's what it's for. Thanks.

BitcoinTalk

Bitcoin 0.3.3 released -- PLEASE UPGRADE

2010-07-25 16:55:09 UTC - -

Please upgrade to 0.3.3! Important security improvements were made in 0.3.2 and 0.3.3.

New features:
- Gavin Andresen's HTTP authentication to secure JSON-RPC
- 5x faster initial block download, under 30 minutes

BitcoinTalk

Re: Stealing Coins

2010-07-25 17:45:22 UTC - -

It's best if you tell it to me privately so it can be fixed first.

I just e-mailed you my e-mail address. (or you could PM me here)

BitcoinTalk

Re: Stealing Coins

2010-07-25 19:06:23 UTC - -

Red, thanks for telling me privately first! Please go ahead and post it (and relieve the suspense for everyone!)

His point is that transactions paid to a Bitcoin Address are only as secure as the hash function. To make Bitcoin Addresses short, they are a hash of the public key, not the public key itself. An attacker would only have to break the hash function, not ECDSA.

BitcoinTalk

Re: Stealing Coins

2010-07-25 20:01:40 UTC - -

Quote from: knightmb on July 25, 2010, 07:44:02 PM
If I figure out that Public Key 123456 generates Hash ABCD
and
Public Key 654321 also generates Hash ABCD
I'm still left without the Private Key.

But from what you are saying, all I need is Public Key 654321 and I can spend coin pretending to be Public Key 123456.

You would still have to sign it with public key 654321. You need to find a collision using a public key for which you know the private key.

When you claim a Bitcoin Address transaction, you give your public key that matches the hash, then you must sign it with that key.

Red's point is that it's easy to quickly generate insecure public keys which you could break and find the private key after you find a collision.

He points out that if the public key was required to be a secure one, one which must have required significant work to find the prime numbers, that would increase the strength above that of the hash function alone. Someone trying to brute force would have to take time generating a key for each attempt.

BitcoinTalk

Re: Stealing Coins

2010-07-25 20:48:01 UTC - -

Quote

Here is a paper that claims to find SHA-1 collisions in 2^52 crypto operations. And optimally secure hash would take 2^80 operations. 2^52 time is still large, but it is getting into cluster and botnet range.

2^80 is if you can use a birthday attack. You can't use a birthday attack for this, so the difficulty is the full 2^160 bits. Although, if you were trying to crack any one of 1 million (2^20) transactions, you could do a partial birthday attack 2^160/2^20 = 2^140.

Bitcoin Addresses are the only place where 160-bit hash is used. Everything else is SHA-256. They're calculated as:

bitcoinaddress = RIPEMD-160(SHA-256(publickey))

Correct me if I'm wrong (please, and I'll gladly eat crow) but I think it would be hard to use an analytical attack on RIPEMD-160 in this case. An analytical attack prescribes a certain range or pattern of inputs to try that will greatly increase your chance of finding a collision. Here, you don't have that kind of control over RIPEMD-160's input, because the input is the output of SHA-256. If an analytical attack helps you find an input to RIPEMD-160 that produces a collision, what are you going to do with it? You still have to get SHA-256 to output that value, so you would still have to break SHA-256 too.

For brute force, RIPEMD-160(SHA-256(x)) is no stronger than RIPEMD-160 alone. But for analytical attack, it seems like you must analytical attack both RIPEMD-160 and SHA-256. If I'm wrong, then the strength is the same as RIPEMD-160 and the SHA-256 only serves as one round of key strengthening.

BitcoinTalk

Re: JSON-RPC password

2010-07-25 21:34:29 UTC - -

Quote from: lachesis on July 25, 2010, 07:52:35 PM

I found what appears to be a bug: with a long enough username and password combination, the base64 encoder in bitcoind produces authorization headers that look like this:

Code:

...

Authorization: Basic YWJiYWJiYWFiYmE6aGVsbG93b3JsZGhlbGxvd29ybGRoZWxsb3dvcmxkaGVsbG93b3JsZGhlbGxvd29ybGRoZWxsb3dvcmxk

It inserts a newline every 64 characters, which obviously breaks the Authorization header, so commands like "bitcoin getinfo" fail. The server still works fine with properly behaving clients.

This can be solved by removing the newlines (and maybe ' 's) from result at the end of the Base64Encode function:

Code:

result.erase(std::remove(result.begin(), result.end(), ' '), result.end());
result.erase(std::remove(result.begin(), result.end(), ' '), result.end());

+1 to you for having such a long password that you found this bug.

Uploaded to SVN as rev 110.

BitcoinTalk

Re: JSON-RPC password

2010-07-25 21:44:16 UTC - -

Quote from: BitLex on July 25, 2010, 08:45:38 PM

i got some problems here too trying to get this run on PHP.
so far i had no luck, neither the wiki-sample (jsonRPCClient trying to fopen(http://username:password@localhost:8332/)), nor my curl-sample (using setopt CURLOPT_HTTPAUTH, CURLAUTH_BASIC) seem to work.

That's strange, didn't someone just say that was supposed to work? (what library was he using?) Post if you figure out what wrong.

I hope it's not going to put up this much of a fight for all PHP users.

Looks like we've got the Fortran scenario already.

BitcoinTalk

Re: JSON-RPC password

2010-07-25 21:51:31 UTC - -

Quote from: gavinandresen on July 25, 2010, 09:38:19 PM
Great catch! Simpler fix is to specify the BIO_FLAGS_BASE64_NO_NL in the rpc.cpp/EncodeBase64 function

SVN rev 111

BitcoinTalk

Re: md5?

2010-07-25 22:06:57 UTC - -

For future reference, here's my public key. It's the same one that's been there since the bitcoin.org site first went up in 2008. Grab it now in case you need it later.

http://www.bitcoin.org/Satoshi_Nakamoto.asc

BitcoinTalk

Re: Stealing Coins

2010-07-25 22:27:36 UTC - -

Sorry, actually it's ECDSA (Elliptic Curve Digital Signature Algorithm) not RSA. I shouldn't have said "prime numbers". ECDSA doesn't take much time to generate a keypair.

BitcoinTalk

bitcoind without wxWidgets

2010-07-26 17:23:33 UTC - -

I replaced the last of the few wxBase dependencies in bitcoind.

bitcoind now compiles without wxWidgets or wxBase in SVN rev 112.

main(int argc, char* argv[]) is added to init.cpp. CMyApp and the Startup folder stuff are moved to ui.cpp. ui.cpp and uibase.cpp aren't linked by bitcoind.

The makefiles have -DGUI to control whether the GUI is used.

I test compiled MinGW, VC and Ubuntu. I don't know if I broke the Mac OSX build, someone will need to check that.

BitcoinTalk

Re: Bitcoin x64 for Windows

2010-07-26 18:41:31 UTC - -

Quote from: Olipro on July 26, 2010, 06:39:17 AM

Credit to tcatm for the caching part of the SHA context - this offers absolutely brilliant performance. Additionally, the Intel compiler really comes into its own here as its parallelisation abilities give a massive performance boost over Visual Studio.

Performance: 4700khash/s on 4 cores, I think that speaks for itself.

I've included both the VS and Intel build, but there's really no comparison, the Intel build craps all over VS.

Is that still starting from Crypto++? Lets get this into the main sourcecode.

BitcoinTalk

Re: Bitcoin x86 for Windows

2010-07-27 01:29:42 UTC - -

Quote from: Olipro on July 26, 2010, 01:04:41 PM

Crypto++ 5.6.0: http://www.cryptopp.com/
Cached SHA256: http://pastebin.com/rJAYZJ32(although I'm pretty sure this is publically submitted elsewhere, I was linked to it on IRC)

I added the cached SHA256 state idea to the SVN, rev 113. The speedup is about 70%. I credited it to tcatm based on your post in the x64 thread.

I can compile the Crypto++ 5.6.0 ASM SHA code with MinGW but as soon as it runs it crashes. It says its for MASM (Microsoft's assembler) and the sample command line they give looks like Visual C++. Does it only work with the MSVC and Intel compilers?

BitcoinTalk

Re: Proof-of-work difficulty increasing

2010-07-27 03:04:58 UTC - -

New difficulty factor 244.213223092
+35%

I updated the first post.

date, difficulty factor, % change
2009 1.00
30/12/2009 1.18 +18%
11/01/2010 1.31 +11%
25/01/2010 1.34 +2%
04/02/2010 1.82 +36%
14/02/2010 2.53 +39%
24/02/2010 3.78 +49%
08/03/2010 4.53 +20%
21/03/2010 4.57 +9%
01/04/2010 6.09 +33%
12/04/2010 7.82 +28%
21/04/2010 11.46 +47%
04/05/2010 12.85 +12%
19/05/2010 11.85 -8%
29/05/2010 16.62 +40%
11/06/2010 17.38 +5%
24/06/2010 19.41 +12%
06/07/2010 23.50 +21%
13/07/2010 45.38 +93%
16/07/2010 181.54 +300%
27/07/2010 244.21 +35%

BitcoinTalk

Re: Bitcoin x86 for Windows

2010-07-27 18:27:30 UTC - -

Quote from: BlackEye on July 25, 2010, 10:12:23 PM

I was able to integrate the SHA256 functionality from Crypto++ 5.6.0 into Bitcoin. This is the fastest SHA256 yet using the SSE2 assembly code. Since Bitcoin was sending unaligned data to the block hash function, I had to change the MOVDQA instruction to MOVDQU.

I think using the SHA256 functionality from Crypto++ 5.6.0 is the way forward right now.

I added a subset of the Crypto++ 5.6.0 library to the SVN. I stripped it down to just SHA and 11 general dependency files. There shouldn't be any other crypto in there other than SHA.

I aligned the data fields and it worked. The ASM SHA-256 is about 48% faster. The combined speedup is about 2.5x faster than version 0.3.3.

I guess it's using SSE2. It automatically sets its build configuration at compile time based on the compiler environment.

It looks like it has some SSE2 detection at runtime, but it's hard to tell if it actually uses it to fall back if it's not available. I want the release builds to have SSE2. SSE2 has been around since the first Pentium 4. A Pentium 3 or older would be so slow, you'd be wasting your electricity trying to generate on it anyway.

This is SVN rev 114.

BitcoinTalk

Re: Bitcoin x86 for Windows

2010-07-27 19:47:42 UTC - -

OK, thanks. I'd also like to know if it runs fine as long as you don't turn on Generate. You'd think as long as it doesn't actually execute any SSE2 instructions, it would still load. At least Pentium 3's could run it without generating.

BitcoinTalk

Re: Having problems specifing -datadir

2010-07-28 20:58:26 UTC - -

It was able to reproduce this. The database doesn't like the relative path.

"bitcoind -datadir=./subdir getinfo" works against a running daemon, but trying to start the daemon as "bitcoind -datadir=./subdir" gets that exception.

I guess we should resolve the full path before passing it to the database.

It looks like you were the first one to ever use -datadir with a relative path.

BitcoinTalk

Re: Build error SVN r115 on my Mac: workaround

2010-07-28 21:23:23 UTC - -

Was that the only thing I broke in the OSX build?! Does it actually work after just that one change?

I had to do that for makefile.vc also. It compiled, but SHA-256 didn't work correctly; it returned the same incorrect hash each time.

We'll disable it now, and if anyone figures out how to fix it, we can re-enable it then. It's still 1.7x faster from the midstate optimisation.

The Crypto++ ASM SHA-256 works with GCC on Linux and Windows (MinGW).

I uploaded this makefile.osx change to SVN. (let me know if that compiles now)

BitcoinTalk

Re: Difficulty

2010-07-29 01:16:23 UTC - -

You were looking at the wrong code. Here's the code that applies:

Code:

```
bool CBlock::CheckBlock() const
{
...
    // Check timestamp
    if (nTime > GetAdjustedTime() + 2 * 60 * 60)
        return error("CheckBlock() : block timestamp too far in the future");
...

bool CBlock::AcceptBlock()
{
   ...
    // Check timestamp against prev
    if (nTime <= pindexPrev->GetMedianTimePast())
        return error("AcceptBlock() : block's timestamp is too early");
```

The timestamp is limited to up to 2 hours in the future. It can be earlier than the previous block, but it must be greater than the median of the last 11 blocks. The reason for doing it that way is so the time can get corrected in the next block if the previous block had the time too far in the future, like what happened.

BitcoinTalk

Re: Scalability and transaction rate

2010-07-29 02:00:38 UTC - -

The current system where every user is a network node is not the intended configuration for large scale. That would be like every Usenet user runs their own NNTP server. The design supports letting users just be users. The more burden it is to run a node, the fewer nodes there will be. Those few nodes will be big server farms. The rest will be client nodes that only do transactions and don't generate.

Quote from: bytemaster on July 28, 2010, 08:59:42 PM
Besides, 10 minutes is too long to verify that payment is good. It needs to be as fast as swiping a credit card is today.

See the snack machine thread, I outline how a payment processor could verify payments well enough, actually really well (much lower fraud rate than credit cards), in something like 10 seconds or less. If you don't believe me or don't get it, I don't have time to try to convince you, sorry.
http://BitcoinTalk.org/index.php?topic=423.msg3819#msg3819

BitcoinTalk

Re: wiki registration email?

2010-07-29 02:10:46 UTC - -

WTF? How did we get on that? AFAIK, the only e-mail is if you tell the forum to do notifications, and I guess the wiki registration. I'd consider turning off the forum notification e-mails, I don't know why we have that.

BitcoinTalk

*** ALERT *** Upgrade to 0.3.6

2010-07-29 19:13:06 UTC - -

Please upgrade to 0.3.6 ASAP! We fixed an implementation bug where it was possible that bogus transactions could be displayed as accepted. Do not accept Bitcoin transactions as payment until you upgrade to version 0.3.6!

If you can't upgrade to 0.3.6 right away, it's best to shut down your Bitcoin node until you do.

Also in 0.3.6, faster hashing:
- midstate cache optimisation thanks to tcatm
- Crypto++ ASM SHA-256 thanks to BlackEye
Total generating speedup 2.4x faster.

Download:
http://sourceforge.net/projects/bitcoin/files/Bitcoin/bitcoin-0.3.6/

Windows and Linux users: if you got 0.3.5 you still need to upgrade to 0.3.6.

BitcoinTalk

Re: *** ALERT *** version 0.3.6

2010-07-29 19:55:51 UTC - -

Haven't had time to update the SVN yet. Wait for 0.3.6, I'm building it now. You can shut down your node in the meantime.

BitcoinTalk

Re: *** ALERT *** version 0.3.6

2010-07-29 20:30:15 UTC - -

SVN is updated with version 0.3.6.

Uploading Windows build of 0.3.6 to Sourceforge now, then will rebuild linux.

BitcoinTalk

Re: *** ALERT *** Upgrade to 0.3.6 ASAP!

2010-07-29 21:20:38 UTC - -

0.3.6 Linux build is back to the old makefile.unix. It static links libjpeg so that shouldn't be a problem.

Is that working better?

If you got 22DbRunRecoveryException and you've used someone else's build before, you may need to delete (or move the files somewhere else) database/log.000000*

Windows and Linux users: if you got 0.3.5 you still need to upgrade to 0.3.6.

BitcoinTalk

Re: *** ALERT *** Upgrade to 0.3.6 ASAP!

2010-07-29 21:43:15 UTC - -

"./bitcoin: /lib64/libc.so.6: version `GLIBC_2.11' not found (required by ./bitcoin)" isn't a new problem that started with 0.3.6 is it? This was built on the same OS installations as 0.3.0.

Unfortunately I upgraded to Ubuntu 10.04 before 0.3.0. I will not upgrade anymore. I don't know when I might have time to reinstall to downgrade, but at least by not upgrading, it'll gradually fix itself.

BitcoinTalk

Re: Implementation bug prior to 0.3.6

2010-07-29 22:04:15 UTC - -

Actually, it works well to just PM me. I'm the one who's going to be fixing it. If you find a security flaw, I would definitely like to hear from you privately to fix it before it goes public.

BitcoinTalk

Re: Transaction disappeared in the void...

2010-07-29 22:08:31 UTC - -

If the transaction didn't go out immediately at first, like if you weren't connected at the time, it may take up to 2 hours to resend it. Long term, it does keep relentlessly sending it.

I'll shorten that length of time in a future version.

You do need to have downloaded the complete block chain (currently 71040 blocks) before you'll see any confirms. Same with the recipient.

BitcoinTalk

Re: Linux distribution download

2010-07-29 22:17:24 UTC - -

Yeah, acutely aware that I should have stayed on 9.04 or 9.10. It's a lot more work to downgrade than upgrade and I've been squeezed for time. Ubuntu is the most popular distro, so I'm staying with that.

BitcoinTalk

Re: *** ALERT *** Upgrade to 0.3.6 ASAP!

2010-07-29 23:12:12 UTC - -

Quote from: lachesis on July 29, 2010, 10:14:36 PM
On Debian testing 32-bit, I get a few build errors, all resembling:
Code:
script.cpp:114: error: \u0091OP_NOP1\u0092 was not declared in this scope
I got these when attempting to "make bitcoind" without "make clean" or "make" first. It looks like the bitcoind build instructions don't compile the headers first, but they also don't delete the headers.h.gch, so the old headers are used if present.

If anyone else gets this error, the simplest solution is to "make clean" and retry the build.

We don't really need pre-compiled header. It only makes it compile slightly faster. I think I'll just get rid of it. Even still, you'd still need to remember to "make -f makefile.unix clean" or delete headers.h.gch one more time to get rid of the leftover file.

Damn that GLIBC_2.11. I thought I'd been careful not to accept any of the updates.

bitcoin-list

[**bitcoin-list**] Alert: upgrade to bitcoin 0.3.6

2010-07-30 06:02:38 UTC - -

Please upgrade to 0.3.6 ASAP to get an important bugfix.

See the bitcoin.org homepage for download links.

BitcoinTalk

Re: Bug: "Immature" coins lost in wallet.dat during transaction

2010-07-30 19:19:05 UTC - -

I don't get how it let you send if it was not matured. Your balance would have been lower than the amount. It would have said balance 0.01, right? If I try that it says "you don't have enough money" or "Insufficient funds" from the command line.

How many blocks did it say it had left to mature when you sent?

There's a chance it might still go through.

Have you copied or moved your wallet.dat in any way?

BitcoinTalk

Re: [PATCH] implement 'listtransactions'

2010-07-30 19:40:54 UTC - -

What are you needing to use listtransactions for?

The reason I didn't implement listtransactions is I want to make sure web programmers don't use it. It would be very easy to latch onto that for watching for received payments. There is no reliable way to do it that way and make sure nothing can slip through the cracks. Until we have solid example code using getreceivedbyaddress and getreceivedbylabel to point to and say "use this! use this! don't use listtransactions!", I don't think we should implement listtransactions.

When we do implement listtransactions, maybe one way to fight that is to make it all text. It should not break down the fields into e.g. comment, confirmations, credit, debit. It could be one pretty formatted string like "0/unconfirmed 0:0:0 date comment debit 4 credit 0" or something so it's hard for programmers to do the wrong thing and process it. It's only for viewing the status of your server. I guess that would be kinda annoying for web interfaces that would rather format it into html columns though.

BitcoinTalk

Re: *** ALERT *** Upgrade to 0.3.6 ASAP!

2010-07-30 19:53:06 UTC - -

Quote from: knightmb on July 30, 2010, 07:24:07 PM

I can only imagine the pain you went through to get these builds because I'm trying to build the program on a Ubuntu 9.04 box and so far I can't seem to find all the dependencies to compile no matter how much I keep installing packages and compiling source, LOL.

I can't understand why you're having so much pain. I just followed the instructions in build-unix.txt. I made a couple little corrections for Boost 1.37, which I'll put on SVN the next time I update it, noted below:

Dependencies
---
sudo apt-get install build-essential
sudo apt-get install libgtk2.0-dev
sudo apt-get install libssl-dev
sudo apt-get install libdb4.7-dev
sudo apt-get install libdb4.7++-dev
sudo apt-get install libboost-all-dev (or libboost1.37-dev)

wxWidgets
---
cd /usr/local
tar -xzvf wxWidgets-2.9.0.tar.gz
cd /usr/local/wxWidgets-2.9.0
mkdir buildgtk
cd buildgtk
../configure --with-gtk --enable-debug --disable-shared --enable-monolithic
make
sudo su
make install
ldconfig

added a comment in makefile.unix:

```
# for boost 1.37, add -mt to the boost libraries
LIBS= \
 -Wl,-Bstatic \
   -l boost_system \
   -l boost_filesystem \
   -l boost_program_options \
   -l boost_thread \
   -l db_cxx \
   -l crypto \
 -Wl,-Bdynamic \
   -l gthread-2.0
```

BitcoinTalk

Re: *** ALERT *** Upgrade to 0.3.6 ASAP!

2010-07-30 21:44:04 UTC - -

> ***Quote from: knightmb on July 30, 2010, 08:04:19 PM***
> *So that last command should simply be*
> ***sudo apt-get install libboost1.37-dev***

Except that wouldn't work for boost 1.40+ (on Ubuntu 10.04), where you need to get libboost-all-dev.

Seems they changed everything around in Boost recently, "-mt" and all that, makes it hard.

BTW, I tried Boost 1.34 but it didn't have the boost.interprocess stuff.

Mac OSX version is available now. See bitcoin.org or the SourceForge link.

BitcoinTalk

Re: 4 hashes parallel on SSE2 CPUs for 0.3.6

2010-07-31 00:29:20 UTC - -

That's amazing...

So are you saying you use 128-bit registers to SIMD four 32-bit data at once? I've wondered about that for a long time, but I didn't think it would be possible due to addition carrying into the neighbour's value.

BitcoinTalk

Webpage idea: Next predicted difficulty change

2010-07-31 01:32:08 UTC - -

It would be neat if someone had a page (like that handy calculator at http://www.alloscomp.com/bitcoin/calculator.php) that projects what the next difficulty adjustment will be.

projected difficulty adjustment multiplier =

```
 blocks_since_last_adjustment / 2016
 -----------------------------------
 time_since_last_adjustment / 14_days
```

For instance, if it already got half way to the next adjustment in only 3.5 days instead of 7, we would expect difficulty to double:

(1008/2016) / (3.5/14) = 0.5/0.25 = 2.0

Also, it could show the predicted time when the next adjustment will occur, and tell when the last adjustment was and how much it changed.

BitcoinTalk

Re: Linux distribution download

2010-07-31 14:38:52 UTC - -

It can be built with Boost 1.37 or later.

BitcoinTalk

Re: Linux version => No GUI after upgrade. WTF?

2010-08-02 17:39:27 UTC - -

Did it print anything to the console? Are you sure you didn't run "bitcoind"?

Try version 0.3.7.

BitcoinTalk

Re: Mac Client Problems Outlined...

2010-08-02 18:02:20 UTC - -

"Minimize to the tray instead of the taskbar" & "Minimize to the tray on close" must not be implemented yet on the Mac. We should grey them out in the next version.

BitcoinTalk

Re: 4 hashes parallel on SSE2 CPUs for 0.3.6

2010-08-02 19:02:46 UTC - -

Is it 2x fast on AMD and 1/2 fast on Intel?

Quote from: tcatm on July 31, 2010, 10:12:38 AM

Btw. Why are you using this alignup<16> function when __attribute__ ((aligned (16))) will tell the compiler to align at compiletime?

Tried that, but it doesn't work for things on the stack. I ran some tests.

It doesn't even cause an error, it just doesn't align it.

BitcoinTalk

Re: Protocol Buffers for Bitcoin

2010-08-02 20:22:08 UTC - -

The reason I didn't use protocol buffers or boost serialization is because they looked too complex to make absolutely airtight and secure. Their code is too large to read and be sure that there's no way to form an input that would do something unexpected.

I hate reinventing the wheel and only resorted to writing my own serialization routines reluctantly. The serialization format we have is as dead simple and flat as possible. There is no extra freedom in the way the input stream is formed. At each point, the next field in the data structure is expected. The only choices given are those that the receiver is expecting. There is versioning so upgrades are possible.

CAddress is about the only object with significant reserved space in it. (about 7 bytes for flags and 12 bytes for possible future IPv6 expansion)

The larger things we have like blocks and transactions can't be optimized much more for size. The bulk of their data is hashes and keys and signatures, which are uncompressible. The serialization overhead is very small, usually 1 byte for size fields.

On Gavin's idea about an existing P2P broadcast infrastructure, I doubt one exists. There are few P2P systems that only need broadcast. There are some libraries like Chord that try to provide a distributed hash table infrastructure, but that's a huge difficult problem that we don't need or want. Those libraries are also much harder to install than ourselves.

BitcoinTalk

Re: Builds for Ubuntu?

2010-08-03 20:56:11 UTC - -

Quote from: nimnul on August 03, 2010, 05:51:15 PM

Is satoshi noWx patch in 0.3.7 already? Before that bitcoind required wx, and I never seen Satoshi announcing that it's in trunk

Yes, 0.3.7 has it. It was in rev 112.

BitcoinTalk

Re: Bitcoind x86 binary for CentOS

2010-08-03 21:05:08 UTC - -

Quote from: sgtstein on August 03, 2010, 05:30:37 PM
I have successfully built it with 4.8, 4.7 never would but with 4.8 bitcoind locks up whenever it dumps the initial block download to disk.

I urge you not to use BDB 4.8. The database/log0000* files will be incompatible if anyone uses your build and then goes back to the official build.

BitcoinTalk

Re: Content-Length header and 500 (was Re: Authentication, JSON RPC and Python)

2010-08-03 21:26:26 UTC - -

Quote from: gavinandresen on August 03, 2010, 06:56:44 PM
Quote from: jgarzik on August 03, 2010, 06:09:08 PM
bitcoin requires the Content-Length header, but several JSON-RPC libraries do not provide it. When the Content-Length header is absent, bitcoin returns 500 Internal Server Error.
Can you be more specific about which JSON libraries don't provide Content-Length ? It'd be nice to document that.

I guess we should try to support the case where there's no Content-Length parameter. I don't want to rip and replace streams though, even if it has to read one character at a time.

Edit: That is, assuming there actually are any libraries that don't support Content-Length.

BitcoinTalk

Re: What happens when network is split for prolonged time and reconnected?

2010-08-03 22:45:07 UTC - -

creighto: I agree with that idea. After a few hours, it should be possible for the client to notice if the flow of blocks has dropped off by more than would be likely just by chance. It could tell if it's not hearing the hum of the world anymore.

Quote from: knightmb on August 03, 2010, 07:02:13 PM
Quote from: gavinandresen on August 03, 2010, 06:38:44 PM

Or if the split lasted long enough (more than 100 blocks), transactions that involve generated coins on the shorter chain would be invalid at the merge.

Interesting info, so other than some double-spending issues, as long as the block chain isn't separated for more than 100 or so blocks (or 16+ hours),

In practice, splits are likely to be very asymmetrical. It would be hard to split the world down the middle. More likely it would be a single country vs the rest of the world, lets say a 1:10 split. In that case, it would take the minority fork 10 times as long to generate 100 blocks, so about 7 days. Also it would be super easy for the client to realize it's hearing way too few blocks and something must be wrong.

Quote from: knightmb on August 03, 2010, 07:02:13 PM

If there a hard coded limit on split delay? Meaning if I had a small network split from the public network, spent some coin around, came back a few days later and got them sync up to the public network (other than coin generation if it happened) transactions should be fine?

There's no time limit. Assuming you weren't spending coins generated in the minority fork, or spending someone's double-spends you received, your transactions can get into the other chain at any time later.

BitcoinTalk

Please upgrade to 0.3.8!

2010-08-03 23:40:18 UTC - -

Version 0.3.8 adds an important security improvement. Everyone should upgrade to get this change.

The new safety feature displays a warning message in the status bar and locks down RPC if it detects a problem that may require an upgrade.

If it sees a longer chain, but it can't process it, then it knows something is wrong. It displays "WARNING: Displayed transactions may not be correct! You may need to upgrade." and makes most RPC commands return an error. It still keeps generating as normal, which is necessary for the stability of the network.

There were important security updates in the versions before this too, so if you haven't upgraded recently, it's extremely important that you upgrade now!

Also, don't forget, we recently added 2.4x faster generating thanks to tcatm's mid-state caching optimisation and BlackEye's help getting ASM SHA-256 working.

