Komodo

An Advanced Blockchain Technology, Focused on Freedom

Introduction to Komodo

The Komodo project focuses on empowering users with Freedom through blockchain technology.

There are many forms of Freedom that Komodo can provide, and we are currently focusing on empowering two types of users: the blockchain entrepreneur, and the average cryptocurrency investor. Together, our community of entrepreneurs, investors, and other users form an economic ecosystem.

The foundational pillar of the Komodo ecosystem is security. Komodo provides a unique and innovative form of security that is as strong as the Bitcoin network, yet does not require the incredible cost. Every member of the Komodo ecosystem receives the benefits of this security. The investor relies on it for everyday use. The entrepreneur relies on it to protect their blockchain innovation at a cost that is affordable even to small businesses and startups.

Another of Komodo's powerful technologies is a new method of trading cryptocurrencies directly from one person to another. It is a new kind of "decentralized exchange." Our decentralized exchange removes all forms of middlemen, vouchers, and escrow services. It relies on an underlying concept called the "atomic swap," and we are the leaders in this technology.

Our atomic-swap powered decentralized exchange serves both the investor and the blockchain entrepreneur.

For the investor, they can trade cryptocurrencies without having to pass through a centralized exchange, which can be an arduous and even dangerous process. They also do not have to use an escrow service, voucher, nor even an intermediary coin—not even Bitcoin. Furthermore, there is no registration process required, nor are there any withdrawal limits. We currently feature approximately one hundred blockchain coins for trading, and we are prepared to scale into the thousands.

For the entrepreneur, our decentralized exchange enables the release of new products to the world without middleman involvement. Furthermore, even entrepreneurs who have previously built other blockchain projects outside our ecosystem can easily feature their coin on our decentralized exchange. The only requirement is that the blockchain product have the proper security elements in the core of the blockchain's code.

Komodo also has powerful privacy features built into our platform. This allows the investor to trade and purchase goods and services within their right to privacy. It also allows the entrepreneur to release their product, and to crowdsource funds, from an audience that may prefer to maintain their privacy.

There are many other technologies and features in the Komodo ecosystem, and we are experiencing a rapid growth of both entrepreneurs and investors.

This Komodo white paper provides an in-depth discussion about Komodo's unique security features, our decentralized exchange, the method of releasing new products on it, and our native privacy features.

We welcome feedback from our readers. If you have any questions or concerns over the course of reading this material, please reach out to our team directly. You may find our contact information on our accompanying website: KomodoPlatform.com

Table of Contents

Part I - Security for the Komodo Ecosystem: Delayed Proof of Work (dPoW)

Part II - The Decentralized Initial Coin Offering

Part III - Komodo's Atomic-Swap Powered, Decentralized Exchange: BarterDEX

Part IV - Privacy in the Komodo Ecosystem: Jumblr

Part V - Additional Information Regarding the Komodo Ecosystem

Part I

Komodo’s Method of Security: Delayed Proof of Work (dPoW)

A Foundational Discussion of Blockchain Security

Komodo's form of providing security is called Delayed Proof of Work technology (dPoW). It builds on the most advanced form of blockchain security in existence, Proof of Work technology (PoW). The latter form of security is the method that the Bitcoin network utilizes. To understand the value of Komodo's dPoW security, we must first explain how PoW works and why it is the most secure method of maintaining a decentralized blockchain. We must also examine PoW's shortcomings, so that we may understand the need for Komodo's dPoW method and the advantages it provides to the blockchain community.

To understand how PoW technology functions, we begin by explaining the roots that make the Bitcoin protocol a viable means of securely transferring value.

What is a Consensus Mechanism?

The "Double Spend" Problem

The creation of blockchain technology stems from the early mathematical studies of encryption using computer technology.

One such example is related to the information-encoding device, "Engima," invented by the Germans at the end of World War I. Alan Turing, a British Intelligence agent, famously beat the Enigma device by inventing the world's first "digital computer." This provided enough computing power to break Enigma's encryption and discover German secret communications.[1]

This early affair with encryption set off a race throughout the world to develop myriad forms of securely transferring information from one party to another via computer technology. While each new form of computer encryption provided more advantages, there remained one problem that prevented encryption from being useful as a means of transferring not just information, but also financial value.

This challenge is known as the "Double Spend" problem. The issue lies in the ability of computers to endlessly duplicate information. In the case of financial value, there are three important things to record: who owns a specific value; the time at which the person owns this value; the wallet address in which the value resides. When transferring financial value from one person to another, it is essential that if Person A sends money to Person B, Person A should not be able to duplicate the same money and send it again to Person C.

The Bitcoin protocol,[2] invented by an anonymous person (or persons) claiming the name of Satoshi Nakamoto, solved the Double Spend problem. The underlying math and computer code is both highly complex and innovative. For the purposes of this paper we need only focus on the one aspect of the Bitcoin protocol that solves the Double Spend problem, the consensus mechanism.

The Consensus Mechanism Provides Security Against a "Double Spend"

The consensus mechanism invented by Nakamoto is perhaps one of the most powerful innovations of the twenty-first century. His invention allows individual devices to work together, using high levels of encryption, to securely and accurately track ownership of digital value (be it financial resources, digital

[1] https://en.wikipedia.org/wiki/Enigma_machine

[2] https://en.wikipedia.org/wiki/Bitcoin_network

real estate, etc.). It performs this in a manner that does not allow anyone on the same network (i.e. the Internet) to spend the same value twice.

Let us suppose a user, Michael, indicates in his digital wallet that he wants to send cryptocurrency money to a friend. Michael's computer now gathers several pieces of information, including any necessary permissions and passwords, the amount that Michael wants to spend, and the receiving address of his friend's wallet. All this information is gathered into a packet of data, called a "transaction," and Michael's device sends the transaction to the Internet.

There are several types of devices that will interact with Michael's transaction on the Internet. These devices will share the transaction information with other devices supporting the cryptocurrency network. For this discussion, we need only focus on one type of device: a cryptocurrency miner.

Note: The following descriptions are simplified explanations of a truly complex byzantine process. There are many other strategies cryptocurrency miners devise to out-mine their competition, and those strategies can vary widely.

A Miner Competes to Add Blocks to the Network's History, in Exchange for a Reward

Step One: Preparing the Preliminary Information

This device is performing an activity called cryptocurrency "mining." Let us focus now on a mining device that captures Michael's raw transaction data. This device is owned by a tech-savvy miner, named Gus, who wants to add Michael's transaction to the permanent history of the Bitcoin network.

If Gus is the first person to properly process Michael's transaction he will receive a financial reward. One key part of this reward is a percentage-based fee, taken from Michael's total transaction amount.

The Mempool is the Collection of All Raw Transactions Waiting to be Processed

Furthermore, Gus does not have just one transaction alone to mine. Rather, he has an entire pool of raw transactions, created by many people across the Internet. The raw data for each of these transactions sits in the local memory bank of each miner's mining device, awaiting the miner's commands. Miners call this pool of transactions, the "mempool." Most miners have automated systems to determine the transaction-selection process, based on estimated profit.

Creating Transaction Hashes

After Gus makes his choices about which transactions he will attempt to mine (and we assume that he includes Michael's transaction), Gus's mining device then begins a series of calculations.

His device will first take each individual transaction's raw data and use mathematical formulas to compress the transaction into a smaller, more manageable form. This new form is called a "transaction hash." For instance, Michael's transaction hash could look like this:

b1fea52486ce0c62bb442b530a3f0132b826c74e473d1f2c220bfa78111c5082

Gus will prepare potentially hundreds of transaction hashes before proceeding to the next step.

One important thing to understand about the compression of data in the Bitcoin protocol, including the transaction hash above, is that calculations herein obey a principle called, The Cascade Effect.

The Cascade Effect: Changing One Bit of Data Changes the Entire Result

The Cascade Effect simply means that were Gus to attempt to change even the smallest bit in the raw data—whether from a desire to cheat, or by mistake, or for any other reason—the entire transaction hash would dramatically change. In this way, the mathematical formulas in the Bitcoin protocol ensure that Gus cannot create an improper history.

Were Gus to attempt to create an incorrect transaction hash, other miners on the network could use the raw transaction data from Michael, perform the proper mathematical formulas in the Bitcoin protocol, and immediately discover that Gus's hashes are incorrect. Thus, all the devices on the network would reject Gus's incorrect attempts, and prevent him from claiming any resulting rewards.

Step One Continued: Finishing the Preliminary Calculations

Now, using more mathematical formulas, Gus takes the transaction hashes he is attempting to process and compresses them into a new manageable piece of data.

