From NatureVault

NatureVault: DigitalCollectibleNetwork

Overview

A digital **collectible** network (DCN) (formerly called digital collectible currency {DCC}**[1]** **[2]** **bitcointalk**) is an alternative to current cryptocurrency technology. The biggest differences are: it does not have a blockchain, mining can be offline, only CPU's can mine - not GPU's(*) or ASIC's, ownership is 100% private by default, transactions are not time-gated and can be near instant and scale-able to infinite transactions per second **[3]**, emission scales perfectly with adoption, and nodes compete on the free market for the customer's trust. Nodes wield full authority over the network. The **customer** has ultimate power over the nodes by deciding which node(s) to **trust** and voting with their feet, money, and work. **Reputation** will be a big part of node selection by customers.

Pitch

Current cryptocurrency is "trust-less" with mechanisms that enforce a rigid and inflexible consensus. However these mechanisms can always be gamed. In the case of typical blockchain, the people with the most hashpower have the most vote, culminating in the possibility of owning 51% of the network and therefore controlling it entirely. Proof-of-stake is similar, with the person(s) owning the most coins having more vote and ability to attack the network.

"Trust-less" might be a good term for it. But "trust" is replaced with "faith". Faith that the bad actors will never be able to overwhelm the good ones, and the good ones will never be fooled. It all boils down to faith in the Rich and Powerful. This faith has shown itself to be unfounded by countless 51% attacks affecting a significant percentage of coins besides bitcoin. State actors, given enough incentive such as killing competition to their own digital currencies, might even attack Bitcoin too someday.

I prefer trust over faith. Let us, as the customers, have the choice of which nodes to **trust**, and provide opportunities for them to win our trust. Some of these opportunities can be nodes **staking** collectibles, getting **vouches**, staying free from **disputes** and maintaining good **customer** service. If a node is shown to be untrustworthy (and thus loose their **staked collectibles** and the staked collectibles of those **vouching** for them), the community would simply switch to other **node(s)** all without any transactions being lost since the database is distributed by nature.

Node selection by the **customer** will be primarily based on **reputation**. Reputation is a combination of many factors, and is based on human **heuristics**, so it is difficult to know exactly what you must do to improve your **reputation** other than being "good". This makes it difficult for the nodes to carry out an attack, since **reputation** is slow to build and fast to loose - and is not formulaic on how to achieve it other than being a consistently good actor that has put a lot at stake in the future of the network.

Collectible

The units of value that we discuss here will be referred to as either tokens or collectibles, these terms are interchangeable. However, three types of tokens/collectibles can exist on this network (**NFT's**, **LFT's**, and **FFT's**, and also **contracts**).

A collectible is just another form of Digital Property **[4]**.

Why collectible and not coin?

Firstly it is advantageous from a regulation standpoint to use collectibles instead of currency since there are less regulations on them. Secondly "collectible" is a more accurate term for our network. Collectibles by nature cannot be divided and each one takes work to produce. This is the case with our network. Each mining reward is indivisible and must be traded whole. With each collectible roughly the **value of a couple sticks of gum**, it is small enough to give pretty good granularity to transactions. Also the **more a collectible has been traded, the lower it's value**, which will make payment possibilities even more granular.

A collectible on our network is like a password. In reality it is the private key (password) of a public key that has a challenge based on it completed with proof of work. The proof of completed challenge (proof of work) is inextricably linked to the given public and private key and cannot be separated from it. Basically a line on the **privatenet** is one collectible on our network.

In online gaming there is an illicit but proven item/account trading methodology that works in a very similar way as our collectible network. What people have done is the person selling an account receives payment usually in the form of a collectible, a gift card that has a special code underneath a scratch-off layer. Often times the buyer, instead of physically sending this collectible card intact, will scratch off the layer and take a picture of the code and send it to the seller. The account seller then deposits this code into their account in Steam or iTunes or similar thus redeeming it so the buyer can no longer use it. In this case usually the seller has reputation and thus the buyer trusts them that once they redeem the code they will send the account password (this is no different than Bitcoin, you have to trust the seller will send your good once you have paid them the Bitcoin). Next the seller sends the password of the account (including the username if it is unknown) to the buyer. Now both the seller and the buyer can log in to the account. So to complete the transfer the buyer now requests a password change so only they can access the account now. This is nearly identical to the **transferal** process in the DCN, which is not an accident, since the DCN **transferal** method is based on this proven account selling method people have been using in online gaming for two decades.

Mining

Overview

Mining does not need an internet connection and miners create and complete their own challenges that meet the **requirements** of the network. Challenges are designed to be completed by an enthusiast PC in one week. The challenge utilizes GNFS **factorization** of large numbers which cannot be sped up with GPU's or FPGA's or ASICS's. GPU's can help support the CPU though and speed it up moderately when working in conjunction.

Every time a **factorization** is completed that meets the **requirements** of the network of nodes (**difficulty** and other **requirements**), one more **collectible** enters circulation; which is the **emission**.

Mining challenge

1. First the miner see's what **bit length** of number is required to be accepted by the network of nodes.

2. The miner then creates a private key.
3. The miner then creates a **public key** from the private key.
4. Next the miner uses a nonce of his choosing, then hashes the nonce with the public key and a BCH block hash (see **time stamping**) using something like Skein 1024 or more easily **SHAKE256-1024** (and converted from **Hex to decimal**) to generate a large random number.
5. Then the miner truncates the number to the **bit length** and other **requirements** that the network of nodes require.
6. The miner then makes sure the number is not prime **[5]**. If it is, or seems to be by not having small factors **[6]**, it can be **saved secretly for use in RSA encryption**.
7. The miner now completes a basic ECM. If it passes (has no factors found), the miner completes a **detailed ECM to optimally 32% of the number length, or 34% if using GPU**, to verify it is semi-prime; which is true if no factors are found. If factors are found the challenge number fails, and the nonce needs to be changed and a new potential challenge number hashed. Currently at ~150 digit numbers it takes ~300 tries to generate an acceptable challenge number.
8. The miner now completes a **factorization** of the number (See **how to mine**) and records it.
9. The miner checks that the **factorization** meets the **requirements of the nodes**.
10. The public key, nonce, and **factorization** constitutes one "**digital collectible**". The **factorization** is the "Proof of Work".
11. This digital collectible can be stored "indefinitely" in secret by the miner. As long as it continues to meet the **requirements** of the **nodes**, then it is valid. Alternatively the miner can submit the **digital collectible** to the **network of nodes** immediately for safekeeping and so it always keeps it's value.
12. The miner can give the private key to another person, that person can then change the private key to something only they know, and they are now the new owner of the collectible. This step requires broadcasting the transaction to trusted **node(s)** for storage on the **ledger**.

How to mine

Software and guides of how to factor large numbers have already been developed and are mature and can easily be found online **See post #3, working!**. Thus our digital **collectible** network does not need to include mining software. But we do need to develop simple software to create strong and secure private and **public key's (PuTTYgen works, use ECDSA nistp521)**, and large random numbers. This can utilize built in operating system features **[7]**. This software should be only used on a computer that is not connected to the internet, so it cannot be hacked.

Making money mining

Miners make money by creating and completing challenges that meet the **requirements** of the **nodes**.

Miners should distribute their collectibles as widely as possible. There will likely multiple networks soft forked from each-other with different transactions on their ledger, however most should have similar difficulty requirements so would likely accept the same collectibles from miners. These different networks, by definition, will not sync their ledgers, so it is up to the miners to send their mining awards to the networks separately. There could be certain unaffiliated or multi-affiliated nodes that provide this service to miners and give you a rundown of what networks are currently accepting what difficulty of mining challenges.

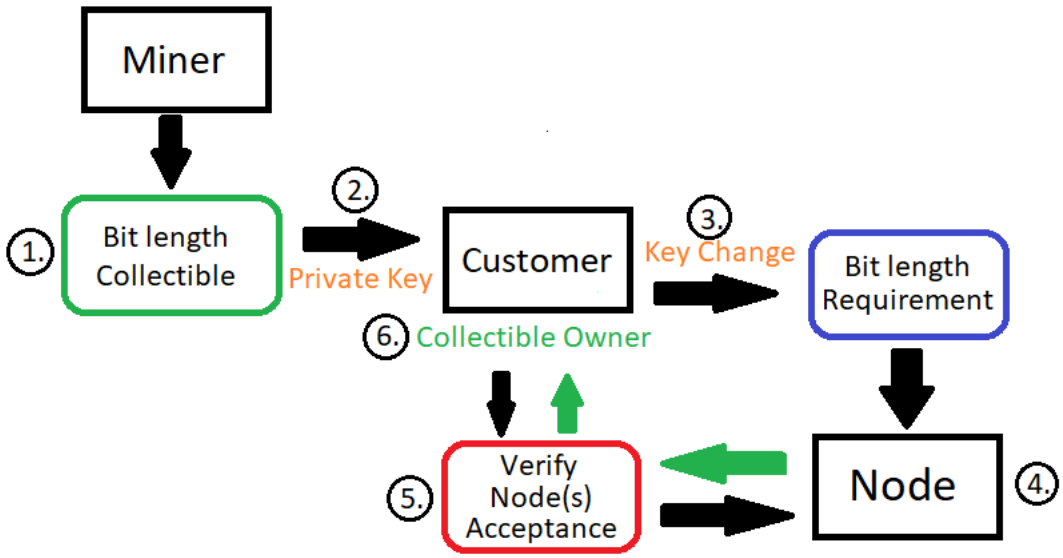
Also even hard-forked networks that have different difficulty requirements you could likely "merge mine" for these too. Say one network requires a difficulty of 153 digit numbers whereas another requires 150 digit difficulty. You could mine a 153 digit number then submit it to both networks since that would fulfill requirements of both.