Download:
http://sourceforge.net/projects/bitcoin/files/Bitcoin/bitcoin-0.3.8/

BitcoinTalk

Re: Bitcoind x86 binary for CentOS

2010-08-04 00:09:32 UTC - -

Quote from: knightmb on August 03, 2010, 11:46:46 PM
There are two versions, one built from stock code, the other modified to accept up to 1,000 nodes (hence the super node name)

I'd rather you didn't make a build of the 1000 node connecting version available. It won't take very many people running that before we have to make another release just to limit the incoming connections.

BitcoinTalk

Re: Please upgrade to 0.3.8!

2010-08-04 00:29:37 UTC - -

I guess SourceForge hasn't updated its mirrors yet. The files are there on the admin side, but not on the user side. I have no idea how long that will take. It's always been immediate in the past.

Edit: SourceForge is updated now.

BitcoinTalk

Re: Building initial transaction trust through "coin ripping"

2010-08-04 00:40:40 UTC - -

The software is designed to support things like this. I was going to post details of the plans for Escrow, but since getting slashdotted I haven't had time.

BitcoinTalk

Re: Flood attack 0.00000001 BC

2010-08-04 16:25:36 UTC - -

Quote from: Insti on August 04, 2010, 02:58:31 PM
It seems to do more harm than good because it prevents micropayment implementations such as the one bytemaster is suggesting.

Bitcoin isn't currently practical for very small micropayments. Not for things like pay per search or per page view without an aggregating mechanism, not things needing to pay less than 0.01. The dust spam limit is a first try at intentionally trying to prevent overly small micropayments like that.

Bitcoin is practical for smaller transactions than are practical with existing payment methods. Small enough to include what you might call the top of the micropayment range. But it doesn't claim to be practical for arbitrarily small micropayments.

BitcoinTalk

Re: Flood attack 0.00000001 BC

2010-08-05 16:03:21 UTC - -

Forgot to add the good part about micropayments. While I don't think Bitcoin is practical for smaller micropayments right now, it will eventually be as storage and bandwidth costs continue to fall. If Bitcoin catches on on a big scale, it may already be the case by that time. Another way they can become more practical is if I implement client-only mode and the number of network nodes consolidates into a smaller number of professional server farms. Whatever size micropayments you need will eventually be practical. I think in 5 or 10 years, the bandwidth and storage will seem trivial.

I am not claiming that the network is impervious to DoS attack. I think most P2P networks can be DoS attacked in numerous ways. (On a side note, I read that the record companies would like to DoS all the file sharing networks, but they don't want to break the anti-hacking/anti-abuse laws.)

If we started getting DoS attacked with loads of wasted transactions back and forth, you would need to start paying a 0.01 minimum transaction fee. 0.1.5 actually had an option to set that, but I took it out to reduce confusion. Free transactions are nice and we can keep it that way if people don't abuse them.

That brings up the question: if there was a minimum 0.01 fee for each transaction, should we automatically add the fee if it's just the minimum 0.01? It would be awfully annoying to ask each time. If you have 50.00 and send 10.00, the recipient would get 10.00 and you'd have 39.99 left. I think it should just add it automatically. It's trivial compared to the fees many other types of services add automatically.

Quote from: FreeMoney on August 04, 2010, 07:30:32 PM
Does including more slow down your hashing rate?

No, not at all.

BitcoinTalk

Re: Flood attack 0.00000001 BC

2010-08-05 16:30:20 UTC - -

Quote from: bytemaster
Payments would generally be advanced, say 1 BTC at a time and when the connection closes any "change" would be returned. This rule makes it impossible to pay for a simple "search query" with no further transactions.

One alternative is to use a round-up system. You pay for, say, 1000 pages or images or downloads or searches or whatever at a time. When you've used up your 1000 pages, you pay for another 1000 pages. If you only use 1 page, then you have 999 left that you may never use, but it's not a big deal because the cost per 1000 is still small.

Or you could pay per day. The first time you access the site on a given day, you pay for 24 hours of access.

Per 1000 or per day may be easier for consumers to get their heads around too. They worry about per item because it's harder to figure if it might add up too fast. Unlimited for 24 hours they know what the cost will be. Or if 1000 seems like plenty, they're not worrying that it's costing more with each click if they figure 1000 is more than they'll probably use.

BitcoinTalk

Re: Flood attack 0.00000001 BC

2010-08-05 16:39:58 UTC - -

Quote from: bytemaster on August 05, 2010, 03:39:19 PM
The only solution to this problem is to make broadcasting of a transaction "non free". Namely, if you want me to include it you have to pay me. The net (no pun intended) result is that each client would need to pay other clients to whom they even send their transaction, not just the individual who gets it in a block. In this way the laws of economics take over and no one gets a free ride on the transaction broadcast system.

I don't know a way to implement that. The transaction fee to the block creator uses a special trick to include the transaction fee without any additional size. If there was a transaction for each transaction fee, then what about the transactions fees for the transaction fee's transaction?

BitcoinTalk

Re: Who's the Spanish jerk draining the Faucet?

2010-08-05 17:06:03 UTC - -

Silently failing would look bad.

Quote from: gavinandresen on August 04, 2010, 08:40:55 PM
1. Rate limit based on the first byte of the IP address (79. or 81. in this case).

Definitely needed. What rate are you thinking of? Ultimately, it's better to rate limit it than to let it all drain out.

Quote from: gavinandresen on August 04, 2010, 08:40:55 PM
3. Rate limit based on last two domains of reverse DNS lookup of the IP address (rima-tde.net in this case).

That might work surprisingly well. If it works, it keeps them from hitting the rate limit, but the rate limit is there as the last line of defence.

Quote from: gavinandresen on August 04, 2010, 08:40:55 PM
4. Make the standard amount given away 0.5 Bitcoins (Bitcoins have gone up 10 times in value since I started the Faucet).

Definitely time to lower it.

BitcoinTalk

Re: bitcoind transaction to ip address

2010-08-05 17:28:40 UTC - -

It's not implemented.

It turned out nobody liked that mode of transfer anyway, so it hasn't had much development attention.

BitcoinTalk

Re: Transaction Overload Solution

2010-08-05 17:38:21 UTC - -

I can't think of a way to implement that. All the transaction fees would be additional transactions. What about the transaction fees for the transaction fee's transaction?

BitcoinTalk

Re: Flood attack 0.00000001 BC

2010-08-05 17:49:43 UTC - -

Quote from: bytemaster on August 05, 2010, 04:46:52 PM
Right now the transaction fee address is left "blank" and the block generator fills it out.
Now you would fill it in with the address of the person you are asking to build the block.

If you're only going to have one person work on building the block, that could take days. Oh, do you mean send a different variation to each node with the tx fee written to them?

The way it is now, it's whoever builds this gets it.

If we needed to, we could have a BitTorrent-esque tit-for-tat for transaction broadcast. Relay paying transactions to me, or I won't relay them to you. It probably won't be an actual problem though. It only takes one node relaying like it should to cancel out 7 others greedily not relaying.

BitcoinTalk

Re: A proposal for a semi-automated Escrow mechanism

2010-08-05 18:08:30 UTC - -

A transaction can be written that requires two signatures to spend it next. You write a payment that requires the signature of both the recipient and the sender to spend it. To release the escrow, you give the recipient the signature for your half, or the payee can return it by giving you his signed half. There's no mediator in this simple case. The recourse is to refuse to ever release it, essentially burning the money.

BitcoinTalk

Re: latency and locality

2010-08-07 16:28:17 UTC - -

Once you get away from a system where each node's influence is proportional to their CPU power, then what else do you use to determine who is (approximately) one person?

BitcoinTalk

Re: Bitcoin minting is thermodynamically perverse

2010-08-07 17:46:09 UTC - -

It's the same situation as gold and gold mining. The marginal cost of gold mining tends to stay near the price of gold. Gold mining is a waste, but that waste is far less than the utility of having gold available as a medium of exchange.

I think the case will be the same for Bitcoin. The utility of the exchanges made possible by Bitcoin will far exceed the cost of electricity used. Therefore, *not* having Bitcoin would be the net waste.

Quote from: gridecon on August 06, 2010, 04:48:00 PM

As an overall point, I also do not agree with the idea that the very high computational burden of coin generation is in fact a necessity of the current system. As I understand it, currency creation is fundamentally metered by TIME - and if that is the fundamental controlling variable, what is the need for everyone to "roll as many dice as posible" within that given time period? The "chain of proof" for coin ownership and transactions doesn't depend on the method for spawning coins.

Each node's influence on the network is proportional to its CPU power. The only way to show the network how much CPU power you have is to actually use it.

If there's something else each person has a finite amount of that we could count for one-person-one-vote, I can't think of it. IP addresses... much easier to get lots of them than CPUs.

I suppose it might be possible to measure CPU power *at certain times*. For instance, if the CPU power challenge was only run for an average of 1 minute every 10 minutes. You could still prove your total power at given times without running it all the time. I'm not sure how that could be implemented though. There's no way for a node that wasn't present at the time to know that a past chain was actually generated in a duty cycle with 9 minute breaks, not back to back.

Proof-of-work has the nice property that it can be relayed through untrusted middlemen. We don't have to worry about a chain of custody of communication. It doesn't matter who tells you a longest chain, the proof-of-work speaks for itself.

BitcoinTalk

Re: A proposal for a semi-automated Escrow mechanism

2010-08-07 20:04:59 UTC - -

Quote from: jgarzik on August 05, 2010, 07:00:30 PM

Due to that recourse, it is unlikely to be used as an escrow mechanism 

Really? Do you think people won't be able to understand the benefit? (If your response is an argument that there's no benefit at all, I guess that will reinforce the case that people won't be able to understand it.)

BitcoinTalk

Escrow

2010-08-07 20:13:52 UTC - -

Here's an outline of the kind of escrow transaction that's possible in software. This is not implemented and I probably won't have time to implement it soon, but just to let you know what's possible.

The basic escrow: The buyer commits a payment to escrow. The seller receives a transaction with the money in escrow, but he can't spend it until the buyer unlocks it. The buyer can release the payment at any time after that, which could be never. This does not allow the buyer to take the money back, but it does give him the option to burn the money out of spite by never releasing it. The seller has the option to release the money back to the buyer.

While this system does not guarantee the parties against loss, it takes the profit out of cheating.

If the seller doesn't send the goods, he doesn't get paid. The buyer would still be out the money, but at least the seller has no monetary motivation to stiff him.

The buyer can't benefit by failing to pay. He can't get the escrow money back. He can't fail to pay due to lack of funds. The seller can see that the funds are committed to his key and can't be sent to anyone else.

Now, an economist would say that a fraudulent seller could start negotiating, such as "release the money and I'll give you half of it back", but at that point, there would be so little trust and so much spite that negotiation is unlikely. Why on earth would the fraudster keep his word and send you half if he's already breaking his word to steal it? I think for modest amounts, almost everyone would refuse on principle alone.

BitcoinTalk

Re: 4 hashes parallel on SSE2 CPUs for 0.3.6

2010-08-07 21:16:01 UTC - -

Quote from: impossible7 on August 06, 2010, 11:37:20 AM

CRITICAL_BLOCK is a macro that contains a for loop. The assertion failure indicates that break has been called inside the body of the loop. The only break statement in this block is in line 2762. In the original source file, there is no break statement in this critical block. I think you must remove lines 2759-2762. The is nothing like that in the original main.cpp.

Sorry about that. CRITICAL_BLOCK isn't perfect. You have to be careful not to break or continue out of it. There's an assert that catches and warns about break. I can be criticized for using it, but the syntax would be so much more bloated and error prone without it.

Is there a chance the SSE2 code is slow on Intel because of some quirk that could be worked around? For instance, if something works but is slow if it's not aligned, or thrashing the cache, or one type of instruction that's really slow? I'm not sure how available it is, but I think Intel used to have a profiler for profiling on a per instruction level. I guess if tcatm doesn't have a system with the slow processor to test with, there's not much hope. But it would be really nice if this was working on most CPUs.

BitcoinTalk

Re: bitcoin generation broken in 0.3.8?

2010-08-09 18:50:41 UTC - -

I found that SSE2 only added a slight 2% speedup, which didn't seem worth the incompatibility. I was trying to take the safer option.

It doesn't look to me like Crypto++ could be deciding whether to use SSE2 at runtime. There's one place where it detects SSE2 for deciding some block count parameter, but the SSE2 stuff is all #ifdef at compile time and I can't see how that would switch at runtime. Maybe I'm not looking in the right place.

Should we enable SSE2 in all the makefiles? It seems like we must in case someone compiles with 64-bit.

I will recompile the 64-bit part of the Linux 0.3.8 release.

BitcoinTalk

Version 0.3.8.1 update for Linux 64-bit

2010-08-09 19:46:58 UTC - -

When we switched to Crypto++ 5.6.0 SHA-256 in version 0.3.6, generation got broken on the Linux 64-bit build. Version 0.3.8.1 is on SourceForge with the 64-bit binary updated.

Download:
http://sourceforge.net/projects/bitcoin/files/Bitcoin/bitcoin-0.3.8/bitcoin-0.3.8.1-linux.tar.gz/download

Future versions after 0.3.8 will probably require SSE2. Anyone have Pentium 3 or older where this would be a problem?

BitcoinTalk

Re: What could be the transition plan to Y2038 compliant Bitcoin?

2010-08-09 20:13:26 UTC - -

unsigned int is good until 2106. Surely the network will have to be totally revamped at least once by then.

There should not be any signed int. If you've found a signed int somewhere, please tell me (within the next 25 years please) and I'll change it to unsigned int.

BitcoinTalk

Re: bitcoin generation broken in 0.3.8? (64-bit)

2010-08-09 20:34:06 UTC - -

I uploaded 0.3.8.1 for Linux with re-built 64-bit. I ran a difficulty 1 test with it and it has generated blocks.

http://BitcoinTalk.org/index.php?topic=765.0

Download:
http://sourceforge.net/projects/bitcoin/files/Bitcoin/bitcoin-0.3.8/bitcoin-0.3.8.1-linux.tar.gz/download

BitcoinTalk

Re: Version 0.3.8.1 update for Linux 64-bit

2010-08-09 20:55:06 UTC - -

That's a good point, I believe you could run with generation off if you don't have SSE2.

How about add to the top of cryptopp/config.h:

```
#if !defined(_M_X64) && !defined(__x86_64__)
#define CRYPTOPP_DISABLE_SSE2  1
#endif
```

that would disable SSE2 for 32-bit builds. (at least with GCC or MSVC)

BitcoinTalk

Connection limits

2010-08-09 20:58:45 UTC - -

SVN rev 125:
- Always make 8 outbound connections even if have 8 inbound
- Limit outbound connections to one per a.b.?.? range
- Switch -maxconnections=#

I added the (currently undocumented) switch -maxconnections=#. You shouldn't use it unless you need to because your router can't maintain a lot of connections, then try -maxconnections=30.

I haven't really tested -maxconnections much, could someone test it?

BitcoinTalk

Re: Bitcoin minting is thermodynamically perverse

2010-08-09 21:28:39 UTC - -

The heat from your computer is not wasted if you need to heat your home. If you're using electric heat where you live, then your computer's heat isn't a waste. It's equal cost if you generate the heat with your computer.

If you have other cheaper heating than electric, then the waste is only the difference in cost.

If it's summer and you're using A/C, then it's twice.

Bitcoin generation should end up where it's cheapest. Maybe that will be in cold climates where there's electric heat, where it would be essentially free.

BitcoinTalk

Re: Version 0.3.8.1 update for Linux 64-bit

2010-08-10 23:46:00 UTC - -

SVN rev 128: disable SSE2 on 32-bit. This may only disable it for MSVC and GCC. Other compilers might have different 64-bit defines.

BitcoinTalk

Re: Not a suggestion

2010-08-11 00:14:22 UTC - -

This is a very interesting topic. If a solution was found, a much better, easier, more convenient implementation of Bitcoin would be possible.

Originally, a coin can be just a chain of signatures. With a timestamp service, the old ones could be dropped eventually before there's too much backtrace fan-out, or coins could be kept individually or in denominations. It's the need to check for the absence of double-spends that requires global knowledge of all transactions.

The challenge is, how do you prove that no other spends exist? It seems a node must know about all transactions to be able to verify that. If it only knows the hash of the in/outpoints, it can't check the signatures to see if an outpoint has been spent before. Do you have any ideas on this?

It's hard to think of how to apply zero-knowledge-proofs in this case.

We're trying to prove the absence of something, which seems to require knowing about all and checking that the something isn't included.

BitcoinTalk

Re: Escrow

2010-08-11 01:30:02 UTC - -

Quote from: jgarzik on August 10, 2010, 06:53:57 PM

Ask some real-world business owners if they want to tell their customers about the chance of the money being lost forever, unrecoverable by either party.

That makes it sound like it might somehow get lost and the parties can't get it even if they want to cooperate.

When you pay for something up front, you can't get it back either. Consumers seem comfortable with that. It's no worse than that.

Either party always has the option to release it to the other.

> ***Quote from: nelisky on August 10, 2010, 08:20:36 PM***
> *But the money burning solution, while great at preventing economically viable fraud, does nothing to prevent revenge and actually makes everyone loose if one side is dishonest. I would certainly not endorse that.*

Then you must also be against the common system of payment up front, where the customer loses.

Payment up front: customer loses, and the thief gets the money.
Simple escrow: customer loses, but the thief doesn't get the money either.

Are you guys saying payment up front is better, because at least the thief gets the money, so at least someone gets it?

Imagine someone stole something from you. You can't get it back, but if you could, if it had a kill switch that could be remote triggered, would you do it? Would it be a good thing for thieves to know that everything you own has a kill switch and if they steal it, it'll be useless to them, although you still lose it too? If they give it back, you can re-activate it.

Imagine if gold turned to lead when stolen. If the thief gives it back, it turns to gold again.

It still seems to me the problem may be one of presenting it the right way. For one thing, not being so blunt about "money burning" for the purposes of game theory discussion. The money is never truly burned. You have the option to release it at any time forever.

BitcoinTalk

Re: Compile error in SVN r127

2010-08-11 01:42:30 UTC - -

Updated SVN. Thanks.

There's little hope of not repeatedly stumbling over that in the future. It doesn't break the compile for me.

BitcoinTalk

Re: Not a suggestion

2010-08-11 21:07:59 UTC - -

Still thinking this idea through...

The only job the network needs to do is to tell whether a spend of an outpoint is the first or not.

If we're willing to have clients keep the history for their own money, then some of the information may not need to be stored by the network, such as:
- the value
- the association of inpoints and outpoints in one transaction

The network would track a bunch of independent outpoints. It doesn't know what transactions or amounts they belong to. A client can find out if an outpoint has been spent, and it can submit a satisfying inpoint to mark it spent. The network keeps the outpoint and the first valid inpoint that proves it spent. The inpoint signs a hash of its associated next outpoint and a salt, so it can privately be shown that the signature signs a particular next outpoint if you know the salt, but publicly the network doesn't know what the next outpoint is.

I believe the clients would have to keep the entire history back to the original generated coins. Someone sending a payment would have to send data to the recipient, as well as still communicating with the network to mark outpoints spent and check that the spend is the first spend. Maybe the data transfer could be done as an e-mail attachment.

The fact that clients have to keep the entire history reduces the privacy benefit. Someone handling a lot of money still gets to see a lot of transaction history. The way it retrospectively fans out, they might end up seeing a majority of the history. Denominations could be made granular to limit fan-out, but a business handling a lot of money might still end up seeing a lot of the history.

BitcoinTalk

Re: Lost large number of bitcoins

2010-08-11 21:46:51 UTC - -

Quote from: sirius-m on August 11, 2010, 02:01:53 AM

I added to the FAQ the warning to back up after each transaction. Is it necessary btw to stop the client before making a backup? That's a bit inconvenient. Automatic backups would be useful indeed.

You can get away with backing up without stopping the client if you don't do anything or receive a payment within a few seconds before the backup. (like 5 seconds)

> ***Quote from: gridecon on August 11, 2010, 08:46:08 PM***
> *Wait, I'm confused again. I thought the essence of the surprise was that Bitcoin is programmed to "empty your wallet" for EACH transaction.*

No, it doesn't usually empty your wallet with each transaction. It uses the smallest set of coins it can find to add up to near the amount. In this case, unfortunately, his wallet had a single 9000 BTC bill in it, and it had to break it to get 1 BTC and 8999 BTC change.

BitcoinTalk

Re: Where is the separate discussion devoted to possible Bitcoin weaknesses.

2010-08-11 22:40:25 UTC - -

It doesn't have to be such a breaking change. New nodes could accept old transactions for a long time until most nodes have already upgraded before starting to refuse transactions without PoW. Or, they could always accept old transactions, but only a limited number per time period.

I've thought about PoW on transactions many times, but usually I end up thinking a 0.01 transaction fee is essentially similar and better. 0.01 is basically a proof of work, but not wasted. But if the problem is validating loads of transactions, then PoW could be checked faster.

A more general umbrella partial solution would be to implement the idea where an unlikely dropoff in blocks received is detected. Then an attacker would still need a substantial portion of the network's power to benefit from a DoS attack.

> ***Quote from: gavinandresen on August 11, 2010, 04:10:56 PM***
> *Bitcoin's p2p network is subject to various kinds of denial of service attacks.*

There, I said it.

+1

Any demonstration tests at this point would only show what we already know, and divert dev time from strengthening the system to operational fire fighting.

BitcoinTalk

Re: Flood attack 0.00000001 BC

2010-08-11 23:28:50 UTC - -

It would be nice to keep the blk*.dat files small as long as we can.

The eventual solution will be to not care how big it gets.

But for now, while it's still small, it's nice to keep it small so new users can get going faster. When I eventually implement client-only mode, that won't matter much anymore.

There's more work to do on transaction fees. In the event of a flood, you would still be able to jump the queue and get your transactions into the next block by paying a 0.01 transaction fee. However, I haven't had time yet to add that option to the UI.

Scale or not, the test network will react in the same ways, but with much less wasted bandwidth and annoyance.

BitcoinTalk

Re: BSD detection

2010-08-12 00:02:06 UTC - -

Quote from: dkaparis on August 11, 2010, 11:00:16 PM
There is this piece of code in headers.h:

```
#ifdef __WXMAC_OSX__
#define __WXMAC__ 1
#define __WXOSX__ 1
#define __BSD__ 1
#endif
#endif
```

That code was a bad idea anyway, I'm deleting it. Any Mac code should only use __WXMAC_OSX__, not __WXMAC__ or __WXOSX__, and we should stop using __BSD__.

Quote

```
#if (defined(__unix__) || defined(unix)) && !defined(USG)
#include <sys/param.h>
#endif
```

Will that definitely cause BSD to be defined on Mac?

BitcoinTalk

Re: Not a suggestion

2010-08-12 02:46:56 UTC - -

Quote from: Red on August 12, 2010, 01:10:19 AM
Quote from: satoshi on August 11, 2010, 09:07:59 PM
I believe the clients would have to keep the entire history back to the original generated coins. The fact that clients have to keep the entire history reduces the privacy benefit.

I thought this too at first. But then I convinced myself otherwise.

Are you back to talking about the existing Bitcoin system here?

I was talking about in the hypothetical system I was describing, if the network doesn't know the values and lineage of the transactions, then it can't verify them and vouch for them, so the clients would have to keep the history all the way back.

If a client wasn't present until recently, the two ways to convince it that a transaction has a valid past is:
1) Show it the entire history back to the original generated coin.
2) Show it a history back to a thoroughly deep block, then trust that if so many nodes all said the history up to then was correct then it must be true.

But if the network didn't know all the values and lineage of the transactions, it couldn't do 2), I don't think.

BitcoinTalk

Re: BSD detection

2010-08-12 21:14:20 UTC - -

This is in SVN rev 130. Check that it compiles right.

Code:
```
#if (defined(__unix__) || defined(unix)) && !defined(USG)
#include <sys/param.h>  // to get BSD define
#endif
#ifdef __WXMAC_OSX__
#ifndef BSD
#define BSD 1
#endif
#endif
```

BitcoinTalk

Bugfixes in SVN rev 130

2010-08-12 21:20:31 UTC - -

Misc bugfixes in rev 130:

fix -datadir with relative path
autostart is now off by default except on windows
fix occasional "vector iterator not dereferencable" assertion when compiled with msvc
fix readlink compile warning on linux build
use sys/param.h and BSD define instead of __BSD__
-paytxfee switch, e.g. -paytxfee=0.01

BitcoinTalk

Re: Bitcoin Watchdog Service

2010-08-12 21:34:44 UTC - -

True, there would probably be someone with a dial-up modem or satellite dish internet. Rarer would be someone who has both that and the wired internet that has the outage, but if it's a big enough segment to matter, out of a million people there's bound to be a multi-home geek.

ISP network cuts are just your local area. If you still have communication with the rest of your area, it would probably be something like 1/1000 of the world or less. Block generation in the segment would take several hours per block.

I favour the plan to monitor if the frequency of blocks received drops too slow. That covers a large range of possibilities.

BitcoinTalk

Re: Having problems specifing -datadir

2010-08-12 21:43:29 UTC - -

Fixed in SVN rev 130.

BitcoinTalk

Re: 4 hashes parallel on SSE2 CPUs for 0.3.6

2010-08-12 22:07:23 UTC - -

That big of a difference in speed, by a factor of 4 or 6, feels like it's likely to be some quirky weak spot or instruction that the old chip is slow with. Unless it's a touted feature of the i5 that they made SSE2 six times faster.

A quick summary:
Xeon Quad 41% slower
Core 2 Duo 55% slower
Core 2 Duo same (vess)
Core 2 Quad 50% slower
Core i5 200% faster (nelisky)
Core i5 100% faster (vess)
AMD Opteron 105% faster

aceat64:
My system went from ~7100 to ~4200.
This particular system has dual Intel Xeon Quad-Core CPUs (E5335) @ 2.00GHz.

impossible7:
on an Intel Core 2 Duo T7300 running x86_64 linux it was 55% slower compared to the stock version (r121)

nelisky:
My Core2Quad (Q6600) slowed down 50%,
my i5 improved ~200%,

impossible7:
on an AMD Opteron 2374 HE running x86_64 linux I got a 105% improvement (!)

BitcoinTalk

Re: Bugfixes in SVN rev 130

2010-08-13 03:15:23 UTC - -

No, that's not what it is.

-paytxfee allows you to include a transaction fee with your transactions. If transaction confirmations become slow, you can get priority by using "-paytxfee=0.01". Any transactions you send would cost an extra 0.01. There's no reason to use more than 0.01.

It's just there in case we need it. It probably won't be needed, and it can be explained more if we do.

BitcoinTalk

Re: Bitcoin Watchdog Service

2010-08-13 17:09:27 UTC - -

Quote
But there will be no irc server to bootstrap from.

Which doesn't matter because you can't access sourceforge to download the software either.

If you've ever been connected before, you don't need IRC to bootstrap anymore. Even if you haven't, you can bootstrap from seed nodes. IRC is completely redundant since 0.3.0.

BitcoinTalk

Version 0.3.9 rc1, please test

2010-08-13 17:40:00 UTC - -

Here's a test build if you'd like to help test before 0.3.9 is released.
(or if you'd rather get upgrading out of the way now instead of waiting)

Downloads: (binaries only)
http://www.bitcoin.org/download/bitcoin-0.3.9.rc1-win32.zip
(http://www.bitcoin.org/download/bitcoin-0.3.9.rc1-linux.tar.gz)

SHA1 a36ea00cce27b4b083755df73a3d1e5e5729884e bitcoin-0.3.9.rc1-win32.zip
SHA1 bbb333b0ea57302740ad1bb9948520d00f884f9d bitcoin-0.3.9.rc1-linux.tar.gz

Edit:
Linux please test rc2 instead. This adds a -4way switch for tcatm's 4-way SSE2. This will only be for Linux:
http://www.bitcoin.org/download/bitcoin-0.3.9.rc2-linux.tar.gz

SHA1 47d9998f7d15fe81234a5c89a542da9d0664df40 bitcoin-0.3.9.rc2-linux.tar.gz

Please report back your results
http://BitcoinTalk.org/index.php?topic=820

BitcoinTalk

Re: Not a suggestion

2010-08-13 19:28:47 UTC - -

I'm not grasping your idea yet. Does it hide any information from the public network? What is the advantage?

If at least 50% of nodes validated transactions enough that old transactions can be discarded, then everyone saw everything and could keep a record of it.

Can public nodes see the values of transactions? Can they see which previous transaction the value came from? If they can, then they know everything. If they can't, then they couldn't verify that the value came from a valid source, so you couldn't take their generated chain as verification of it.

Does it hide the bitcoin addresses? Is that it? OK, maybe now I see, if that's it.