This is called, "the merkle root." It represents all the transactions that Gus hopes to process, and from which he hopes to gain a reward. Gus's merkle root could look like this:

7dac2c5666815c17a3b36427de37bb9d2e2c5ccec3f8633eb91a4205cb4c10ff

Finally, Gus will gather information provided from the last miner that successfully added to the permanent blockchain history. This information is called, "the block header." It contains a large amount of complex data, and we won't go into all the details. The one important element to note is that the block header gives Gus clues about how to properly add the next piece of information to the permanent Bitcoin history. One of these hints could look like this:

"difficulty" : 1.00000000

We will return to this clue further on.

Having all this information, Gus is nearly prepared. His next step is where the real challenge begins.

Step Two: The Race to Finish First

Gus's computer is going to gather all the above information and collect it into a set of data called a "block." Mining this block and adding it to the list of blocks that came before is the process of creating a "chain" of blocks—hence the industry title, "blockchain."

However, adding blocks to the blockchain is not so easy. While Gus may have everything up to this point correctly prepared, the Bitcoin protocol does not yet give Gus the right to add his proposed block to the chain.

The consensus mechanism is designed to force the miners to compete for this right. By requiring the miners to work for the right to mine a new valid block, competition spreads across the network. This provides many benefits, including time for the transactions of users (like Michael) to disseminate around the world, thus providing a level of decentralization to the network.

Therefore, although Gus would prefer to immediately create a new valid block and collect his reward, he cannot. He must win the competition by performing the proper work first. This is the source of the title of the Bitcoin-protocol consensus mechanism, “Proof of Work.”

The competition that Gus must win is to be the first person to find an answer to a simple mathematical puzzle, designed by Satoshi Nakamoto. To solve the puzzle, Gus guesses at random numbers until he discovers a correct number. The correct number is determined by the internal complex formulas of the consensus mechanism, and cannot be discovered by any means other than guessing. Bitcoin miners call this number a “nonce,” which is short for “a ‘number’ you use ‘once.’”

Gus’s mining device will make random guesses at the nonce, one after another, until a correct nonce is found. With each attempt, Gus will first insert the proposed nonce into the rest of his block. To find out if his guess is correct, he will next use mathematical formulas (like those he used earlier) to compress his attempt into a “block hash.”

A block hash is a small and manageable form of data that represents the entire history of the Bitcoin blockchain and all the information in Gus’s proposed block. A block hash can look like this:

000000000019d6689c085ae165831e934ff763ae46a2a6c172b3f1b60a8ce26f

Recall now The Cascade Effect, and how it states that changing one small number in the data before performing the mathematical computations creates a vastly different outcome.

Since Gus is continually including new guesses at the nonce with each computation of a block hash, each of Gus’s block-hash attempts will produce a widely different sequence of numbers.

Miners on the Bitcoin network know when a miner, such as Gus, solves the puzzle; by observing the clues that were provided earlier. Recall that the last time a miner successfully added data to the blockchain, they provided these clues in their block header. One of the clues from the previous block header can look like this:

"difficulty" : 1.00000000

This detail, “difficulty,” simply tells miners how many zeros should be at the front of the next valid block hash. When the difficulty setting is the level displayed above, it tells miners that there should be exactly ten zeros.

Observe Gus’s attempted block hash once again, which he created after making a guess at a nonce, adding this proposed nonce into his block, and performing the mathematical formulas:

000000000019d6689c085ae165831e934ff763ae46a2a6c172b3f1b60a8ce26f

The block hash above has ten zeros at the beginning, which matches the number of zeros in the difficulty level.

Therefore, the hash that Gus proposed is correct. This must mean that he guessed a correct nonce. All the miners on the network can prove for themselves that Gus was correct by taking all the same information from their mempools, adding Gus’s nonce, and performing the mathematical calculations. They will receive the same result, and therefore Gus is the winner of this round.

On the other hand, due to The Cascade Effect, if Gus's attempted nonce had produced a block hash with the incorrect number of zeros at the front, his block hash would be invalid. The network would not afford him the right to add an incorrect block hash to the network, and all the miners would continue searching.

Step Three: Gus Finds the Nonce

Once a miner discovers a nonce that produces a valid block hash, the miner has "found a new block," and can send the signal across the Internet. The consensus mechanism running on every other mining device can verify for themselves the calculations. Once verified, the consensus mechanism grants the miner the right both to add the proposed block to the blockchain, and to receive the reward.

Let us return to Gus's machine, having just guessed a correct nonce, and thus holding a valid block hash. Gus's machine instantly sends out the winning information across the Internet, and Gus collects his reward from the Bitcoin network.

All the other miners must readjust. Earlier, they were searching for the correct nonce based off the information from the previous block header. However, Gus's new valid block includes a new block header. All the other miners on the network abandon their current work, adopt Gus's new block header, make many recalculations in their underlying data, and begin their search for the next nonce.

There is no sympathy in the Bitcoin protocol for any miner's wasted efforts. Suppose another machine on the network was also trying to mine Michael's transaction, and lost to Gus in the race. Only Gus earns the reward from Michael's transaction, and the other miner receives nothing in return for their costs and time.

For Michael, this process seems simple. He first indicated the wallet address of his friend, and sent cryptocurrency. After a certain amount of time, his friend received the money. Michael can ignore the byzantine process of the miners that occurred between these two events. Michael may not realize it, but the PoW consensus mechanism provides the foundation of security upon which he relies.

PoW is Currently the Most Secure Form of Consensus Mechanisms

There are several reasons why PoW networks, especially Bitcoin, continue to dominate in terms of security and blockchain success. A simple, preliminary reason is that PoW networks foster ever-increasing speed and computer power. Miners must constantly update and innovate above their competitors to continue earning rewards.

There are yet more reasons behind PoW's success, and The Longest Chain Rule is one of the most notable. This rule can also be dangerous to the unwary and unprepared entrepreneur of a new blockchain product.

Speed and Power are of the Essence

Among miners, having a faster and more powerful computer can mean earning rewards more frequently. For miners seeking to maximize profit, competition requires constant upgrades to machinery and to a miner's customized underlying code.

The frequency at which a device can create proposed block hashes is called "hash power." The more hash power a collective PoW network has across all miners mining the blockchain, the more secure the network. This competitive pressure provides one important advantage in security to PoW networks, compared to alternate consensus mechanisms.

The Network Effect: Bitcoin's Ability to Dominate Begins

A high level of security fosters a sense of trust among users, and this can grow a PoW network's audience. As the audience grows, both the number of transactions and the price of the coin increase. This attracts more miners. The rising level of miners provides greater overall hash rate to the network, which in turn fosters a stronger sense of trust. This increased sense of security can raise the number of users on the network, which can increase the number of miners, and the cycle repeats.

In economics, this is classified as a "Network Effect," where a cycle of behavior encourages more of the same behavior, with compounding interest. Due to the Network Effect, and the fact that Bitcoin is the oldest PoW network, Bitcoin is increasing its security at a rate faster than the rate of other PoW networks.

Furthermore, consider the effect caused when the price of a PoW-blockchain coin rises. Before the rise, assume the blockchain coin is worth one dollar. A miner is justified in spending the necessary money (on equipment, upgrades, and electrical costs, etc.) to justify one dollar's worth of hash rate. If the price shifts upwards to two dollars, the miner must upgrade their entire business to justify two dollars' worth of a matching hash rate. If the miner does not upgrade, their competitor will, and then the miner will no longer be able to compete for rewards.

The Longest Chain Rule: The "Secret Sauce" of PoW Domination

There are many more reasons why PoW networks continue to dominate in security. Yet, for our discussion, there is one element that rises above all others. It is called, "The Longest Chain Rule," and some can argue that it is "the secret sauce" that fuels PoW's strength.

The Longest Chain Rule is the determining factor whenever two competing versions of the blockchain history arise on the network. The rule simply states that whichever of the two versions grows longer first, wins. The other version is overwritten, and therefore all transactions and rewards on that version are erased. The simplicity of this rule is a key to understanding why PoW consensus mechanisms continue to outperform their competition.

The Simple Effects of The Longest Chain Rule

On a surface level, this rule prevents a double spend by a network user. For instance, consider a husband and wife accidentally attempting to spend the same money at the exact same time, while each person is traveling in a different part of the world.

Komodo Team Note: For the sake of the discussion, we are oversimplifying the following actions so that they take place within only a few milliseconds. We also oversimplify the technical details, for clarity. The full explanation of this process is provided in the Bitcoin wiki, for those who would like to gain a deeper understanding.

A Tale of Two Blockchains

Let us suppose that the husband is in Asia and the wife is in the Americas. Both are purchasing a car. The husband uses all the funds from the family Bitcoin wallet to purchase a car at precisely 8:00 PM (UTC). The wife makes her purchase at the exact same moment, for a similar amount.

After making his purchase, the husband's transaction hash is immediately sent to a mining device in China, where it is held in the miner's local mempool (recall that a mempool is a collection of all raw transaction data across the network).