Physical collectibles

If a miner is well trusted, another thing they could do is sell physical collectibles. The way this should be done is a trusted miner randomly and securely generates a private key, public key, and completes the mining challenge. They then should submit the collectible to trusted nodes to be registered. However the private key could be saved physically on a paper or QR code or something and this hidden inside a physical item such as a ceramic disk (like a poker chip) or other item that must be destroyed to retrieve the code. Of course the miner could potentially "spend" the collectible without your knowledge so a miner would have to develop a good reputation to be trusted with this. Also an **expiration date of 7 years** after it was registered should be put on the physical collectible as a "trade by" date.

Transferal of Collectible

After a **collectible** is mined it goes either directly to a node for recording, is secretly saved for later use, or given to a **customer** who then submits it to the network. How this transferal of a **collectible** works is displayed in the diagram below.



Transfer of an unclaimed collectible from miner to customer, and putting the collectible on the network so the customer becomes the owner.

Private keys

Chromaton was designed as a convenient and efficient way to transfer private keys.

If the collectible is already on the network then only the private key needs to be given during a trade. The reason for this is using the private key, the holder of the private key can recreate the public key so they know what the public key is. Also the nonce and factor are all on the ledger already. So if someone gets a private key then they own the crypto-collectible as long as they can then change the private key before the previous owner can. Therefore the reciever of the collectible will need to have an internet connection to verify their ownership of the collectible. The giver does not need an internet connection, but needs to have had an internet connection at some point so they could have gotten the collectible onto the network. Otherwise if they mined the collectible and never submitted it, in addition to the private key they would also need to give the new owner the nonce and factors proving work so the new owner can submit it to the network.

Layer 2: Fully offline trading (Cash)

In addition to **physical collectibles** for offline trading, another thing to introduce is encrypted trading, basically an app or program that encrypts the collectibles private key so while you are holding it, you can't access the private key. This shouldn't be the standard but it would be good for when you want to trade them offline and not require an internet connection to trade them or to receive them. Perhaps the app verifies it online once and then it can be traded and the giver and receiver both don't know the private key so they can't double spend. The app or program maintains and transfers ownership. So basically solving the double spending problem totally offline by encrypting the private key so both the giver and receiver don't know it and only an app intermediary does. For example, You could send a collectible to your phone app. Now the phone, when online, submits it to the network with a key change that only it knows. Now the phone has the new private key offline encrypted and you don't know what it is. Now when you transfer to a friend offline via Bluetooth for example, your app signs the key over to your friends app. Now both apps hold the (encrypted) private key but the app programming now eliminates the original app from double spending. Now your friend can do the same process to someone else. To cash out from your app, when you do get online, ask the app to give the collectibles to you (via a QR code or something) and now once that is done they can no longer be traded in the app itself without resending them to the app again and the app changing the private key again to something only it knows.

In order to prevent hacked apps bieng used where users can double spend, recieving app would verify the integrity of the sending app by verifying its hash.

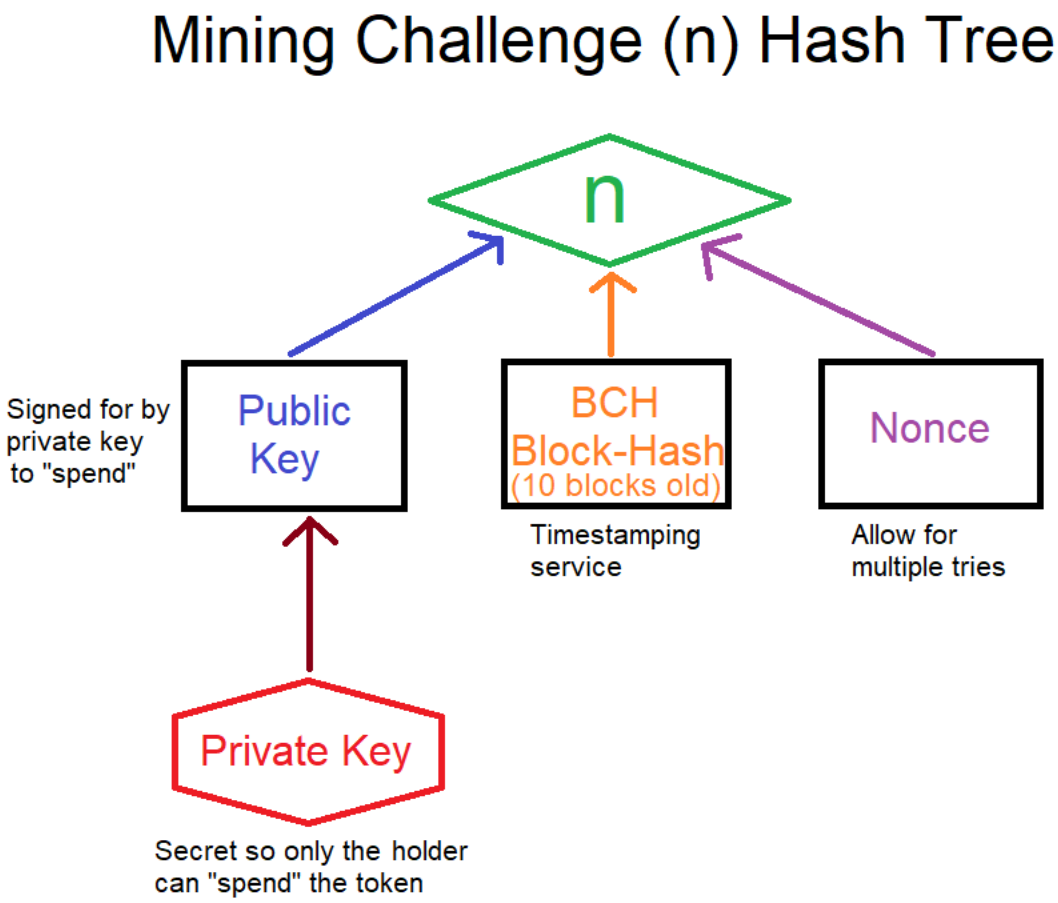
Time-stamping collectibles

Since we have **deterioration** of collectibles, we will probably want a way to determine how old a collectible is without having to rely on archives of nodes records of when they received it. One way we can have collectibles timestamped is to use the timestamping function of a blockchain. Bitcoin Cash would be an ideal candidate since it is based on the trusted bitcoin code and also has a rolling checkpoint every 10 blocks. What this means for us is when we generate a random number to factor, we can seed it with the block-hash of a 10 block old bitcoin cash block (actually should be the 11th which is immutable). This means that we know that the collectible mined is newer than that block in the bitcoin cash chain. So a node couldn't claim that the collectible needs to be deleted since it is 49 years old, in reality it is only 1 year old and the node is lying. It can be proven exactly how old that collectible is, regardless of the nodes recollection. Well, the block could be younger than it is claimed, but when we are talking about deterioration, the newer the better anyway so the miner is incentivized to use the newest bitcoin cash block (that they know will never experience a restructuring and change) that they can so their collectible lasts the longest on the network.