Crypto may offer a way to do "key blinding". I did some research and it was obscure, but there may be something there. "group signatures" may be related.

There's something here in the general area:
http://www.users.zetnet.co.uk/hopwood/crypto/rh/

What we need is a way to generate additional blinded variations of a public key. The blinded variations would have the same properties as the root public key, such that the private key could generate a signature for any one of them. Others could not tell if a blinded key is related to the root key, or other blinded keys from the same root key. These are the properties of blinding. Blinding, in a nutshell, is x = (x * large_random_int) mod m.

When paying to a bitcoin address, you would generate a new blinded key for each use.

Then you need to be able to sign a signature such that you can't tell that two signatures came from the same private key. I'm not sure if always signing a different blinded public key would already give you this property. If not, I think that's where group signatures comes in. With group signatures, it is possible for something to be signed but not know who signed it.

As an example, say some unpopular military attack has to be ordered, but nobody wants to go down in history as the one who ordered it. If 10 leaders have private keys, one of them could sign the order and you wouldn't know who did it.

BitcoinTalk

Re: Proposed change to sendtoaddress API call

2010-08-13 23:39:14 UTC - -

It's too soon to start junking up the API for backward compatibility at all costs.

Just return "<txid>".

BitcoinTalk

Re: 4 hashes parallel on SSE2 CPUs for 0.3.6

2010-08-14 00:49:18 UTC - -

MinGW on Windows has trouble compiling it:

```
g++ -c -mthreads -O2 -w -Wno-invalid-offsetof -Wformat -g -D__WXDEBUG__ -
DWIN32 -D__WXMSW__ -D_WINDOWS -DNOPCH -I"/boost" -I"/db/build_unix" -
I"/openssl/include" -I"/wxwidgets/lib/gcc_lib/mswud" -I"/wxwidgets/include" -msse2 -
O3 -o obj/sha256.o sha256.cpp

sha256.cpp: In function `long long int __vector__ Ch(long long int __vector__, long long
int __vector__, long long int __vector__)':
sha256.cpp:31: internal compiler error: in perform_integral_promotions, at
cp/typeck.c:1454
Please submit a full bug report,
with preprocessed source if appropriate.
See <URL:http://www.mingw.org/bugs.shtml> for instructions.
make: *** [obj/sha256.o] Error 1
```

BitcoinTalk

Re: 4 hashes parallel on SSE2 CPUs for 0.3.6

2010-08-14 04:22:29 UTC - -

If you haven't already, try aligning thash. It might matter. Couldn't hurt.

Quote from: tcatm on August 14, 2010, 12:53:07 AM

Looks like we're triggering a compiler bug in the tree optimizer. Can you try to compile it -O0?

No help from -O0, same error.

MinGW is GCC 3.4.5. Probably the problem.

I'll see if I can get a newer version of MinGW.

BitcoinTalk

Re: 4 hashes parallel on SSE2 CPUs for 0.3.6

2010-08-14 17:55:37 UTC - -

Got the test working on 32-bit with MinGW GCC 4.5. Exactly 50% slower than stock with Core 2.

BitcoinTalk

Re: 4 hashes parallel on SSE2 CPUs for 0.3.6

2010-08-14 22:06:13 UTC - -

MinGW GCC 4.5.0:
Crypto++ doesn't work, X86_SHA256_HashBlocks() never returns
I only got 4-way working with test.cpp but not when called by BitcoinMiner

MinGW GCC 4.4.1:
Crypto++ works
4-way SIGSEGV

GCC is definitely not aligning __m128i.

Even if we align our own __m128i variables, the compiler may decide to use a __m128i behind the scenes as a temporary variable.

By making our __m128i variables aligned and changing these inlines to defines, I was able to get it to work on 4.4.1 with -O0 only:
```
#define Ch(b, c, d)  ((b & c) ^ (~b & d))
#define Maj(b, c, d)  ((b & c) ^ (b & d) ^ (c & d))
#define ROTR(x, n) (_mm_srli_epi32(x, n) | _mm_slli_epi32(x, 32 - n))
#define SHR(x, n)  _mm_srli_epi32(x, n)
```

But that's with -O0.

BitcoinTalk

Re: 4 hashes parallel on SSE2 CPUs for 0.3.6

2010-08-15 03:40:29 UTC - -

On both MinGW GCC 4.4.1 and 4.5.0 I have it working with test.cpp but SIGSEGV when called by BitcoinMiner. So now it doesn't look like it's the version of GCC, it's something else, maybe just the luck of how the stack is aligned.

I have it working fine on GCC 4.3.3 on Ubuntu 32-bit.

I found the problem with Crypto++ on MinGW 4.5.0. Here's the patch for that:

Code:

```
--- \old\sha.cpp Mon Jul 26 13:31:11 2010
+++ ew\sha.cpp Sat Aug 14 20:21:08 2010
@@ -336,7 +336,7 @@
  ROUND(14, 0, eax, ecx, edi, edx)
  ROUND(15, 0, ecx, eax, edx, edi)

- ASL(1)
+    ASL(label1)   // Bitcoin: fix for MinGW GCC 4.5
  AS2(add WORD_REG(si), 4*16)
  ROUND(0, 1, eax, ecx, edi, edx)
  ROUND(1, 1, ecx, eax, edx, edi)
@@ -355,7 +355,7 @@
  ROUND(14, 1, eax, ecx, edi, edx)
  ROUND(15, 1, ecx, eax, edx, edi)
  AS2( cmp  WORD_REG(si), K_END)
- ASJ( jne, 1, b)
+    ASJ(    jne,    label1,  )   // Bitcoin: fix for MinGW GCC 4.5

  AS2( mov  WORD_REG(dx), DATA_SAVE)
  AS2( add  WORD_REG(dx), 64)
```

BitcoinTalk

tcatm's 4-way SSE2 for Linux 32/64-bit is in 0.3.10

2010-08-15 15:52:09 UTC - -

0.3.10 has tcatm's 4-way SSE2 as an option switch.

Use the switch "-4way" to turn it on. Without the switch you get Crypto++ ASM SHA-256.

I could only get this working with Linux.

Download:
Get 0.3.10 from http://BitcoinTalk.org/index.php?topic=827.0

Please report back your CPU and results! I think it's pretty clear that Core 2 and lower are slower, i5 faster. I don't think we've heard any i7 results yet. We need to know about the different models of AMD or other less common CPUs.

BitcoinTalk

Re: Potential disaster scenario

2010-08-15 16:37:16 UTC - -

Some places where generation will gravitate to:
1) places where it's cheapest or free
2) people who want to help for idealogical reasons
3) people who want to get some coins without the inconvenience of doing a transaction to buy them

There are legitimate places where it's free. Generation is basically free anywhere that has electric heat, since your computer's heat is offsetting your baseboard electric heating. Many small flats have electric heat out of convenience.

How expensive is heating oil? With the price of oil so high, if it's actually more expensive than electric, then generating would have negative cost.

There's also kids putting it on their parent's power bill, employees their employer, botnets, etc.

Case 3 comes into play for small amounts. The overhead of doing an exchange doesn't make sense if you just need a small bit of pocket change for incidental micropayments. I think this is a nice advantage vs fiat currency, instead of all the seigniorage going to one big entity, let it go in convenience amounts to people who need to scrape up a small amount of change.

BitcoinTalk

Re: Version 0.3.9 rc1, please test

2010-08-15 18:11:41 UTC - -

Quote from: jgarzik on August 15, 2010, 05:46:27 PM
the extended-help might have been based on my idea, but the code was somewhat different.

The idea was the main part. When you posted your patch, I realized it should have been done that way instead of "-?". I always had reservations about "-?" because it intrudes on the possible parameter values, and the help response is based on the version of the caller instead of the server.

BitcoinTalk

Re: tcatm's 4-way SSE2 for Linux 32/64-bit 0.3.9 rc2

2010-08-15 18:23:26 UTC - -

I hope someone can test an i5 or AMD to check that I built it right. I don't have either to test with.

I'm also curious if it performs much worse on 32-bit linux vs 64-bit.

BitcoinTalk

Re: tcatm's 4-way SSE2 for Linux 32/64-bit 0.3.9 rc2

2010-08-15 18:43:27 UTC - -

I just uploaded a quick build so testers can check if I built it right. (I don't have an i5 or AMD) If it checks out, I'll put together the full package and do all the release stuff.

bitcoin-list

[**bitcoin-list**] ALERT - we are investigating a problem

2010-08-15 20:38:33 UTC - -

*** WARNING *** We are investigating a problem. DO NOT TRUST ANY TRANSACTIONS THAT HAPPENED AFTER 15.08.2010 17:05 UTC (block 74638) until the issue is resolved.

BitcoinTalk

Re: overflow bug SERIOUS

2010-08-15 20:59:09 UTC - -

Here's the preliminary change. Look right? I have more changes to make, this isn't all of it. Will SVN shortly.

Code:

```
    bool CheckTransaction() const
{
  // Basic checks that don't depend on any context
  if (vin.empty() || vout.empty())
    return error("CTransaction::CheckTransaction() : vin or vout empty");

  // Check for negative and overflow values
```

```
    int64 nTotal = 0;
    foreach(const CTxOut& txout, vout)
    {
        if (txout.nValue < 0)
            return error("CTransaction::CheckTransaction() : txout.nValue negative");
        if (txout.nValue > 21000000 * COIN)
            return error("CTransaction::CheckTransaction() : txout.nValue too high");
        nTotal += txout.nValue;
        if (nTotal > 21000000 * COIN)
            return error("CTransaction::CheckTransaction() : txout total too high");
    }

    if (IsCoinBase())
    {
        if (vin[0].scriptSig.size() < 2 || vin[0].scriptSig.size() > 100)
            return error("CTransaction::CheckTransaction() : coinbase script size");
    }
    else
    {
        foreach(const CTxIn& txin, vin)
            if (txin.prevout.IsNull())
                return error("CTransaction::CheckTransaction() : prevout is null");
    }

    return true;
}
```

Don't sticky the topic, nobody looks up there. There'll be enough posts to bump.

BitcoinTalk

Re: overflow bug SERIOUS

2010-08-15 21:06:45 UTC - -

It would help if people stop generating. We will probably need to re-do a branch around the current one, and the less you generate the faster that will be.

A first patch will be in SVN rev 132. It's not uploaded yet. I'm pushing some other misc changes out of the way first, then I'll upload the patch for this.

BitcoinTalk

Re: overflow bug SERIOUS

2010-08-15 21:23:55 UTC - -

Once you have an update, you could download knightmb's block chain. You'll want one that's old enough that it ends *before* block 74000 so the most recent security lockin will check it. Can someone find the link for that?

BitcoinTalk

Re: overflow bug SERIOUS

2010-08-15 21:40:19 UTC - -

Patch is uploaded to SVN rev 132!

For now, recommended steps:
1) Shut down.
2) Download knightmb's blk files. (replace your blk0001.dat and blkindex.dat files)
3) Upgrade.
4) It should start out with less than 74000 blocks. Let it redownload the rest.

If you don't want to use knightmb's files, you could just delete your blk*.dat files, but it's going to be a lot of load on the network if everyone is downloading the whole block index at once.

I'll build releases shortly.

BitcoinTalk

Re: overflow bug SERIOUS

2010-08-15 22:58:08 UTC - -

Don't update the block chain download. When you take someone's block chain download, you don't want it right up to the end. A somewhat old one is better so it can download and verify the most recent blocks.

tcatm's 4-way SSE2 SHA-256 is in the file sha256.cpp and already uploaded a few revs ago.

I just now uploaded rev 134 which is the makefile.unix that enables building with it on Linux. If you build rev 134 on Linux now you'll get the -4way switch.

If you have problems building because of it, then edit makefile.unix and:
- remove -DFOURWAYSSE2

- remove obj/sha256.o from the end of these lines:
bitcoin: $(OBJS) obj/ui.o obj/uibase.o obj/sha256.o
bitcoind: $(OBJS:obj/%=obj/nogui/%) obj/sha256.o

The 0.3.10 linux build *will* have the -4way option when I build it.

Here are the patch downloads for Windows:

http://www.bitcoin.org/download/bitcoin-0.3.10-win32-setup.exe
http://www.bitcoin.org/download/bitcoin-0.3.10-win32.zip

SHA1 16645ec5fcdb35bc54bc7195309a1a81105242bb bitcoin-0.3.10-win32-setup.exe
SHA1 4f35ad7711a38fe8c880c6c9beab430824c426d3 bitcoin-0.3.10-win32.zip

Steps:
1) Shut down.
2) Download knightmb's blk files and replace your blk0001.dat and blkindex.dat files.
http://knightmb.dyndns.org/files/bitcoin/blocks/
http://rapidshare.com/files/413168038/BitcoinBlocks.torrent
3) Upgrade to 0.3.10.
4) It should start out with less than 74000 blocks and redownload the rest.

Or if you don't want to mess with downloading blk files, you can just do this:

1) Shut down.
2) Delete (or move) blk*.dat
3) Upgrade to 0.3.10.
4) It redownloads all blocks, probably take about an hour.

BitcoinTalk

Re: overflow bug SERIOUS

2010-08-15 23:17:24 UTC - -

Quote from: knightmb on August 15, 2010, 10:59:04 PM
[edit] *Just saw your post, I'll build one to less than 74,000 then, should at least save you technical people a few minutes of downloading the new chain.* 😉

Just leave the old one alone! Older is better. What block number is it? Anywhere from 60000-74000 is good. The one that you've had available for a while has been vetted and is the best choice.

BitcoinTalk

Re: overflow bug SERIOUS

2010-08-15 23:36:10 UTC - -

Starting at 67000 is *perfect*.

Yeah, at the moment you'll stop at 74638. It should start slowly creeping up as more nodes upgrade and generate.

Linux build links below.

The Linux version includes tcatm's 4-way SSE2 SHA-256 that makes generating faster on i5 and AMD CPU's. Use the "-4way" switch to enable it and check if it's faster for you.

Download links:
http://www.bitcoin.org/download/bitcoin-0.3.10-win32-setup.exe
http://www.bitcoin.org/download/bitcoin-0.3.10-win32.zip
http://www.bitcoin.org/download/bitcoin-0.3.10-linux.tar.gz

SHA1 16645ec5fcdb35bc54bc7195309a1a81105242bb bitcoin-0.3.10-win32-setup.exe
SHA1 4f35ad7711a38fe8c880c6c9beab430824c426d3 bitcoin-0.3.10-win32.zip
SHA1 e3fda1ddb31b0d5c35156cacd80dee6ea6ae6423 bitcoin-0.3.10-linux.tar.gz

BitcoinTalk

Re: overflow bug SERIOUS

2010-08-15 23:37:07 UTC - -

Quote from: Joozero on August 15, 2010, 11:32:43 PM
I think that you should add something about this: http://BitcoinTalk.org/index.php?topic=259.0
There must be a label on the client that show a warning message if needed
Now everyone have always to check the website, and I think that this is bad.

Agree, wanted to do that for a long time, haven't had time to do it.

For now, you could also subscribe to the **bitcoin-list** mailing list. It rarely gets used except for announcements like this and major new versions.

Subscribe/unsubscribe page:
http://lists.sourceforge.net/mailman/listinfo/**bitcoin-list**

BitcoinTalk

Version 0.3.10 - block 74638 overflow PATCH!

2010-08-15 23:48:22 UTC - -

Version 0.3.10 patches the block 74638 overflow bug. http://BitcoinTalk.org/index.php?topic=823

The Linux version includes tcatm's 4-way SSE2 SHA-256 that makes generating faster on i5, i7 (with hyperthreading) and AMD CPU's. Try the "-4way" switch to enable it and check if it's faster for you.

Download from sourceforge:
http://sourceforge.net/projects/bitcoin/files/Bitcoin/bitcoin-0.3.10/

SHA1 16645ec5fcdb35bc54bc7195309a1a81105242bb bitcoin-0.3.10-win32-setup.exe
SHA1 4f35ad7711a38fe8c880c6c9beab430824c426d3 bitcoin-0.3.10-win32.zip
SHA1 e3fda1ddb31b0d5c35156cacd80dee6ea6ae6423 bitcoin-0.3.10-linux.tar.gz
SHA1 b812ccff4881778b9090f7c0b0255bcba7b078ac bitcoin-0.3.10-macosx.zip

It is no longer necessary to delete blk*.dat. The good block chain has overtaken the bad block chain, so you can just upgrade and it'll automatically reorg away the bad block chain.

BitcoinTalk

Re: 0.3.10.1 Question on where block should be

2010-08-16 00:28:28 UTC - -

I suspect there's some difficulty receiving blocks if all the nodes you're connected to are 0.3.9 or lower. We need enough of us so that at least one node you connect to will be 0.3.10. The problem will start to go away when we make up more than 1/8th of the network.

It'll help if you port forward so you can get lots of connections.

BitcoinTalk

Re: 0.3.10.1 Question on where block should be

2010-08-16 00:37:20 UTC - -

For now, can some people running 0.3.10 with static IP who can receive incoming connections post their IP? Then we can -addnode= them and make sure to connect to at least one 0.3.10 node.

BitcoinTalk

Re: overflow bug SERIOUS

2010-08-16 01:00:45 UTC - -

Quote from: Ground Loop on August 16, 2010, 12:29:55 AM
*Question about fallout: I had a **transaction** that I submitted after the bad block, using the bad block chain.*

What is the status of that transaction?
From what I can tell, my (updated) sending client wallet shows the deducted amount.

Will it get reincorporated into the fixed chain, and will the recipient be able to spend it?

Right, it will get reincorporated into the fixed chain. The transaction won't disappear, it'll still be visible on both sides, but the confirmation count will jump back to 0 and start counting up again.

It's only if you generated a block in the bad chain after block 74638 that the 50 BTC from that will disappear. Any blocks in the bad chain wouldn't have matured yet.

BitcoinTalk

Re: overflow bug SERIOUS

2010-08-16 01:02:24 UTC - -

Quote from: kosovito on August 16, 2010, 12:39:17 AM
I did all steps, now my client is 0.3.10 and it stopped at block 74638. Is all fine?

If you still show 74638 blocks then you aren't connected to any 0.3.10 nodes.

For today, try adding these parameters:
-addnode=75.158.131.108 -addnode=99.27.237.13 -addnode=68.68.99.14

See
http://BitcoinTalk.org/index.php?topic=828

BitcoinTalk

Re: overflow bug SERIOUS

2010-08-16 01:12:05 UTC - -

Quote from: trebronics on August 16, 2010, 01:02:35 AM

Most people running clients are not reading this message thread. So... Silly questions:

1) How will this continue to affect version 3.8.1 (pre-catastrophe) clients with bad block chain?
2) How will this affect clients that upgrade to 3.8.10 but don't remove their block chain files?

1) Once more than 50% of the node power is upgraded and the good chain overtakes the bad, the 0.3.10 nodes will make it hard for any bad transactions to get any confirmations.
2) If you didn't remove your blk*.dat files, you're not helping to contribute to that 50%, and you'll still show bad transactions until the good chain overtakes the bad chain.

BitcoinTalk

Re: overflow bug SERIOUS

2010-08-16 02:16:10 UTC - -

The bad chain is also slowed down as more nodes upgrade.

We've already generated 14 blocks since 74638. The builds of 0.3.10 were uploaded about 2 and 3 hours ago. Of the nodes I'm connected to, more than half are already 0.3.10. I would say we probably already have more power than the bad chain.

BitcoinTalk

Re: overflow bug SERIOUS

2010-08-16 02:38:21 UTC - -

On Windows, findstr /c:"version message" debug.log

It looks like the bad chain was on block 74678 recently. Can't wait to overtake it.

On the stats at http://nullvoid.org/bitcoin/statistix.php there's been 5 blocks per hour in the last 3 hours. We had a difficulty adjustment about a day ago that should have put it back to 6 blocks per hour.

Re: tcatm's 4-way SSE2 for Linux 32/64-bit 0.3.9 rc2

2010-08-16 02:57:57 UTC - -

Quote from: tcatm on August 16, 2010, 12:43:39 AM
I propose to compile sha256.cpp with -O3 -march=amdfamk10 (will work on 32bit and 64bit) as only CPUs supporting this instruction set (AMD Phenom, Intel i5 and newer) benefit from -4way and it'll improve performance by ~9%.

GCC 4.3.3 doesn't support -march=amdfamk10. I get:
sha256.cpp:1: error: bad value (amdfamk10) for -march= switch

Quote from: NewLibertyStandard on August 16, 2010, 01:49:01 AM
With 4way, I get significantly better performance when I have all my virtual cores enabled. I think I get about the same amount of hashes when hyper threading is turned off with or without 4way.

Hey, you may be onto something!

hyperthreading didn't help before because all the work was in the arithmetic and logic units, which the hyperthreads share.

tcatm's SSE2 code must be a mix of normal x86 instructions and SSE2 instructions, so while one is doing x86 code, the other can do SSE2.

How much of an improvement do you get with hyperthreading?

Some numbers? What CPU is that?

BitcoinTalk

Re: tcatm's 4-way SSE2 for Linux 32/64-bit 0.3.9 rc2

2010-08-16 03:23:04 UTC - -

Quote from: Vasiliev on August 16, 2010, 03:17:07 AM
try -march=amdfam10

That works.

That's strange... are we sure that's the same thing? tcatm, try amdfam10 and make sure you get the same speed measurement.

BitcoinTalk

Re: tcatm's 4-way SSE2 for Linux 32/64-bit is in 0.3.10

2010-08-16 04:36:59 UTC - -

Quote from: jgarzik on August 16, 2010, 03:35:28 AM

Code:

cpu family : 6
model : 26
model name : Genuine Intel(R) CPU 000 @ 3.20GHz
stepping : 4

cpu family 6 model 26 stepping 4 is an Intel Core i7.
That's a 23% speedup with -4way, 63% total speedup with -4way + hyperthreading.
33% faster with hyperthreading than without it.

BitcoinTalk

Re: overflow bug SERIOUS

2010-08-16 12:59:38 UTC - -

It looks like we overtook the bad chain somewhere around 74689. 0.3.9 and lower nodes have been responding with the current block number for some hours now.

That means it's no longer necessary to delete blk*.dat before upgrading. You can just upgrade and it'll reorg away the bad block chain.

Thanks to everyone for the quick response!

BitcoinTalk

Re: tcatm's 4-way SSE2 for Linux 32/64-bit is in 0.3.10

2010-08-16 13:38:01 UTC - -

I wrapped sha256.cpp in
#ifdef FOURWAYSSE2
#endif // FOURWAYSSE2

try it now.

BitcoinTalk

Re: [PATCH] Automatic block validation

2010-08-16 15:25:54 UTC - -

That's a difficult approach.

We need to cause a reorg, which will disconnect the invalid chain.

This is code that will rarely ever get tested, and is fairly intricate, so something simple and safe is best.

Here's what I was thinking of. (I haven't tested this yet) It checks all the blocks in the main chain. If it finds a bad one, it sets all that chain's bnChainWork to 0 so it can't win best chain again, and it reduces best chain work to the fork level so any new block after the fork will cause a reorg. (It can't change pindexBest without actually doing a reorg)

This isn't perfect yet. It still needs to receive one valid block to trigger the reorg.

It would probably be possible to initiate an AddToBlockIndex or Reorganize after the check, but it would require a lot more careful attention. I probably should break out part of AddToBlockIndex that sets the new best block. I'll probably end up doing that instead of the code below.

Code:

```
bool CTxDB::LoadBlockIndex()
{
    ...

    // Verify blocks in the main chain
    vector<CBlockIndex*> vChain;
    for (CBlockIndex* pindex = pindexBest; pindex && pindex->pprev; pindex = pindex->pprev)
    {
        vChain.push_back(pindex);
        CBlock block;
        if (!block.ReadFromDisk(pindex))
            return error("LoadBlockIndex() : block.ReadFromDisk failed");
        if (!block.CheckBlock())
        {
            bnBestChainWork = pindex->pprev->bnChainWork;
            foreach(CBlockIndex* pindex2, vChain)
                pindex2->bnChainWork = 0;
        }
    }

    return true;
}
```

BitcoinTalk

blocks minus 1

2010-08-16 15:59:25 UTC - -

I'd like to reduce the number of blocks displayed in the status bar by 1. When you first load the program, it'll display 0 blocks instead of 1:
"0 connections 0 blocks 0 transactions"

It's always been "nBestHeight + 1" because it's counting the genesis block. Technically, yes, the genesis block is a block. It's a hardcoded block that you start out with. You can't *not* have the genesis block. Maybe think of it as a reference coin that you measure other coins against. The block count people are looking for is the number of blocks they've downloaded.

The main benefit is that blocks will be equal to the block number of the current best block. If blocks is 10, then the highest block number you have is 10. It means you have block 10 and you don't have block 11.

It would reduce the confusion we had here:

> ***Quote from: kencausey on August 15, 2010, 11:45:26 PM***
> ***Quote from: davidonpda on August 15, 2010, 11:31:37 PM***
> *... It already is on block 74638. I assume that means that block is now a good one?*

I had some confusion on this myself and got clarification in #bitcoin-dev:

The bad block was number 74638, the last good one was 74637. The numbers start at 0, so when your client shows there are 74638 blocks then that means you have up to block number 74637, the last good one.

BitcoinTalk

Re: blocks minus 1

2010-08-16 17:06:27 UTC - -

Done in SVN rev 137

BitcoinTalk

Re: [PATCH] Automatic block validation

2010-08-16 17:08:02 UTC - -

Quote from: satoshi on August 16, 2010, 03:25:54 PM

It would probably be possible to initiate an AddToBlockIndex or Reorganize after the check, but it would require a lot more careful attention. I probably should break out part of AddToBlockIndex that sets the new best block. I'll probably end up doing that instead of the code below.

This is what I ended up doing in SVN rev 139.

Instead of deleting the bad chain, I added an extra CheckBlock to ConnectBlock so bad blocks can't get back into the best chain once they're kicked out.

BitcoinTalk

Checking the block chain on load

2010-08-16 20:07:46 UTC - -

SVN rev 139 does a basic check of the block chain after loading.

With this we wouldn't have needed to delete blk*.dat, it would have automatically done a reorg back to the fork. There wasn't time to do a careful implementation of this at the time.

It might take longer than we want, since it has to load all the blocks. If it's too slow, we could have it only go back to a certain block number.

BitcoinTalk

Re: checkpointing the block chain

2010-08-16 20:20:53 UTC - -

There is no way for the software to automatically know if one chain is better than another except by the greatest proof-of-work. In the design it was necessary for it to switch to a longer chain no matter how far back it has to go.

The only exception to that is the manual checkpoints I've added. If it weren't for those, it would be able to reorg all the way back to the first block.

BitcoinTalk

Re: overflow bug SERIOUS

2010-08-16 22:54:55 UTC - -

Un-upgraded nodes have the correct chain most of the time, but they are still trying to include the overflow transaction in every block, so they're continually trying to fork and generate invalid blocks. If an old version node is restarted, its transaction pool is emptied, so it may generate valid blocks for a while until the transaction gets broadcast again. 0.3.9 and lower nodes still must upgrade.

The SVN now has the code we needed to automatically reorg the block chain without having to delete the blk*.dat files manually. I knew I couldn't write that code fast and carefully enough yesterday, so I went with the quick manual option.

BitcoinTalk

Re: checkpointing the block chain

2010-08-16 23:01:48 UTC - -

Quote from: NewLibertyStandard on August 16, 2010, 10:42:28 PM
How is the strength of the chain calculated?

Total proof-of-work.

BitcoinTalk

Re: New screenshots to the front page?

2010-08-18 16:58:44 UTC - -

Definitely. The old screenshots of 0.1 are very outdated.

Windows Aero is a good choice. Windows is still the largest user group. Mind what's behind it for the transparent parts.

What to have displayed in the transaction list? Not completely filled up with stuff, just a few things.

BitcoinTalk

Re: Difficulty: More nodes active, or faster nodes?

2010-08-18 18:01:40 UTC - -

The performance numbers posted from a VIA C7's hardware SHA-256 weren't astronomical. Only in the 1500 khash/s range. If you think about it, just because it's implemented in hardware doesn't mean it's crazy fast. It still has to do all the steps. It's only if simplifying it down to single-purpose hardware makes it small enough to fit many in parallel. That's not necessarily easy or a given.

BitcoinTalk

Re: Checking the block chain on load

2010-08-18 18:28:28 UTC - -

In the next SVN rev, I'll make it only go back to the last checkpoint at block 74000. If we need to correct a problem in the future, we can always make sure it goes back at least as far back as the problem. Also, I'm adding code to verify the block index, which means the proof-of-work chain is checked.