Let us suppose that the husband's transaction arrives in the Chinese miner's mempool at the exact moment that the Chinese mining equipment finds a correct nonce and a valid block hash. The Chinese miner declares the winning information, mines a new block, and collects a reward. All the miners in his local (Asian) vicinity (who receive the winning information faster than in the Americas, due to proximity) complete the block verification process, increase the length of the blockchain, and begin searching for the next valid block hash.

On the opposite side of the world, essentially the exact same actions happen. The wife's transaction is sent to the nearest miner, this time located in Washington state of the United States. Just as the transaction enters the Washington state miner's mempool, the miner discovers a valid block hash. He sends out the signal, mines a new block, and also collects the reward (this is the same reward that the Chinese miner is attempting to claim). All the miners in the local (US) vicinity verify the information immediately, and begin searching for a new valid block hash based on the Washington state miner's recent block.

An Internal Conflict of Interest Arises Within the Bitcoin Network

Note the paradox here. There are now two versions of the Bitcoin history that are valid, yet different.

These two versions make their way across the Internet, around the world, each to the other side. When the competing messages arrive, the Bitcoin protocol sees that there is a conflict: the same money was spent twice.

Consider how on each side of the world the miners are spending their financial and temporal resources to further their own interests. There is no economic incentive for either side to submit to the other, by nature. Therefore, there is a conflict of interest within the Bitcoin network itself. The Bitcoin network would swiftly fail, were it not for The Longest Chain Rule.

The Longest Chain Rule: The History Which is Longer First, Wins

The Longest Chain Rule simply declares that whichever of the two competing blockchains grows longer first, wins. The consensus mechanism erases the other version.

Let us suppose that the Chinese mining equipment is superior in this instance, and the Chinese miner manages to discover the next valid block hash and send out the signal before the Washington state miner can do likewise. Across the world, the moment the information arrives that the Chinese miner completed yet another valid block, the Bitcoin protocol erases the Washington state miner's version of the Bitcoin history.

There is no sympathy for any wasted efforts, nor for any misunderstandings between the wife and her car dealer. The Bitcoin protocol's consensus mechanism simply presses forward. The Washington state miner's rewards disappear, as though they never occurred. The wife's purchase of a car likewise evaporates.

(Typically, a normal and prepared car dealer utilizing cryptocurrency would not consider a customer's transactions acceptable until several new blocks were added to the blockchain. In this manner,

cryptocurrency users can ensure that a transaction is beyond contestation before the customer can, for example, drive a new car off the lot.)

The Washington state miner gets a raw deal in this scenario, but the network benefits as a whole. The Longest Chain Rule provides the necessary security to prevent a Double Spend. The network accurately recorded one family member's purchase of a car, prevented the mistaken double spend, and ensured that the most competitive miner received a just reward.

This example illuminates the importance of The Longest Chain Rule. However, there is a dark side to this rule for the unsuspecting and unprepared blockchain developer.

The "Easy" Way to Destroy a PoW Network: The 51% Attack

Here's where intrigue enters the picture. The "easiest" way to steal money on a PoW blockchain (such as Bitcoin) is to perform a 51% Attack.

In this attack, the malicious actor first spends cryptocurrency in exchange for something of value, which they take from their victim. Next, the malicious actor creates an alternate version of the PoW network's history wherein those transactions never took place. Using advanced mining equipment, the malicious actor then "attacks" the PoW network by mining blocks to this "false" history faster than the rate at which other miners on the PoW network can mine blocks to the "true" history.

Assuming the malicious actor has a sufficient hash rate, as this "false" history grows longer than the "true" history, The Longest Chain Rule will cause the consensus mechanism to overwrite the "true" version. The earlier transactions the malicious actor made would be as though they never occurred. Therefore, the malicious actor would keep both their original funds and whatever item of value they exacted from their victim.

This is known as The 51% Attack. The number 51% derives from the fact that to successfully perform this attack, the attacker must add enough hashing power to the overall PoW network to form a majority of the hash rate.

Size is Yet Another Reason Behind Bitcoin's Current Success Among PoW Networks

Today, Bitcoin's overall hash rate is enormous. The collective of computers around the world mining Bitcoin is effectively the largest supercomputer ever created by man. As of the writing of this paper, some estimate that the Bitcoin network consumes more electricity than the entire country of Denmark, and the number of miners continues to grow.

Therefore, to attempt a 51% Attack on the Bitcoin network could cost millions, if not billions of dollars in computer hardware. It would also require a sustained consumption of electricity that is likely unfeasible for a single geographical location. So long as the miners of Bitcoin remain interested in the Bitcoin network, therefore, Bitcoin has a level of security that is nigh impenetrable.

We will return to the proposition of the miners' ability to choose a different network to mine, later.

The “Hard” Way to Destroy a PoW Network: The Genesis Attack

A Genesis Attack on the Bitcoin Network

Recall that according to the original version of the Bitcoin protocol, sometimes called the “vanilla” version,[3] The Longest Chain Rule only requires that the blocks in the longest chain all be properly mined. Furthermore, recall that computers can endlessly duplicate code.

Finally, note that during our explanation, when describing a malicious actor’s attempt to create an empty, meaningless blockchain history, we use quotation marks when employing the word, “false.” Likewise, when describing the blockchain history trusted by the people on the network, we include the word “true” in quotations.

We do this because at the core level, the consensus mechanism is purposefully blind regarding any human user’s preference between “truth” and “false.” The code only sees “truth” in terms of properly forged blocks, and overall blockchain length. Nothing more.

Now suppose the existence of a supercomputer a thousand times more powerful than the entirety of the Bitcoin miner network. This powerful supercomputer could, in theory, stealthily re-create and execute the initial code that spawned the very first block of the Bitcoin blockchain—the “Genesis Block.” The supercomputer could then grind out block hashes, one-by-one, mining meaningless blocks and adding them to this empty, “false” version of the Bitcoin history.

Once this meaningless blockchain’s length sufficiently exceed the so-called “true” blockchain used today, the supercomputer could then release its “false” version to the Internet.

Throughout the world, (assuming the vanilla protocol) the Bitcoin network would automatically recognize the “false” blockchain as the correct blockchain! This would all be according to the code. The so-called “false” blocks would be properly mined, and the length would be longer than the chain that users currently trust. The vanilla protocol would, in theory, replace the so-called “true” history with the empty variant.

It might seem to users like a virus being uploaded to the Internet. It could destroy all human trust in the current version of the Bitcoin protocol, wreaking financial havoc throughout the cryptocurrency realm. While users of the Bitcoin protocol would naturally protest, the entire operation would be entirely in agreement with the underlying code.

Nevertheless, when observing Bitcoin’s current hash power, the creation of such an anti-Bitcoin supercomputer is clearly not feasible in the immediate future. Assuming Bitcoin miners remain interested in the Bitcoin network, the risk of a Genesis Attack on Bitcoin is essentially non-existent.

However, consider the implications of the Genesis Attack on unsuspecting or underprepared smaller PoW blockchain projects.

The More Realistic Dangers of The Genesis Attack

Let us assume a naïve blockchain entrepreneur building a new product. They are generally aware that malicious actors throughout the world are likely to attack their blockchain, stealing funds and otherwise

[3] https://www.worldcryptoindex.com/bitcoin-scaling-problem-explained/

causing trouble. Therefore, the naïve entrepreneur decides to implement what they believe is the most secure method of a blockchain consensus mechanism, PoW, and they offer ample financial rewards to miners to incentivize a secure network.

The entrepreneur and their entire audience may not realize it, but so long as their network's overall hash rate remains below the threshold of an attack by even an average supercomputer, their entire blockchain history is vulnerable to complete annihilation. A technically astute competitor, seeing the vulnerability, and possessing ownership of the requisite computer hardware, would be able to create an empty and longer version of the same blockchain code and vaporize their competitor's financial records.

The cryptocurrency industry is young, and few but the most advanced of developers understand the many ways in which blockchain competition can be technically eliminated. Therefore, we have seen but a few serious cases of The Genesis Attack. One notable instance occurred when an original Bitcoin developer, Luke-jr, used a variation of the attack to destroy a blockchain project called Coiledcoin. Luke-jr performed this attack out of a belief that Coiledcoin was a disingenuous project.[4] Setting aside any human sentiment on either side of the event, the fact stands that Luke-jr's variation of The Genesis Attack was the end of the Coiledcoin network.

The complexity in establishing a secure PoW blockchain remains a challenge for would-be entrepreneurs. Furthermore, there are existing PoW developers that are not fully aware of their vulnerability. Likewise, there are would-be malicious actors that have yet to realize the many methods available to cause frustration. The potential danger surrounding the issue of The Genesis Attack shows the relative youthfulness of the cryptocurrency industry.

For a PoW blockchain network to maintain Bitcoin-level security, therefore, it must maintain a hash rate that is high enough to constantly mine blocks faster than a potential competitor could either perform The 51% Attack (destroying the most recent of transactions), or the deadly Genesis Attack (complete annihilation).