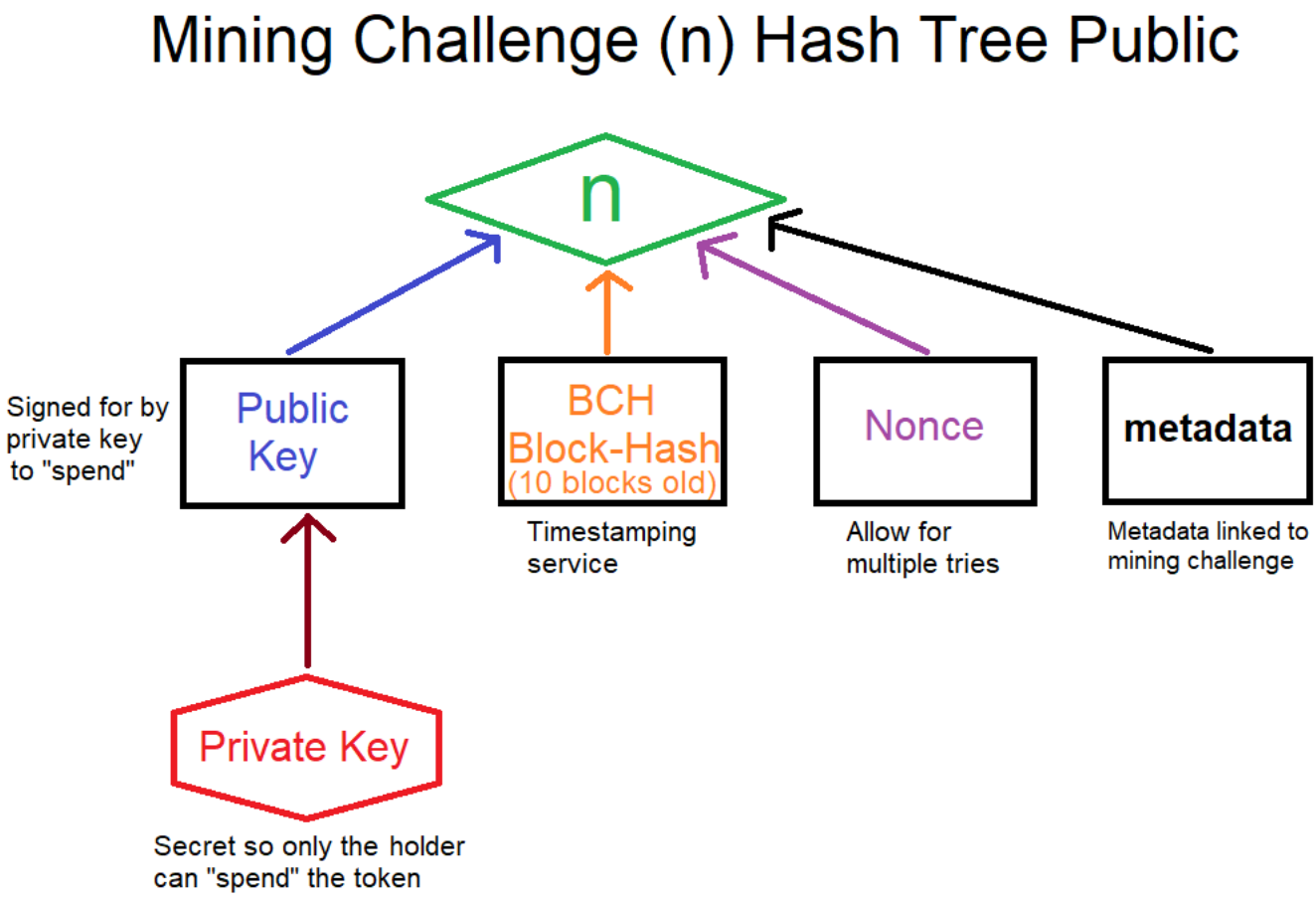
This means that **nodes** on the network will need to keep the block headers of Bitcoin Cash blocks in their memory (hard drive) which shouldn't take up too much space since the blocks themselves are not needed. Basically the node keeps time via bitcoin cash headers.

If for some reason Bitcoin Cash network ever fails, then we can switch to using another blockchain-based cryptocurrency as a time keeper. We simply would continue to hold the bitcoin cash block headers up to the time the network failed, and switch to a new cryptocurrency's block headers for new collectibles.

To do this when generating a large random number could be as follows. Create a private key that only you know. Hash that private key using whatever algorithm is chosen by the network (ECDSA or SHA-3 or something, we **chose to go with ECDSA nistp521**) to get a public key. Next hash that public key with a 10 block old bitcoin cash block-hash. Then hash that with Skien-1024 or something to generate our large random number and truncate to the amount of digits required, etc. Knowing the block-hash and nonce that was used (recorded in our database) you can sign to prove you own the given public key.



All three, the Public key, BCH Block-hash, and Nonce (as well as the number (n)) are all known and recorded in the database. The private key is not, it is a secret recorded and known only by the owner of the token. They use this private key to sign for the public key to prove ownership for transferring the token. However, they do not need to sign to get the number accepted to the network. This is because someone can make an unspendable token if they want, with the public key a **burn** address. Also signing for a public key may reduce security in the case that the hashing algorithm is ever cracked. So, signing should only ever be done once, and at the time of transfer when they are assigning the token to a new public key altogether.



As you can see above, if we want to make an **NFT** or **LFT** we can include the metadata within the mining challenge. However I also want it to be possible to later link metadata to an **FFT** that is in the **privatenet** as seen **here**.

Real World Value of the DCN Mining Challenge

Determining security of a given bit length

Knowing the speed of factoring large numbers, and the market price this is valued at, is of extreme importance to security researchers creating encryption standards. Researchers are willing to pay tens of thousands of dollars for this information **[8]**, and our network will provide them constant real time data of these exact metrics, which is priceless.

Prime number discovery

Our network will also help discover large prime numbers, which when kept secret, are digital commodities **[9]**. Once in a while a miner will generate a number to factor that is, in fact, prime. When this happens it can be **saved secretly** to create encryption yourself or given secretly to security researchers for use in encryption like RSA. If this is not desired, it could be published on another digital collectible network (DCN) that uses primes instead of **factorizations** as it's proof of work (which network again would be valuable to security researchers, so they can see the value of large prime numbers).

Also the knowledge of which large numbers are composites, can help narrow down the search for primes by security researchers. For example researchers could check the primality of numbers near proven composite numbers, since the primality of these numbers would probably have a higher likelihood than any random number **[10]** **[11]**. Alternatively, if using the sieve of eratosthenes, these composites can be added to the sieve, speeding it up **[12]**

People power

The practice of factoring large numbers is a valuable tool both for creating our own strong encryption or breaking that of bad actors. The People having this power helps keep bad actors and those who would exert power over others in check. Power to the people.

Dis-incentivizing bad uses of technology

Factoring computers could be used nefariously to try to crack high value keys, perhaps even keys that hold lots of bitcoin for example **[13]**. By giving monetary awards for factoring random challenges, there would be less risk to encryption keys being attacked for the value they hold or for other monetary compensation. It would likely prove more valuable to create your own challenge on the network than to try to brute force a key. For example, each key on our network will contain only 1 collectible, and breaking the key would be near infinitely harder than just mining your own collectible yourself. So our network incentivizes factoring computers to act in a good way instead of bad.

Nodes

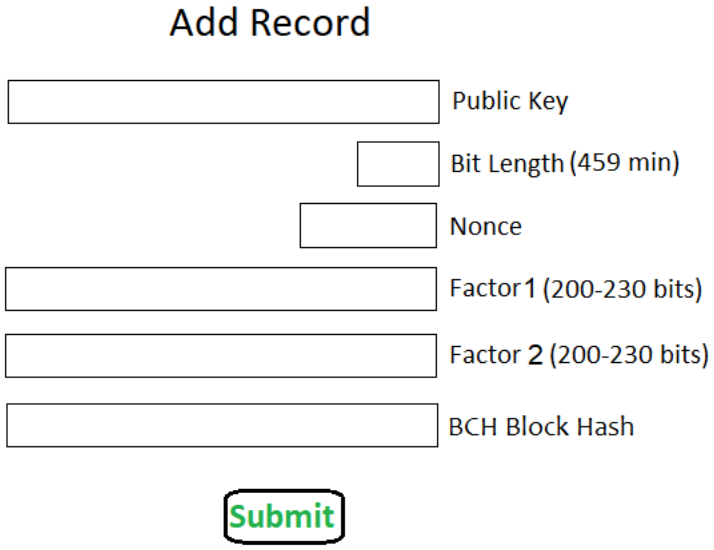
Nodes are not miners and have a huge and important role in a Digital **Collectible** Network. The nodes are the forefront of the network and compete with each-other on the free market to be chosen by customers as a **trusted node**. What this means is when a miner gives the collectible they mined to the **customer**, the customer has to ask node(s) to accept the collectible and then change the key associated with it. Nodes will likely ask a **fee for this service**, and are incentivized to share the info they gain with other nodes to increase their **trustworthiness**. They are also incentivized to stake collectibles (see **publicnet**) that can be blacklisted if they misbehave, also to increase the **trust** people have in them. They are also incentivized to come to a consensus with the other nodes about what rules they impose on **collectible requirements**.

Below we will propose some example ways for miners/customers to interact with nodes. Of course these things can be automated if desired so this will be a simplification for illustration.

Submitting new collectibles to node

As seen in the picture below, a miner or a primary owner of a **collectible** (a collectible that has never been recorded before in a node) simply needs to submit to his preferred node(s) the Public key, recent BCH block hash, Nonce, bit length of the random number that was factored, and the proof of work prime factors. These would then be verified by the node that they meet all the **requirements**, and the user would be notified if the submission was accepted or what requirement was breached. The node should notify the user of the accepted public key so they can verify the node received the request properly.

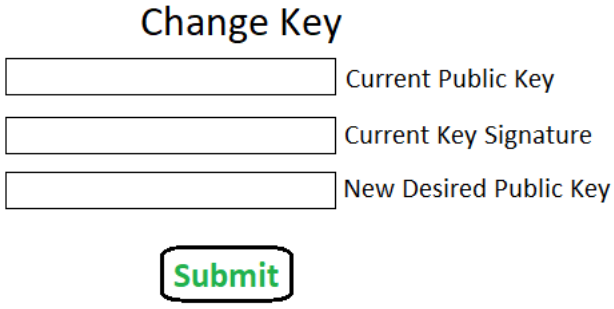
You do not ever need to submit the private key to the network. Keep the private key secret (referencing the public key) and you will only give this to someone else (secretly, ie: encrypted) when you want to transfer the collectible to another person.



Changing collectible ownership request

Firstly you cannot simply tell the network to give a **collectible** to someone else's key, unlike digital coins. With collectibles, you must actually transfer the private key to someone else. This can be done via Bluetooth or RFID or QR code, or actually mailing the key or sending over encrypted online communication. This is actually an all-around benefit, since it can be done offline if desired, and also instantly.

As seen in the picture below, for a collectible that is already recorded in the network, in order to request a change in ownership is as follows. The person enters the current public key of the collectible, signs for it proving they have the private key of the collectible(*), and then requests a new public key of their choosing. If everything meets the requirements of the node, the user will be notified that their submission was successful. The node should notify the user of the new public key so they can verify the node received the request properly. If the new public key has no private key, as in the case of **proof of burn**, then no one can ever spend the collectible again.