Still, the system won't be entirely secure against your blk*.dat files. You are trusting someone if you use a copy of their blk files.

BitcoinTalk

Re: Convert Bitcoin to GTK: Yes? No? wx is better?

2010-08-19 18:44:36 UTC - -

Quote from: BioMike on August 19, 2010, 08:05:18 AM

WxWidgets is not really a problem. My problem is the version that is used (2.9), which is considered unstable by many distro packagers (although the WxWidgets devs say it isn't). On the other side, as far as I know WxWidgets uses gtk under Linux for drawing the whole stuff and makes it for the bitcoins devs easy to make things cross platform.

wxWidgets 2.9 is their first UTF-8 version. We are UTF-8 on all platforms including Windows.

The distro packages of 2.8 are UTF-16, so they just trip people up. People had endless build problems with 2.8 and its wxString UTF-16/ANSI conditional build options until we standardized on 2.9. Also, to use 2.8, we were using ANSI, which was just a temporary stopgap until wxWidgets supported UTF-8.

This is a problem that will solve itself. With time, 2.9 will become a more mainline release.

BitcoinTalk

Re: HOWTO: Compiling Bitcoin on Ubuntu 10.04 (Karmic)

2010-08-19 18:55:48 UTC - -

That's a really well written walkthough. Someone should confirm if they followed it and didn't run into any snags.

BitcoinTalk

Re: tcatm's 4-way SSE2 for Linux 32/64-bit is in 0.3.10

2010-08-19 19:07:43 UTC - -

Quote from: Ground Loop on August 18, 2010, 11:14:26 PM
Any non-Mac i5 love?
Windows i5 64-bit got slower here.

That's the first I've heard anyone say i5 was slower. Everyone else has said 4way was faster on i5. Moreso with hyperthreading enabled.

Quote from: nelisky on August 18, 2010, 11:02:25 PM
And i5, at least on my macbookpro

Good, so I take it that's a confirmation that it's working on Mac as well?

Laszlo told me he did compile in the -4way stuff on Mac, so the -4way switch is also available to try on Mac. I don't think makefile.osx on SVN has it yet, just the built version.

BitcoinTalk

Re: 28 days without generation, i have 4200khash/s

2010-08-19 19:40:30 UTC - -

Make sure your computer's date and time are correct.

BitcoinTalk

Need a post writing up some things users should know

2010-08-19 20:14:01 UTC - -

I'm not sure what to call it, but we could use a post that lists these things users should know. If someone has time to write it, here's the list:

- Make sure your clock is set correctly.

- Microsoft Security Essentials. This never got written up proper.

- Warning not to mess around with your wallet.dat file. It's a database file, it's not as simple as you think. In this Beta version, we haven't had time to try and tinker-proof it yet. It may not work as expected if you start swapping it around.

BitcoinTalk

Re: Hypothetical question on lost coins / transfers

2010-08-19 20:28:50 UTC - -

That's right. You don't need to be re-broadcasting your transactions for it to work.

When any node disconnects a fork, it dumps all the transactions from the fork back into the transaction pool to add to the new chain. The entire network is making sure to re-integrate your transactions again. All you should see is that your number of confirmations starts over from 0.

In some types of forks, your transaction would have gotten into both forks already, so you're already good either way.

BitcoinTalk

Re: Need a post writing up some things users should know

2010-08-22 22:51:00 UTC - -

The clock part will be covered in the next release (0.3.11 or higher). SVN rev 141 pops up a message box if your clock is too far off.

BitcoinTalk

Re: 28 days without generation, i have 4200khash/s

2010-08-22 23:01:02 UTC - -

Search debug.log for "proof-of-work found". If you find any, then check for any errors right after that.

> *Quote from: davidonpda on August 19, 2010, 07:43:01 PM*
> *How big of a margin on the time is allowed for things to work right.*

The margin is 2 hours.

This should be solved in SVN rev 141 and the next release (0.3.11+). It'll pop up a message box alerting you if your clock is off by more than an hour.

BitcoinTalk

Re: tcatm's 4-way SSE2 for Linux 32/64-bit is in 0.3.10

2010-08-22 23:21:50 UTC - -

Thanks for clearing that up. I read the link someone posted about AMD making that change around 2007, but I didn't know what the story was for Intel.

There's no hope for Core/Core2 then. They only have half the SSE2 hardware.

Strange that Intel has 3 128bit units, but AMD with 2 128bit units is the faster one.

BitcoinTalk

Development of alert system

2010-08-22 23:55:06 UTC - -

I've been working on writing the alert system. Alerts are broadcast through the network and apply to a range of version numbers. Alert messages are signed with a private key that only I have.

Nodes can do two things in response to an alert:
- Put a warning message on the status bar.
- Make the money handling methods of the json-rpc interface return an error.

In cases like the overflow bug or a fork where users may not be able to trust received payments, the alert should keep old versions mostly safe until they upgrade. Manual users should notice the status bar warning when looking for received payments, and the json-rpc safe mode stops automated websites from making any more trades until they're upgraded.

The json-rpc methods that return errors during an alert are:
sendtoaddress
getbalance
getreceivedbyaddress
getreceivedbylabel

listreceivedbyaddress
listreceivedbylabel

BitcoinTalk

Re: integrating digital payments into p2p protocols

2010-08-22 23:57:32 UTC - -

Hey Zooko!

I wanted to thank you for posting about Bitcoin on your blog a year or two ago, back when I announced it on the **Cryptography Mailing List**.

BitcoinTalk

Re: tcatm's 4-way SSE2 for Linux 32/64-bit is in 0.3.10

2010-08-24 22:43:56 UTC - -

Quote from: ArtForz on August 21, 2010, 04:56:31 PM

- *AMD K10: 2 128bit units*
- *intel nehalem: 3 128bit units*

This probably explains why hyperthreading increases performance with -4way. If three SSE2 units is excessive, then hyperthreading would help keep them all busy.

BitcoinTalk

Re: Development of alert system

2010-08-24 23:51:12 UTC - -

If you're so paranoid that you're getting hysterical over this, then surely you're paranoid enough that if a warning message displays on the status bar, you'll check the website and forum.

I think if another bug like the overflow bug occurs, it's important that automated websites stop trading until their admins can check out what's going on and decide what to do. If you decide it's a false alarm and want to take your chances, you can use the "-disablesafemode" switch.

BitcoinTalk

Re: Development of alert system

2010-08-25 00:06:36 UTC - -

This is in SVN rev 142 as version 0.3.11.

BitcoinTalk

Re: Development of alert system

2010-08-25 15:17:37 UTC - -

It can't do arbitrary actions remotely. Maybe some of you are responding to other posters who suggested the alert system should do more?

If there is an alert, the following json-rpc methods return an error:
sendtoaddress
getbalance
getreceivedbyaddress
getreceivedbylabel
listreceivedbyaddress
listreceivedbylabel

The remaining 14 methods function as normal.

I believe the safer option should be enabled by default. If you want your server to keep trading and ignore an alert saying the money its receiving might be like the money from the overflow bug, then you can use the switch and not blame anyone else if you lose your money.

Worst case if you leave alerts enabled, your site stops trading until you upgrade or add the -disablesafemode switch.

Getting surprised by some temporary down time when your node would otherwise be at risk is better than getting surprised by a thief draining all your inventory.

Someday when we haven't found any new bugs for a long time and it has been thoroughly security reviewed without finding anything, this can be scaled back. I'm not arguing that this is the permanent way of things forever. It's still beta software.

BitcoinTalk

Re: Development of alert system

2010-08-25 16:40:20 UTC - -

I changed the switch name to -disablesafemode.

BitcoinTalk

Re: Development of alert system

2010-08-25 16:56:15 UTC - -

Quote from: jimbobway on August 25, 2010, 04:45:22 PM
Quote from: BioMike on August 23, 2010, 05:15:43 AM
@mizerydearia, I think the quote button is easier to find then the reply one.

So, theoretical this is a first control system where <some goverment> can arrest satoshi and demand
that he hands over his key (or get it from his computer) and shut down the complete network?

Or is that not possible? How far would <some goverment> get?

A few rhetorical questions for satoshi:

Can you resist waterboarding?
Can you endure electric shock?
All forms of torture?
Lastly, are you Jack Bauer by any chance? Seriously.

WRT the alert system, who cares? The most the key can do is temporarily disable six json-rpc commands until the site owners either add the -disablesafemode switch or upgrade. All nodes keep running and generating, the network stays up. If I'm not available, any script kiddie can figure out how to add two characters and make a new version that disables the alert system. It would be a temporary inconvenience only.

Quote from: BioMike on August 23, 2010, 05:15:43 AM
So, theoretical this is a first control system where <some goverment> can arrest satoshi and demand
that he hands over his key (or get it from his computer) and shut down the complete network?

This is what makes me think the people objecting don't know what they're talking about. It can't "shut down the complete network".

BitcoinTalk

Re: Development of alert system

2010-08-25 17:59:30 UTC - -

Quote from: nelisky on August 25, 2010, 01:28:32 AM

So what kind of warning do admins get from bitcoind? Is there something we can grep from debug.log? Or will rpc calls raise some specific error? Is there a way to locally force this to happen, for unittesting services?

getinfo has a new field that shows any alert messages or other errors that would be displayed on the status bar.

The rpc methods return a json-rpc error with the error description "Safe mode: " followed by additional text specified by the alert.

I added the switch "-testsafemode" for you. SVN rev 145.

This stuff is very new and may still be subject to change.

Quote from: mizerydearia on August 25, 2010, 12:11:50 AM

I just discovered http://www.bitcoin.org/wiki/doku.php?id=man_pageand don't see any reference to -disablesafemode. Perhaps it should be added! Also others liek -4way should be added as well.

Many switches are intentionally undocumented, like if their functionality is still under construction or I haven't settled on their name yet, or just test code not intended for release.

-4way should eventually be replaced by an auto-detect.

BitcoinTalk

Re: Development of alert system

2010-08-26 00:08:12 UTC - -

Quote from: BioMike on August 25, 2010, 06:23:45 PM
Quote from: satoshi on August 25, 2010, 04:56:15 PM
Quote from: BioMike on August 23, 2010, 05:15:43 AM

So, theoretical this is a first control system where <some goverment> can arrest satoshi and demand
that he hands over his key (or get it from his computer) and shut down the complete network?

Or is that not possible? How far would <some goverment> get?

This is what makes me think the people objecting don't know what they're talking about. It can't "shut down the complete network".

I've never objected this change/idea, just asking if this was possible and to what extent.

What's wrong with getting informed? 

My apologies, your post was indeed a question not a statement.

BitcoinTalk

Re: RFC: remove DB_PRIVATE flag

2010-08-26 00:33:28 UTC - -

Can you provide more details about what removing DB_PRIVATE does?

I can't remember if I had a specific reason for DB_PRIVATE, or if I just copied the flags from some example code. Does removing DB_PRIVATE make it safe for other processes to open the database simultaneously? That may be an improvement, depending what the side effects are. Does it substantially reduce performance by making it have to write out every change immediately or do other coordination? Are there additional locking or coordination files then? What else changes? You could test by timing an initial block download with and without DB_PRIVATE, preferably -connect-ing to a local machine so network isn't a factor.

Apparently, DB_PRIVATE doesn't do what you would hope it would do, which is prevent other processes from being able to open the database. It still lets them, it just screws up if they do. Another option, if there's a way, would be to make it lock the database files so they can't be accessed by other processes.

BitcoinTalk

Re: Need a post writing up some things users should know

2010-08-26 00:44:05 UTC - -

Any backup process/procedure would just be a stopgap until there's time to properly work on coding solutions in software. We can try to use words to help the situation until code gets there.

The main backup improvement will be pre-made pool of keys, and a rescan at load to scrape missed transactions from the block history. Then a backup will last forward for a long time.

BitcoinTalk

Re: auto backing up of wallet.dat

2010-08-26 00:57:40 UTC - -

I started posting in the other topic but I'll repeat here, this thread seems more specific to the topic.

The main backup improvement will be a pre-generated pool of keys and a rescan at load to scrape missed transactions from the block history. Then a backup will last forward for a long time.

I was starting to post the same idea you said nelisky.

How about a json-rpc command that locks the wallet, flushes it, copies wallet.dat to a location you specified, then unlocks it? That would be a smaller project than the pooled keys, so maybe it could be done first.

What's the simplest portable way to copy a file? Is there something in Boost?

What should it be named? maybe:
backupwallet <destination>

BitcoinTalk

Re: Gentoo Linux Ebuild

2010-08-27 00:49:43 UTC - -

Try -datadir=

Last time I tried $(shell /usr/bin/wx-config), there was immediate hollering about build problems with it. There wasn't time to investigate at the time.

One problem with $(shell /usr/bin/wx-config) is it will pick up any version (wx 2.8) and any configuration (non-UTF-8) of wxWidgets that happens to be there. -lwx_gtk2ud-2.9 only matches the right configuration. It fails if wxWidgets was built with the wrong configuration.

Quote

Iirc, chatting in #wxwidgets on freenode, the devs there were baffled why that was used.

Did they say why they were baffled?

Quote

This is because on my system the path is /usr/include/wx-2.9/wx/wx.h

Why is it there? Was it included by the OS, or did you have to build it? If you built it, I wonder why it would put itself in a different place.

Has wxWidgets 2.9 finally started to become available as a debian package?

Maybe we should do this:

```
INCLUDEPATHS= \
 -I"/usr/local/include/wx-2.9" \
 -I"/usr/local/lib/wx/include/gtk2-unicode-debug-static-2.9" \
 -I"/usr/include/wx-2.9" \
 -I"/usr/lib/wx/include/gtk2-unicode-debug-static-2.9"
```

Again, those paths help make sure it's only 2.9 and will fail with 2.8.

wxWidgets 2.8 comes in ANSI and UTF-16, both wrong for us. It's tempting because it's so easily available as a package; a lot of people were frustrated by it until we started hardcoding 2.9 into the makefile.

BitcoinTalk

Re: auto backing up of wallet.dat

2010-08-27 01:13:42 UTC - -

If you read it into memory and write it out, it could fail in tight memory situations.

I'm looking for something like copyfile(const char* from, const char* to) or copyfile(path from, path to), preferably something in Boost if it has it. If you find it for me, it's more likely I'll get to implementing it.

Quote from: nelisky on August 26, 2010, 01:21:57 AM
As for the file copy, why add to the boost dependency? I for one would love to get a core lib with very little deps.

We require Boost for JSON and a dozen things replacing dependencies on wxWidgets. Boost is good, portable stuff, we should not shy away from it.

BitcoinTalk

Re: auto backing up of wallet.dat

2010-08-27 02:54:07 UTC - -

I doubt there's an mmap(2) on Windows. I'd rather call an existing file copy function than make and test my own.

> ***Quote from: nelisky on August 27, 2010, 01:21:09 AM***
> *But if you are already using features from boost::filesystem you can use copy_file from that. I just think that, if not already required for something else, it's a tad overkill.*

Thanks. I thought it would be in there somewhere.

We already use boost::filesystem in a dozen places. It's not a new added dependency. It gives us a lot of portable stuff that we would otherwise have to have a #ifdef for each OS and test everywhere.

BitcoinTalk

Re: auto backing up of wallet.dat

2010-08-27 15:47:57 UTC - -

Sorry, I've been so busy lately I've been skimming messages and I still can't keep up.

We want to avoid Windows API calls whenever possible. They usually take about 6-8 parameters and a lot of testing to get right, it takes a page of code to do something simple.

I usually shy away from iostreams. Seems like I too often hit limitations. They kind of botched the C++ streams standard in the 90's, which is too bad, streams can be very powerful and useful when done right. Using it in rpc.cpp may still turn out to be a mistake.

Bottom line is I'd rather call an existing file copy function than make and test my own.

BitcoinTalk

Re: New web service: obtain dump of bitcoin block NNNN

2010-08-27 16:13:16 UTC - -

That's kind of interesting as an upside-down bar chart of how many blocks were produced each day. The target is 144 blocks per day.

BitcoinTalk

Re: Bitcoins are most like shares of common stock

2010-08-27 16:39:26 UTC - -

Bitcoins have no dividend or potential future dividend, therefore not like a stock.

More like a collectible or commodity.

BitcoinTalk

Re: Bitcoin does NOT violate Mises' Regression Theorem

2010-08-27 17:32:07 UTC - -

As a thought experiment, imagine there was a base metal as scarce as gold but with the following properties:
- boring grey in colour
- not a good conductor of electricity
- not particularly strong, but not ductile or easily malleable either
- not useful for any practical or ornamental purpose

and one special, magical property:
- can be transported over a communications channel

If it somehow acquired any value at all for whatever reason, then anyone wanting to transfer wealth over a long distance could buy some, transmit it, and have the recipient sell it.

Maybe it could get an initial value circularly as you've suggested, by people foreseeing its potential usefulness for exchange. (I would definitely want some) Maybe collectors, any random reason could spark it.

I think the traditional qualifications for money were written with the assumption that there are so many competing objects in the world that are scarce, an object with the automatic bootstrap of intrinsic value will surely win out over those without intrinsic value. But if there were nothing in the world with intrinsic value that could be used as money, only scarce but no intrinsic value, I think people would still take up something.

(I'm using the word scarce here to only mean limited potential supply)

BitcoinTalk

Version 0.3.11 with upgrade alerts

2010-08-27 21:54:12 UTC - -

Version 0.3.11 is now available.

Changes:
- Some blk*.dat checking on load
- Built the -4way code with -march=amdfam10, which makes it a little faster
- Warning if your clock is too far off
- Warnings/errors/alerts can also be seen in the getinfo command
- Alert system

The alert system can display notifications on the status bar to alert you if you're running a version that needs to be upgraded for an important security update.

In response to an alert, your node may also go into safe mode, which disables the following json-rpc commands (used by automated websites) to protect it from losing money until you get a chance to upgrade:
sendtoaddress
getbalance
getreceivedbyaddress
getreceivedbylabel
listreceivedbyaddress
listreceivedbylabel

If you decide it's a false alarm and want to take your chances, you can use the switch -disablesafemode to re-enable them.

This is an important safety improvement. For a large segment of possible problems, this can warn everyone immediately once a problem is discovered and prevent them from acting on bad information.

Nodes keep operating and do not stop generating in response to an alert, so old versions may still try to make a fork, but the alert system can make sure users are warned not to act on anything in the fork.

Download:
http://sourceforge.net/projects/bitcoin/files/Bitcoin/bitcoin-0.3.11/

BitcoinTalk

Re: tcatm's 4-way SSE2 for Linux 32/64-bit is in 0.3.10

2010-08-28 14:27:15 UTC - -

The simplification is intentional. There will only be more than one thash[7]=0 in one out of 134,217,728 cases. It only makes it 0.0000007% slower.

BitcoinTalk

Re: Version 0.3.11 with upgrade alerts

2010-08-28 14:54:04 UTC - -

> ***Quote from: torservers on August 28, 2010, 01:00:37 PM***
> *The "About" dialog still shows 0.3.10.1 beta.*

What OS? I ran the Windows and 64-bit Linux version and checked the about dialog.

The Mac version is still 0.3.10.1.

> ***Quote from: pavelo on August 28, 2010, 07:36:07 AM***
> *iirc, it is possible to specify -march on a per-function basis using some gcc attribute . That way, only the function in question would be optimized, and if the user doesn't specify -4way, everything else should be ok.*

I updated the first post to be more specific. Only the -4way code is compiled this way.

BitcoinTalk

Re: Big endian code problems

2010-08-29 22:14:36 UTC - -

The code assumes little-endian throughout and was written with the intention of never being ported to big-endian. Every integer that is sent over the network would have to be byte swapped, in addition to many dozens of other places in code. It would not be worth the extra sourcecode bloat.

Big-endian is on its way out anyway.

BitcoinTalk

Re: CryptoPP Assertion Error

2010-09-05 23:25:32 UTC - -

You can probably just comment out the line
cryptopp/secblock.h:187
//assert(false);

Let me know if it works, and watch if it memory leaks.

It looks like a template class to make sure the derived class defines its own version of allocate and deallocate. It would be weird if that was the actual problem and it made it all the way to release. Probably a false alarm.

BitcoinTalk

Re: Warning : Check your system (Help me)

2010-09-05 23:36:20 UTC - -

Any suggestions for better text to put for this error message so the next person will be less likely to be confused?

It's trying to tell them their clock is wrong and they need to correct it.

It's relying on 3 time sources:
1) the system clock
2) the other nodes, if within an hour of the system clock
if those disagree, then
3) the user (asking the user to fix the system clock)

I've thought about NTP, but this is more secure.

BitcoinTalk

Re: HTTP status codes from the JSON-RPC api

2010-09-06 21:21:21 UTC - -

This is in SVN rev 147.

This is more standard, and although json-rpc 1.0 didn't specify the format of error objects, it did specify that they would be *objects* not strings or other values, so we needed to change this to be correct. The code/message members have become standard in later json-rpc specs.

If you have code that checks the error and expects a string, you'll need to change it. When there is an error, the error member is now an object not a string.

Also in SVN rev 147:
- The command line json-rpc returns the error code as its exit code. Exit codes can only be 0-255 on unix, so it's abs(code)%256.
- The "backupwallet <destination>" command that was discussed in another thread. It locks the wallet and copies it, so you can be sure you get a correct copy.

BitcoinTalk

Re: Warning : Check your system (Help me)

2010-09-06 21:41:06 UTC - -

Quote from: Insti on September 06, 2010, 12:51:37 PM
Quote from: satoshi on September 05, 2010, 11:36:20 PM
Any suggestions for better text to put for this error message so the next person will be less likely to be confused?
"Please check that your computer's date and time are correct. If your clock is wrong Bitcoin will not work properly."

Thanks.

BitcoinTalk

Re: auto backing up of wallet.dat

2010-09-06 21:45:10 UTC - -

rpc backupwallet <destination> is in SVN rev 147.

BitcoinTalk

Re: bitcoind as daemon in OSX

2010-09-06 21:52:45 UTC - -

Can you build?

Try changing line 78 of init.cpp from:
#ifdef __WXGTK__

to:
#ifndef __WXMSW__

If that works, I'll change the source. It should work.

BitcoinTalk

Re: Always pay transaction fee?

2010-09-07 16:32:21 UTC - -

Another option is to reduce the number of free transactions allowed per block before transaction fees are required. Nodes only take so many KB of free transactions per block before they start requiring at least 0.01 transaction fee.

The threshold should probably be lower than it currently is.

I don't think the threshold should ever be 0. We should always allow at least some free transactions.

BitcoinTalk

Version 0.3.12

2010-09-07 19:17:55 UTC - -

Version 0.3.12 is now available.

Features:
- json-rpc errors return a more standard error object. (thanks to Gavin Andresen)
- json-rpc command line returns exit codes.
- json-rpc "backupwallet" command.
- Recovers and continues if an exception is caused by a message you received. Other nodes shouldn't be able to cause an exception, and it hasn't happened before, but if a way is found to cause an exception, this would keep it from being used to stop network nodes.

If you have json-rpc code that checks the contents of the error string, you need to change it to expect error objects of the form {"code":<number>,"message":<string>}, which is the standard. See this thread:
http://BitcoinTalk.org/index.php?topic=969.0

Download:
http://sourceforge.net/projects/bitcoin/files/Bitcoin/bitcoin-0.3.12/

BitcoinTalk

Re: Always pay transaction fee?

2010-09-08 17:30:14 UTC - -

Currently, paying a fee is controlled manually with the -paytxfee switch. It would be very easy to make the software automatically check the size of recent blocks to see if it should pay a fee. We're so far from reaching the threshold, we don't need that yet. It's a good idea to see how things go with controlling it manually first anyway.

It's not a big deal if we reach the threshold. Free transactions would just take longer to get into a block.

I did a rough tally of 4000 blocks from around 74000-78000. This is excluding the block reward transactions:

There were average 2 transactions per block, 17 transactions per hour, 400 transactions per day.

Average transaction bytes per block was 428 bytes, or 214 bytes per transaction.

The current threshold is 200KB per block, or about 1000 transactions per block. I think it should be lowered to 50KB per block. That would still be more than 100 times the average transactions per block.

The threshold can easily be changed in the future. We can decide to increase it when the time comes. It's a good idea to keep it lower as a circuit breaker and increase it as needed. If we hit the threshold now, it would almost certainly be some kind of flood and not actual use. Keeping the threshold lower would help limit the amount of wasted disk space in that event.

BitcoinTalk

Re: Version 0.3.12

2010-09-08 18:06:04 UTC - -

Bitcoin clients currently only create and recognize transactions that match two possible templates.

Those are some quick tests that loosely check if transactions fit some general metrics that those standard transactions fit. Nodes will only work on adding those transactions to their block.

In the future, if we add more templates to the existing 2 types of transactions, we can change the "rather not work on nonstandard transactions" test to accept them.

BitcoinTalk

Re: Bitcoin Blogger: Is It Better To Buy Or Generate Bitcoins?

2010-09-08 20:27:39 UTC - -

Quote from: BitLex on September 07, 2010, 08:10:54 PM

AMD X3 @2.8ghz
->stock client
~3800khs ~150Watt

Did you try -4way?

Quote

How many hashes can I expect with a 24 core machine? I have a quad-core generating 4,300 hashes-per-second, so I am estimating a 24-core machine could mine bitcoins at 25,000 hashes-per-second.

AMD Phenom (I think 4-core) CPUs are doing about 11,000khps with -4way, about 100% speedup. 24 cores should get 66,000khps. AMD is the best choice because it has the best SSE2 implementation. (or maybe because tcatm had an AMD and optimised his code for that)

There's been so much else to do that I haven't had time to make -4way automatic. For now you still have to do it manually.
http://BitcoinTalk.org/index.php?topic=820.0

BitcoinTalk

Auto-detect for 128-bit 4-way SSE2

2010-09-09 01:04:05 UTC - -

SVN rev 150 has some code to try to auto-detect whether to use 4-way SSE2. We need this because it's only faster on certain newer CPUs that have 128-bit SSE2 and not ones with 64-bit SSE2.

It uses the CPUID instruction to get the CPU brand, family, model number and stepping. That's the easy part. Knowing what to do with the model number is the hard part. I was not able to find any table of family, model and stepping numbers for CPUs. I had to go by various random reports I saw.

Here's what I ended up with:

Code:

```
 // We need Intel Nehalem or AMD K10 or better for 128bit SSE2
// Nehalem = i3/i5/i7 and some Xeon
// K10 = Opterons with 4 or more cores, Phenom, Phenom II, Athlon II
//  Intel Core i5  family 6, model 26 or 30
//  Intel Core i7  family 6, model 26 or 30
//  Intel Core i3  family 6, model 37
//  AMD Phenom    family 16, model 10
bool fUseSSE2 = ((fIntel && nFamily * 10000 + nModel >=  60026) ||
                 (fAMD   && nFamily * 10000 + nModel >= 160010));
```

I saw some sporadic inconsistent model numbers for AMD CPUs, so I'm not sure if this will catch all capable AMDs.

If it's wrong, you can still override it with -4way or -4way=0.

It prints what it finds in debug.log. Search on CPUID.

This is only enabled if built with GCC.

BitcoinTalk

Re: Won't let me send coins because it requires a transaction fee?

2010-09-10 00:23:24 UTC - -

What version is the one where this happened? Release build, or built it yourself? Which operating system?

Were you sending by IP or by Bitcoin Address?

When you sent 49.99, did it prompt you to pay a 0.01 fee?

There was a change in GetMinFee, but I can't see how it would cause this. It only starts to apply when a block gets huge.

The reason for the difference in block number is the number displayed was reduced by 1 in 0.3.11 because it made more sense that way.

BitcoinTalk

Re: Won't let me send coins because it requires a transaction fee?

2010-09-10 00:46:37 UTC - -

I think I know what happened. Doubleclick on the generated transaction. It probably has a sub-0.01 transaction fee in it.

Someone has been paying a 0.00000010 transaction fee. I don't think you can even set that with -paytxfee, I think you'd have to modify the code to do it. Your generated block is worth 50.00000010, so when you try to send the whole thing you have 0.00000010 left over for the change, which triggers the dust spam 0.01 fee.

It would normally be harmless except in this corner case. I should add a special case to CreateTransaction to handle this.

BitcoinTalk

Re: Won't let me send coins because it requires a transaction fee?

2010-09-10 17:12:33 UTC - -

The fix is in SVN rev 151.

You will be able to send your stuck 0.01 (actually 0.01000010) when you next upgrade.