The Financial and Eco-Unfriendly Problems with All PoW Networks

The problems with young PoW networks do not stop there, and furthermore, even Bitcoin's PoW network has issues: the security of a PoW network comes at a high cost to the environment, and miners have no obligation to mine any particular network.

PoW Networks Are Expensive

Some estimate that by 2020, the Bitcoin network alone will consume more electricity than the entire world currently consumes (as of 2017).[5] Having just one PoW network in existence, therefore, is already strain enough on our environment. It is also a burden on our infrastructure and our worldwide economy.

[4] https://bitcointalk.org/index.php?topic=56675.msg678006#msg678006

[5] https://arstechnica.com/tech-policy/2017/12/bitcoins-insane-energy-consumption-explained/

On the one hand, adding additional PoW blockchains to the world can serve the purpose of forcing free-market competition on the Bitcoin developers, encouraging ethical and innovative behavior. Therefore, some competition among PoW networks is likely useful.

However, as a human species, we can consider that there are more financially sound and eco-friendly methods of innovating with blockchain technology without always directly competing with Bitcoin PoW security. Our innovation, delayed Proof of Work, is one response to this fact, as we will soon discuss.

Miners are Free to Mine Other Networks

In November of 2017, for a few hours the majority of Bitcoin network miners switched their hash power to a competitor's PoW network, the "Bitcoin Cash" network.[6] This switch was the result of clever software engineering on the part of the Bitcoin Cash team.

The team recognized that most miners on the Bitcoin network are set to automatically mine whichever network is most profitable. Therefore, the team conducted a calculated change in their underlying protocol that caused the profitability of the Bitcoin Cash network to dramatically increase. The majority of the world's Bitcoin mining equipment, running via automation, recognized the higher profitability and switched to the Bitcoin Cash network automatically.

While Bitcoin Cash's play for a majority hash rate only proved effective for a matter of hours, their accomplishment raised awareness to a tacit principle in the network: Bitcoin's hash rate is not bound to Bitcoin. The hardware is free to serve any compatible network the miners choose.

At the time of the writing of this paper, between Bitcoin and Bitcoin Cash, ~80% of the available hash rate is aligned with the former, and ~20% with the latter. There is speculation in the industry that if the Bitcoin Cash network creates a more favorable position, the balance of hashing power could change on a long-term basis. Furthermore, there are many other blockchain competitors who may gain the attention of Bitcoin's miners in the future.

Were a shift in the balance of hash rate to occur, Bitcoin would no longer be the leader of security in the cryptocurrency realm. The price of Bitcoin would likely drop as users realized the resulting lack of security leadership. This might cause more miners to switch to a more profitable network to cover the cost of operating their expensive hardware. As miners abandon Bitcoin, and as users continue to leave, the situation becomes a reversal of The Network Effect. The Bitcoin network would come crashing downwards at an ever-compounding rate.

This is all theoretical, but it raises yet another concern that we need to illuminate: the security of a blockchain depends on many things, including the potentially fickle support of human blockchain miners. Our innovation, delayed Proof of Work (dPoW), takes this fact into account as we empower members of the Komodo ecosystem with Bitcoin-level security. Before we finally turn to our own solution, we must discuss the primary competitor to the PoW consensus mechanism, Proof of Stake (PoS).

[6] https://www.coinwarz.com/network-hashrate-charts/bitcoincash-network-hashrate-chart

The Primary Alternative Consensus Mechanism: Proof of Stake

Perhaps the most popular alternative consensus mechanism is Proof of Stake (PoS). In this mechanism, blocks are mined not by miners performing work, but rather by any user "staking" their coins on the open network for the right to mine blocks.

The meaning of "staking" has different variations depending on the specific rules set forth by the developers of the unique variant of the PoS consensus mechanism. In general, staking one's coins means placing them as collateral on the open network in exchange for the right to mine new blocks.

Users who stake their coins, thereby, can periodically extract a portion of the mempool, mine new blocks, and earn rewards. There is no need to perform any hardware-expensive proof-of-work calculations, as the user's incentive to be honest is encouraged by the fact that their own wealth hangs in the balance.

The Security Risks and Shortcomings of PoS

The downside to PoS is that a user who simply leaves a large portion of wealth staked (and therefore continually claims rewards) gradually becomes a centralized point of wealth through the power of compound interest. On PoS networks, monopolies are a constant danger. The owner of a monopoly has power over the well-being of the network.

Once a majority of the supply is obtained, the owner gains a position known as "Nothing at Stake." The owner can mine "false" blocks to the PoS blockchain and use their own majority supply over the network to declare these "false" blocks valid. All other stakeholders on the network must adopt these "false" blocks, lest the majority holder use their strength to declare competing blockchain versions as invalid. If a non-majority holder attempts to challenge the monopoly holder's version, the non-majority holder can achieve little more than the loss of coins they placed at stake.

While PoS can be a useful alternative, it has yet to produce a network with the same level of security as the PoW consensus mechanism. The latter does not suffer from the risk of a monopoly, as majority holders gain no unique control over the mining of new blocks.

There are, however, scenarios in which PoS can be useful for entrepreneurs. In the Komodo ecosystem, our dPoW consensus mechanism can provide security to both types of networks on our platform.

Following this summary, we finally turn our attention to our dPoW consensus mechanism.

A Summary of the PoW Consensus Mechanism

In short, the PoW consensus mechanism, as designed by Satoshi Nakamoto, is currently the soundest method of blockchain security. It solves the Double Spend problem and creates a secure network, capable of transferring financial value. Furthermore, competition among miners and The Longest Chain Rule create fairness on the blockchain. The Longest Chain Rule provides a high level of defense against two of the most dangerous methods of blockchain destruction—The 51% Attack and The Genesis Attack—assuming a strong overall hash rate on the network.

New PoW blockchains can opt to compete directly with Bitcoin's hash rate, and some level of competition is good for the ethical values and innovative power of the cryptocurrency industry. However, it is not necessary, cost-effective, nor eco-friendly that every new blockchain innovation requiring security should attempt to compete directly with Bitcoin. Not only is this unsustainable, but it is also unreliable, as it depends on the arbitrary choices of the decentralized network of miners around the world.

The Komodo Solution

Abstract of the Delayed Proof of Work Consensus Mechanism (dPoW)

Komodo presents a technology, the delayed Proof of Work consensus mechanism, that solves the problems described above. Komodo's unique consensus mechanism provides the same level of security as the strongest PoW network, without attempting direct competition. Instead, Komodo's consensus mechanism uses the chosen PoW network as a storage space for "backups" of Komodo transactions. By this method, in the event of a devastating attack, a single surviving copy of the Komodo main chain can serve to rebuild the entire Komodo ecosystem.

In a key difference separating Komodo from regular PoW networks, our dPoW consensus mechanism does not recognize The Longest Chain Rule. Instead, to resolve a conflict in the Komodo network, the dPoW consensus mechanism looks to backups it inserted previously into the chosen PoW blockchain.

Furthermore, asset chains in the Komodo ecosystem can likewise elect to have backups of their own records inserted into the Komodo main chain. These asset chains also do not recognize The Longest Chain Rule, but instead look to the Komodo main chain to resolve conflicts.

Therefore, to destroy even the smallest asset chain that is employing Komodo's dPoW security, the attacker would have to destroy: a) all existing copies of the asset chain; b) all copies of the Komodo main chain; c) the accompanying PoW security network into which the dPoW backups are inserted. This endows the Komodo ecosystem with Bitcoin-level security, while avoiding the excessive financial and eco-unfriendly costs.

Democratically elected notary nodes have the freedom to switch notarization to another PoW network. Reasons the notary nodes might elect to switch networks could include the event that worldwide miners' hashing power changes to another PoW network, or the cost of notarization to the current PoW network becomes more than necessary. Through this flexibility, the Komodo ecosystem maintains both a superior level of security and a more flexible and adaptive nature than Bitcoin itself.

A Note About Komodo's Iguana Core Technology

All the following processes are supported by a deeper Komodo technology called Iguana Core. Readers of our entire white paper will note that Iguana Core is featured in each section. This is because Iguana Core is the heart of the underlying technology that enables the vast Komodo ecosystem work together. The Iguana Core code itself is complex and to fully explain would require a separate white paper.

In short, Iguana Core is a collection of code that serves many purposes. One function of Iguana Core is to empower the blockchain technologies Komodo either builds or adopts to act in coordination with each other. Often, Iguana Core can advance their initial capabilities beyond original expectations. In the case of dPoW, the code that underlies notary-node functionality spawned from Iguana Core technology.

Iguana Core is coded in the C programming language—the language of choice of our lead developer, JL777.

The Notarization Process

The First Step: Gathering the Appropriate Data

The process of inserting backups of Komodo transactions into a secure PoW network we call "notarization." The democratically elected notary nodes in the Komodo ecosystem perform this process via the automation provided by Iguana Core.