Joining collectibles

If there is a collision, and the new desired public key already exists, then it can either be denied by the node, depending on node rules, or it can be merged into the record of that public key. In the latter case the public key would then contain 2 **collectibles**, but each containing their own separate proof of work. These can be called "Joined Keys" or "Joined Collectibles". Again, it is undesirable to have multiple collectibles under one public key except for reasons of **staking**. The more collectibles in a public key, the less fungible it is since it likely cannot be easily split up into smaller pieces ever again without having traceable links to the other collectibles it was at one time joined to. Also even if it is never split, there are more opportunities to link transactions to the same owner. Before linking collectibles to the same public key for **customers**, nodes should give a warning that the collectible(s) would be forever relegated to the **publicnet** and may not be able to be split up again and will likely loose value and fungibility. Some nodes may refuse to have non-staked collectibles linked to the same public key, some may allow this as a **premium paid service** since they would have to move the collectible(s) to the **publicnet**. This is for the network to decide and hopefully eventually reach consensus.

- In order to prevent a man-in-the-middle attack during transit, the digital signature could be that of the hash of the old public key combined with the new public key. This way someone intercepting the transmission can't use your signature as a blank check and change the new public key you requested to a different value without you knowing.

- Joined keys are very similar to **staked collectibles**, the only differences are slightly different format in the ledger and also they are not necessarily owned by the hosting **node**. Joined keys can also be used as **staked collectibles** for nodes.

Responsibilities

Source Control

Nodes are basically required to keep source control of their **ledger** over time (such as Subversion) likely of SQL files and host them with at least daily updates and provide real-time syncing of databases. What this does is allow new nodes to get the latest SQL and get nearly all the entries, then real-time sync to get fully up to date and stay in communication for real time changes.

Source control is especially important so that the other nodes and/or public can trace back to where a problem happened in a given ledger. Also for **vouching** it is critical so that a node can vouch for your ledger on a certain date and this can be traced back to what exactly a node is vouching for.

Displaying Trust

Nodes can stake collectibles, forge reliable trusting connections with other trusted nodes, get other nodes to vouch for them, set good policies, and be responsive to **customers**; in order to build trust with the community.

Rights

Setting public key requirements

Nodes should set a public key requirement that is different than any other currency to prevent industry wide collision searches **[14]**. I don't see why this requirement cannot change over time, to make network wide collision searches even less effective. This changing of the key format over time could also lead to old collectibles having a higher value than new ones due to rarity. This fits with the spirit of collectibles but would harm fungibility. Again this is up to the nodes to decide. To start we are just using basic ECDSA nistp521 public keys which can be easily generated using **PuTTYgen**.

Setting collectible requirements

The nodes set the requirements for **collectibles** they add to their **ledger** for **miners** to abide by. The nodes verify that the proposed collectibles presented by the **miners** meet all the following requirements. The nodes set the rules and ultimately the **customers** are liable to make sure their trusted nodes are setting good rules. Here are some highly suggested requirements the nodes can set, some tweaks might be needed or more rules implemented as needed to prevent miners from minimizing the difficulty of their challenge.

1. Number to factor needs to meet **bit length** requirements.

2. Number to factor cannot have any leading or trailing zeroes or fives.

3. Number to factor must be odd.

4. **Factorization** consists of finding **two** prime factors, the smallest of which is within 28% of 1/2 the bit length (0.36 x digit length, rounded up).

Setting bit length requirements

The bit length of a number to factor is a variable that is controlled by the nodes. The bit length requirement of the numbers to factor is the Difficulty. Here are some rules the nodes should enforce.

1. Difficulty should be reevaluated every set interval and adjusted if needed. This interval should be between 1 month and 1 year and should be consistent.

2. Difficulty should be set so that an enthusiast PC working full time can complete 1 challenge and thus earn one collectible every week.

3. Requirements should be set so that GPU's cannot be used to significantly speed up completing the challenge, and that GPU's cannot be used solely to complete a challenge in anywhere near the time a CPU can do it. For example, the bit length of the factors required should not be so low as to allow GPU's to ECM to find acceptable factors in less than twice the time that a CPU could find an acceptable factor. Basically we want using CPU's to be the only reasonable option. GPU's can help by speeding up ECM, but we want the majority of time taken to complete the challenge to require the use of GNFS which is CPU only. Of course if a better algorithm is found for large number factorization that works even better than GNFS on CPU's, then by all means use that **[15]**, and adjust difficulty accordingly.

4. Difficulty, which is the bit length of the number to factor to be accepted by the network, should only be adjusted up or down a maximum of 2 decimal digits per adjustment interval. So if difficulty is adjusted yearly, then if the digit length requirement this year is 151+, then next year it should be set to either 149+ or 153+ or somewhere in between. This means each period would have a max of roughly ~30% difference in work required than the previous period **[16]**. Miners can speculate by mining numbers that are not currently acceptable but that they think would be in the future. However, chances are that the difficulty will always increase and not decrease. If certain nodes go up and down on difficulty often, they may be colluding with speculating miners and should be suspect.

Note: In 2021 it appears it will take factoring a roughly 442 bit (134 digit at 3.3 bit per decimal digit**[17]**) number to achieve our "one collectible per PC-week" **emission** goal **[18]**. Or perhaps around a 150 digit number **[19]**. We are going to start with 151 digits.

Determining Factorization

A full "prime factorization" of a Semi-Prime number will likely be needed to maintain high difficulty. So we would need to find a number with only 2 factors that are somewhat near half the length of the number to factor. So basically we are looking for RSA-like numbers. Therefore we need to find two prime factors near half the size of the number to ensure, not only that GNFS is used (which cannot be sped up with GPU/FPGA/ASIC, and is super-polynomial but sub-exponential **[20]**) as the majority

of the computation effort (since finding small factors can be sped up with GPU [21]), but also that each challenge takes roughly the same amount of time. The reason we can't use a single prime factor, is the rest of the factors can be small and lead to some numbers being much faster than others. With requiring two large factors, we know that any number that meets those requirements is of similar difficulty. RSA numbers would fit this requirement nicely.

We can say, for example, that a semi-prime with two prime factors, the smallest of which within (less than or equal to) 28% of half the size of the number, needs to be found (just multiply the digit requirement, say 151 times 0.36 and round up to find the minimum factor digit length for the smallest of the 2 factors. IE: 151*0.36 = 54.36 -> 55). Both the size of the factors, or the **bit length** of the challenge number could be changed to change the difficulty of course. However I strongly suggest the size of the smaller of the two factors be set to something within 28% of half the **bit length**. Also I suggest the minimum and/or maximum size of the factors required (the percentage number) is not changed unless absolutely necessary so that there is only one variable to change for difficulty adjustments (**bit length** of the number to factor) to make it easier for nodes to reach consensus on the rules. So for example if we wanted to increase difficulty significantly, then since we were requiring 151 digit numbers this year, next year we would require 153 digit numbers, but we would keep the 28% requirement the same. So for 151 digit number the 28% requirement would mean a min factor at least 55 digits, and for a 153 digit number that would mean a min factor of at least 56 digits (always round up). The 28% rule for a 151 digit number would mean that the smallest factor needs to be 151/2 = 75.5 * .72 = 54.36 rounded up to 55 digits. The 28% is set to make it so it wouldn't be favorable to just use a GPU to ECM a bunch of candidates up to say 52 digits instead of using GNFS. Requiring factors to be within 28% of half the bit length is set to ensure that the maximum amount of CPU time is required, which prevents significant GPU speed up. If GPU's are found to be gaining a significant advantage over CPU's, then this percentage away from half the bitlength should be lowered to keep CPU's necessary and fastest at completing challenges. So if we had a challenge of 151 digits and GPU's could ECM 55 digit factors too fast, then perhaps lowering the percentage to 25% to require 57 digit factors would help. However in general to keep pace with new CPU's coming out, only the bitlength of the challenge number would be changed, not the percentage length of the factor.

Determining Emission

The nodes, by setting and enforcing the **bit length** of the number that miners factor (difficulty), emission of the **collectible** is controlled.

The emission has no upper limit. We believe that miners, nodes, and **customers** are all essential, and always essential. And if emission is ever throttled, then that is saying we want miners to quit, which we never do. Emission must be able to scale up with more people joining/using the network and demand, just like if there is more gold demand there will be more gold mined.

Emission is controlled that for every enthusiast PC worth of computing power that is on earth mining, there will be one collectible added to the ledger per week. So if the world has 5,000 PC's worth of power mining the collectible, then 5,000 collectibles will be added to the ledger every week. See **ledger** for some calculations predicting emission. This will keep the value of the collectible quite stable as long as the nodes are controlling difficulty well and the amount of miners will scale up with demand for the collectibles.

Determining Price

The price is not controlled directly but I expect each **collectible** to hold a value somewhere between $0.10 and $1.00 each in 2021 dollars. This is based on the **emission** peg at one PC-week worth of work per collectible and how much that is valued on the Monero blockchain right now which is a predominantly CPU mined coin. One PC-week currently earns you about 0.0029 coins of Monero which, at $150 per coin, earns you about 43 cents.