BitcoinTalk

Re: Auto-detect for 128-bit 4-way SSE2

2010-09-10 18:11:06 UTC - -

Quote from: teknohog on September 09, 2010, 07:32:05 PM
Since the function CallCPUID function contains x86 assembler, it breaks the build on other architectures. I've changed line 2770 in main.cpp to

#if defined(__GNUC__) && defined(CRYPTOPP_X86_ASM_AVAILABLE)

to make it compile again, at least on ARM.

Added in SVN rev 152

BitcoinTalk

Re: Running on a port other than 8333

2010-09-12 17:40:20 UTC - -

Quote from: lachesis on August 10, 2010, 03:24:55 PM
~~Also, does Bitcoin open the BerkeleyDB as exclusive, precluding the need for a file lock?~~It does not -- did my own tests.
Is there a way to open BerkeleyDB exclusive?

DB_PRIVATE is the worst of both worlds. DB_PRIVATE is not exclusive, but it does make it get screwed up if another process tries to access it at the same time.

I've dropped the DB_PRIVATE flag in rev 153.

BitcoinTalk

Re: RFC: remove DB_PRIVATE flag

2010-09-12 18:00:39 UTC - -

Trying it without the DB_PRIVATE flag in rev 153. We need to keep an eye on what's different.

On Windows at least, it creates six __db.001 - __db.006 files with sizes from 24K to 4MB. It doesn't delete them on exit, it just leaves them behind.

The docs say it uses memory mapped files. I assume they have the same file permissions as the database files, so the same user access restrictions apply.

Tests on Windows private LAN download of 78500 blocks:
with DB_PRIVATE 20 minutes 51 seconds
without DB_PRIVATE 20 minutes 51 seconds

I wasn't expecting them to come out exactly the same.

BitcoinTalk

Re: Switch to GPL

2010-09-12 19:24:53 UTC - -

If the only library is closed source, then there's a project to make an open source one.

If the only library is GPL, then there's a project to make a non-GPL one.

If the best library is MIT, Boost, new-BSD or public domain, then we can stop re-writing it.

I don't question that GPL is a good license for operating systems, especially since non-GPL code is allowed to interface with the OS. For smaller projects, I think the fear of a closed-source takeover is overdone.

BitcoinTalk

Re: Memory leak

2010-09-19 17:22:03 UTC - -

Bouncing between 0 and 2 connections could be if it's connecting to itself. Are you using the "-connect" switch?

Did you compile it or is this a release build, and what version?

I'm not sure how the 200Kb/sec, since it waits at least a half second between connection attempts. How fast is it flickering between 0 and 2 connections? Faster than twice a

second?

The wait function on linux is:

```
inline void Sleep(int64 n)
{
   boost::thread::sleep(boost::get_system_time() + boost::posix_time::milliseconds(n));
}
```

If that doesn't work right, then it would be possible for it to spin through the loop as fast as it can.

BitcoinTalk

Re: Issues building bitcoin on Windows 7

2010-09-19 18:46:46 UTC - -

The lines it's tripping on:
Code:

```
ERROR extern map<string, string> mapAddressBook;
ERROR extern CCriticalSection cs_mapAddressBook;
ERROR extern vector<unsigned char> vchDefaultKey;
OK extern bool fClient;
OK extern int nBestHeight;


OK extern unsigned int nWalletDBUpdated;
ERROR extern DbEnv dbenv;
```

So it's acting like nothing is defined, not even map and vector.

Yet, db.h is included by headers.h (and only there, nowhere else) which includes vector, map, util.h and everything before db.h.

Is VC trying to use precompiled headers and screwing it up? Could there be some leftover precompiled header files in your directory from previously failed attempts that it's finding and using?

There's an installer package now that makes it really easy to install MinGW. Don't use the latest version 4.5.0, use a few versions back like 4.4.1 (1.908.0) or 1.812.0. A setup program completely installs everything, it's not hard like it used to be. I think the only thing I had to do was rename make*.exe something to make.exe.
http://tdm-gcc.tdragon.net/

Off topic, but: It would be nice if someone would hack on getting tcatm's 4-way 128-bit SSE2 code working on Windows. There's something with MinGW's optimisation, I'm

not sure but maybe a problem with 16-byte alignment on the stack, that makes it segfault. With some fiddling, I was able to get his code to work in a test program, but not in Bitcoin itself for some reason.

BitcoinTalk

Re: Bug? /usr/bin/bitcoind ""

2010-09-19 19:58:11 UTC - -

I don't know anything about any of the bug trackers. If we were to have one, we would have to make a thoroughly researched choice.

We're managing pretty well just using the forum. I'm more likely to see bugs posted in the forum, and I think other users are much more likely to help resolve and ask follow up questions here than if they were in a bug tracker. A key step is other users helping resolve the simple stuff that's not really a bug but some misunderstanding or confusion.

I keep a list of all unresolved bugs I've seen on the forum. In some cases, I'm still thinking about the best design for the fix. This isn't the kind of software where we can leave so many unresolved bugs that we need a tracker for them.

BitcoinTalk

Re: The case for removing IP transactions

2010-09-19 21:49:30 UTC - -

Probably best to disable receiving by IP unless you specifically intend to use it. This is a lot of surface area that nobody uses that doesn't need to be open by default.

In storefront cases, you would typically only want customers to send payments through your automated system that only hands out bitcoin addresses associated with particular orders and accounts. Random unidentified payments volunteered to the server's IP address would be unhelpful.

In general, sending by IP has limited useful cases. If connecting directly without a proxy, the man-in-the-middle risk may be tolerable, but no privacy. If you use a privacy proxy, man-in-the-middle risk is unacceptably high. If we went to all the work of implementing SSL, only large storefronts usually go to the trouble of getting a CA cert, but most of those cases would still be better off to use bitcoin addresses.

I uploaded this change to SVN rev 156. The switch to enable is "-allowreceivebyip".

Senders with this version will get the error "Recipient is not accepting transactions sent by IP address". Older version senders will get "Transfer was not accepted".

I used a different name for the switch because "-allowiptransactions" sounds like it includes sending. If there's a better name for the switch, we can change it again.

BitcoinTalk

Re: Message Encryption as a built-in feature?

2010-09-19 22:47:00 UTC - -

Theymos already said this... ECDSA does not support encrypting messages. Only digital signatures.

BitcoinTalk

Re: Always pay transaction fee?

2010-09-23 16:08:35 UTC - -

Quote from: satoshi on September 08, 2010, 05:30:14 PM

The current threshold is 200KB per block, or about 1000 transactions per block. I think it should be lowered to 50KB per block. That would still be more than 100 times the average transactions per block.

I implemented this change in SVN rev 157.

The reason I previously made it so high was to allow very large transactions without hitting the transaction fee. The threshold was around 26,000 BTC for transactions made of 50 BTC generated coins. Even though it was 100 times easier to generate back then, only a few people ever encountered the fee at that level. The new threshold puts it at around 11,000 BTC for sending generated coins. It would mostly only be reached with generated bitcoins. If you bought your bitcoins, they'll be denominated in larger transactions and won't be anywhere near the fee limit, unless you bought them in several hundred separate transactions. Even if you do reach the fee level, you only have to pay it once to bundle your little transactions together.

BitcoinTalk

Internal version number

2010-09-23 16:19:08 UTC - -

In the next release (0.3.13), I'm going to change the format of the internal version number integer from 313 to 31300, for instance 31305 = 0.3.13.5. The last number represents changes on the SVN between releases and ought to be properly represented in

the version number. Otherwise, it would be a pain if we had a mistake or something in one of the sub versions that needed to be worked around.

BitcoinTalk

Re: How To Make a Distributed BitCoin Escrow Service

2010-09-26 17:34:26 UTC - -

It's not implemented yet, but the network can support a transaction that requires two signatures. It's described here:
http://BitcoinTalk.org/index.php?topic=750.0

It's absolutely safer than a straight payment without escrow, but not as good as a human arbitrated escrow, assuming you trust the human enough.

In this kind of escrow, a cheater can't win, but it's still possible for you to lose. It at least takes away the profit motive for cheating you. The seller is assured that the money is reserved for him, while the buyer retains the leverage that the seller hasn't been paid yet until completion.

BitcoinTalk

Re: I broke my wallet, sends never confirm now.

2010-09-30 16:38:53 UTC - -

As you figured out, the root problem is we shouldn't be counting or spending transactions until they have at least 1 confirmation. 0/unconfirmed transactions are very much second class citizens. At most, they are advice that something has been received, but counting them as balance or spending them is premature.

I made changes so they show up in lighter print, with the credit amount in square brackets like [+1.23], and the amount not counted towards your balance and not available for spending. This doesn't apply to transactions you sent, which you implicitly trust, since you wrote them.

I didn't make it (+1.23) because parenthesis in accounting means negative. I hope square brackets is different enough to be clear what is meant.

The JSON-RPC interface can still see 0/unconfirmed if it wants by specifying 0 confirmations.

I uploaded the changes to SVN rev 158. I will post a 0.3.13 RC shortly.

If you have any of these transactions in your wallet, do not send any payments until

you've upgraded to 0.3.13, which will be coming soon.

If you've already sent any of these transactions, or you're the creator of them, then use theymos' patch or make the following change and use it to send your clean transactions to a new wallet to clean things up.

change:
```
    if (pcoin->GetDepthInMainChain() < 1 && pcoin->GetDebit() <= 0)
        continue;
```
to:
```
    if (pcoin->GetDepthInMainChain() < 1)
        continue;
```

BitcoinTalk

0.3.13 RC1 for Windows, please test

2010-09-30 17:04:15 UTC - -

0.3.13 release candidate, to be released soon so please test:
http://www.bitcoin.org/download/bitcoin-0.3.13-rc1-win32-setup.exe

- don't count or spend payments until they have 1 confirmation
  http://BitcoinTalk.org/index.php?topic=1306.0
- internal version number from 312 to 31300
- only accept transactions sent by IP address if -allowreceivebyip is specified
- dropped DB_PRIVATE Berkeley DB flag
- fix problem sending the last cent with sub-cent fractional change
- auto-detect whether to use 128-bit 4-way SSE2 on Linux

Gavin Andresen:
- option -rpcallowip= to accept json-rpc connections from another machine
- clean shutdown on SIGTERM on Linux

BitcoinTalk

Re: BitCoin Wikipedia page DELETED!!!

2010-09-30 17:50:32 UTC - -

If you do, I think it should be a very brief, single paragraph article like 100 words or less that simply identifies what Bitcoin is.

I wish rather than deleting the article, they put a length restriction. If something is not famous enough, there could at least be a stub article identifying what it is. I often come across annoying red links of things that Wiki ought to at least have heard of.

The article could be as simple as something like:
"Bitcoin is a peer-to-peer decentralised /link/electronic currency/link/."

The more standard Wiki thing to do is that we should have a paragraph in one of the more general categories that we are an instance of, like Electronic Currency or Electronic Cash. We can probably establish a paragraph there. Again, keep it short. Just identifying what it is.

BitcoinTalk

Re: Prioritized transactions, and tx fees

2010-09-30 18:11:56 UTC - -

It ramps up the fee requirement as the block fills up:

<50KB free
50KB 0.01
250KB 0.02
333KB 0.03
375KB 0.04
etc.

It's a typical pricing mechanism. After the first 50KB sells out, the price is raised to 0.01. After 250KB is sold, it goes up to 0.02. At some price, you can pretty much always get in if you're willing to outbid the other customers.

Just including the minimum 0.01 goes a long way.

BitcoinTalk

Re: Prioritized transactions, and tx fees

2010-09-30 18:22:22 UTC - -

True, the switch should be something more dynamic that pays per KB. It's harder to think of how to explain it.

BitcoinTalk

Re: Remote RPC access

2010-09-30 18:27:41 UTC - -

It can be safe if you're using it over your own LAN, like if you have multiple servers at a location that talk to each other.

0.3.13 RC1 is available for Windows:
http://www.bitcoin.org/download/bitcoin-0.3.13-rc1-win32-setup.exe

BitcoinTalk

Re: 0.3.13 RC1 for Windows, please test

2010-10-01 00:32:46 UTC - -

Too late for 0.3.13, but I'll try to find time to add it to the next version.

BitcoinTalk

Version 0.3.13, please upgrade

2010-10-01 00:34:35 UTC - -

Version 0.3.13 is now available. You should upgrade to prevent potential problems with 0/unconfirmed transactions. Note: 0.3.13 prevents problems if you haven't already spent a 0/unconfirmed transaction, but if that already happened, you need 0.3.13.2.

Changes:
- Don't count or spend payments until they have 1 confirmation.
- Internal version number from 312 to 31300.
- Only accept transactions sent by IP address if -allowreceivebyip is specified.
- Dropped DB_PRIVATE Berkeley DB flag.
- Fix problem sending the last cent with sub-cent fractional change.
- Auto-detect whether to use 128-bit 4-way SSE2 on Linux.
Gavin Andresen:
- Option -rpcallowip= to accept json-rpc connections from another machine.
- Clean shutdown on SIGTERM on Linux.

Download:
http://sourceforge.net/projects/bitcoin/files/Bitcoin/bitcoin-0.3.13/

(Thanks Laszlo for the Mac OSX build!)

Note:
The SSE2 auto-detect in the Linux 64-bit version doesn't work with AMD in 64-bit mode. Please try this instead and let me know if it gets it right:
http://www.bitcoin.org/download/bitcoin-0.3.13.1-specialbuild-linux64.tar.gz

You can still control the SSE2 use manually with -4way and -4way=0.

Version 0.3.13.2 (SVN rev 161) has improvements for the case where you already had 0/unconfirmed transactions that you might have already spent. Here's a Windows build of it:
http://www.bitcoin.org/download/bitcoin-0.3.13.2-win32-setup.exe

BitcoinTalk

Re: Version 0.3.13

2010-10-03 18:17:06 UTC - -

Quote from: ShadowOfHarbringer on October 02, 2010, 01:00:07 PM
That's nice, however the automatic 4way detection is not working on my Gentoo AMD 64 version client.

I still have to add the "-4way" switch.

Forgot to say, I suspected the detect might not work on 64-bit AMD. I found it hard to believe but AMD reports a different model number in 64-bit mode.

Could you grep CPUID your debug.log and tell me what it says? (and anyone else with 64-bit AMD) And what AMD chip do you have?

Do all AMDs that support 64-bit have the better SSE2 hardware also?

BitcoinTalk

Re: Version 0.3.13, please upgrade

2010-10-03 19:39:06 UTC - -

Could a few people please run this special build? It'll amnesty the dust spam transactions, which will clear up the 0/unconfirmed problem for now. We really just need one block letting them through to clear up the previous transactions. Post if you generate a block with this.

These are binaries only. The linux version is 64-bit only.
http://www.bitcoin.org/download/bitcoin-0.3.13.1-specialbuild-win32.zip
http://www.bitcoin.org/download/bitcoin-0.3.13.1-specialbuild-linux64.tar.gz

SHA1 fb7c66270281ed058c570627cf7baff0bdc16e5d bitcoin-0.3.13.1-specialbuild-win32.zip
SHA1 9fc44ea5f2109618073e2cfd887e2cc266eb31a9 bitcoin-0.3.13.1-specialbuild-linux64.tar.gz

The linux 64-bit version includes a change to the cpuid 4-way 128-bit SSE2 autodetect for AMD in 64-bit mode, if you'd like to test that and see if that's better.

BitcoinTalk

Re: Version 0.3.13, please upgrade

2010-10-03 19:49:32 UTC - -

Quote from: tcatm on October 03, 2010, 07:45:45 PM
983 Mhash/s box.

Seriously? What hardware is that?

BitcoinTalk

Re: Version 0.3.13, please upgrade

2010-10-03 20:02:24 UTC - -

Code:

```
diff -u old\main.cpp new\main.cpp
--- old\main.cpp Sun Oct 03 20:57:20 2010
+++ new\main.cpp Sun Oct 03 20:57:54 2010
@@ -2831,6 +2831,10 @@
    bool fUseSSE2 = ((fIntel && nFamily * 10000 + nModel >=  60026) ||
                     (fAMD   && nFamily * 10000 + nModel >= 160010));

+    // AMD reports a lower model number in 64-bit mode
+    if (fAMD && sizeof(void*) > 4 && nFamily * 10000 + nModel >= 160004)
+        fUseSSE2 = true;
+
    static bool fPrinted;
    if (!fPrinted)
    {
@@ -2989,6 +2993,17 @@

                // Transaction fee based on block size
                int64 nMinFee = tx.GetMinFee(nBlockSize);
+                //////// temporary code
+                if (nBlockSize < MAX_BLOCK_SIZE_GEN / 10 &&
GetWarnings("statusbar") == "")
+                {
+                    if (nBestHeight < 91000)
+                        nMinFee = 0;
+                    if (nBestHeight < 100000 && nTxSize < 2000)
```

```
+                    nMinFee = 0;
+                if (nBestHeight < 110000 && nBestHeight % 10 == 0)
+                    nMinFee = 0;
+            }
+            //////// temporary code

            map<uint256, CTxIndex> mapTestPoolTmp(mapTestPool);
            if (!tx.ConnectInputs(txdb, mapTestPoolTmp, CDiskTxPos(1,1,1),
pindexPrev, nFees, false, true, nMinFee))
diff -u old\serialize.h new\serialize.h
--- old\serialize.h Sun Oct 03 20:57:45 2010
+++ new\serialize.h Sun Oct 03 20:57:54 2010
@@ -22,8 +22,8 @@
 class CAutoFile;
 static const unsigned int MAX_SIZE = 0x02000000;

-static const int VERSION = 31300;
-static const char* pszSubVer = "";
+static const int VERSION = 31301;
+static const char* pszSubVer = " test1";
```

BitcoinTalk

Re: Version 0.3.13, please upgrade

2010-10-03 20:54:07 UTC - -

Quote from: theymos on October 03, 2010, 08:09:51 PM

ArtForz is already running with no fees, and he has 20-30% of the network's CPU power. The person who originally sent the broken transactions deleted his wallet, though, and the network has forgotten these historical transactions, so any transactions based on this won't confirm.

Transactions aren't accepted or displayed as 0/unconfirmed until your node has a path of transactions back to the block chain.

Any transactions in your wallet also have bundled with them all unrecorded transactions required to reach the block chain. If you have a transaction that is displayed as 0/unconfirmed, then you have all the previous unrecorded transactions it depends on and you will also rebroadcast those transactions when you rebroadcast yours.

If a no-fee block has already been generated and hasn't helped, then I need to look at what's wrong. It's a part of code that doesn't get much use. They should be recorded in the wallets of everyone who has a transaction depending on them.

Quote from: theymos on October 03, 2010, 08:09:51 PM

The person who originally sent the broken transactions deleted his wallet

Sigh... why delete a wallet instead of moving it aside and keeping the old copy just in case? You should never delete a wallet.

> *Quote from: tcatm on October 03, 2010, 08:10:47 PM*
> *It's running. Should find a block within 3 hours.*

It may take a while to collect re-broadcast transactions. It'll help if you can accept inbound connections so you'll be listening to more nodes. Even if you find a block in 3 hours, keep it running continuously for a few days at least.

BitcoinTalk

Re: [PATCH] increase block size limit

2010-10-03 21:07:28 UTC - -

> *Quote from: theymos on October 03, 2010, 08:28:39 PM*
> *Applying this patch will make you incompatible with other Bitcoin clients.*
> *+1 theymos. Don't use this patch, it'll make you incompatible with the network, to your own detriment.*

We can phase in a change later if we get closer to needing it.

BitcoinTalk

Re: How to overthrow the GPU Oligarchs

2010-10-03 21:30:04 UTC - -

> *Quote from: theymos on October 02, 2010, 06:11:11 AM*
> *Quote from: lzsaver on October 02, 2010, 05:49:47 AM*
> *Can you tell more about it:*
> *"they have to do weird things with extraNonce, which increases the size of the block header".*
> *When you generate, you calculate hashes of the block header. Hashing more data is slower than hashing less data, so the block header is critically of a fixed size for everyone, with one exception.*

This is the point of confusion. extraNonce is not part of the block header, it is part of the first transaction. It does not slow down your hashing. It does not change the size of the header.

We need to be vigilant and nip in the bud any misconception that the contents of your block slows down your hash speed. It doesn't.

extraNonce never needs to be very big. We could reset it every second whenever the time changes if we wanted. Worst case, if you didn't want to keep track of incrementing it, extraNonce could be 4 random bytes and the chance of wasting time from collision would be negligible.

Separate machines are automatically collision proof because they have different generated public keys in the first transaction. That also goes for each thread too.

BitcoinTalk

Re: Version 0.3.13, please upgrade

2010-10-03 21:43:20 UTC - -

ShadowOfHarbringer, is yours faster with -4way?

If it is, then I'm thinking that any AMD that supports 64-bit has 128-bit SSE2.

The specialbuild version I posted here looks for model 4 or higher. If yours is faster with -4way, then I should change it to always use SSE2 with any AMD with 64-bit.

BitcoinTalk

Re: Memory leak

2010-10-03 22:07:00 UTC - -

You're connecting to yourself. All 21 connection attempts were to a node with version 31300 (0.3.13). Not everyone has 0.3.13 yet.

IRC seems to be working. It ought to have other nodes to try.

There may be something I need to do to make sure it doesn't try to connect to itself again right away after disconnecting. I can't see how it's happening though, it should be resetting nLastTry which would put it to the back of the queue, but the log doesn't show it.

You can try moving addr.dat aside. Maybe there's something wrong in it.

Are you using -addnode?

BitcoinTalk

Re: Version 0.3.13, please upgrade

2010-10-03 23:46:19 UTC - -

Make sure you keep your node online so it'll keep rebroadcasting transaction b412a0. It haven't seen it rebroadcast since 29/09/2010 16:41.

BitcoinTalk

Re: Website and software translations

2010-10-04 01:44:41 UTC - -

Thanks eurekafag, Russian translation added to SVN rev 160.

BitcoinTalk

Re: Website and software translations

2010-10-04 19:21:01 UTC - -

Quote from: eurekafag on October 04, 2010, 10:55:56 AM

Where can I find the latest English .po file to keep the translation up-to-date?

poedit does it. Either get the src directory from a release, or download it with SVN. Place your .po file 3 directories deep under the src directory. Open it with poedit and do Catalog->Update from sources.

So for example, you have:
src
srcase58.h
srcignum.h
...
src\util.cpp
src\util.h
src\xpm
src\locale u\LC_MESSAGESitcoin.po

Open bitcoin.po with poedit, do Catalog->Update from sources. It looks for the sourcecode up 3 directories (..\..\..) from where bitcoin.po is.

This updates your existing .po file you already worked on and adds any news strings. It may try to match close strings, so check things over and make sure it didn't make any bad guesses.

Make sure you use the .po file I uploaded to SVN or in a release, because I always fix up at least a few things. I'm attaching your Russian one to this message.

BitcoinTalk

Re: [PATCH] increase block size limit

2010-10-04 19:48:40 UTC - -

It can be phased in, like:

```
if (blocknumber > 115000)
   maxblocksize = largerlimit
```

It can start being in versions way ahead, so by the time it reaches that block number and goes into effect, the older versions that don't have it are already obsolete.

When we're near the cutoff block number, I can put an alert to old versions to make sure they know they have to upgrade.

BitcoinTalk

Re: Website and software translations

2010-10-06 15:42:39 UTC - -

poedit reorganised the file for some reason. I re-ran update from sources and it put it back in the original order so it's fine now. Did you run it on a drive where files aren't sorted alphabetically, like a FAT drive or USB flash drive?

Strings aren't added or changed very often. It's months before enough changes build up.

I uploaded the changes.

This Windows build has the Russian translation in it:
http://www.bitcoin.org/download/bitcoin-0.3.13.2-win32-setup.exe

BitcoinTalk

Re: I broke my wallet, sends never confirm now.

2010-10-06 16:54:23 UTC - -

That's going to be more of a SelectCoins thing.

SVN rev 161 has a refinement to recursively determine if your own unconfirmed transactions can be spent. This is needed because you should be able to spend your own

change right away.

The new recursive determination is: 0/unconfirmed can be spent if it's yours and all its dependencies are either in a block or also yours.

Here's a Windows build:
http://www.bitcoin.org/download/bitcoin-0.3.13.2-win32-setup.exe

This version is an improvement if you already had a 0/unconfirmed transaction and might have already spent it. If you were the original creator of a 0/unconfirmed transaction, you still need theymos' patch instead.

BitcoinTalk

Re: Tor connections not working reliably, many seednodes offline

2010-10-06 17:36:41 UTC - -

Maybe you were just unlucky to have an exit node without reverse lookup.

The IRC server's response doesn't look like it was disconnecting you for that. It's supposed to go IRC SENDING: NICK after that, and it doesn't so it gets timed out.

I see the problem. The IRC code is looking for various phrases to see when the server is ready to receive your NICK, but it's not looking for that particular phrase. I'll fix it.

I don't know if it's really required to wait for the server to finish looking up hostname before sending nick.

How long did it take to get connected with TOR the first time, having to use the seed nodes?

BitcoinTalk

Re: The Niche List

2010-10-06 23:10:31 UTC - -

Quote from: kiba on September 23, 2010, 04:00:16 PM
1. Download site like rapidshare and other crappy host. Inconvenient captcha and required paypal. Bitcoin can possibly take both roles and streamline the whole process.

Repeating myself here, but there is open source software for that, so it would just be a matter of bolting on a Bitcoin payment mechanism. One good one I found was Mihalism Multi Host. It's designed as a free host, so it would just need a few tweaks to loosen up restrictions consistent with paid use.

BitcoinTalk

Key pool feature for safer wallet backup

2010-10-09 20:19:33 UTC - -

SVN rev 163 (ver 0.3.13.3) has the key pool feature. Pre-generated new keys are aged in a queue before use, so that backups of wallet.dat hold keys you'll use in the future.

For now I made the default pool size 100. It can be configured with -keypool=. Be aware, it takes a little time to increase the pool size, so don't go crazy with it. Disk space is about 1K per key.

I have not addressed the recovery side of this yet. If you actually did restore an old wallet.dat, I think you may have to delete blk*.dat to rediscover your own transactions during the redownload.

I've only tested this moderately. You might not want to use this for a website server until it's had some more testing.

BitcoinTalk

Version 0.3.14

2010-10-21 16:39:27 UTC - -

Version 0.3.14 is now available
http://sourceforge.net/projects/bitcoin/files/Bitcoin/bitcoin-0.3.14/

Changes:
- Key pool feature for safer wallet backup
Gavin Andresen:
- TEST network mode with switch -testnet
- Option to use SSL for JSON-RPC connections on unix/osx
- validateaddress RPC command
eurekafag:
- Russian translation

BitcoinTalk

Re: Website and software translations

2010-10-21 22:50:47 UTC - -

The order matters not to the program, but it matters to me maintaining it. If it jumbles the order of the .po file then I can't diff for changes. I have to update all 7 translation files when I change the English text in the program, and it's easier when they're all in the same order.

I can still put it back into normal order by making poedit rescan it.

It is normal that untranslated strings are shown on top.

Quote from: eurekafag on October 06, 2010, 07:39:36 PM
By the way, there are some similar lines that possibly may be replaced by one. They are very close by meaning and differs only by 1-2 words. Just a suggestion of course.

I know, but not easily without complicating the sourcecode.

BitcoinTalk

Re: ERROR - PLEASE HELP ME!

2010-10-23 18:22:49 UTC - -

Quote from: theymos on October 21, 2010, 10:00:26 PM
his block count remains "stuck" at 1698.

He was generating invalid blocks at difficulty 1.0. He must have a corrupted entry in his blk0001.dat or blkindex.dat file. He just needs to delete blk*.dat and let it redownload.

The safety lockdown detected the problem and was displaying "WARNING: Displayed transactions may not be correct!" because it saw a longer chain existed that it was unable to accept. The safety lockdown cannot stop generation or it would create an attack possibility.

Quote from: gavinandresen on October 22, 2010, 02:25:14 PM
The Bitcoin client really shouldn't allow coin generation until you have all of the blocks up to the last block checkpoint.

Good idea, I made a change to make sure it won't generate before checkpoint block 74000.

BitcoinTalk

Re: ERROR - PLEASE HELP ME!

2010-10-23 18:38:04 UTC - -

OK, if it really won't get past block 1698 on redownload, then we're in stranger territory.

Yes, possibly he has antivirus software or even a router or filewall that is pattern matching a sequence of bytes and censoring it.

It would be instructive to get knightmb's blk*.dat and see if that gets him past that point.