The process of notarization is simple. Roughly every ten minutes, the notary nodes perform a special block hash mined on the Komodo blockchain and take note of the overall Komodo blockchain "height" (i.e. the number of total blocks in the Komodo blockchain since inception). The notary nodes process this specific block in such a manner that their signatures are cryptographically included within the content of the notarized data.

The pieces going into the notarization process could look like this:[7]

0a88371cc63969d29492110592189f839557e606db6f2b418ecfe8af24451c07

- This is the "block hash" from the KMD blockchain—mined and cryptographically signed by the notary nodes

Block 607240

- This is the blockchain "height" of the Komodo blockchain at the time of notarization (i.e. the total number of KMD blocks ever created)

KMD

- The letters "KMD" are added into the notarization mixture to indicate the name of the blockchain to which this notarization belongs

The notary nodes will take these three pieces of information and compress them into a format that is more computer-friendly. The result will look like this:

6a28071c4524afe8cf8e412b6fdb06e65795839f189205119294d26939c61c37880a084409004b4d4400

The above number can be said to be a cryptographic representation of all that has happened on the Komodo blockchain up to this point in time. According to The Cascade Effect, were an attacker to attempt to go back in the history of the Komodo blockchain and change even a single character of data, and then attempt to recreate the same numerical representation according to the correct Komodo code, the number above would dramatically change.

This makes the notary nodes' notarization a useful backup, assuming this number is in a safe location where anyone on the Internet can view and verify it. It enables a single surviving copy of the "true" Komodo main chain to identify itself to the rest of the Komodo network as the correct version, as only the "true" data can produce the same result. On the other hand, an incorrect history of the Komodo

[7] All examples herein are estimated based off this actual KMD notarization to the BTC network: https://www.blocktrail.com/BTC/tx/313031a1ed2dbe12a20706dff48d3dffb0e39d15e3e4ff936d01f091fb3b8556#tx_messages

network will not be able to produce the same notarization. Therefore, all nodes on the Komodo network will align with the “true” blockchain history and overwrite any malicious actors’ “false” attempts.

Step Two: Notarizing the Data to a Secure Location

Naturally, for security purposes this number cannot simply be saved to one person’s local computer, or written down on a piece of paper. Were the number to be in such a centralized location, a would-be attacker could simply destroy the backup, or replace it with a “false” version. For the number to be useful, it must be placed in a secure and decentralized location.

Here is where Komodo adopts security from another network: Komodo will perform a simple transaction in which it writes the above number into the data history of the strongest PoW blockchain (currently Bitcoin). This location is as secure as the miners’ hash rate makes it, and the location is decentralized, by nature.

To place this information in the accompanying PoW network, the notarizing nodes will use a feature that exists at the core of the Bitcoin protocol when making a transaction. The feature is called “OP_RETURN,” and it allows for a message to be added to the blockchain, permanently, alongside the transaction hash.

A notable use of the ability to write messages to a blockchain is found in the first actions of Satoshi Nakamoto himself (themselves). In the first Bitcoin block ever mined, Satoshi used a feature like OP_RETURN[8] to include this message:

03/Jan/2009 Chancellor on brink of second bailout for banks[9]

Readers who have downloaded the Bitcoin blockchain to their local computer, and who possess the knowledge necessary to inspect the raw Bitcoin data, can discover these very words written to their own hard drive. The important thing to understand for our discussion is that any message written to a secure and decentralized PoW blockchain is viewable and verifiable to all.

The permanence and security of OP_RETURN messages are a core aspect of dPoW’s security. In the event of a devastating attack on the Komodo network, there need be no argument over the correct notarized marker upon which the ecosystem members should rely. The Iguana Core code and Komodo blockchain can automatically turn to the chosen PoW network to rebuild.

The Third Step: Notarizing the PoW Network Information Back to the KMD Main Chain

One final step remains to complete the loop of security between the KMD main chain and the chosen PoW network. The KMD blockchain must record within its own records the specific location where it placed this backup into the PoW blockchain. This serves to remind the Komodo ecosystem itself where to look for the proper information.

[8] Nakamoto used a feature called “coinbase,” which is similar to OP_RETURN. A primary difference between coinbase and OP_RETURN is that coinbase is used by miners when mining a block, whereas OP_RETURN can be used by regular users when performing transactions.

[9] https://en.bitcoin.it/wiki/Genesis_block

To create this reminder, the notary nodes will now gather one more piece of information, this time drawn from the accompanying PoW network: the transaction hash (txid) identifying the location of the first notarization. This information could look like this:

313031a1ed2dbe12a20706dff48d3dffb0e39d15e3e4ff936d01f091fb3b8556

The notary nodes will combine it with all the information that has come before. The result will be transformed, again, into a computer-friendly version:

**6a28071c4524afe8cf8e412b6fdb06e65795839f189205119294d26939c61c37880a0844090056853bfb91
f0016d93ffe4e3159de3b0ff3d8df4df0607a212be2deda13130314b4d4400**

Just as before, this number is a compressed cryptographic representation of everything that has happened in the Komodo ecosystem up to this point in time.

This notarization is placed as a transaction message directly into the KMD main chain itself. This enables the Komodo ecosystem to know how to find a reference of its own history, should the need ever arise.

A Brief Discussion of Komodo's Protective Measures in Action

There are myriad ways that an attacker can assail a blockchain project, and the Komodo ecosystem is well prepared. In this foundational paper, we only discuss two of the most crucial attacks—The 51% Attack and The Genesis Attack.

In a separate technical white paper, written by our lead developer, we provide several more discussions on how Komodo responds to many other forms of attack. Some mentioned therein include The Sybil Attack, The Eclipse Attack, The Double Spend Attack, and more. We encourage any reader searching for information about the deepest levels of Komodo security not only to read the accompanying white paper, but also to reach out to our team directly on our Slack channel.

Notary Nodes and Iguana Core Provide a Defense Against The 51% Attack

The Komodo network on a surface-level is a minable network, like other PoW networks. Any technically savvy user can activate a device capable of mining the Komodo network, and thereby mine blocks and receive rewards. For these miners, the Bitcoin protocol functions normally. With each block header, clues are provided for miners to find the next valid block hash. The specific clue, "difficulty," changes with each block header.

Understanding "Difficulty" on a PoW Chain

On a regular PoW chain, the "difficulty" clue is even more pertinent to our discussion. With each block header, the difficulty level can change. The Bitcoin protocol itself decides what the difficulty for the next valid block should be.

The difficulty is decided based on the amount of overall hash power mining the network. If many miners are mining the network, then the hash rate is high, and the Bitcoin protocol sets the difficulty to a higher number. On the other hand, if the hash rate is low, then the protocol sets the difficulty to a lower number.

Recall that the "difficulty" level determines the number of zeros at the beginning of the next valid block hash. The more zeros at the beginning of a valid block hash, the more unlikely each attempt at finding a valid block hash will be.

When the Bitcoin protocol was in its infancy, the difficulty setting was easy. In fact, the block hash we used earlier as an example is, in truth, the very first block hash ever created—by Satoshi Nakamoto himself.

000000000019d6689c085ae165831e934ff763ae46a2a6c172b3f1b60a8ce26f

He (they) designed the difficulty setting to encourage the network to find new block hashes once every ten minutes, on average.

For a computer, to guess within ten minutes a nonce that will produce a block hash beginning with ten zeros is relatively easy. It is so simple, in fact, no special computer is required. Early Bitcoin miners could use nothing more than the average desktop machine, having the CPU—the small heart of the computer—performing the calculations.

As more miners joined the network, however, the Bitcoin protocol automatically increased the difficulty. This maintained the speed at which the pool of all miners discovered new blocks, despite the increased size of the pool. Stabilizing the speed ensured an amount of economic predictability upon which users could rely, among many other benefits.

Today, at Bitcoin's current level of overall hash power, a valid block hash requires a much higher level of difficulty. Here is a recent successful block hash:

0000000000000000002d08398d6f21f038019600266b419bad5ab88add5b638d

There are seventeen zeros, and to find a valid block hash at this level requires a prodigious effort.

In the race to win blockchain rewards, miners all over the world have built entire farms of specialized equipment for mining. The small CPU of a desktop is no longer useful, and the time of "easy difficulty" on Bitcoin has passed.

dPoW Has Democratically Elected Notary Nodes

Here is where our dPoW consensus mechanism diverges from the Bitcoin protocol's limitations and creates new levels of security for the Komodo ecosystem.

The code of our dPoW consensus mechanism allows for sixty-four democratically elected "notary nodes." They are a special type of blockchain miner, having certain features in their underlying code that enable them to maintain an effective and cost-efficient blockchain. The periodic elections are held by popular vote.