Unlike in **RPOW** where each collectible is valued in terms of how much work it took to create and thus future miners will be able to mine value easier, our collectibles (on the **privatenet**) are all of the same value (Fully Fungible Tokens - FFT) because no matter how much processing power they took to create, they took 1 pc week of work to create when they were accepted onto the network. This means collectibles mined with a ryzen 7 in 2021 would still be worth the same as those mined with a ryzen 29 in the year 2041. Also since you can't create a super large proof of work like in RPOW and break it into smaller ones, supercomputers or quantum computers won't be able to outcompete smaller miners since they would have to spend their time creating lots of small collectibles just like everyone else.

Premium paid service

Nodes will make income the following ways:

- Accepting **submissions** and **transactions** directly from customers. Nodes would update their ledgers based on other nodes for free because they want to have the full database, but for a customer to submit to them directly they may require a small fee or a subscription. Some nodes may do one or both of these things for free though to increase their **reputation**.

- Allowing **contracts** from customers. Allowing customers to submit contracts is another way for nodes to make income. Contracts are sort of like premium transactions and are in the publicnet so it would make sense for nodes to require payment or subscription for a **customer** to submit these. Nodes would need to sync with other node's contracts for free though to maintain a complete database.

- Joining multiple collectibles to the same public key. **Joining collectibles** also would likely be something that nodes either charge a fee or require the customer have a subscription before performing. Joining collectibles takes collectibles from the privatenet or new collectibles and joins them to one public key, thus rendering them unfit for privatenet and moved to the publicnet. This makes all those collectibles linked to one common identity so should be discouraged by the nodes due to loss of fungibility; and requiring a fee or high tier subscription makes sense.

    - Joining collectibles can also be done directly by miners by using the same public key and using different nonces to create new proofs of work. This would be common in **staked collectibles**. These would also likely be treated by nodes similarly to a **customer** requested joining since the nodes would have to join the proofs of work to the public key themselves as well, especially if a miner submitted them separately.

    - Unjoining collectibles can also be a service that nodes perform for customers, but as far as I can predict it would require the collectibles maintain a link to the others they were once joined to. This is especially true if they were shared across the network, and nodes would have prior record of them being joined in the past leading to traceability.

- Allowing customers to **dispute** transactions. Now we are getting into something I think few nodes would be willing to do for free. Disputing transactions put the reputation of the node on the line as well as likely taking actual human work to review for veracity of the claims. Also disputes are not something that other nodes need to sync with unless they too want to enter into the dispute. The node would likely only allow customers to submit disputes for a fee or especially a high tier subscription, especially if the **customer** has many disputes or the customer's disputes are important enough for the node to stake collectibles on them.

As you can probably tell, one way for a new node to gain **reputation** quickly would be to offer some or all of the above services for free. **Customers** are going to be willing to give a node the benefit of the doubt and try to build a trusting relationship with a node that is willing to give them a good deal.

Ledger

The ledger is the distributed database of accepted **collectibles** stored and served by the **nodes**. It consists of a **Privatenet** and **Publicnet**.

Doing some calculations based on Monero blockchain, they appear to have about 244,000 PC's worth of mining power. This would mean if we scaled to the size of Monero, we would be adding 244,000 entries into our database per week, 12.7 million per year. At 100 bytes per entry **[22]** this equates to only 1.27 GB per year, much less than Bitcoin or other cryptocurrencies. Our protocol is very light. Here are some resources about database size solutions **[23] [24]**.

Privatenet

Privatenet is the default network of a digital collectible network (DCN). It is 100% private and untraceable, and as such, 100% fungible. Each **collectible** has its own public key, so is un-linkable with other collectibles. A collectible never moves to another existing address, so using chain analytics to track a collectible from one owner to the next, is impossible (except potentially in **certain circumstances**, but that would move the collectible to the **publicnet**). **Collectibles** should be given in person, such as payment via bluetooth or card like RFID, QR code, or magnetic strip; or sent encrypted over the internet via wallet-to-wallet or email etc. But even if someone snoops on the transmission, unless they change the key of the collectible before the receiver of the transaction, they do not know the new key.

Privatenet format

Privatenet

Public key	Bit length	Nonce	Factor 1	Factor 2	BCH Block Hash
XvyJ1e47mN9W...	460	28457294657	346573627849895...	5728949022712...	XceR95Pw3qRt42...
XvyKe45Le89seD... Xvy67jU021Bvwf.../eK36Lw49X...	463	36578	572384759598372...	3892305683022...	Xc457PeWcV5Bx4...
Xvy5kL01eDcX98...	490	5783746	477588937256759...	85693810582372...	Xw67UpQwecb71...

Notice how the second entry has 2 keys. The first key was for the original proof of work, and the second is for the changed key. The new owner chooses a public key to change the old one to, and the node allows them to sign for both the new public key and old public key combined using the old private key (notice the signature next to the new public key after the "/"). Presumably they know the private key for the new public key they selected, otherwise is now **burned** and cannot be spent again.

Also the new public key should be a hash of the old public key and the new private key. This way the chain of keys cannot be deleted but must all be kept in order for everything to be verified. This is important so a bad node couldn't just steal ownership by making false entries.

Also notice also that the bit length is different for each collectible. This is fine and denotes that they were an acceptable bit length at the time that they were accepted on the network. They all have the same value to the network.

Proof of Burn

- This section needs to be revjsited and rewritten

Proof of burn is proving that you made a **collectible** un-spendable.

Proof of burn is an elegant solution to a complex problem. Say you want to create a search engine where websites can bid on their search rankings for different search terms. Great. So who gets the collectibles that were bid? Instead of giving the collectibles to a 3rd party and making someone rich off this competition, proof of burn means that in essence the bid value is split between every holder of collectibles on the network by making collectibles more rare.

Proof of burn is accomplished by **transferring** the collectible to a public key that provably no one could own the private key to. Public keys are hashed from private keys, so if you create the public key "PaulWalkerisStillAlive1", it is impossible that you just so happen to have the private key to that public key. So signing that you change the key to that burn address, you are verifying that the collectible is lost forever. Proof of burn can even be used in this way for posting short messages for just the cost of burning a collectible.

Because of proof of burn (including **disputes**) and lost keys, and because the emission is adjusted on Moore's law and not price; the price of each collectible should steadily increase slowly over time, slightly above inflation. This is while it stays the same difficulty to mine forever, namely an enthusiast PC-week of work per collectible.

Publicnet

Publicnet is an opt-in area of the DCN and should only be used rarely for broadcasting contracts and claiming ownership of collectibles for purposes of **node** staking for example. **Collectibles** on the publicnet are not fully fungible, as a feature, due to their trace-ability. The publicnet of a node consists primarily of that single individual nodes **staked collectibles**, **vouches**, **disputes**, **joined keys**, and **contracts**. Out of these, **joined keys** and **contracts** are a little different; as all contracts and joined keys on the network are recorded, not just that one node's.

If someone wants to verify that an address (public key) in the publicnet belongs to the node, they can request a signature verification. The more times something is signed, the worse security gets, so nodes can set a limit of one signature per day or something like that. In order to help offset this, other nodes can vouch for you. Basically the vouching node would need to only sign once to vouch that ALL your claims in your publicnet are true (or at least the ones they are vouching for) and they have verified them. This way if someone wants to see if a certain node's claims are true, instead of asking for a signature verification for every claim, they could instead find another node who vouches for them and only require 1 signature.

Components of the Publicnet

Staked collectibles

Staked collectibles are **collectibles** that a certain node is claiming ownership of. If you ping a node and ask for verification of their staked collectibles they should provide this (sign for their key(s)) so that they come off as **trustworthy**. See also **vouches**. **Joined keys** are nearly identical to staked collectibles however the format in the **ledger** is a bit different and they are not necessarily owned by the hosting node. **Joined keys** can also be used as staked collectibles by a node.

Staked Collectible Format

Staked Collectibles

Public key	Bit length	Nonce	Factor
Xvy4jYt159sX...	463	547328947	4726485950066...
	462	4783993	54363457567565..
	467	3760349346	34539819671250...
XvyUp40QL6B...	503	4785	34276954607893...
	507	6849340	88574020430348...

Notice that this node has 5 collectibles staked between 2 different addresses. Staking is one reason why you would want to have many collectibles under one key, that way you only have to sign once for multiple collectibles. If you are determined to be a "bad node" your staked collectibles would be blacklisted from the other nodes and lost forever, similar to **proof of burn**.

Vouches

A vouch means that a node is willing to stake their own reputation that another node's publicnet is truthful and that they have personally verified every claim. A vouch is useful because you can get confidence in a new node (new to you) with only 1 signature instead of having to request a dozen signatures from the new node. The signature of a vouch likely would be the signature of the public key of staked collectibles, or just the fact that a node vouches for another node in their publicnet is also self evident that the node vouches for the other node, and thus have verified their claims so that you don't have to. Of course, that is if you trust the vouching node. In order to trust a vouching node even if you haven't worked with them, is to ask for the signature of the vouching nodes staked collectibles. In the picture below, we see that the node staked certain collectibles on the reputation of the node they are vouching for. If the node they are vouching for ever becomes a bad node, then not only will the bad node loose all their staked collectibles, but also other nodes that vouched for the bad node would also loose the collectibles they staked on the bad node. You cannot vouch the same collectibles on two different nodes reputation at the same time, including your own. The reason for this is if you and your friend get banned at the same time, your loss would be limited which isn't fair.