BitcoinTalk

Re: Win7 64bit since last patch Tues now crashes

2010-10-23 18:52:02 UTC - -

Quote from: Odin on October 22, 2010, 09:24:38 PM

Fault Module Name: mingwm10.dll

This is the important clue. I believe it's saying it crashed in that. Maybe there are other versions of it to try. mingwm10.dll is just a simple placeholder thing that satisfies some callback requirement for multithreaded apps.

Is anyone else running OK on Windows 64-bit?

BitcoinTalk

Re: Suggestion: Allow short messages to be sent together with bitcoins ?

2010-10-23 19:02:57 UTC - -

ECDSA can't encrypt messages, only sign signatures.

It would be unwise to have permanently recorded plaintext messages for everyone to see. It would be an accident waiting to happen.

If there's going to be a message system, it should be a separate system parallel to the bitcoin network. Messages should not be recorded in the block chain. The messages could be signed with the bitcoin address keypairs to prove who they're from.

BitcoinTalk

Re: Multiple Wallets, one computer

2010-10-24 19:17:51 UTC - -

I have the beginning of something like this. It's mostly like what Gavin described.

Some more rpc interface:

move <fromaccount> <toaccount> <amount>
Move from one internal account to another. I think blank account name ("") will be your default account. If you sell something to a user, you could do move "theiraccount" "" 123.45.
Is "move" the best name for this? I shied away from "transfer" because that sounds too close to sending a transaction.

I'm thinking a new function getaccountaddress instead of overloading getnewaddress:

getaccountaddress <account>
Gives you an address allocated from getnewaddress <account>. It'll keep giving the same address until something is received on the address, then it allocates a new address. (It automatically does what the sample code I posted some time ago did)

Would these commands make it possible in simple cases to implement your website without needing a database of your own?

BitcoinTalk

Re: Multiple Wallets, one computer

2010-10-25 16:53:53 UTC - -

Here's some pseudocode of how you would use the account based commands. It sure makes website integration a lot easier.

```
print "send to " + getaccountaddress(username) + " to fund your account"
print "balance: " + getbalance(username, 0)
print "available balance: " + getbalance(username, 6)

// if you make a sale, move the money out of their account
move(username, "", amount, 6)

// withdrawal
sendfrom(username, bitcoinaddress, amount, 6)
```

BitcoinTalk

Re: Win7 64bit since last patch Tues now crashes

2010-10-25 17:27:47 UTC - -

The only thing I can think of is to see if there are other versions of mingwm10.dll you can get. mingwm10.dll is a tiny little DLL that came with the MinGW compiler that you need when you build for multi-thread. I don't know exactly what it does, but it probably just says something like "yes Windows, see I'm in a DLL like you insisted."

The end of your debug.log file might show the last thing it was doing before it crashed.

BitcoinTalk

Re: New icon/logo

2010-11-13 00:55:51 UTC - -

I'm happy if someone with artistic skill wants to contribute alternatives. The icon/logo was meant to be good as an icon at the 16x16 and 20x20 pixel sizes. I think it's the best program icon, but there's room for improvement at larger sizes for a graphic for use on websites.

It'll be a lot simpler if authors could make their graphics public domain.

BitcoinTalk

Re: Some testing that I did on the testnetwork, my findings.

2010-11-13 23:25:26 UTC - -

Thank you for limiting flood tests to the testnet.

Version 0.3.15 combines several features to help legitimate transactions jump the queue during a flood attack. The key was Gavin's idea for prioritising transactions based on the age of their dependencies. Every coin is entitled to turn over so often. The longer waited, the more priority accumulates. Priority is sum(valuein * age) / txsize. Transaction fee still takes precedence over priority, and priority determines the order of processing within a fee strata.

In support of the priority feature, SelectCoins only uses your own 0 conf transactions only as a last resort if that's all you have left. This helps keep you from turning your coins over rapidly unless you're forcing it by actually turning all your coins over rapidly.

BitcoinTalk

Version 0.3.15

2010-11-13 23:26:40 UTC - -

Version 0.3.15 is now available.

Changes:
- paytxfee switch is now per KB, so it adds the correct fee for large transactions
- sending avoids using coins with less than 6 confirmations if it can
- BitcoinMiner processes transactions in priority order based on age of dependencies
- make sure generation doesn't start before block 74000 downloaded
- bugfixes by Dean Gores
- testnet, keypoololdest and paytxfee added to getinfo

BitcoinTalk

Re: Some testing that I did on the testnetwork, my findings.

2010-11-14 16:53:19 UTC - -

Quote from: ByteCoin on November 13, 2010, 11:55:11 PM

*Of course, if the network is not being flooded and you're not overly concerned about the current transaction getting held up then it's probably worth preferring to use your 0 conf transactions so that you can "save" the higher priority coins for when the network **is** being flooded.*

You should use at least some priority in case a flood comes along before the next block.

As long as all dependencies have at least 1 conf, if the transaction doesn't have enough priority at first, the dependencies will age until it does.

Quote

Gaming the system by including 1000 or so recently turned over BTC to bump the priority as described in my post above still works of course!

Or managing how much priority you spend on a transaction. The software would have to know your future plans to know whether to spend your priority now or save it for later. I don't think we'll need to get into that much detail though. There's a wide enough difference between normal users and flooders.

Priority doesn't have to do everything. Once you know there's a flood, you can add -paytxfee=0.01. Hopefully with priority, your transactions before that should be at worst slow, not stuck.

BitcoinTalk

Re: Need OP_BLOCKNUMBER to allow "time" limited transactions

2010-11-15 18:37:44 UTC - -

We can't safely do OP_BLOCKNUMBER. In the event of a block chain reorg after a segmentation, transactions need to be able to get into the chain in a later block. The OP_BLOCKNUMBER transaction and all its dependants would become invalid. This wouldn't be fair to later owners of the coins who weren't involved in the time limited transaction.

nTimeLock does the reverse. It's an open transaction that can be replaced with new versions until the deadline. It can't be recorded until it locks. The highest version when the deadline hits gets recorded. It could be used, for example, to write an escrow transaction that will automatically permanently lock and go through unless it is revoked before the deadline. The feature isn't enabled or used yet, but the support is there so it could be implemented later.

BitcoinTalk

Re: Transaction / spam flood attack currently under way

2010-11-19 23:50:24 UTC - -

Quote from: creighto on November 19, 2010, 08:29:12 PM

Perhaps in addition to the age priority rule recently implimented, there should be a minimum age rule without a transaction fee. Said another way, perhaps a generation rule that says that a free transaction must be 3 blocks deep before it can be transfered again for free. This will still allow real users to immediately spend new funds if they have to, while still permitting real users to reshuffle funds to suit their needs without an overhead cost. I think that this would significantly inhibit the type of spamming attack that is currently underway.

I'm doing something like that. Priority is a more formalised version of the concept you're describing.

Quote from: FreeMoney on November 19, 2010, 05:39:44 PM

As it stands now 3.15 has a lot of free transaction space and that space is given first to transactions with the highest [age][value]/[size] correct? Would it be reasonable to make some arbitrary portion of the free space require [age]*[value]/[size] > C ?*

Maybe set C so that a standard 1BTC transaction can get into the main free area on the next block. And a .1 can get in after waiting about 10 blocks. And make the area which allows [age][value]/[size] < C to let in about a dozen transactions or so.*

Yes, like this. And the no-priority-requirement area is 3K, about a dozen transactions per block.

I just uploaded SVN rev 185 which has a minimal priority requirement for free transactions. Transaction floods are made up of coins that are re-spent over and over, so they depend on their own 0 conf transactions repeatedly. 0 conf transactions have 0 priority, so free transactions like that will have to wait for one transaction to get into a block at a time.

Version 0.3.15 doesn't write transactions using 0 conf dependencies unless that's all it has left, so normal users shouldn't usually have a problem with this.

I think this is a good compromise short of making the default fee 0.01. It's not so much to ask that free transactions can only be used to turn coins over so often. If you're using free transactions, you're taking charity and there has to be some limit on how often you can use it with the same coins.

We've always said free transactions may be processed more slowly. You can help ensure your transactions go through quickly by adding -paytxfee=0.01.

BitcoinTalk

Re: OpenCL miner for the masses

2010-11-20 17:24:20 UTC - -

Quote from: m0mchil on November 20, 2010, 10:16:19 AM
updated to SVN 186

Thanks m0mchil for keeping up on the updates!

GPU miners, please upgrade as soon as possible to shut down the free transaction abuse! This version has the new priority-based limit on free transaction spam.

Quote from: m0mchil on November 16, 2010, 10:30:41 AM
Just updated to SVN 181 and fixed getwork patch to wait 60 seconds between rebuilding the block with new transactions. This is actually the behavior of the original client, was forgotten in the patch by mistake. Fixes heavy CPU usage on every getwork request (this became obvious with recent heavy transaction spam). Please upgrade.

Before SVN 184, compiling transactions into a block used an n^2 algorithm. The new efficient single-pass algorithm is orders of magnitude quicker. (O(n) vs O(n^2)/2 algorithm, n=200 maybe 10 to 100 times quicker)

New getwork

2010-11-23 19:50:12 UTC - -

I uploaded a redesign of m0mchil's getwork to SVN rev 189 (version 31601)

m0mchil's external bitcoin miner idea has solved a lot of problems. GPU programming is immature and hard to compile, and I didn't want to add additional dependencies to the build. getwork allows these problems to be solved separately, with different programs for different hardware and OSes. It's also convenient that server farms can run a single Bitcoin node and the rest only run getwork clients.

The interface has a few changes:

getwork [data]
If [data] is not specified, returns formatted hash data to work on:
"midstate" : precomputed hash state after hashing the first half of the data
"data" : block data
"hash1" : formatted hash buffer for second hash
"target" : little endian hash target
If [data] is specified, tries to solve the block and returns true if it was successful. [data] is the same 128 byte block data that was returned in the "data" field, but with the nonce changed.

Notes:
- It does not return work when you submit a possible hit, only when called without parameter.
- The block field has been separated into data and hash1.
- data is 128 bytes, which includes the first half that's already hashed by midstate.
- hash1 is always the same, but included for convenience.
- Logging of "ThreadRPCServer method=getwork" is disabled, it would be too much junk in the log.

BitcoinTalk

Re: New getwork

2010-11-23 20:55:27 UTC - -

It's not an exact drop-in replacement. I wanted to clean up the interface a little. It only requires a few changes.

ScanHash_ functions aren't going away. BTW, the interface of this is designed to mirror the parameters of that (midstate, data, hash1).

BitcoinTalk

Re: New getwork

2010-11-24 17:21:01 UTC - -

Quote from: jgarzik on November 24, 2010, 04:47:42 AM
I suspect something weird going on with ByteReverse (or lack thereof). It's quite unclear whether or not 'data' and 'nonce' must be byte-reversed, and in what way.

getwork does the byte-reversing. midstate, data and hash1 are already big-endian, and you pass data back still big-endian, so you work in big-endian and don't have to do any byte-reversing. They're the same data that is passed to the ScanHash_ functions. You can take midstate, data and hash1, put them in 16-byte aligned buffers and pass them to a ScanHash_ function, like ScanHash(pmidstate, pdata + 64, phash1, nHashesDone). If a nonce is found, patch it into data and call getwork.

I should probably change the ScanHash_ functions to use pdata instead of pdata + 64 so they're consistent.

target is little endian, it's supposed to be the same as how m0mchil's did it. (if it's not, then it should be fixed) That's the only case where you would use byte reverse. I think you do it like: if ByteReverse((unsigned int*)hash[6]) < (unsigned int*)target[6].

Quote from: DiabloD3 on November 24, 2010, 11:31:11 AM
Satoshi, please fix your implementation of getwork so it complies with m0mchill's specification

This is the new spec. It shouldn't be hard to update your miner to use it.

The changes are:
- It does not return work when you submit a possible hit, only when called without parameter.
- The block field has been split into data and hash1.
- state renamed to midstate for consistency.
- extranonce not needed.

BitcoinTalk

Re: OpenCL miner for the masses

2010-11-24 17:53:09 UTC - -

A revised version of getwork is now in the official client, but the miners need to be updated a little to use it.

BitcoinTalk

Re: RFC: ship block chain 1-74000 with release tarballs?

2010-11-25 17:51:39 UTC - -

It's not the downloading that takes the time, it's verifying and indexing it.

Bandwidthwise, it's more efficient than if you downloaded an archive. Bitcoin only downloads the data in blk0001.dat, which is currently 55MB, and builds blkindex.dat itself, which is 47MB. Building blkindex.dat is what causes all the disk activity.

During the block download, it only flushes the database to disk every 500 blocks. You may see the block count pause at ??499 and ??999. That's when it's flushing.

Doing your own verifying and indexing is the only way to be sure your index data is secure. If you copy blk0001.dat and blkindex.dat from an untrusted source, there's no way to know if you can trust all the contents in them.

Maybe Berkeley DB has some tweaks we can make to enable or increase cache memory.

BitcoinTalk

Version 0.3.17

2010-11-25 20:07:36 UTC - -

Version 0.3.17 is now available.

Changes:
- new getwork, thanks m0mchil
- added transaction fee setting in UI options menu
- free transaction limits
- sendtoaddress returns transaction id instead of "sent"
- getaccountaddress <account>

The UI transaction fee setting was easy since it was still there from 0.1.5 and all I had to do was re-enable it.

The accounts-based commands: move, sendfrom and getbalance <account> will be in the next release. We still have some more changes to make first.

Downloads:
http://sourceforge.net/projects/bitcoin/files/Bitcoin/bitcoin-0.3.17/

BitcoinTalk

Re: RFC: ship block chain 1-74000 with release tarballs?

2010-11-26 17:32:01 UTC - -

I tested it on a slow 7 year old drive, where bandwidth and CPU were clearly not the bottleneck. Initial download took 1 hour 20 minutes.

If it's taking a lot longer than that, certainly 24 hours, then it must be downloading from a very slow node, or your connection is much slower than around 15KB per sec (120kbps), or something else is wrong. It would be nice to know what appears to be the bottleneck when that happens.

Every 10 minutes or so when the latest block is sent, it should have the chance to change to a faster node. When the latest block is broadcast, it requests the next 500 blocks from other nodes, and continues the download from the one that sends it fastest. At least, that's how it should work.

Quote from: jgarzik on November 26, 2010, 02:07:43 AM
Quote from: satoshi on November 25, 2010, 05:51:39 PM
Maybe Berkeley DB has some tweaks we can make to enable or increase cache memory.
Which of the ACID properties do you need, while downloading?

It may only need more read caching. It has to read randomly all over blk0001.dat and blkindex.dat to index. It can't assume the file is smaller than memory, although it currently still is. Caching would be effective, since most dependencies are recent.

Someone should experiment with different Berkeley DB settings and see if there's something that makes the download substantially faster. If something substantial is discovered, then we can work out the particulars.

Quote
Adding BDB records is simply appending to a log file, until you issue a checkpoint. The checkpoint then updates the main database file.

We checkpoint every 500 blocks.

BitcoinTalk

Re: Version 0.3.17

2010-11-26 18:23:30 UTC - -

Laszlo does them, but I haven't asked him to do one for a while because there wasn't anything major. I'll ask him to do this version.

BitcoinTalk

Re: New getwork

2010-11-26 21:31:13 UTC - -

That's what it does, it returns true/false.

BitcoinTalk

Re: New demonstration CPU miner available

2010-11-26 22:02:41 UTC - -

You should try it with tcatm's 4-way SSE2 SHA in sha256.cpp. It compiles fine as a C file, just rename sha256.cpp to sha256.c. I was able to get it to work in simple tests on Windows, but not when linked in with Bitcoin. It may have a better chance of working as part of a C program instead of C++.

Currently it's only enabled in the Linux build, so if you get it to work you could make it available to Windows users. It's about 100% speedup on AMD CPUs.

BitcoinTalk

Re: Cooperative mining

2010-11-28 16:03:30 UTC - -

ribuck's description is spot on.

Pool operators can modify their getwork to take one additional parameter, the address to send your share to.

The easy way for the pool operator would be to wait until the next block is found and divy it up proportionally as:
user's near-hits/total near-hits from everyone

That would be easier and safer to start up. It also has the advantage that multiple hits from the same user can be combined into one transaction. A lot of your hits will usually be from the same people.

The instant gratification way would be to pay a fixed amount for each near-hit immediately, and the operator takes the risk from randomness of having more or less near-hits before a block is found.

Either way, the user who submits the hit that solves the block should get an extra amount off the top, like 10 BTC.

New users wouldn't really even need the Bitcoin software. They could download a miner, create an account on mtgox or mybitcoin, enter their deposit address into the miner and point it at anyone's pool server. When the miner says it found something, a while later a few coins show up in their account.

Miner writers better make sure they never false-positive near-hits. Users will depend on that to check if the pool operator is cheating them. If the miner wrongly says it found something, users will look in their account, not find anything, and get mad at the pool operator.

BitcoinTalk

Re: RFC: ship block chain 1-74000 with release tarballs?

2010-11-28 17:13:01 UTC - -

Despite everything else said, the current next step is:

Quote

Someone should experiment with different Berkeley DB settings and see if there's something that makes the download substantially faster. If something substantial is discovered, then we can work out the particulars.

In particular, I suspect that more read caching might help a lot.

Quote from: jgarzik on November 28, 2010, 02:33:29 AM

Another new user on IRC, Linux this time, was downloading at a rate of 1 block every 4 seconds -- estimated total download time around 4 days.

Then something more specific was wrong. That's not due to normal initial download time. Without more details, it can't be diagnosed. If it was due to slow download, did it speed up after 10-20 minutes when the next block broadcast should have made it switch to a faster source? debug.log might have clues. How fast is their Internet connection? Was it steadily slow, or just slow down at one point?

Quote

We have the hashes for genesis block through block 74000 hardcoded (compiled) into bitcoin, so there's no reason why we shouldn't be able to automatically download a compressed zipfile of the block database from anywhere, unpack it, verify it, and start running.

The 74000 checkpoint is not enough to protect you, and does nothing if the download is already past 74000. -checkblocks does more, but is still easily defeated. You still must trust the supplier of the zipfile.

If there was a "verify it" step, that would take as long as the current normal initial download, in which it is the indexing, not the data download, that is the bottleneck.

> ***Quote from: jgarzik on November 28, 2010, 07:33:55 AM***
> *Presumably at some point there will be a lightweight client that only downloads block headers, but there will still be hundreds of thousands of those...*

80 bytes per header and no indexing work. Might take 1 minute.

> ***Quote***
> *uncompressed data using a protocol (bitcoin P2P) that wasn't designed for bulk data transfer.*

The data is mostly hashes and keys and signatures that are uncompressible.

The speed of initial download is not a reflection of the bulk data transfer rate of the protocol. The gating factor is the indexing while it downloads.

BitcoinTalk

Re: Is safe running bitcoins with the same wallet on more computers simultaneously?

2010-11-28 18:06:39 UTC - -

> ***Quote***
> *Will it be synchronized automatically?*

Very much not. Using multiple copies of wallet.dat is not recommended or supported, in fact all of Bitcoin is designed to defeat that. Both copies will get screwed up.

If you're trying to consolidate your generated coins into one wallet, a better solution now is to run getwork miners on the additional systems. jgarzik has a CPU miner, and it supports tcatm's 4-way SSE2, so on Windows it's up to twice as fast as the built-in SHA if you have an AMD or recent Intel (core 3, 5 or 7).

New demonstration CPU miner available:
http://BitcoinTalk.org/index.php?topic=1925.0

BitcoinTalk

Re: RFC: ship block chain 1-74000 with release tarballs?

2010-11-29 20:19:12 UTC - -

It seems like you're inclined to assume everything is wrong more than is actually so.

Writing the block index is light work. Building the tx index is much more random access per block. I suspect reading all the prev txins is what's slow. Read caching would help that. It's best if the DB does that. Maybe it has a setting for how much cache memory to use.

Quote

1) bitcoin should be opening databases, not just environment, at program startup, and closing database at program shutdown.

Already does that. See CDB. The lifetime of the (for instance) CTxDB object is only to support database transactions and to know if anything is still using the database at shutdown.

Quote

And, additionally, bitcoin forces a database checkpoint, pushing all transactions from log into main database.

If it was doing that it would be much slower. It's supposed to be only once a minute or 500 blocks:

```
   if (strFile == "blkindex.dat" && IsInitialBlockDownload() && nBestHeight % 500 != 0)
       nMinutes = 1;
   dbenv.txn_checkpoint(0, nMinutes, 0);
```

Probably should add this:

```
   if (!fReadOnly)
       dbenv.txn_checkpoint(0, nMinutes, 0);
```

Quote

2) For the initial block download, txn commit should occur once every N records, not every record. I suggest N=1000.

Does transaction commit imply flush? That seems surprising to me. I assume a database op wrapped in a transaction would be logged like any other database op. Many database applications need to wrap almost every pair of ops in a transaction, such as moving money from one account to another. (debit a, credit b) I can't imagine they're required to batch all their stuff up themselves.

In the following cases, would case 1 flush once and case 2 flush twice?

case 1:

write
write
write
write
checkpoint

case 2:
begin transaction
write
write
commit transaction
begin transaction
write
write
commit transaction
checkpoint

Contorting our database usage will not be the right approach. It's going to be BDB settings and caching.

BitcoinTalk

Re: Incompatible wallet format with latest bitcoin-git ?

2010-11-30 19:02:31 UTC - -

What was this wallet used with? An early accounts patch or git build?

It's while loading the wallet. I assume it must be in this:

```
else if (strType == "acentry")
{
    string strAccount;
    ssKey >> strAccount;
    uint64 nNumber;
    ssKey >> nNumber;
    if (nNumber > nAccountingEntryNumber)
        nAccountingEntryNumber = nNumber;
}
```

You could check that with this:

```
else if (strType == "acentry")
{
    string strAccount;
    assert(!ssKey.empty());
    ssKey >> strAccount;
```

```
        uint64 nNumber;
        if (ssKey.size() != 8 )
            printf("***** %s %d ", strAccount.c_str(), ssKey.size());
        assert(ssKey.empty() == false);
        ssKey >> nNumber;
        if (nNumber > nAccountingEntryNumber)
            nAccountingEntryNumber = nNumber;
    }
```

Was there an interim version of accounts on git at some point that had just ("acentry", "account") for the key?

If you have gdb, you could run it in gdb and do a backtrace.

gdb --args bitcoin ...
run
(wait for exception)
bt

BitcoinTalk

Re: RFC: ship block chain 1-74000 with release tarballs?

2010-12-01 21:25:39 UTC - -

That's a good optimisation. I'll add that next time I update SVN.

More generally, we could also consider this:

```
        dbenv.set_lk_max_objects(10000);
        dbenv.set_errfile(fopen(strErrorFile.c_str(), "a")); /// debug
        dbenv.set_flags(DB_AUTO_COMMIT, 1);
+       dbenv.set_flags(DB_TXN_NOSYNC, 1);
        ret = dbenv.open(strDataDir.c_str(),
                         DB_CREATE     |
                         DB_INIT_LOCK  |
                         DB_INIT_LOG   |
```

We would then rely on dbenv.txn_checkpoint(0, 0, 0) in CDB::Close() to flush after wallet writes.

BitcoinTalk

Re: Wikileaks contact info?

2010-12-05 09:08:08 UTC - -

Quote from: RHorning on December 04, 2010, 10:17:44 PM

Basically, bring it on. Let's encourage Wikileaks to use Bitcoins and I'm willing to face any risk or fallout from that act.

No, don't "bring it on".

The project needs to grow gradually so the software can be strengthened along the way.

I make this appeal to WikiLeaks not to try to use Bitcoin. Bitcoin is a small beta community in its infancy. You would not stand to get more than pocket change, and the heat you would bring would likely destroy us at this stage.

BitcoinTalk

Re: JSON-RPC method idea: list transactions newer than a given txid

2010-12-08 20:21:49 UTC - -

It's not safe to use listtransactions this way.

I know I've been criticized for being reluctant about listtransactions. Let me explain my reluctance.

Transactions are dynamic. Past transactions can become unconfirmed, go away and come back, become invalid and disappear, or be replaced by a different double-spend. Their date can change, their order can change.

Programmers are naturally inclined to want to use listtransactions like this: feed me the new transactions since I last asked, and I'll keep my own tally or static record of them. This will seem to work in all regular use, but if you use the amounts for anything, it is highly exploitable:
1) How do you know if a past transaction becomes invalid and disappears?
2) When there's a block-chain reorg, it would be easy to double-count transactions when they get confirmed again.
3) A transaction can be replaced by a double-spend with a different txid. You would count both spends.

The model where you assume you only need to see new transactions because you've already seen previous transactions is not true. Old transactions can change at any time.

Any time you take an action based on payment amounts received, you always need to go back to bitcoin and ask for a current balance total (or use move or sendfrom), and be ready for the possibility that it can go down.

Now that we have the Accounts feature making it easier to do it the right way, we're better prepared to have listtransactions.

BitcoinTalk

Re: JSON-RPC method idea: list transactions newer than a given txid

2010-12-08 22:36:45 UTC - -

Then how do you cope with the issues I listed in the message you quoted?

bitcoin-list

[**bitcoin-list**] Bitcoin 0.3.18 is released

2010-12-08 23:11:55 UTC - -

Version 0.3.18 is now available.

Changes:
- Fixed a wallet.dat compatibility problem if you downgraded from 0.3.17 and then upgraded again
- IsStandard() check to only include known transaction types in blocks
- Jgarzik's optimisation to speed up the initial block download a little

The main addition in this release is the Accounts-based JSON-RPC commands that Gavin's been working on (more details at http://www.bitcoin.org/smf/index.php?topic=1886.0).
- getaccountaddress
- sendfrom
- move
- getbalance
- listtransactions

Download:
http://sourceforge.net/projects/bitcoin/files/Bitcoin/bitcoin-0.3.18/

BitcoinTalk

Version 0.3.18

2010-12-08 23:19:24 UTC - -

Changes:
- Fixed a wallet.dat compatibility problem if you downgraded from 0.3.17 and then upgraded again
- IsStandard() check to only include known transaction types in blocks
- Jgarzik's optimisation to speed up the initial block download a little

The main addition in this release is the Accounts-Based JSON-RPC commands that Gavin's been working on (more details at http://BitcoinTalk.org/index.php?topic=1886.0).
- getaccountaddress
- sendfrom
- move
- getbalance
- listtransactions

Download:
http://sourceforge.net/projects/bitcoin/files/Bitcoin/bitcoin-0.3.18/

BitcoinTalk

Re: JSON-RPC method idea: list transactions newer than a given txid

2010-12-09 00:12:17 UTC - -

I'm not talking about the normal risk for a given minconf level, I'm talking about additional pitfalls from listtransactions when used this way.

Quote from: satoshi on December 08, 2010, 10:36:45 PM

2) When there's a block-chain reorg, it would be easy to double-count transactions when they get confirmed again.

The OP's example of listtransactions <account> [count=10] [txid] seems to imply and it would be very easy for programmers to assume that if they pass in the last txid of the previous call to listtransactions, they will never see the same transaction more than once, which is not the case. It would be very easy to double-count payments if you don't maintain your own persistent map or dictionary to track which txid's you've already accepted.

It doesn't seem right to have a function that seems tailor made to be used a certain obvious way, and that way is a non-obvious trap.

Quote from: jgarzik on December 08, 2010, 11:07:22 PM
Quote from: satoshi on December 08, 2010, 10:36:45 PM

3) A transaction can be replaced by a double-spend with a different txid. You would count both spends.

listtransactions does not add anything to this problem, beyond that which is already vulnerable through listreceivedbyaddress.

Suppose both spends are to the same address. getreceivedbyaddress would always count only one or the other spend at any given time, never both.

Using listtransactions, it would be very easy to count both. You see the first spend, you count it. You see the second spend, you count it. Total is double counted.

BitcoinTalk

Re: Version 0.3.18

2010-12-09 14:37:05 UTC - -

New transaction templates can be added as needed. Within a few days, there will be plenty of GPU power that accepts and works on it. Network support will be thorough *long before* there'll be enough clients who understand how to receive and interpret the new transaction.

Timestamp hashes are still already possible:

txin: 0.01
txout: 0.00 <appid, hash> OP_CHECKSIG
fee: 0.01

If there's an actual application like BitDNS getting ready to actually start inserting hashes, we can always add a specific transaction template for timestamps.