The Komodo blockchain endows notary nodes with many additional abilities. These abilities help to maintain cost-effectiveness and an eco-friendly nature in the ecosystem. They also prevent the ecosystem from falling into the trap of directly competing with other networks for hash-rate security status.

Each Notary Node Gets One Chance Per Every Sixty-Five Blocks to Mine on Easy

Notary nodes have a specific ability that enables them to protect against The 51% Attack. Each individual node periodically receives the privilege to mine a block on "easy difficulty." It has this ability until it mines one "easy" block, and then the Iguana Core code removes the "easy difficulty" for that notary node for the next sixty-four blocks. After the sixty-four-block period passes, the notary node can once again attempt to capture a block on "easy difficulty."

Therefore, while everyone else on the network mines at an adjustable level of difficulty according to the normal PoW consensus mechanism (which keeps the overall speed of the Komodo network stable) the notary nodes have a chance to step outside the normal rules. For every sixty-five-block period on the Komodo blockchain, the odds that a block will be mined by a notary node, as opposed to a normal miner, are essentially 3:1.

Since the rest of the miners have an adjustable difficulty ratio, it does not matter how many more miners attempt to mine Komodo. Most of the valid blocks will always be found by the sixty-four democratically elected notary nodes, even were the entire hash power of the Bitcoin network to somehow switch all its attention to mining Komodo.

The Free-for-All Period

Every 2000 blocks, the Iguana Core code removes the easy-difficulty mining ability from all notary nodes for a sixty-four-block period. This gives the entire ecosystem the chance to freely mine the Komodo blockchain—and therefore the chance to control the "truth" of the Komodo history.

The Combination of Features Protects Against The 51% Attack

To brute force a 51% majority over the network, therefore, a malicious actor would have two challenges. They would first have to compromise most of the democratically elected notary nodes, and then provide a 51% majority over the network's hash throughout the period of the Free-for-All.

Were a malicious actor to compromise the notary nodes, the most the actor could do would be to prevent further notarization—and therefore destroy their own network. Were the notary nodes to attempt to falsify the blockchain history, the Free-for-All period would allow the normal miners on the network to correct the record.

So long as Komodo-ecosystem members choose their notary nodes wisely, The 51% Attack is essentially impossible to perform against Komodo.

The Ultimate Defense Against The Genesis Attack

One of the deadliest attacks against a vanilla version of the Bitcoin protocol is The Genesis Attack, as we previously discussed. The key to successfully performing a Genesis Attack is to recreate the genesis block of the victim's blockchain, mine a blockchain height that is greater than the so-called "true" blockchain, and release this meaningless variant on the Internet. The underlying Bitcoin protocol can immediately vaporize the so-called "true" blockchain, and thus the decentralized record of transactions on the network are lost.

The key vulnerability that this attack exploits is the very rule that, under other circumstances, makes a PoW network so powerful: The Longest Chain Rule. On one hand, assuming a strong hash rate, The Longest Chain Rule provides security against attacks such as a Double Spend. This feeling of security encourages growth in the user base, which incentivizes growth in the miner base, and this cycle creates a Network Effect. However, if the hash rate of the PoW network is low, The Longest Chain Rule creates the vulnerability of chain vaporization.

In the Komodo project, we are not attempting to compete with PoW networks, such as Bitcoin, for overall hash-rate security services. Our end goal is to create a secure, powerful, and open-source blockchain ecosystem, with the option of privacy features. Bitcoin-level security is necessary to achieve our goals, but it does not need to be an endeavor unto itself. Therefore, we simply chose to turn off The Longest Chain Rule in our code, and instead adopt the strength of the strongest PoW network.

If an outsider was to attempt The Genesis Attack on our ecosystem, the Komodo blockchain would simply ignore this “false” blockchain history. The Iguana Core code resolves conflicts by looking to the chosen PoW network for the historical KMD markers. Therefore, the Komodo ecosystem maintains Bitcoin-level security from the deadliest of attacks without obeying The Longest Chain Rule.

The dPoW Consensus Mechanism is Inherent in All Komodo Asset Chains

These security features extend to any asset chain of the Komodo ecosystem. The primary difference between an asset chain and the main chain is that the main chain notarizes to an exterior PoW network (Bitcoin), whereas the asset chain notarizes to the Komodo main chain.

An entrepreneur in our ecosystem has the option of Bitcoin-level security, at only a fraction of the cost. The entrepreneur need only employ the notary nodes’ services to notarize backups of their blockchain to the Komodo network, and thereby the entrepreneur is protected from even the most devastating of blockchain attacks.

In the following section, Part II, we begin our discussion of an entrepreneur’s formation and distribution of a Komodo asset chain.

Part II

The Decentralized Initial Coin Offering

Abstract of the Decentralized Initial Coin Offering

There lies a great power in the idea that any person, regardless of nationality, creed, or background, can obtain funding to innovate and prosper. An integral tenet of blockchain technology is "decentralization." By decentralizing systems, we reduce the number of control points that can be compromised and manipulated. Decentralization plays a more common role in our new cryptocurrency economy, but there is one area of the market that remains centralized and vulnerable: the initial coin offering (ICO). The cryptocurrency industry needs a solution, and Komodo presents an answer with our decentralized initial coin offering (dICO).

In today's common ICO model, the high level of centralization creates many problems. Third-parties can block or manipulate entrepreneurs' efforts to innovate and prosper. The centralized location of releasing the ICO blockchain product is vulnerable, allowing whales, hackers, and human error to corrupt or destroy an entrepreneur's efforts. The negative experience of users in these situations can also impact the perception and adoption of cryptocurrency. Furthermore, the traceable nature of an ICO prevents society from crowdsourcing and purchasing within our inherent right to barter in private.

The dICO model, as created by the Komodo project, overcomes these challenges. It provides the necessary technology to create and release a blockchain product to the world with the full power of decentralization.

Entrepreneurs building on our platform begin by creating an asset chain, and our technology simplifies this process. One need only install the necessary software, execute a few commands on a command prompt, and then establish a connection between two or more Komodo-enabled devices. Komodo's core technology will do the rest of the work necessary to create a fully independent blockchain, empowered with an array of Komodo features.

Our **dPoW** technology is a key feature, as explained in Part I. dPoW provides the necessary security to protect the integrity of the blockchain. Use of dPoW is optional, and since asset chains in the Komodo ecosystem are independent by nature, entrepreneurs can discontinue dPoW services at will.

Having thus created the blockchain, the entrepreneur then uses our **decentralized exchange** to release the project to the world in a decentralized manner. Our decentralized exchange is called, **BarterDEX**, and it is thoroughly explained in Part III of this paper. Because BarterDEX is a decentralized exchange, and through our **atomic-swap** technology (also explained in Part III), no third-party manipulators can prevent the entrepreneur from their crowdsourcing and innovation endeavors.

Through our privacy technology, **Jumblr**, dICO participants can purchase the product within their inherent right to barter in private. A detailed explanation of Jumblr and its method of providing privacy is provided in Part IV of this paper.

The Challenges in Current ICO Platforms

Specific Weaknesses in the Centralized ICO Model

There are many weaknesses present in today's Initial Coin Offering (ICO) process. Several notable weaknesses include third-party discrimination, "whale" manipulation, the vulnerability to theft and human error, and a lack of privacy.

Third-Party Discrimination

An entrepreneur seeking to serve their intended audience may experience adverse intervention from a third party. The antagonists may display personal and malicious intent, regardless of the value of the entrepreneur's innovation.

Centralization of Technology: "Whale" Manipulation, Theft, and Human Error

During the initial stages of a blockchain's release to the public, users who are wealthy and tech-savvy (often referred to as "whales") have an unequal advantage: they can rapidly purchase a majority of the coin supply while it is inexpensive. Thereafter, they can manipulate the market price at the expense of less established ICO participants.

Furthermore, today's ICOs are generally conducted in escrow, where the purchasers must transfer money to one node for holding. This typically occurs through a single website, and the cryptocurrency funds are held on a single server. They must then wait while the ICO administrators first verify the transactions and distribute the coins. During this time the funding is centralized, and therefore vulnerable to thieves and human error.

Lack of Privacy

Because ICO transactions are highly traceable it is difficult, if not impossible, to perform ICOs within our right to barter in private.

Third-Party Discrimination via the Centralized ICO

One weakness of the ICO process is, paradoxically, rooted in a great strength of blockchain technology: its borderless nature. A key power of any blockchain is that any human capable of accessing the technology can activate the blockchain, regardless of their geographical location or social status. Thus, anyone can provide yet another verifiable record of the transaction history, and this decentralization provides a crucial element of security to the blockchain.

An ICO innovator, therefore, may prefer to use a blockchain platform that transcends man-made barriers, to protect their innovation. Circumventing man-made barriers could be integral to the blockchain's survival, because the element of decentralization prevents malicious actors from creating subjective borders around the blockchain records and then using authority to falsify and manipulate.