Vouches Format

Vouches

I vouch for

IP Address	Date last verified	My stake
123.86.674.23	2009-06-15T13:45:30	Xvy65Jc5bN89Wex...
453.75.930.34	2010-06-15T14:46:01	XvyuY45mv2JLq123...

Vouches for me

IP Address
583.83.902.18
189.56.980.39

Notice that there is a time listed in the picture above. Nodes can't be expected to vouch for another in real time, so they list a date and time they last checked the other node's accuracy. This requires node **source control** so one can find what a ledger looked like when a node vouched for them.

Contracts

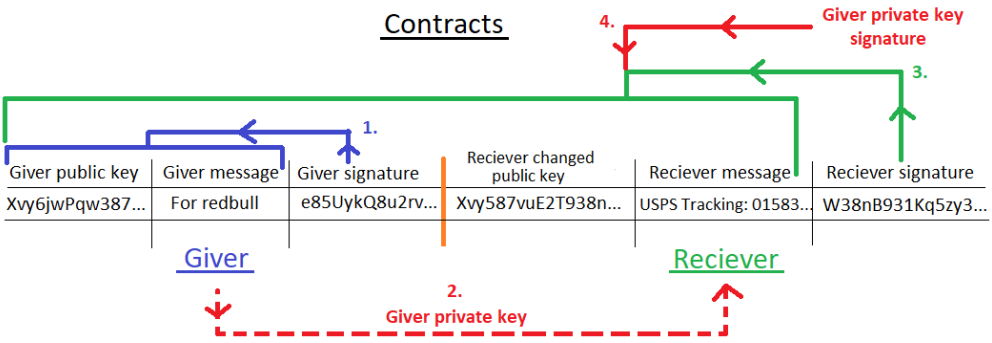
Contracts are critical for enforcing promises on the network. For example, a contract could be as simple as the "memo line" on a check. So in the publicnet you can sign to the public key of a **collectible** "giving to 'node y' for expenses". Then when node y receives the private key for the **collectible** from you, if they sign for the transfer then they are acknowledging the contract. Node y would then need to put this in their publicnet as well as a signed acknowledgement of the contract and by accepting the payment they are accepting the terms. The reciever should leave a memo like: "accepting as full payment from customer".

Since contracts do not need to be signed for on-demand, unlike vouches and staked collectibles, they could be in a separate part of the publicnet. The only reason it needs to be in the publicnet is because these collectibles are now tied to identifying information and, as such, are not fully fungible. Also all contracts on the entire network need to be recorded by every node. This is in contrast with **staked collectibles**, **vouches**, and **disputes**; which only pertain to one **node**.

Either one of the nodes or both could publish contracts on behalf of one of their **customers**. In other words, you do not need to own your own node to make contracts, but you do need your own node to have **staked collectibles** or **vouches** or **disputes**. The reason for this is contracts are linked to a collectible and published whereas **staked collectibles** and **vouches** and **disputes** are linked to a a node's public ip/identity.

If a contract is breached, a **dispute** can be logged by node(s) and appropriate action taken. It should be noted though that a dispute should not revert ownership of the collectible back to the giver, but should rather just blacklist the collectible entirely. This is to remove incentive of a giver to fraudulently dispute transactions. In this way the settlement for a dispute is splitting the disputed value among all collectible holders, in the same way as **proof of burn**.

Contracts format



In addition to what is listed in the diagram above:

- If this was a node to node contract then there can also be the node IP address or other identifying info listed in the contract, and/or this can go in the part of the publicnet that the node is signing to verify.
- The giver might be able to sign one more message afterwards about his feedback of the receiver. Whether or not if this is allowed, the giver can always ask their trusted **node** to **dispute** the transaction.

Here is an explanation of the diagram:

1. First the giver has a collectible with a public key that he owns the private key to. Then he writes a message. Next he signs the message and public key with his private key. He sends this to the node(s).
2. Next the giver sends his private key of the collectible to the receiver. This is done outside of the network hence the dashed line.
3. The receiver then creates a new desired public key for the collectible. She also writes a message. She then signs the entire transaction with the private key of her newly desired public key. This signing with her private key is optional, so a burn address can be used as the new public key if desired.
4. The receiver also signs it with the giver's private key to prove she owns it.

Smart contracts

Since we will have immutable **timekeeping** in the form of the collectibles accepted into the network, we can use these bitcoin cash block hashes in order to keep time and execute time based contracts.

Disputes

To be a trustworthy node, you have to be willing to go to war for the truth and your **customers**. Agreeableness is great and can enhance consensus, but consensus on a lie is worthless and harms everyone.

Disputes are like reverse vouches. If a node or contractee fails to meet obligations in the eyes of another node, that node can dispute the party. In the case of a non-node contract, the disputing node would blacklist (**burn**) the public key of the bad actors collectible. In the case of a node to node dispute, the disputing node would blacklist the bad nodes staked collectible address(es) and even potentially blacklist the staked collectibles of the nodes vouching for the bad node.

Depending on the severity of the transgression, the disputing node could also blacklist the offending IP temporarily because they do not trust syncing transactions with the bad node. I suggest the IP ban is not permanent since there are only a limited amount of IP's available. A nine month IP ban should suffice and/or when the rest of the nodes concur and blacklist the bad node's staked collectibles.

The disputing node would write a reasoning for the dispute. Just like a nodes vouches and staked collectibles, a dispute should be signed for on-demand by the disputing node.

Nodes can also stake collectibles on a dispute, saying they are confident that the node is a bad actor to stake their own collectibles on the claim. If the dispute is accepted by other nodes, they would likewise blacklist the offending collectibles public key, or node's staked collectibles etc as the case may be. If the dispute claim is rejected by other nodes on the network, they could blacklist the disputers staked collectibles instead of the node being accused of wrongdoing. Transactions should not (never say never) be reversed to prevent incentivizing fraudulent disputes, rather just the collectibles **burned**. Fork's in the network are a risk of disputes, which are more important to prevent, than for a **customer** to get their collectible(s) back.

If consensus is not reached and the outcome is unfair enough, this could cause a fork in the network, which is OK however value/fungibility of collectibles added to the ledgers after the fork might be affected depending on how much of the network went one way or another and whether they start setting different **collectible requirements**. However this can be much less upsetting than a fork in a current cryptocurrency like bitcoin, because in a DCN, miners can still provide their mined collectibles to both of the networks, presuming the nodes do not change the public key requirements of the collectibles they accept. In any case, a fork in a DCN network does not necessarily imply a fork in the mining power supporting the network. If a fork occurs yet both forks maintain the same **collectible requirements** for miners and thus the miners support both networks, this is considered a soft fork. If the network itself and the consensus thereof proves itself to be bad, and it is too onerous to maintain a fork that complies with **collectible requirement** of the other fork(s), a new network/hard fork needs to be created and good customers, miners, and nodes should move to and support the new network exclusively.

Disputes are another service that nodes can provide their customers. If a **customer** has a bad experience with a node or contractee, they can make the case to their trusted node to dispute the party. The affected customer should provide as much evidence as possible to the node to make a decision on the veracity of the claims. Accepting

disputes from customers can be a **premium offering** from nodes and would likely be a paid feature and also there could be a limit to the number of disputes a node will hear from a customer in a given time period.

Nodes are constantly at war and that shows that the network is alive and robust.

Dispute format

Disputes

IP Address	Date last verified	My stake	Message
123.86.674.23	2009-06-15T13:45:30	Xvy65Jc5bN89Wex...	Node refuses... http://...
453.75.930.34	2010-06-15T14:46:01	XvyuY45mv2JLq123...	Node refuses... http://...

Collectible	Date last verified	My stake	Message
Xvy5je7WRuQ1...	2010-06-15T14:46:01	Xvy75Je9Cxz287vM...	Cust. didn't recieve good

Forks

Where there are **disputes**, there will be forks! This is OK. I will describe what I expect to see in terms of forks.

Soft forks may be common. A soft fork occurs when groups of nodes cannot agree on what the proper ledger should look like. However they do roughly agree what difficulty mining should be. Miners then can continue to support both networks. It is in everyone's best interest to try to not split miners work, because the more miners the more widely accepted your network will be.

Hard forks really need to be deliberate. I foresee some groups that just want their own ecosystem, which is totally fine. One way to achieve this is to set the difficulty to completely different levels. So for example they could set their difficulty target to 1 collectible mined for every 2 weeks of PC-work, instead of 1 week. This choice is arbitrary, but I chose 1 week so people would feel incentivized to mine if even just with their laptop or even smartphone. Another group could set it to even 1 day of PC-work. While this would definitely incentivize people with slower devices to mine, it also would be cumbersome with how many collectibles need to be transferred for each purchase. I think 1 week is a happy medium. Everyone likes to get paid weekly.

You shouldn't hard fork unless you have a group of miners that really want to mine on your chosen difficulty. In any case, miners will likely still be able to contribute to your new network, unless your difficulty is higher than the majority of other networks. If the main network is 1 pc-week of work, and you set yours to 10, not many miners are going to accommodate you. But if instead of 1 week of pc-work you set it to 3 days of pc-work, then miners would likely contribute their collectibles to the main network, and also to yours since theirs would qualify on your network too. Kind of like merged mining. Miners will tend to keep the networks and nodes in some sort of consensus on difficulty.