I like Hal Finney's idea for user-friendly timestamping. Convert the hash of a file to a bitcoin address and send 0.01 to it:

Quote from: Hal on December 05, 2010, 11:43:56 PM

I thought of a simple way to implement the timestamp concept I mentioned above. Run sha1sum on the file you want to timestamp. Convert the result to a Bitcoin address, such as via http://blockexplorer.com/q/hashtoaddress . Then send a small payment to that address.

The money will be lost forever, as there is no way to spend it further, but the timestamp Bitcoin address will remain in the block chain as a record of the file's existence.

I understand that this is arguably not a good use of the Bitcoin distributed database, but nothing stops people from doing this so we should be aware that it may be done.

BitcoinTalk

Re: Version 0.3.18

2010-12-09 15:17:53 UTC - -

I came to agree with Gavin about whitelisting when I realized how quickly new transaction types can be added.

> ***Quote from: nanotube on December 09, 2010, 06:19:05 AM***
> *why not make it easier on everyone and just allow say, 64 or 128 bytes of random data in a transaction?*

That's already possible. <pubkey> OP_CHECKSIG. <pubkey> can be 33 to 120 bytes.

I also support a third transaction type for timestamp hash sized arbitrary data. There's no point not having one since you can already do it anyway. It would tell nodes they don't need to bother to index it.

BitcoinTalk

Re: JSON-RPC method idea: list transactions newer than a given txid

2010-12-09 18:08:08 UTC - -

> ***Quote from: jgarzik on December 09, 2010, 12:58:05 AM***
> *I agree with you and satoshi about "txs after <txid>". My listtransactions (now xlisttransactions) patch pointedly does not have that feature, and never has.*

As long as the interface is designed for things like showing the user the last N transactions history, it's fine, now that we have the Accounts feature making it easier to do payment detection the right way.

Gavin, could listtransactions have an option to list transactions for all accounts?

I'm not sure what the interface could be, maybe:
listtransactions <JSON null type> [count]

It would be hard to do that from the command line though.

I can't think of a good solution for the interface, that's the problem. Maybe "*" special case like "" is. Everyone would have to make sure no user can create account name "*".

> ***Quote from: jgarzik on December 09, 2010, 04:13:50 PM***
> *Sure, and that's easy enough to track with transactions.*

I don't get how that's "easy" to track with transactions.

BitcoinTalk

Re: Automated nightly builds

2010-12-09 18:28:45 UTC - -

Thanks for setting this up Cdecker.

Is there any chance of getting it to build the GUI version also? If this is Ubuntu, if you get wxWidgets 2.9.0 it should just be a matter of following the steps in build-unix.txt exactly. Is this an environment where you can build wxWidgets once and leave it there and just keep using it?

BitcoinTalk

Re: BitDNS and Generalizing Bitcoin

2010-12-09 21:02:42 UTC - -

I think it would be possible for BitDNS to be a completely separate network and separate block chain, yet share CPU power with Bitcoin. The only overlap is to make it so miners can search for proof-of-work for both networks simultaneously.

The networks wouldn't need any coordination. Miners would subscribe to both networks in parallel. They would scan SHA such that if they get a hit, they potentially solve both at once. A solution may be for just one of the networks if one network has a lower difficulty.

I think an external miner could call getwork on both programs and combine the work. Maybe call Bitcoin, get work from it, hand it to BitDNS getwork to combine into a combined work.

Instead of fragmentation, networks share and augment each other's total CPU power. This would solve the problem that if there are multiple networks, they are a danger to each other if the available CPU power gangs up on one. Instead, all networks in the world would share combined CPU power, increasing the total strength. It would make it easier for small networks to get started by tapping into a ready base of miners.

BitcoinTalk

Re: BitDNS and Generalizing Bitcoin

2010-12-09 22:46:50 UTC - -

Quote from: nanotube on December 09, 2010, 09:20:40 PM

308

seems that the miner would have to basically do "extra work". and if there's no reward from the bitdns mining from the extra work (which of course, slows down the main bitcoin work), what would be a miner's incentive to include bitdns (and whatever other side chains) ?

The incentive is to get the rewards from the extra side chains also for the same work.

While you are generating bitcoins, why not also get free domain names for the *same work*?

If you currently generate 50 BTC per week, now you could get 50 BTC and some domain names too.

You have one piece of work. If you solve it, it will solve a block from both Bitcoin and BitDNS. In concept, they're tied together by a Merkle Tree. To hand it in to Bitcoin, you break off the BitDNS branch, and to hand it in to BitDNS, you break off the Bitcoin branch.

In practice, to retrofit it for Bitcoin, the BitDNS side would have to have maybe ~200 extra bytes, but that's not a big deal. You've been talking about 50 domains per block, which would dwarf that little 200 bytes per block for backward compatibility. We could potentially schedule a far in future block when Bitcoin would upgrade to a modernised arrangement with the Merkle Tree on top, if we care enough about saving a few bytes.

Note that the chains are below this new Merkle Tree. That is, each of Bitcoin and BitDNS have their own chain links inside their blocks. This is inverted from the common timestamp server arrangement, where the chain is on top and then the Merkle Tree, because that creates one common master chain. This is two timestamp servers not sharing a chain.

BitcoinTalk

Re: Fees in BitDNS confusion

2010-12-09 23:58:54 UTC - -

Not locktime.

There's a possible design for far in the future:

You intentionally write a double-spend. You write it with the same inputs and outputs, but this time with a fee. When your double-spend gets into a block, the first spend becomes invalid. The payee does not really notice, because at the moment the new transaction becomes valid, the old one becomes invalid, and the new transaction simply takes its place.

It's easier said than implemented. There would be a fair amount of work to make a client that correctly writes the double-spend, manages the two versions in the wallet until one is chosen, handles all the corner cases. Every assumption in the existing code is that you're not trying to write double-spends.

There would need to be some changes on the Bitcoin Miner side also, to make the possibility to accept a double-spend into the transaction pool, but only strictly if the inputs and outputs match and the transaction fee is higher. Currently, double-spends are never accepted into the transaction pool, so every node bears witness to which transaction it saw first by working to put it into a block.

BitcoinTalk

Re: BitDNS and Generalizing Bitcoin

2010-12-10 17:29:28 UTC - -

Piling every proof-of-work quorum system in the world into one dataset doesn't scale.

Bitcoin and BitDNS can be used separately. Users shouldn't have to download all of both to use one or the other. BitDNS users may not want to download everything the next several unrelated networks decide to pile in either.

The networks need to have separate fates. BitDNS users might be completely liberal about adding any large data features since relatively few domain registrars are needed, while Bitcoin users might get increasingly tyrannical about limiting the size of the chain so it's easy for lots of users and small devices.

Fears about securely buying domains with Bitcoins are a red herring. It's easy to trade Bitcoins for other non-repudiable commodities.

If you're still worried about it, it's cryptographically possible to make a risk free trade. The two parties would set up transactions on both sides such that when they both sign the transactions, the second signer's signature triggers the release of both. The second signer can't release one without releasing the other.

BitcoinTalk

Accounts example code

2010-12-10 19:21:03 UTC - -

Some sample pseudocode using the new Accounts based commands in 0.3.18.

```
print "send to " + getaccountaddress(username) + " to fund your account"
```

```
print "balance: " + getbalance(username, 0)
print "available balance: " + getbalance(username, 6)

// if you make a sale, move the money from their account to your "" account
if (move(username, "", amount, 6, "purchased item"))
    SendTheGoods()

// withdrawal
sendfrom(username, bitcoinaddress, amount, 6, "withdrawal by user")
```

You can use listtransactions(username) to show them a list of their recent transactions.

BitcoinTalk

Re: BitDNS and Generalizing Bitcoin

2010-12-10 19:55:12 UTC - -

Quote from: Hal on December 10, 2010, 07:14:04 PM

additional block chains would each create their own flavor of coins, which would trade with bitcoins on exchanges? These chain-specific coins would be used to reward miners on those chains, and to purchase some kinds of rights or privileges within the domain of that chain?

Right, the exchange rate between domains and bitcoins would float.

A longer interval than 10 minutes would be appropriate for BitDNS.

So far in this discussion there's already a lot of housekeeping data required. It will be much easier if you can freely use all the space you need without worrying about paying fees for expensive space in Bitcoin's chain. Some transactions:

Changing the IP record.

Name change. A domain object could entitle you to one domain, and you could change it at will to any name that isn't taken. This would encourage users to free up names they don't want anymore. Generated domains start out blank and the miner sells it to someone who changes it to what they want.

Renewal. Could be free, or maybe require consuming another domain object to renew. In that case, domain objects (domaincoins?) could represent the right to own a domain for a year. The spent fee goes to the miners in the next block fee.

BitcoinTalk

Re: BitDNS and Generalizing Bitcoin

2010-12-10 20:19:39 UTC - -

I agree. All transactions, IP changes, renewals, etc. should have some fee that goes to the miners.

You might consider a certain amount of work to generate a domain, instead of a fixed total circulation. The work per domain could be on a schedule that grows with Moore's Law. That way the number of domains would grow with demand and the number of people using it.

BitcoinTalk

Re: BitDNS and Generalizing Bitcoin

2010-12-11 13:08:30 UTC - -

@dtvan: all 3 excellent points.
1) IP records don't need to be in the chain, just do registrar function not DNS. And CA problem solved, neat.
2) Pick one TLD, .web +1.
3) Expiration and significant renewal costs, very important.

Quote from: joe on December 11, 2010, 10:53:58 AM
However, thinking more about this now I support inclusion of additional coinbases / tracking systems in the main network. The reason for doing this is so as not to water down CPU power into multiple networks. We want one strong network, so the network should be versatile.

Avoiding CPU power fragmentation is no longer a reason. Independent networks/chains can share CPU power without sharing much else. See: http://BitcoinTalk.org/index.php?topic=1790.msg28696#msg28696 and http://BitcoinTalk.org/index.php?topic=1790.msg28715#msg28715

BitcoinTalk

Re: Bitcoin and buffer overflow attacks

2010-12-11 13:32:37 UTC - -

Quote from: da2ce7 on December 11, 2010, 05:49:22 AM
direct to IP address transfers seems like a obvious surface area to attack.

If you ever find anyone who turned it on. It's disabled by default.

Quote from: witchspace on December 11, 2010, 09:59:40 AM

There is no way to be absolutely sure that there are no buffer overflow attacks. Although it would help to implement the client in a language that doesn't have buffer overflows because it checks array indices (Python, Java, C#, ...).

It's all STL. There are almost no buffers.

BitcoinTalk

Re: minimalistic bitcoin client on D language?

2010-12-11 22:07:04 UTC - -

Quote from: Hal on December 11, 2010, 08:08:45 PM

I'd like to hear some specific criticisms of the code. To me it looks like an impressive job, although I'd wish for more comments. Now I've mostly studied the init, main, script and a bit of net modules. This is some powerful machinery.

That means a lot coming from you, Hal. Thanks.

BitcoinTalk

Re: PC World Article on Bitcoin

2010-12-11 23:39:16 UTC - -

It would have been nice to get this attention in any other context. WikiLeaks has kicked the hornet's nest, and the swarm is headed towards us.

BitcoinTalk

Added some DoS limits, removed safe mode (0.3.19)

2010-12-12 18:22:33 UTC - -

There's more work to do on DoS, but I'm doing a quick build of what I have so far in case it's needed, before venturing into more complex ideas. The build for this is version 0.3.19.

- Added some DoS controls
As Gavin and I have said clearly before, the software is not at all resistant to DoS attack. This is one improvement, but there are still more ways to attack than I can count.

I'm leaving the -limitfreerelay part as a switch for now and it's there if you need it.

- Removed "safe mode" alerts
"safe mode" alerts was a temporary measure after the 0.3.9 overflow bug. We can say all we want that users can just run with "-disablesafemode", but it's better just not to have it for the sake of appearances. It was never intended as a long term feature. Safe mode can still be triggered by seeing a longer (greater total PoW) invalid block chain.

Builds:
http://sourceforge.net/projects/bitcoin/files/Bitcoin/bitcoin-0.3.19/

bitcoin-list

[**bitcoin-list**] Bitcoin 0.3.19 is released

2010-12-13 16:12:09 UTC - -

This is a minor release to add some DoS protection.

Changes:
- Added some DoS limits, though it's still far from DoS resistant.
- Removed "safe mode" alerts.

http://www.bitcoin.org/smf/index.php?topic=2228.0

Download:
http://sourceforge.net/projects/bitcoin/files/Bitcoin/bitcoin-0.3.19/

Mike Hearn *<mike@plan99.net> Mon, Dec 27, 2010 at 8:21 PM*
To: Satoshi Nakamoto <satoshin@gmx.com>

Happy Christmas Satoshi, assuming you celebrate it wherever you are in the world :-)

I have been working on a Java implementation of the simplified payment verification, with an eye to building a client that runs on Android phones. So I've been thinking a lot about storage requirements and the scalability of BitCoin, which led to some questions that the paper did not answer (maybe there could be a new version of the paper at some point, as I think aspects of it are now out of date).

Specifically, BitCoin has a variety of magic numbers and neither the code nor the paper explain where they came from. For example, the fact that inflation ceases when 21 million coins have been issued. This

number must have been arrived at somehow, but I can't see how.

Another is the 10 minute block target. I understand this was chosen to allow transactions to propagate through the network. However existing large P2P networks like BGP can propagate new data worldwide in <1 minute.

The final number I'm interested in is the 500kb limit on block sizes. According to Wikipedia, Visa alone processed 62 billion transactions in 2009. Dividing through we get an average of 2000 transactions per second, so peak rate is probably around double that at 4000 transactions/sec. With a ten minute block target, at peak a block might need to contain 2.4 million transactions, which just won't fit into 500kb. Is this 500kb a temporary limitation that will be slowly removed over time from the official client or something more fundamental?

Satoshi Nakamoto <satoshin@gmx.com> Wed, Dec 29, 2010 at 10:42 PM

To: Mike Hearn <mike@plan99.net>

I have been working on a Java implementation of the simplified payment verification, with an eye to building a client that runs on Android phones. So I've been thinking a lot about storage requirements and the scalability of BitCoin, which led to some questions that the paper did not answer (maybe there could be a new version of the paper at some point, as I think aspects of it are now out of date).

The simplified payment verification in the paper imagined you would receive transactions directly, as with sending to IP address which nobody uses, or a node would index all transactions by public key and you could download them like downloading mail from a mail server.

Instead, I think client-only nodes should receive full blocks so they can scan them for their own transactions. They don't need to store them or index them. For the initial download, they only need to download headers, since there couldn't be any payments before the first time the program was run (a header download command was added in 0.3.18). From then on, they download full blocks (but only store the headers).

Code for client-only mode is mostly implemented. There's a feature branch on github with it, also I'm attaching the patch to this message.

Here's some more about it:

"Here's my client-mode implementation so far. Client-only mode only records block headers and doesn't use the tx index. It can't generate, but it can still send and receive transactions. It's not fully finished for use by end-users, but it doesn't matter because it's a complete no-op if fClient is not enabled. At this point it's mainly documentation showing the cut-lines for client-only re-implementers.

With fClient=true, I've only tested the header-only initial download.

A little background. CBlockIndex contains all the information of the block header, so to operate with headers only, I just maintain the CBlockIndex structure as usual. The nFile/nBlockPos are null, since the full block is not recorded on disk.

The code to gracefully switch between client-mode on/off without deleting blk*.dat in between is not implemented yet. It would mostly be a matter of having non-client LoadBlockIndex ignore block index entries with null block pos. That would make it re-download those as full blocks. Switching back to client-mode is no problem, it doesn't mind if the full blocks are there.

If the initial block download becomes too long, we'll want client mode as an option so new users can get running quickly. With graceful switch-off of client mode, they can later turn off client mode and have it download the full blocks if they want to start generating. They should rather just use a getwork miner to join a pool instead.

Client-only re-implementations would not need to implement EvalScript at all, or at most just implement the five ops used by the standard transaction templates."

Specifically, BitCoin has a variety of magic numbers and neither the code nor the paper explain where they came from. For example, the fact that inflation ceases when 21 million coins have been issued. This number must have been arrived at somehow, but I can't see how.

Educated guess, and the maths work out to round numbers. I wanted something that would be not too low if it was very popular and not too high if it wasn't.

Another is the 10 minute block target. I understand this was chosen to allow transactions to propagate through the network. However existing large P2P networks like BGP can propagate new data worldwide in <1 minute.

If propagation is 1 minute, then 10 minutes was a good guess. Then nodes are only losing 10% of their work (1 minute/10 minutes). If the CPU time wasted by latency was a more significant share, there may be weaknesses I haven't thought of. An attacker would not be affected by latency, since he's chaining his own blocks, so he would have an advantage. The chain would temporarily fork more often due to

latency.

The final number I'm interested in is the 500kb limit on block sizes. According to Wikipedia, Visa alone processed 62 billion transactions in 2009. Dividing through we get an average of 2000 transactions per second, so peak rate is probably around double that at 4000 transactions/sec. With a ten minute block target, at peak a block might need to contain 2.4 million transactions, which just won't fit into 500kb. Is this 500kb a temporary limitation that will be slowly removed over time from the official client or something more fundamental?

A higher limit can be phased in once we have actual use closer to the limit and make sure it's working OK.

Eventually when we have client-only implementations, the block chain size won't matter much. Until then, while all users still have to download the entire block chain to start, it's nice if we can keep it down to a reasonable size.

With very high transaction volume, network nodes would consolidate and there would be more pooled mining and GPU farms, and users would run client-only. With dev work on optimising and parallelising, it can keep scaling up.

Whatever the current capacity of the software is, it automatically grows at the rate of Moore's Law, about 60% per year.

```
diff -u old\db.cpp new\db.cpp
--- old\db.cpp  Sat Dec 18 18:35:59 2010
+++ new\db.cpp  Sun Dec 19 20:53:59 2010
@@ -464,29 +464,32 @@
     ReadBestInvalidWork(bnBestInvalidWork);

     // Verify blocks in the best chain
-    CBlockIndex* pindexFork = NULL;
-    for (CBlockIndex* pindex = pindexBest; pindex && pindex->pprev; pindex =
pindex->pprev)
+    if (!fClient)
     {
-        if (pindex->nHeight < nBestHeight-2500 && !mapArgs.count("-
checkblocks"))
-            break;
-        CBlock block;
-        if (!block.ReadFromDisk(pindex))
-            return error("LoadBlockIndex() : block.ReadFromDisk failed");
```

```
-       if (!block.CheckBlock())
+       CBlockIndex* pindexFork = NULL;
+       for (CBlockIndex* pindex = pindexBest; pindex && pindex->pprev; pindex =
pindex->pprev)
        {
-           printf("LoadBlockIndex() : *** found bad block at %d, hash=%s\n",
pindex->nHeight, pindex->GetBlockHash().ToString().c_str());
-           pindexFork = pindex->pprev;
+           if (pindex->nHeight < nBestHeight-2500 && !mapArgs.count("-
checkblocks"))
+               break;
+           CBlock block;
+           if (!block.ReadFromDisk(pindex))
+               return error("LoadBlockIndex() : block.ReadFromDisk failed");
+           if (!block.CheckBlock())
+           {
+               printf("LoadBlockIndex() : *** found bad block at %d, hash=%s\n",
pindex->nHeight, pindex->GetBlockHash().ToString().c_str());
+               pindexFork = pindex->pprev;
+           }
+       }
+       if (pindexFork)
+       {
+           // Reorg back to the fork
+           printf("LoadBlockIndex() : *** moving best chain pointer back to block
%d\n", pindexFork->nHeight);
+           CBlock block;
+           if (!block.ReadFromDisk(pindexFork))
+               return error("LoadBlockIndex() : block.ReadFromDisk failed");
+           CTxDB txdb;
+           block.SetBestChain(txdb, pindexFork);
        }
-   }
-   if (pindexFork)
-   {
-       // Reorg back to the fork
-       printf("LoadBlockIndex() : *** moving best chain pointer back to block
%d\n", pindexFork->nHeight);
-       CBlock block;
-       if (!block.ReadFromDisk(pindexFork))
-           return error("LoadBlockIndex() : block.ReadFromDisk failed");
-       CTxDB txdb;
-       block.SetBestChain(txdb, pindexFork);
    }

    return true;
```

```
diff -u old\main.cpp new\main.cpp
--- old\main.cpp        Sat Dec 18 18:35:59 2010
+++ new\main.cpp        Sun Dec 19 20:53:59 2010
@@ -637,6 +637,9 @@
     if (!IsStandard())
         return error("AcceptToMemoryPool() : nonstandard transaction type");

+    if (fClient)
+        return true;
+
     // Do we already have it?
     uint256 hash = GetHash();
     CRITICAL_BLOCK(cs_mapTransactions)
@@ -1308,23 +1311,26 @@
     if (!CheckBlock())
         return false;

-    //// issue here: it doesn't know the version
-    unsigned int nTxPos = pindex->nBlockPos + ::GetSerializeSize(CBlock(),
SER_DISK) - 1 + GetSizeOfCompactSize(vtx.size());
-
-    map<uint256, CTxIndex> mapUnused;
-    int64 nFees = 0;
-    foreach(CTransaction& tx, vtx)
+    if (!fClient)
     {
-        CDiskTxPos posThisTx(pindex->nFile, pindex->nBlockPos, nTxPos);
-        nTxPos += ::GetSerializeSize(tx, SER_DISK);
+        //// issue here: it doesn't know the version
+        unsigned int nTxPos = pindex->nBlockPos + ::GetSerializeSize(CBlock(),
SER_DISK) - 1 + GetSizeOfCompactSize(vtx.size());
+
+        map<uint256, CTxIndex> mapUnused;
+        int64 nFees = 0;
+        foreach(CTransaction& tx, vtx)
+        {
+            CDiskTxPos posThisTx(pindex->nFile, pindex->nBlockPos, nTxPos);
+            nTxPos += ::GetSerializeSize(tx, SER_DISK);

-        if (!tx.ConnectInputs(txdb, mapUnused, posThisTx, pindex, nFees, true, false))
+            if (!tx.ConnectInputs(txdb, mapUnused, posThisTx, pindex, nFees, true,
false))
                 return false;
+        }
+
+        if (vtx[0].GetValueOut() > GetBlockValue(pindex->nHeight, nFees))
```

```
            return false;
    }

-   if (vtx[0].GetValueOut() > GetBlockValue(pindex->nHeight, nFees))
-       return false;
-
    // Update block index on disk without changing it in memory.
    // The memory index structure will be changed after the db commits.
    if (pindex->pprev)
@@ -1378,7 +1384,7 @@
    foreach(CBlockIndex* pindex, vDisconnect)
    {
        CBlock block;
-       if (!block.ReadFromDisk(pindex))
+       if (!block.ReadFromDisk(pindex, !fClient))
            return error("Reorganize() : ReadFromDisk for disconnect failed");
        if (!block.DisconnectBlock(txdb, pindex))
            return error("Reorganize() : DisconnectBlock failed");
@@ -1395,7 +1401,7 @@
    {
        CBlockIndex* pindex = vConnect[i];
        CBlock block;
-       if (!block.ReadFromDisk(pindex))
+       if (!block.ReadFromDisk(pindex, !fClient))
            return error("Reorganize() : ReadFromDisk for connect failed");
        if (!block.ConnectBlock(txdb, pindex))
        {
@@ -1526,7 +1532,7 @@

    txdb.Close();

-   if (pindexNew == pindexBest)
+   if (!fClient && pindexNew == pindexBest)
    {
        // Notify UI to display prev block's coinbase if it was ours
        static uint256 hashPrevBestCoinBase;
@@ -1547,10 +1553,6 @@
    // These are checks that are independent of context
    // that can be verified before saving an orphan block.

-   // Size limits
-   if (vtx.empty() || vtx.size() > MAX_BLOCK_SIZE || ::GetSerializeSize(*this,
SER_NETWORK) > MAX_BLOCK_SIZE)
-       return error("CheckBlock() : size limits failed");
-
    // Check proof of work matches claimed amount
```

```
    if (!CheckProofOfWork(GetHash(), nBits))
        return error("CheckBlock() : proof of work failed");
@@ -1559,6 +1561,13 @@
    if (GetBlockTime() > GetAdjustedTime() + 2 * 60 * 60)
        return error("CheckBlock() : block timestamp too far in the future");

+    if (fClient && vtx.empty())
+        return true;
+
+    // Size limits
+    if (vtx.empty() || vtx.size() > MAX_BLOCK_SIZE || ::GetSerializeSize(*this,
SER_NETWORK) > MAX_BLOCK_SIZE)
+        return error("CheckBlock() : size limits failed");
+
    // First transaction must be coinbase, the rest must not be
    if (vtx.empty() || !vtx[0].IsCoinBase())
        return error("CheckBlock() : first tx is not coinbase");
@@ -1623,13 +1632,14 @@
        return error("AcceptBlock() : out of disk space");
    unsigned int nFile = -1;
    unsigned int nBlockPos = 0;
-    if (!WriteToDisk(nFile, nBlockPos))
-        return error("AcceptBlock() : WriteToDisk failed");
+    if (!fClient)
+        if (!WriteToDisk(nFile, nBlockPos))
+            return error("AcceptBlock() : WriteToDisk failed");
    if (!AddToBlockIndex(nFile, nBlockPos))
        return error("AcceptBlock() : AddToBlockIndex failed");

    // Relay inventory, but don't relay old inventory during initial block download
-    if (hashBestChain == hash)
+    if (!fClient && hashBestChain == hash)
        CRITICAL_BLOCK(cs_vNodes)
            foreach(CNode* pnode, vNodes)
                if (nBestHeight > (pnode->nStartingHeight != -1 ? pnode-
>nStartingHeight - 2000 : 55000))
@@ -2405,6 +2415,8 @@
        {
            if (fShutdown)
                return true;
+            if (fClient && inv.type == MSG_TX)
+                continue;
            pfrom->AddInventoryKnown(inv);

            bool fAlreadyHave = AlreadyHave(txdb, inv);
@@ -2441,6 +2453,9 @@
```

```
          if (inv.type == MSG_BLOCK)
          {
+             if (fClient)
+                 return true;
+
              // Send block from disk
              map<uint256, CBlockIndex*>::iterator mi =
mapBlockIndex.find(inv.hash);
              if (mi != mapBlockIndex.end())
@@ -2486,6 +2501,8 @@

    else if (strCommand == "getblocks")
    {
+       if (fClient)
+           return true;
        CBlockLocator locator;
        uint256 hashStop;
        vRecv >> locator >> hashStop;
@@ -2556,6 +2573,8 @@

    else if (strCommand == "tx")
    {
+       if (fClient)
+           return true;
        vector<uint256> vWorkQueue;
        CDataStream vMsg(vRecv);
        CTransaction tx;
@@ -2620,6 +2639,33 @@

        if (ProcessBlock(pfrom, &block))
            mapAlreadyAskedFor.erase(inv);
+   }
+
+
+   else if (strCommand == "headers")
+   {
+       if (!fClient)
+           return true;
+       vector<CBlock> vHeaders;
+       vRecv >> vHeaders;
+
+       uint256 hashBestBefore = hashBestChain;
+       foreach(CBlock& block, vHeaders)
+       {
+           block.vtx.clear();
```

```
+
+           printf("received header %s\n",
block.GetHash().ToString().substr(0,20).c_str());
+
+           CInv inv(MSG_BLOCK, block.GetHash());
+           pfrom->AddInventoryKnown(inv);
+
+           if (ProcessBlock(pfrom, &block))
+               mapAlreadyAskedFor.erase(inv);
+       }
+
+       // Request next batch
+       if (hashBestChain != hashBestBefore)
+           pfrom->PushGetBlocks(pindexBest, uint256(0));
    }


diff -u old\main.h new\main.h
--- old\main.h  Sat Dec 18 18:35:59 2010
+++ new\main.h  Sun Dec 19 20:53:59 2010
@@ -619,6 +619,8 @@

    bool ReadFromDisk(CDiskTxPos pos, FILE** pfileRet=NULL)
    {
+       assert(!fClient);
+
        CAutoFile filein = OpenBlockFile(pos.nFile, 0, pfileRet ? "rb+" : "rb");
        if (!filein)
            return error("CTransaction::ReadFromDisk() : OpenBlockFile failed");
@@ -1174,6 +1176,7 @@

    bool ReadFromDisk(unsigned int nFile, unsigned int nBlockPos, bool
fReadTransactions=true)
    {
+       assert(!fClient);
        SetNull();

        // Open history file to read
@@ -1231,7 +1234,7 @@


 //
-// The block chain is a tree shaped structure starting with the
+// The block index is a tree shaped structure starting with the
 // genesis block at the root, with each block potentially having multiple
 // candidates to be the next block.  pprev and pnext link a path through the
```

```
 // main/longest chain.  A blockindex may have multiple pprev pointing back
diff -u old\net.cpp new\net.cpp
--- old\net.cpp Wed Dec 15 22:33:09 2010
+++ new\net.cpp Sun Dec 19 21:51:27 2010
@@ -51,7 +51,15 @@
     pindexLastGetBlocksBegin = pindexBegin;
     hashLastGetBlocksEnd = hashEnd;

-    PushMessage("getblocks", CBlockLocator(pindexBegin), hashEnd);
+    /// Client todo: After the initial block header download, start using getblocks
+    /// here instead of getheaders.  For blocks generated after the first time the
+    /// program was run, we need to download full blocks to watch for received
+    /// transactions in them.  We're able to download headers only for blocks
+    /// generated before we ever ran because they can't contain txes for us.
+    if (::fClient)
+        PushMessage("getheaders", CBlockLocator(pindexBegin), hashEnd);
+    else
+        PushMessage("getblocks", CBlockLocator(pindexBegin), hashEnd);
 }
```

client-mode.patch
11K

Mike Hearn <mike@plan99.net> Thu, Dec 30, 2010 at 12:27 AM
To: Satoshi Nakamoto <satoshin@gmx.com>

Thanks for the info.