This creates a conundrum, however. As a human race, we also find strength and empowerment in subjectively defining our own demographics for various reasons, whether they be to form companies, cultures, communities, etc. While we find the ability to create subjective demographics useful, it contrasts with the borderless nature of blockchain technology. Members of one demographic may desire to participate in a specific ICO, but another demographic may find this unfavorable. Therefore, the second party might try to forestall progress. The paradox lies in the fact that for the underlying blockchain product to maintain its integrity, it must serve both communities without regard to any man-made barrier between them.

The problem compounds even further as we observe that on a decentralized blockchain platform, a new ICO product is capable of functioning anywhere there is access to the underlying technology. Therefore, on a decentralized platform, once a new blockchain product is released any person from either demographic is now able to utilize it regardless of the overall sentiment of either demographic. The problem becomes most pronounced if members of a competing group attempt to even maliciously prevent an innovation out of selfish reasons. Thus, it is imperative that the innovator have the option of protection against would-be malicious competitors.

The overall centralized nature of today's ICO process, therefore, presents a problem. Entrepreneurs who are not able to navigate the adverse effects of an inhibiting third party may be unable to realize their creative potential.

Centralization of ICO Technology: Whales, Hackers, and Human Error

Yet another issue plaguing ICOs is that the technology upon which an ICO is released is also centralized. This presents a vulnerability to human foibles.

The Manipulative Behavior of Whales

The centralization of the point of purchase creates an unequal playing field in favor of wealthy, tech-savvy users (referred to as "whales" in the cryptocurrency community). To understand this problem, one must comprehend that "nodes" (computer devices which compute the buying and selling of cryptocurrencies) take orders from ICO purchasers one-by-one. Presently, ICOs are released on only one node — for example, the purchase could take place through a single website, wherein the gathered funds are held on a single server.

Because the node can only process one transaction at a time, the person whose order arrives first will receive an advantage over the coin's future value. If the initial purchaser is both wealthy and able to program sophisticated "bots" (custom-designed programs that automate the trading of cryptocurrencies), the whale can buy a controlling interest in the supply before less wealthy or less technologically savvy people have a chance to participate.

In our current market, often the people who would most benefit from an ICO are unable to participate before the supply evaporates. Meanwhile, this whale now has sufficient control on the overall supply to act as a centralized market manipulator. Buying and selling in large quantities forces fluctuations in the whale's favor.

Hackers and Human Error

Because all coins of an ICO typically process through one node during the purchasing period, the entire supply is vulnerable to any person with access to the node. Therefore, both malicious and clumsy human agents can destroy an ICO. The data holding the cryptocurrency can be damaged, stolen, or simply lost through incompetence.

An entrepreneur can also consider that in today's ICO model both the funding provided by the purchasers, as well as the actual ICO coins that the entrepreneur intends to sell, remain on the centralized node for a long period of time. It is not just one side of the crowdsourcing endeavor that is at risk, but both.

This central point of failure can be catastrophic for all participants.

The Right to Barter in Private

Finally, the lack of current privacy options in the ICO process inhibits blockchain participants from purchasing within our right to barter in private. This right to privately exchange goods and services extends further into history than the written word. We have, as a species, utilized this right to organize into communities, institutions, and even nations.

Many of humanity's most meaningful advancements in art, technology, and other human endeavors began in situations where the creator had the security of privacy in which to explore, to discover, to make mistakes, and to learn thereby.

The right to barter in private, however, is under modern threat as the recent monumental and historical phenomenon, "The Internet of Information," permits many kinds of people to quietly and without inhibition; monitor other people's shopping and bartering behavior. This is a dangerous development, as it destroys the privacy that empowers much of humanity's personal growth. We must reserve our right to barter in private, for we observe that there are myriad ways in which a common person may explore personal growth in an economic environment.

Yet, the highly traceable nature of today's centralized ICO model is in direct contradiction to this human need.

The Blockchain Industry Needs a Solution, and Komodo Presents an Answer

Together, these issues show that the current state of the ICO market is plagued with limitations that inhibit freedom, security, entrepreneurship, and even human growth. The cryptocurrency industry needs a solution to these problems, and Komodo presents an answer.

The Komodo Solution

The Decentralized Initial Coin Offering

The Komodo ecosystem presents a solution, the decentralized the initial coin offering (dICO), that solves these issues and even adds new possibilities to the cryptocurrency market. The decentralized nature of the dICO enables the entrepreneur to release a blockchain product beyond the reach of a malicious third-party influencer. Furthermore, through our decentralized exchange, BarterDEX, the dICO allows an entrepreneur to release their product in a manner that mitigates and even eliminates many of the issues regarding whales, hackers, and human error. With the advantage of Komodo's privacy technology, Jumblr, the participants in a dICO are empowered with their right to barter in private.

Our decentralized exchange, BarterDEX, is explain in detail in part III. An in-depth discussion of our privacy technology, Jumblr, is provided in Part IV.

The Process of Creating a New Blockchain in the Komodo Ecosystem

Note: If you are interested in performing your own dICO to become a part of the Komodo ecosystem, please reach out to our team directly on our website, komodoplatform.com. We are actively seeking new partnerships.

Formerly, coding and generating the blockchain itself were a most difficult aspect of the development process. Now, the Komodo team has simplified the process into easy steps. Through Komodo's Iguana Core technology (introduced in Part I), the entrepreneur can create a new independent blockchain by entering just two simple commands in the command prompt interface of their computer.

The following steps rely on one of Komodo's underlying software processes that run in the background on a user's computer. The name of this software is the "Komodo daemon," or Komodod, for short. Komodod is rooted in Iguana Core technology.

The First Command to Create a New Coin

```
./komodod -ac_name=[ENTREPRENEUR'S COIN] -ac_supply=[TOTAL COIN SUPPLY] -gen
```

The first part of the command, [./komodod], initiates a new instance of Komodod.

By default, the initial [./komodod] command executed alone would launch the Komodo main chain, KMD, on the user's computer. However, the next part of the command tells Komodod to behave differently.

```
-ac_name=[ENTREPRENEUR'S COIN]
```

This command tells Komodod to look for a coin with the inserted name

```
-ac_supply=[TOTAL COIN SUPPLY]
```

This tells Komodod how many total coins there should be in this chain

```
-gen
```

This tells Komodod that the user desires to mine this network

The underlying code of Iguana Core can now make several decisions. First, it will check its connection to the Komodo ecosystem to see if there is a coin by the name of [ENTREPRENEUR'S COIN], having a coin supply of [TOTAL COIN SUPPLY]. If the coin name and total supply are not found, Komodod will assume that the user is attempting to create a new coin, and the [-gen] command tells Komodod that the user wants to mine it.

Komodod now begins the automated process of creating a new asset chain in the Komodo ecosystem. Komodod will first make a fresh and empty clone of the KMD main chain (though it will not yet generate the actual coins), with only a few differences to the underlying nature of the chain.

The Features of the New Asset Chain

There are several primary differences between an asset chain and the main Komodo chain. For example, the asset chain will not automatically generate the 5% APR reward for all wallet addresses holding coins, unlike the main chain. Furthermore, the asset chain's dPoW consensus mechanism is built to notarize to the main chain (as explained in Part I).

Some of the differences reveal strong advantages held by members of the Komodo ecosystem. By design, this asset chain is capable of automatically adopting any updates that the Komodo core development team add to the framework. The asset chain also has a built-in capacity within the framework to allow the entrepreneur to code new rules.

For example, the entrepreneur may decide not to use a PoW consensus mechanism, but may instead prefer PoS (discussed in Part I). Other changes can also be made, according to the entrepreneur's imagination and developer knowledge. So long as the new code that the entrepreneur adds to the asset chain does not interfere with the overall framework, the asset chain will smoothly integrate with the rest of the Komodo ecosystem. We provide more details on this topic in Part V's section regarding **Smart Contracts**.

For the purposes of our discussion, this new asset chain is otherwise the same as the Komodo main chain, including the features to communicate natively with other chains in the Komodo ecosystem via BarterDEX.

The reader may note that this new Komodo asset chain is not a colored-token running on top of a parent blockchain, as is often the case in other blockchain ecosystems (consider the ERC20 token of the Ethereum platform). Instead, this asset chain is an entirely unique and independent blockchain unto itself.

This empowers the entrepreneur with significant advantages over other blockchain ecosystems. The asset chain can run on its own nodes, act according to whatever rules the entrepreneur can imagine, and can scale according to its own audience. Should an asset chain in the Komodo network experience a sudden explosion of activity, the sudden change will not negatively impact the overall Komodo ecosystem. This independence grants a significant competitive advantage in the form of overall security, speed, and ease of use.

Consider the advantage of developing an entrepreneurial product as a fully independent blockchain. Should the entrepreneur desire at a future point to leave the Komodo ecosystem for any reason, they are free to take their blockchain product with them.