Key verification

[25]

Homomorphic swaps

[26]

Value modulation

Deterioration

I believe a fun, interesting, and functional mechanism could be collectible deterioration. Since collectibles would be able to be mined indefinitely with no dampers, having deterioration of preexisting collectibles would make sense. What we could do is have each trade of a collectible reduce it's value. This makes sense because every time a collectible is traded it expands the database. So when it is traded 1000 times, it is deleted. So this would mean a collectible that has only ever been added to a database and never traded would have a value of 1000/1000. One that has been traded 500 times would have a value of 500/1000 which is 1/2. So this is one way also that partial collectibles could be traded. Lets say you want to pay someone 2.5 collectibles, you could give them 2 untraded collectibles and one that has been traded 500 times. This will improve granularity of payments. Many people would likely seek out partial collectibles so they can more easily make more exact (you can always overshoot someone's asking price) payments.

This would also make "new" collectibles more valuable (more collectible) which is a fun thing to consider.

This will also help prevent **dust attacks** where collectibles are transferred back and forth for no reason other than overload the network.

Sabbat

Another thing that could be done is a collectible that has not been transferred for 7 years, would be deleted, no matter how much value it has left. So if there is a collectible that has 1 use left (1/1000 value) and doesn't change hands in order to be destroyed, would get destroyed if it sat at that value for 7 years. This would also incentivize people to use and not hoard collectibles. Software could favor old collectibles to be spent from a persons stash before new ones, so that a person always maintains a fresh stock of collectibles.

If a collectible is only transferred once every 7 years, it could theoretically last 7,000 years before deletion.

Publicnet considerations: Jubilee

In general, deletion can happen on the **privatenet** and **publicnet**. However some might want their collectibles to last longer and not be deleted at all on the publicnet (privatenet collectibles will always deteriorate and have sabbat; thus limited lifespan). In this case this is a premium service and you would likely have to pay your hosting node a recurrent fee to keep hosting your collectible forever, but other nodes might refuse to host them after a certain period since they aren't getting paid. Because of this uncertainty (we can't pay every node forever to maintain consensus!), we need to have the publicnet assets face deletion. The way publicnet assets will be, is that they will not face value deterioration, but they will face a sabbat and jubilee. They must be transferred to a new public key once every 7 years or they will be deleted (which is the sabbat). If they are transfered at least once every 7 years, they will be active for 7x7 years or 49 years (a jubilee). After that period, they will be deleted altogether. You would have to resubmit the asset to get it reinstated onto the network after it is deleted. What this means is that every 49 years a different person would get the opportunity to try to claim ownership of the **publicnet** collectible (in essence a claim to the metadata). The exact time that the asset is scheduled to be deleted must be fully public knowledge and will be known based on the BCH Blockhash that was used to generate the number. Bad nodes could attempt to take control of any expired assets, so the people wanting to claim ownership must broadcast far and wide to all nodes their attempt to claim the asset at that time. Also there could be a period of time to send in a "public bid" (say a day or week) where the largest proof of work submitted wins the ownership of the collectible, not just the first one to submit any proof of work that fulfills node **requirements**. Also we could require that the proof of work factorization began no earlier than 1 year before the expiration date (again by looking at the BCH block hash used to generate the number n) so that people can fairly compete for the expiring asset.

Since publicnet collectibles do not face deterioration, we need to create a way to prevent dust spam, trading the collectible back and forth for no reason but to spam the network. So transferring publicnet collectibles should incur a small fee by the node you send the transaction to, to prevent dust spam. **Privatenet** collectible transfers need not incur a network fee for transfers since a small bit of the value (1/1000th) is burned every transfer.

Warfare

While war is allowed in current cryptocurrencies to fight for truth, it is a very symmetric war. In order to fight the only option you have is to "get rich" and buy more mining power or hard fork. This is not ideal, and has almost never worked to reverse fraud. Manual intervention (hard forks) have been the only way hacked coins have been restored in current cryptocurrencies.

In a DCN, asymmetrical war can be waged. Various tactics can be used such as mining decisions, increasing your node's trustworthiness, convincing other nodes to vouch for you, creating convincing disputes, soft forks, and hard forks as a last resort, etc to turn the tide of the network in the favor of truth and fairness.

Customer

While it is a nice thought to think that everyone that uses the network will be miners and/or nodes, this is not only impractical, but also undesirable. If that were the case, no one would be bringing in outside value and thus there would be little incentive to continue the network. Customers are the essence of any endeavor. They compensate the nodes for maintaining the network, and the miners for creating collectibles. They are both the customer of miners, and also the customer of nodes; and receive value from both. This is similar to gold buyers; they are the customer of gold miners and the customer of safekeeping services (banks/safety deposit boxes). They are the lifeblood of the network and have ultimate authority over it. They have veto power over node decisions by choosing which nodes to trust, and if there are no trustworthy nodes, to start one themselves or compensate someone else to do it.

'Hodl'ing customers want nodes to increase requirements to reduce emission. Buying customers want nodes to reduce requirements to increase emission. This balance is hopefully what keeps emission constant and based fairly on PC-work/Moore's law.

NFT's and Tokens

DCN is compatible with creating Tokens (**LFT's**) and NFT's as a part of the **publicnet**. Since it is a collectible network, each piece of value is unique and can be attributed to a real world product or be distributed as a token. The only catch is you have to mine every collectible you want to distribute. Different forks (or even just different sections) of the DCN could have a low PoW requirement (say 1 day or even 1 hour of a computer mining instead of **1 week**) to facilitate token creation. So "gas" can be created by each person for very little effort. There would likely still be a small fee imposed by the nodes to accept your collectibles (Tokens or NFT's etc) and they would be placed in the **publicnet**. Also I want to say that everything in the **privatenet** is a token so you really don't need to add metadata or put the token in the publicnet to have tokens that you can distribute at will. This section is mostly for Non-Fungible Token's (NFT's) or less fungible tokens (LFT's) ie: tokens linked to metadata. In LFT's the metadata can be the same for a set of tokens, for example if they are linked to a company. The **privatenet** is entirely made of fully fungible tokens with no link to anything.

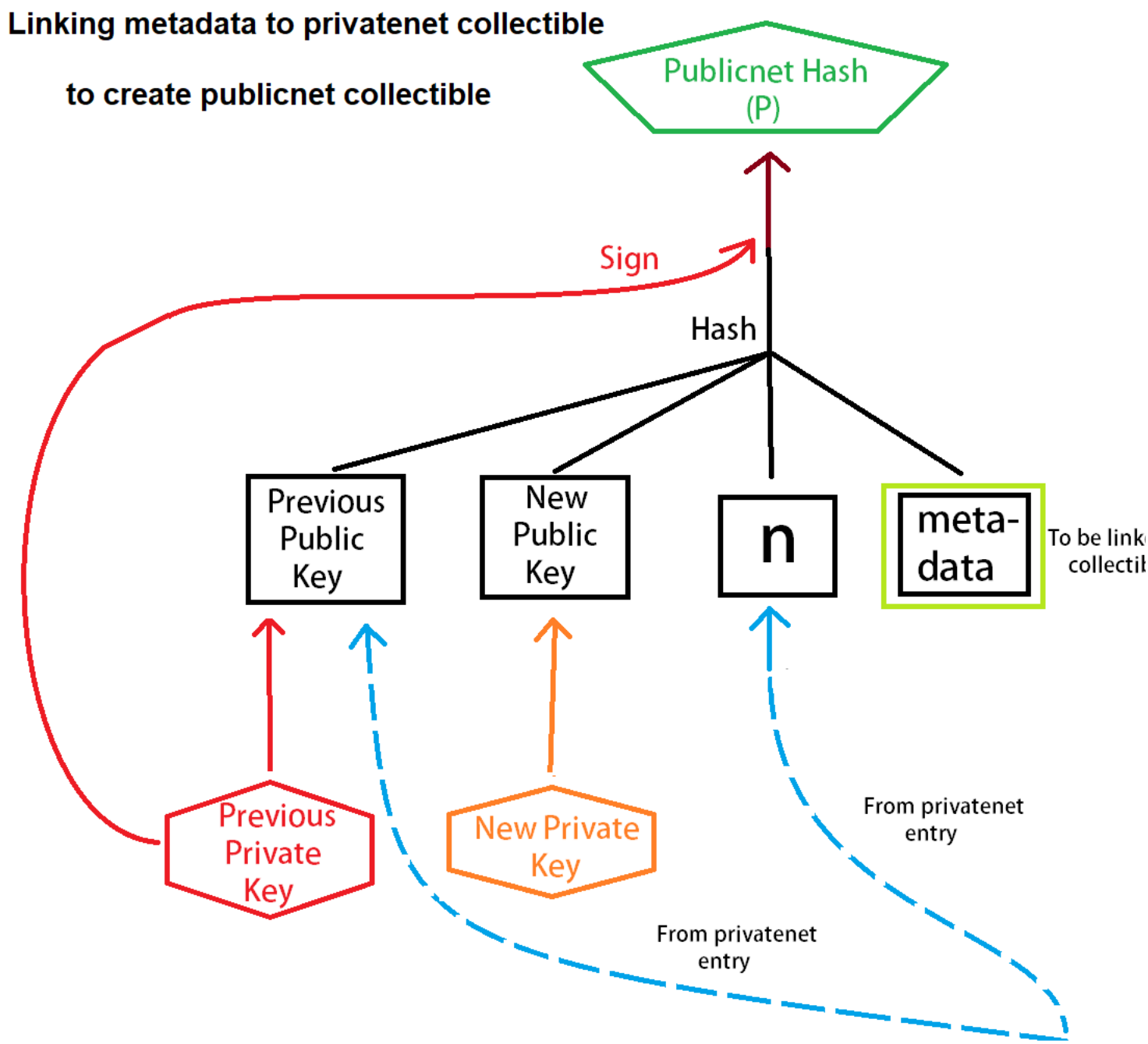
If the public key already exists in the **privatenet** then it would be deleted there and moved to the **publicnet** when it is linked to metadata. Of course the public key would need to be signed for by the owner of the corresponding private key to verify the transfer.