I reached the same conclusions about client only nodes and this is what I've been implementing. I'm nearly there ..... I have block chain download, parsing and verification of the blocks/transactions done, with creation of spend transactions almost done.

v1 will basically do as you propose, with the possible optimization of storing only the blocks needed to form the block locator (with the exponential thinning). As Android provides local storage that is private to the app, you don't need to store the entire block chain to be able to accept new blocks ... just enough to ensure you can always stay on the longest chain.

By the way, your code is easy to read and has been an invaluable reference. So thanks for that.

In v2 I'm thinking of showing transactions before they are integrated into the block chain by running secure/locked down relay nodes that send messages to the phones when a transaction is accepted into the memory pool. Android provides a secure, low power back channel to every phone. Messages are stored server side if the device is offline and apps are automatically started on the phone to handle incoming messages.

So as long as the relay nodes are unhacked, this system should give enough trust that low value transactions can be shown in the UI immediately. It introduces some centralization/single points of failure, but if the relay mechanism dies or is hacked, the damage only lasts for 10 minutes until the new blocks are downloaded.

> Client-only re-implementations would not need to implement EvalScript at
> *all, or at most just implement the five ops used by the standard transaction*
> *templates."*

Indeed, there's no point in client-only implementations implementing EvalScript because they can't verify transactions aren't being double spent without storing and indexing the entire block chain. My code parses the scripts and then relies on them having a standard structure, but doesn't actually run them.

> Educated guess, and the maths work out to round numbers. I wanted something
> that would be not too low if it was very popular and not too high if it
> wasn't.

It'd be interesting to see the working for this. In some sense the number of coins is arbitrary as the nanocoin representation means the issuance is so huge it's practically infinite.

> A higher limit can be phased in once we have actual use closer to the limit
> and make sure it's working OK.

It'd be worth implementing some kind of more robust auto update mechanism, or a schedule for the phase in of this, if only because when people evaluate "is BitCoin worth my time and effort" a solid plan for scaling up is good to have written down.

I'm not worried about the physical capabilities of the hardware, but more protocol ossification as the app is reimplemented and nodes which

don't auto-update themselves increase in number. Client only reimplementations pose no problems of course, but other systems like SMTP have proven impossible to globally upgrade despite having extension mechanisms built in .... just too many implementations and too many installations.

Satoshi Nakamoto <satoshin@gmx.com> Fri, Jan 7, 2011 at 1:00 PM

To: Mike Hearn <mike@plan99.net>

I reached the same conclusions about client only nodes and this is what I've been implementing. I'm nearly there ..... I have block chain download, parsing and verification of the blocks/transactions done, with creation of spend transactions almost done.

That's great! The first client-only implementation will really start to move things to the next step. Is it going to be open source, or Google proprietary?

Mike Hearn <mike@plan99.net> Fri, Jan 7, 2011 at 1:24 PM

To: Satoshi Nakamoto <satoshin@gmx.com>

> That's great! The first client-only implementation will really start to
> move things to the next step. Is it going to be open source, or Google
> proprietary?

Open source. It has to be - I am developing it as a personal project in my spare time and Googles policy is that this is only allowed if you open source the results. But I would have done that anyway.

I managed to spend my first coins on the testnet with my app a few days ago, hopefully will get another chance to make progress this weekend. Probably will have something to show publically sometime in Feb, touch wood.

Satoshi Nakamoto <satoshin@gmx.com> Mon, Jan 10, 2011 at 4:34 PM

To: Mike Hearn <mike@plan99.net>

Open source.

Perfect. Once your code shows how to simplify it down, other authors can follow your lead. Client is a less daunting challenge than full implementation. If it's within reach of more developers, they'll come up with more polished UI and other things I

didn't think of. I expect the original software will become the industrial old thing used by GPU farms and pool servers.

BTW, later a good feature for a client version is to keep your private keys encrypted and you give your password each time you send.

I managed to spend my first coins on the testnet with my app a few days ago, hopefully will get another chance to make progress this weekend. Probably will have something to show publically sometime in Feb, touch wood.

Great, keep me updated.

I wanted something that would be not too low if it was very popular and not too high if it wasn't.
It'd be interesting to see the working for this. In some sense the number of coins is arbitrary as the nanocoin representation means the issuance is so huge it's practically infinite.

It works out to an even 10 minutes per block:
21000000 / (50 BTC * 24hrs * 365days * 4years * 2) = 5.99 blocks/hour

I fudged it to 364.58333 days/year. The halving of 50 BTC to 25 BTC is after 210000 blocks or around 3.9954 years, which is approximate anyway based on the retargeting mechanism's best effort.

I thought about 100 BTC and 42 million, but 42 million seemed high.

I wanted typical amounts to be in a familiar range. If you're tossing around 100000 units, it doesn't feel scarce. The brain is better able to work with numbers from 0.01 to 1000.

If it gets really big, the decimal can move two places and cents become the new coins.

Mike Hearn <mike@plan99.net> Mon, Jan 10, 2011 at 4:48 PM
To: Satoshi Nakamoto <satoshin@gmx.com>

Ah, of course, that makes sense.

By the way, if you didn't see it already, there's a discussion on the security of

secp256k1 on the forum:

http://www.bitcoin.org/smf/index.php?topic=2699.0

Hal (i presume this is Hal Finney) seems to think the curve is at higher risk of attack than random curves. I guess you chose secp256k1 for the mentioned performance improvement?

[Quoted text hidden]

Satoshi Nakamoto <satoshin@gmx.com> Mon, Jan 10, 2011 at 8:47 PM

To: Mike Hearn <mike@plan99.net>

By the way, if you didn't see it already, there's a discussion on the security of secp256k1 on the forum:

http://www.bitcoin.org/smf/index.php?topic=2699.0

Hal (i presume this is Hal Finney)

Yes, it's him. He was supportive on the Cryptography list and ran one of the first nodes.

seems to think the curve is at higher risk of attack than random curves. I guess you chose secp256k1 for the mentioned performance improvement?

I must admit, this project was 2 years of development before release, and I could only spend so much time on each of the many issues. I found guidance on the recommended size for SHA and RSA, but nothing on ECDSA which was relatively new. I took the recommended key size for RSA and converted to equivalent key size for ECDSA, but then increased it so the whole app could be said to be 256-bit security. I didn't find anything to recommend a curve type so I just... picked one. Hopefully there is enough key size to make up for any deficiency.

At the time, I was concerned whether the bandwidth and storage sizes would be practical even with ECDSA. RSA's huge keys were out of the question. Storage and bandwidth seemed tighter back then. I felt the size was either only just becoming practical, or would be soon. When I presented it, I was surprised nobody else was concerned about size, though I was also surprised how many issues they argued, and more surprised that every single one was something I had thought of and solved.

As it turns out, ECDSA verification time may be the greater bottleneck. (In my tests, OpenSSL was taking 3.5ms per ECDSA verify, or about 285 verifies per second) Client versions bypass the problem.

As things have evolved, the number of people who need to run full nodes is less than I originally imagined. The network would be fine with a small number of nodes if processing load becomes heavy.

Open sourced my Java SPV impl
10 messages

Mike Hearn <mike@plan99.net> Mon, Mar 7, 2011 at 2:13 PM
To: Satoshi Nakamoto <satoshin@gmx.com>

Hi Satoshi,

I hope you are doing well. I finally got all the lawyers happy enough to release BitCoinJ under the Google name using the Apache 2 license:

http://www.bitcoin.org/smf/index.php?topic=4236.0

It's incomplete - notably it doesn't properly handle block chain splits yet - but the rest is coming. I put a lot of work into documentation and comments so hopefully it'll open up BitCoin to a new audience who weren't able to understand/build the current code. Over the next month or two I'll be finishing off some of the bigger missing pieces for a full client-mode implementation.

I know you are busy right now but I'm hoping you can find time to answer a few questions I had.

As part of doing full SPV I'm thinking of adding a getmerklebranch message to the protocol. This would return a set of {blockhash, branch} pairs given tx hashes, so allowing verification of a broadcast transaction before it was incorporated into a block without storing the full chain. Does that approach sound good to you?

Also, I've been thinking of exploring different transaction types lately, eg by removing the IsStandard() checks for the testnet. It's clear you put a lot of thought into transactions beyond simply moving coins around up front, but unfortunately none of it was in the paper or documented in the code. Escrow, multi-pay and so on are all interesting but I was wondering if you could compile a list of ideas for things we can do with the scripting language at some point.

Finally, the code that allows for transaction replacement has been disabled but the comment doesn't say why. Is this just to reduce the attack surface/complexity or is there a deeper reason? I haven't fully

understood why sequence numbers are a property of the tx inputs rather than the tx itself.

Hope you can find the time/energy to rejoin us soon! I don't know if you've seen this:

http://bitcoin.sipa.be/speed-lin.png

but it's exciting times for the network!

thanks!
/mike

Satoshi Nakamoto <satoshin@gmx.com> Wed, Mar 9, 2011 at 5:15 PM

To: Mike Hearn <mike@plan99.net>

I hope you are doing well. I finally got all the lawyers happy enough to release BitCoinJ under the Google name using the Apache 2 license:

It's incomplete - notably it doesn't properly handle block chain splits yet - but the rest is coming. I put a lot of work into documentation and comments so hopefully it'll open up BitCoin to a new audience who weren't able to understand/build the current code. Over the next month or two I'll be finishing off some of the bigger missing pieces for a full client-mode implementation.

That's great news! Much complexity can be left behind in a clean rewrite with only client requirements, and it opens it to Java developers too.

I know you are busy right now but I'm hoping you can find time to answer a few questions I had.

I'm happy to answer any questions.

As part of doing full SPV I'm thinking of adding a getmerklebranch message to the protocol. This would return a set of {blockhash, branch} pairs

That's a CMerkleTx

given tx hashes, so allowing verification of a broadcast transaction before it was incorporated into a block without storing the full chain. Does that approach sound good to you?

I don't understand. A merkle branch links a tx back to a block, which only has significance if the block exhibits proof-of-work. Linking back to an as-yet unsolved block proves nothing.

Network nodes are able to verify 0-conf txes because they have the complete tx index, so they can:
1) verify signatures against dependencies.
2) say that they haven't seen another spend yet, because they know about every tx in existence.

Are you talking about CMerkleTxes for the tx's dependencies? That would get part 1), but not part 2).

If you don't know about all txes in existence, I don't know how to do 2). You could only rely on trusting other nodes for that. That trust can be distributed over multiple nodes. Nodes only relay transactions they accept as valid. If you receive inv messages for a tx from all the nodes you're connected to, they're attesting that it's valid and the first spend they saw.

Also, I've been thinking of exploring different transaction types lately, eg by removing the IsStandard() checks for the testnet.

Very good idea. That should definitely be allowed on -testnet.

It's clear you put a lot of thought into transactions beyond simply moving coins around up front, but unfortunately none of it was in the paper or documented in the code. Escrow, multi-pay and so on are all interesting but I was wondering if you could compile a list of ideas for things we can do with the scripting language at some point.

Finally, the code that allows for transaction replacement has been disabled but the comment doesn't say why. Is this just to reduce the attack surface/complexity or is there a deeper reason?

Just to reduce surface area. It wouldn't help with increasing tx fee. A tx starts being valid at nLockTime. It wouldn't work to have a tx that stops being valid at a certain time; once a tx ever becomes valid, it must stay valid permanently.

See these threads:
http://www.bitcoin.org/smf/index.php?topic=1786.msg22119#msg22119
http://www.bitcoin.org/smf/index.php?topic=2181.msg28729#msg28729

I haven't fully understood why sequence numbers are a property of the tx inputs rather than the tx itself.

It's for contracts. An unrecorded open transaction can keep being replaced until nLockTime. It may contain payments by multiple parties. Each input owner signs their input. For a new version to be written, each must sign a higher sequence number (see IsNewerThan). By signing, an input owner says "I agree to put my money in, if everyone puts their money in and the outputs are this." There are other options in SignatureHash such as SIGHASH_SINGLE which means "I agree, as long as this one output (i.e. mine) is what I want, I don't care what you do with the other outputs.". If that's written with a high nSequenceNumber, the party can bow out of the negotiation except for that one stipulation, or sign SIGHASH_NONE and bow out completely.

The parties could create a pre-agreed default option by creating a higher nSequenceNumber tx using OP_CHECKMULTISIG that requires a subset of parties to sign to complete the signature. The parties hold this tx in reserve and if need be, pass it around until it has enough signatures.

One use of nLockTime is high frequency trades between a set of parties. They can keep updating a tx by unanimous agreement. The party giving money would be the first to sign the next version. If one party stops agreeing to changes, then the last state will be recorded at nLockTime. If desired, a default transaction can be prepared after each version so n-1 parties can push an unresponsive party out. Intermediate transactions do not need to be broadcast. Only the final outcome gets recorded by the network. Just before nLockTime, the parties and a few witness nodes broadcast the highest sequence tx they saw.

Mike Hearn <mike@plan99.net> Wed, Mar 9, 2011 at 5:39 PM

To: Satoshi Nakamoto <satoshin@gmx.com>

If you don't know about all txes in existence, I don't know how to do 2). You could only rely on trusting other nodes for that. That trust can be distributed over multiple nodes. Nodes only relay transactions they accept as valid. If you receive inv messages for a tx from all the nodes you're connected to, they're attesting that it's valid and the first spend they saw.

Good point. I was talking about verifying the inputs yes, but it is indeed pointless unless you hear about all open transactions as well. So being able to fetch a CMerkleTx is not important.

Just to reduce surface area. It wouldn't help with increasing tx fee. A tx starts being valid at nLockTime. It wouldn't work to have a tx that stops being valid at a certain time; once a tx ever becomes valid, it must stay valid permanently.

See these threads:

http://www.bitcoin.org/smf/index.php?topic=1786.msg22119#msg22119
http://www.bitcoin.org/smf/index.php?topic=2181.msg28729#msg28729

I see. So right now fees are tricky because you have to decide up front what the fee should be, and if you guess too low, there's no way to correct the transaction and though the network will eventually forget it, your wallet still records that you spent the coins. This has already started happening.

It's for contracts.

Ah ha. A whole unexplored area of the system opens up before my eyes :-) The concept of forming distributed contracts and escrow transactions without needing to trust an intermediary is a concept nearly as novel as BitCoin itself, I think.

I have more questions!

There's an unfinished part of the protocol that deals with setting up publisher/subscriber channels for distributed routing via the network. What was the purpose of this? Was the idea to have a p2p market or did it have some kind of lower level function, like perhaps broadcasting expected tx fees?

There was an interesting discussion of generalizing BitCoin some months ago, but we struggled to fully understand how you planned to achieve it. I think I understood the concept of placing another merkle tree on top of multiple separate chains:

http://www.bitcoin.org/smf/index.php?topic=3414.msg48171#msg48171

But I didn't understand your comment about having 200 bytes for backwards compatibility. Also, I guess this is obvious, but to be super clear - in your idea the alternative chains would share exactly the same format and sets of verification rules as BitCoin (the same script language etc), so all miners can verify all blocks even if they are non-financial in nature? And then the point of having separate block chains is simply to manage storage costs and bandwidth for client-mode implementations?

Thanks!

Satoshi Nakamoto <satoshin@gmx.com> Wed, Mar 9, 2011 at 7:39 PM

To: Mike Hearn <mike@plan99.net>

See these threads:
http://www.bitcoin.org/smf/index.php?topic=1786.msg22119#msg22119
http://www.bitcoin.org/smf/index.php?topic=2181.msg28729#msg28729

I see. So right now fees are tricky because you have to decide up front what the fee should be, and if you guess too low, there's no way to correct the transaction and though the network will eventually forget it, your wallet still records that you spent the coins. This has already started happening.

The network won't forget, and the owner's client will keep rebroadcasting it. The overflow transaction was remembered by the network for several months even as the remaining 0.3.8 nodes diminished.

Priority includes age, so as a transaction waits it ages and will eventually have enough priority.

See one of the links above where I contemplate sending an honest double-spend to increase the fee. It's a lot of work but could be done. I don't think it's worth it right now.

The current system, where nodes make sure to include enough fee for current conditions and the network makes sure all transactions get processed eventually, works well enough. Gavin is fixing the oversight where nodes didn't check the priority of their own transactions when writing them.

Users still worried about processing speed uncertainty should think of it as encouragement to include a fee.

There's an unfinished part of the protocol that deals with setting up publisher/subscriber channels for distributed routing via the network. What was the purpose of this? Was the idea to have a p2p market or did it have some kind of lower level function, like perhaps broadcasting expected tx fees?

I was trying to implement an eBay style marketplace built in to the client. Publish/subscribe would be used for broadcasting product offers and ratings/reviews. Your reviews would be weighted by the blocks you've generated. I rightly abandoned it in favour of JSON-RPC, so other authors could implement it externally. The publish/subscribe "meet in the middle" mechanism was an interesting concept, but nothing remains that uses it.

It was part of writing code to explore the most technically demanding use cases and make sure Bitcoin could support everything that might be needed in the future, given the locked-in nature of the rules once the block chain started.

There was an interesting discussion of generalizing BitCoin some months ago, but we struggled to fully understand how you planned to achieve it. I think I understood the concept of placing another merkle tree on top of multiple separate chains:

http://www.bitcoin.org/smf/index.php?topic=3414.msg48171#msg48171

But I didn't understand your comment about having 200 bytes for backwards compatibility. Also, I guess this is obvious, but to be super clear - in your idea the alternative chains would share exactly the same format and sets of verification rules as BitCoin (the same script language etc), so all miners can verify all blocks even if they are non-financial in nature? And then the point of having separate block chains is simply to manage storage costs and bandwidth for client-mode implementations?

No, other chains do not follow Bitcoin's rules. They are completely independent chains. They share nothing except the miners. The other network's definition of proof-of-work is to make a solved (according to their own chain's difficulty) Bitcoin-format block that has a hash of their own block in it. They don't care if the Bitcoin block is valid or used by Bitcoin, but it allows miners to work both chains at once.

Procedure to hash a BitDNS block:
- hash the BitDNS block
- construct a Bitcoin block
- insert the BitDNS hash into the scriptSig of tx 0 in the Bitcoin block
- hash the Bitcoin block

The BitDNS block is valid if that hash is below BitDNS's target.

The BitDNS block needs to have with it about 200 bytes of data needed to reconstruct the Bitcoin block used in the hash:
- the Bitcoin block header
- the merkle branch to tx 0
- tx 0 (btw, tx 0's prev hash is always 0 so leaving that out saves 32 bytes)

Note that it doesn't matter if the fodder "Bitcoin block" was actually valid in the Bitcoin chain, though it could have been. To BitDNS, it's just a bunch of salt necessary to do its convoluted hash calculation. If a miner is only mining for BitDNS and doesn't care about Bitcoin, it would use a blank Bitcoin block of all zeroes (except the nonce).

To further expand the idea for extensibility, consider instead of putting the BitDNS block hash in tx 0, you put the root of a merkle tree that includes BitDNS. This is the merkle tree that is conceptually at the top.

Mike Hearn <mike@plan99.net> Wed, Mar 9, 2011 at 7:52 PM
To: Satoshi Nakamoto <satoshin@gmx.com>

Thanks again.

Hal speculated that you intended to stash the new merkle root in the tx0

scriptSig. Good to know at least he had the right idea :-)

Holding coins in an unspendable state for a rolling time window
6 messages

Mike Hearn <mike@plan99.net> Mon, Apr 18, 2011 at 11:14 PM
To: Satoshi Nakamoto <satoshin@gmx.com>

Hello Satoshi,

I hope this mail finds you well. Recently I've been thinking about how BitCoin can help handle internet abuse and would appreciate your thoughts.

My "day job" is on the Google abuse team. We make extensive use of phone verification to control outbound spam from our network. Facebook and Craigslist do the same. Phone verification works well because phone numbers are something most people have access to at least one or two of, but rarely more. Yet it has significant downsides - it's expensive (for us), flaky, some people don't like the privacy implications, and some spam is profitable enough that buying lots of SIM cards is worth it.

It would be ideal if BitCoins could be put up as collateral against an account. The amount put up would help determine the limits the system placed on your behavior (eg how much mail you can send), in an anonymous and private way. But how to implement this?

Burning coins forever is easy, just set the only output to be <pubkey> OP_FALSE. Now you can sign some server-provided challenge with that key and prove you did indeed burn those coins. A key would only be usable with one account so spammers cannot simply put up a huge collateral and then resell signatures generated with that key. If the account was found to be abused it'd be terminated like today, and the coins would be "gone".

But people do come and go from these big networks and the thought of losing the coins if you quit Google to run your own mail is unappealing. It would be ideal if coins could be locked up for a period of time such that they cannot be spent until time X, where X can be constantly pushed into the future if the owner desires it but otherwise the coins eventually become spendable again. To verify your Google account, you would take some amount of coins (say 10) and set it up so you cannot spend them for 6 months.

The script language has no concept of time. OP_BLOCKNUMBER was ruled out because re-orgs could potentially invalidate entire chains of transactions. But is an OP_DAY feasible? I'm thinking of an opcode that returns the timestamp from the block header, but rounded to the nearest day to handle the natural clock drift seen in the block chain. If it could work then a TX that ties up coins until past a certain day is easy to construct. Updating it so the deadline is constantly moving is harder. A simple brute force solution is to require the user to put up 2x the coins such that at the point the first tx is about to expire and become spendable again, the second tx is created. In this way you always have at least one tx of sufficiently distant deadline to act as collateral. But this is inelegant. A better way would be to introduce a new rule allowing a tx to connect to such an output before the deadline has passed, as long as the output of that tx is once again a deadlined output of the same form. However this is less general than the scripting language so is also somewhat inelegant.

What do you think?

Satoshi Nakamoto <satoshin@gmx.com> Wed, Apr 20, 2011 at 11:39 AM

To: Mike Hearn <mike@plan99.net>

If the script language is not stateless, if it has access to any outside information that changes or varies between nodes, attackers can use it to fork the chain. The only exception is if it is always false before a certain time and permanently true after, which is implemented with nLockTime.

Since Google is trusted, couldn't users pay a token deposit to Google and Google pays them back when they close the account?

To answer your question though, yes it can be done without using trust:

Tx 1 from User pays to a script that requires the signature of both Google and User to spend.

Tx 2 (the contract) spends Tx 1 and pays it to User. nLockTime is the time to release the money.

Steps:
1) Google gives User a pubkey to use in creating Tx 1.
2) User privately creates Tx 1, does not broadcast it yet.
3) User gives the hash of Tx 1 to Google.
4) Google signs its part of Tx 2, with nLockTime set, and gives it to User.

5) User broadcasts Tx 1.
6) User signs his half of Tx 2 and broadcasts it.

With these steps, the user already has Google's signed half of Tx 2 in hand before he broadcasts Tx 1, so he is assured of what bargain he is signing the money to.

This is the general pattern for safely signing contracts. Tx 2 is prepared first so the parties know what they're paying into. Tx 1 is broadcast to lock up the money and assign it to Tx 2. In other words, all parties assign their money to a pool that is controlled by the unanimous agreement of the group, but first the group has already signed agreement for the default action to take with the money, or partially signed multiple available options that a party can complete by adding the last signature.

By mutual agreement, the parties can always write another version of Tx 2 that releases the money immediately.

[Quoted text hidden]

Mike Hearn <mike@plan99.net> Wed, Apr 20, 2011 at 1:55 PM

To: Satoshi Nakamoto <satoshin@gmx.com>

Thanks, that's helpful. I'm understanding contracts better now.

So the issue with having an OP_TIME/OP_BLOCKNUMBER opcode is not only that the results can change after a re-org (you said that previously), but also that people could use it to produce transactions that cease to be valid entirely after a certain time and cause a fork. Kind of obvious in hindsight.

Since Google is trusted, couldn't users pay a token deposit to Google and Google pays them back when they close the account?

Google and trust is a complicated issue. Lots of people use our services despite having little trust in us. Some people start out trusting us but then read (often sensationalist or wrong) stories in the media that change their minds, and so on. This is one of the problems we have with phone verification ... a few people don't want to give us their phone number.

For this case it'd probably be OK because trust around data privacy is different to trust around obeying contracts. I'm sure nobody would doubt that Google will pay them back - I bet we'd have an even better credit rating than the US Government in that sense :-) But we have quite a high rate of false positives with our verifications and some people would suspect we were accidentally verifying users in order to accumulate big piles of coins with which to earn interest. I've seen much sillier conspiracy theories gain traction.

Besides, avoiding the need to trust big, complex institutions is much more BitCoin-ish. And I correctly suspected there was a way to do it I didn't understand yet so it's a good chance to learn more about BitCoin.

To answer your question though, yes it can be done without using trust

If I wrote a wiki page on how to build contracts with BitCoin, would you mind reviewing it?

I'm thinking it might be a good idea to re-enable transaction replacement soon because as the network grows, it will become harder and harder to upgrade. In one sense this is good as it makes it hard to change the fundamental rules of the system. On the other hand, we risk having a protocol which has many unused features because they aren't widely supported enough. HTTP suffered this fate with many of its verbs as well as features like pipelining.

Did you have any list of tasks for re-activation, some kind of audit or finishing off some code?

I had a few other things on my mind (as always). One is, are you planning on rejoining the community at some point (eg for code reviews), or is your plan to permanently step back from the limelight? One reason I'm peppering you with questions is I worry that much of BitCoins potential lies in careful use of currently inactive features, but there's little guidance on how to do it. And frankly, I don't think I'm smart enough to figure it all out on my own. Maybe theymos is, he seems to understand it well. But if one day you leave entirely, parts of the protocol might fall into disuse, which would be a shame.

Another is the economics of mining after the transition to a fully fee based system. Right now difficulty is roughly a function of the USD/BTC exchange rate and per-block inflation. When mining reward is set by the market, it might be possible for a "Tragedy of the commons" to occur in which everyone benefits from a high difficulty, but nobody specifically wants to pay fees to get it. Besides, valuing difficulty is quite hard as you never know what the capabilities of attackers are until it's too late. Would it be possible for fees to trend towards zero over time as some miner is always willing to accept cheaper transactions and as miners drop out, the difficulty adjusts so the delays never get too bad to tolerate?

As always thanks for your insights.

Satoshi Nakamoto <satoshin@gmx.com> Sat, Apr 23, 2011 at 3:40 PM

To: Mike Hearn <mike@plan99.net>

I had a few other things on my mind (as always). One is, are you planning on

rejoining the community at some point (eg for code reviews), or is your plan to permanently step back from the limelight?

I've moved on to other things. It's in good hands with Gavin and everyone.

I do hope your BitcoinJ continues to be developed into an alternative client. It gives Java devs something to work on, and it's easier with a simpler foundation that doesn't have to do everything. It'll get critical mass when impatient new users can get started using it while the other one is still downloading the block chain.

P2P Foundation

Bitcoin open source implementation of P2P currency

2014-03-07 01:17:00 UTC - -

I am not Dorian Nakamoto.

Editor's note: this comment was apparently made in response to a March 6, 2014 Newsweek article that Dorian Nakamoto was the real Satoshi Nakamoto of Bitcoin fame. The post is debated as to its authenticity. Some say that Satoshi's account was hacked and that this was not actually written by Satoshi Nakamoto.