Tokens

Public key	Bit length	Nonce	Factor	Metadata
XvyJ1e47mN9W...	460	28457294657	346573627849895...	IPFS:eWo56cVbWQ...
XvyKe45Le89seD... Xvy67jU021Bvwf.../eK36Lw49X...	463	36578	572384759598372...	IPFS:38dF8Tp24gwE...
Xvy5kL01eDcX98...	490	5783746	477588937256759...	IPFS:34GftR92vQ32hN...

Notice in line 2 there are two keys. This is because the collectible has transferred ownership and there is a new owner. In this case there is the IPFS of the data of the token/NFT that the digital collectible is referring to, and the IPFS doesn't change with a change in ownership since it is immutable.

Another option for Line 2 is that it is a **multi-key asset** which requires signatures of all of the keys listed to sign for it. In general though, the last key listed is the one used to sign for the asset for the next transfer.



Above is the process for linking metadata to a previously made FFT, entry in the privatenet. This way people can mine FFT's exclusively if they want and sell them to others who want to convert them into NFT's or LFT's. If the miner is also the one wanting to make NFT's or LFT's, then they can use the process **here** instead.

Tokens of real assets

A digital collectible on the network can be linked to a real world asset by using a third party metadata service like IPFS to link the public key of the collectible to metadata and/or by using memo's in the **publicnet**. Metadata can be added to the DCN ledger but many would probably want to use a third party service like IPNS to keep records straightforward. In the case of using IPNS, the metadata of the real asset would be listed in the IPFS database, and the link to the IPFS entry would be added to the **publicnet**.

Multi-key signature

A real asset can be linked to multiple digital collectibles, where each digital collectible will need to be signed for in order to sign for the real asset. These multiple signatures required is a multi-key sig. The uses for this can be that multiple people are all needed to sign for something, or just to enhance security that if one of the keys was broken by hacking, the hacker cannot sign without hacking the other keys as well.

LFT's

LFT stands for Less Fungible Token. This is a token with a set metadata (so is a part of the **publicnet**) but the metadata is not fully unique. The purpose is they are redeemable for a certain product, so they are fungible between each other, but not fungible between tokens of a different product. For example TEEF Powder could make tokens, and each token is redeemable once for a TEEF Powder product. So each TEEF Powder token would have the same value (in theory but some people may pay more for older tokens). But they would not have the same value as BeeBop tokens. This is why they are called "Less Fungible".

For example if TEEF Powder made tokens, they would put TEEFPOWDER in the metadata section (see **NFT**) followed by a unique number. For example one LFT could be TEEFPOWDER00000012. As the business owner I know that I created this LFT and have this entry in my own personal database. This way if someone else created a coin with metadata TEEFPOWDER0958, it wouldn't be valid in my own database since I don't have that number recorded. For the ones I create, I keep a database and customers can redeem their token for my product. The way LFT's will degrade is the same as an NFT, if they are not traded for 7 years. If they are traded at least once every 7 years, they will degrade after 49 years. So when the 49 years is up, the token will be deleted from all databases. Then there will be a window where you can bid on the token with proof of work. The highest proof of work submitted with that metadata over a time period, say 1 week, gets to claim that metadata for their chosen public key with corresponding proof of work. So the morale of the story is, if I create tokens that are good for one of my products, the potential is that they will never go away. As the business owner you would have to publish your database of what tokens YOU deem still redeemable for your product. Even if the token is **burned**, someone can still claim ownership of it. So this is why only the business owner can determine which tokens are still redeemable.

For integrity of the network, each node should verify that the metadata of each submitted token is fully unique.

FFT's

FFT refers to Fully Fungible Tokens and are the native token of the **privatenet** and thus are the same as NFT's and LFT's however they contain no metadata.

FFT's can be converted to an **NFT** or **LFT** but only if it has never been traded. So businesses and artist would buy brand new FFT's from miners much like they would buy canvasses to paint on.

DeFi

Contracts allow for Decentralized Finance on the DCN. Since each piece of value has a unique code, it is easy to contractually obligate certain collectibles to someone at a set time or if conditions are met. The node will execute the contract such as locking collectibles. However the private key for each collectible are needed to be able to transfer ownership so there may need to be custodial services for private keys in contracts.

Multi-sig (centralized) collectibles

Games (or anything/anyone else) could use multi signature tokens to keep them only tradable on their network. For example a public key can be provided by the game client and also the player to hash and create the challenge number, and require authentication by both before it can be transferred. This can help games and programs like social media or anything else, maintain compliance of not letting value be traded outside the program.

This tech can be integrated into games or other programs or things. For example when a player crafts ingredients into an item, a challenge can be created and completed by their own computer while the player waits, in order to mint the item.

Uses

The DCN design is fully open source and no limits are placed on the idea's use. It can be used for any purpose and integrated into any system desired. It can be used in games, in border-less or local payment processors or other transactions including tipping, in internal company or technological systems, in social media for monetization of the platform or of creators content, as a replacement for traditional authority based systems or causing their obsolescence, and any other use.

As sort of a standard branding, companies, groups, or organizations can refer to a DCN that they run as "CollectX" if desired. For example TEEF powder brand can refer to their DCN as "CollectTEEF".

A brand can absolutely "run their own" DCN and that is encouraged, but in order to actually be a DCN they of course have to leave it open to have outsiders mining collectibles on the network and outsiders running nodes and **customers** choosing which nodes to trust completely unhindered. If either of these three things are limited or throttled or pay-walled in any way, then the system is no longer a DCN, it is then a Central Collectible Network (CCN). It is in the best interest of the brand to leave participation wide open and therefore widely promote the brand and increase the value and ubiquity of the branded collectibles. Imagine the marketing impact of seeing other brands accepting your brand's collectibles as payment!

Attack vectors

Mining

Searching for easy to factor numbers **[27]**. This would mean we should make the process of generating a number perhaps a bit time consuming.

Other resources on DCN:

Toppling the Blockchain

Bitcoin vs DCN

Links

Reddit gnfs on gpu **[28]**

GPU can help with polynomial selection, 10-30× faster, lots of fast cores best for sieving **[29]**

GNFS polynomial selection on GPU, sieving on CPU, and post processing on clusters for multithreaded linear algebra (post #21) **[30]**

PoS vs PoW **[31]**

XRP trusted ledger consensus **[32]**

Working Mining guide, see post #3 **[33]**

Online factoring applet for testing things **[34]**

Registering a public key with IPNS **[35]**

CADO vs GGNFS vs Msieve **[36]**.

Factoring as a service rsa-512 **[37]**.

Adjusting parameters necessary for numbers over RSA-160 **[38]**.

The great factoring challenge **[39]**.

Easy to factor numbers **[40]**.

DCN post on the mersenne forum **[41]**

DCN post on the bitcointalk forum **[42]**.

ECDLP is not secure from GPU and other speedup attacks like RSA is **[43]**.

Generate random decimal numbers for testing **[44]**.

The economic practice of jubilee **[45]**.

GGNFS help, may need to do ECM before GNFS on a random number **[46]**.

New Factoring Possibility, not prone to speedup **[47]**.

Multithreaded ECM driver to weed out numbers with small factors **[48]**.

ECM to NFS crossover digits **[49]** forum **[50]**.

ECM parameters for certain digits **[51]**.

Semi-prime (2 factor) rarity **[52]** and properties **[53]**.

Estimate prime rarity **[54]** around 1 prime every 392 numbers at the 10^153 level **[55]**.

References

B-money

Bit gold

Bitcoin [txt]

The cypherpunks

Open Coin

History of Collectibles

RPOW

Disclaimer

This is a living document, both because it is a wiki, and also because I haven't thought of everything. The whole purpose of the network is the nodes call the shots, not me. The nodes need to be willing to take a risk to stand up for the truth against the other nodes if necessary. The customers have to hold the node's feet to the fire so that the nodes serve the customers and the good of the network, which is to everyone's benefit. To the miners, try to be accommodating and service as many networks as you can, and keep mining :) ~**NatureHacker**.

FAQ

1. Can I send a collectible over the network to someone elses public key, so that I don't have to send a bunch of private keys over encrypted communication, one for each collectible I send? Basically, can I just tell the network that a public key now owns all of my 1000 collectibles?

Yes, this is called **"joining"** (see also **RPOW**) where you can join proof of work collectibles to a public key that was not linked to the collectibles original proof of work. This is discouraged however because 1. It makes the collectibles more vulnerable since if the one public key were to be cracked, then all the joined collectibles would also be stolen. 2. These joined collectibles would have to be forever in the **publicnet**, since they can no longer *ever* be traded with their own private key and must be signed for by the current public key owner. 3. It makes them easier to link to an individual since multiple collectibles are linked to one public key. 4. It would be in the publicnet which will likely cost more to host by a node for various reasons.

Category: **Cryptics**