Generating and Mining the New Coins

Let us return now to the moment after the entrepreneur executes the first command in the command prompt, and Komodod creates a fresh and empty clone of the Komodo main chain. While the instance of the Komodod program (running on the entrepreneur's local computer device) will create the necessary code for the new asset chain, Komodod will not yet generate the coin supply itself. Komodod instead will wait for the next few steps to occur.

The reason for the wait is that a blockchain's essence depends upon existing not in isolation, but in a network of multiple devices connected. This is the nature of decentralization. Komodod will wait until it receives a signal from another device, thus indicating that it has a peer with which to form the asset-chain network.

The Entire Coin Supply is Distributed in the Genesis Block

It is imperative to note that in the Komodod process, the entire coin supply is created and distributed immediately to the device that mines the first block, the Genesis Block. The code performs this distribution as a one-time reward for discovering the first valid block hash (as explained in Part I).

Due to the sensitive nature of this step, we recommend that the entrepreneur use a Virtual Private Server (VPS) service. This allows two secure devices to connect to each other with little, if any, risk of a third-party actor mining the first block (which would thus enable a would-be thief to acquire the entire coin supply before distribution).

Having established a secure connection with a second device, the entrepreneur will enter the following command on the second device.

```
./komodod -ac_name=[ENTREPRENUER'S COIN] -ac_supply=[TOTAL COIN SUPPLY] -addnode=[INSERT IP ADDRESS OF FIRST DEVICE]
```

Note that the first three elements of the command, [./komodod], [-ac_name], and [-ac_supply], are the same. It is important that the parameters inserted into these commands match exactly. Otherwise, the instances of Komodod running on the separate devices will ignore each other, and the coin will not be mined.

Note also that the [-gen] command is not present. In this circumstance, we are assuming that the entrepreneur wants to capture the entire coin supply on the first device. Technically speaking, assuming the entrepreneur has ownership over both devices, it does not matter if both devices initiate the [-gen] command. Both devices will attempt to mine the first block and the superior device will receive the coin supply.

There is one key difference in the command.

```
-addnode=[INSERT IP ADDRESS OF FIRST DEVICE]
```

An "IP address" can be compared to a human being's home mailing address, where the IP address is designed for computers to be able to geographically find each other.

With the execution of the IP address command, the second device knows to look across the available connection (the Internet, VPS service, etc.) for the first device, which is already running an instance of

Komodod and the new coin. The command here simply tells the computer the proper IP address of the first device.

As soon as these two devices connect, having all the proper Komodod software running and set in place, the mining begins. One of the devices will mine the first block and instantly receive the total coin supply of the entire blockchain into the user's chosen wallet.

Both devices sync this information to each other, and the [ENTREPRENEUR'S COIN] now exists in the world. The entrepreneur can also add more and more devices to the network.

Notarizing to the Komodo Main Chain

To receive the security of the dPoW consensus mechanism, the entrepreneur simply needs to have the democratically elected notary nodes add the [ENTREPENEUR'S COIN] to their internal list of coins to notarize. This will empower the entrepreneur's product with the same verifiable and decentralized security of the Komodo parent chain.

The process of adding a new notarization service can be executed by the notary nodes with just a simple command. While we are at this early stage of development, this sign-up process for new dICO products is not yet automated. In the future, we intend to automate as much of this process as possible.

There is a fee for receiving notarization services. This helps to cover the business costs associated with notarization (recall that all notarizations are financial transactions, by nature).

Our first partners, Monaize, are already successfully notarizing to the Komodo main chain. We are actively seeking our next partners, and we encourage the reader to reach out to our team directly for inquiries.

Entrepreneurs are thus able to use the asset chain's native dPoW consensus mechanism to notarize to the Komodo main chain to create a secure backup of the coin's history. Even in the event of an attack at this early state of existence the entrepreneur can rest assured that their product will survive, so long as one copy of the blockchain's history exists.

Everything is set on the backend for the entrepreneur, and they are now fully prepared to begin the dICO process. Naturally, we understand that for many potential entrepreneurs in the Komodo ecosystem, this process is unfamiliar territory. We encourage interested entrepreneurs to reach out to our team for guidance during development.

The Distribution of Coins

The Trials and Travails of the Centralized ICO Method

Previously, the entrepreneur at this point would have been required to go through a centralized ICO process.

This could have required several cumbersome and possibly dangerous steps. For example, the entrepreneur would begin gathering cryptocurrencies from their audience to personally hold in escrow while the process of matching purchases to the new blockchain coin were verified.

To distribute these coins, the entrepreneur had two primary options. They could have created and distributed a digital software wallet capable of holding the entrepreneur's coins. This would require their audience to download the software. The entrepreneur would then have to send all the appropriate coins to each wallet address, according to the process they established during their ICO.

Or, the entrepreneur would have to make formal arrangements with another service to manage this process, such as with a centralized exchange. This would require a successful negotiation with this third party, likely paying fees as a part of the agreement. The entrepreneur would then be required to act within the centralized exchange's arbitrary framework.

The centralized ICO process can be arduous and, at times, disastrous.

Enter The dICO

Powered by Komodo's BarterDEX & Jumblr Technology

The Komodo dICO model is an extension of Komodo's BarterDEX technology. BarterDEX is an atomic-swap powered, decentralized exchange. It enables users to directly exchange cryptocurrencies from one person to another without third-party involvement (i.e. no centralized exchanges, escrow services, vouchers, etc.). Furthermore, as the dICO model is entirely decentralized, anyone can use it at will. There are no centralized authority figures capable of creating artificial control points that can be manipulated at the expense of the users. Please turn to Part III for more details.

To begin the distribution process, the entrepreneur first chooses how many nodes they would like to use for the distribution. Nodes can be any type of machine capable of connecting to BarterDEX. Typically, a small-business entrepreneur may choose to use server machines. Server capacity can be rented online, and the servers can be distributed geographically throughout the world, if desired. As an example, our e-banking platform partner, Monaize, is working with us to perform the world's first official dICO launch. Monaize is employing 70+ servers throughout the world for the duration of their dICO.

While renting a multiplicity of servers may be the method of choice for an established small-business, it is not a requirement. An owner of an even smaller business, operating on a low budget, can simply use their own computer(s), geographically stationed nearby for convenience. On the other hand, a large corporation could use the server capacity they already own. The number and strength of the machines is a choice made by the entrepreneur.

Having decided the method of distribution, the entrepreneur will then prepare the total supply of coins. (We are assuming the coins are still located on the first device that mined the entrepreneur's genesis block.) The entrepreneur will first break down the total collection of coins into smaller digital pouches. These small bags of coins are ultimately what will be traded on BarterDEX with their audience. The size of the bags is chosen by the entrepreneur, and therefore the entrepreneur can choose a size that is agreeable to their outlook on any KYC legal requirements. For a detailed explanation of the process of breaking down the total collection into smaller bags of coins, we also recommend reading about UTXO technology in Part III of this paper.

Having created these bags of coins, the entrepreneur then sends them to all chosen nodes throughout the BarterDEX network. Coins are distributed to each node's wallet(s) by a normal transaction.

With the coins distributed as desired, the entrepreneur sets the time and date when each bag of coins will be available for purchase. When a bag of coins becomes available on BarterDEX for trading, members of the Komodo ecosystem simply purchase the coins. Please see our discussion on atomic-swap technology in Part III for more details.

The Many Solutions of the dICO Model: Security, Privacy, Decentralization, and Freedom

This method of conducting a decentralized initial coin offering mitigates and circumvents the issues found in a centralized ICO. The entire process is conducted in a decentralized manner. The dICO entrepreneur has direct access to their audience, as there are no centralized human authorities acting as middlemen.

Because the bags of coins can be distributed across a vast range of nodes, and because the entrepreneur can program the time at which each bag of coins becomes available, it is possible to prevent a "whale" from seizing a majority control in one swooping moment of the dICO. The whale will have to compete to purchase their desired amount one transaction at a time, just like the other members of the ecosystem.

Furthermore, BarterDEX has advanced trading features that provide additional whale resistance. For example, BarterDEX can perform ten to twenty trades at once, unlike a normal node in the typical ICO model. Therefore, even if the whale were able to place large orders on every node of a dICO, BarterDEX would still be performing orders simultaneously for other members of the Komodo ecosystem.

Concerning theft, the dICO provides solutions to both methods of theft in the centralized ICO. Unlike the centralized ICO, once the distribution of the bags takes place the effect of their distribution adds a layer of security from a would-be hacker. The hacker can only steal funds at the node they manage to penetrate. Were the hacker to steal coins before the actual dICO, the entrepreneur would have the option to simply create a [NEW ENTREPRENEUR'S COIN] again, without losing any personal wealth.

Furthermore, since the trades happen instantaneously with each bag available for sale, the entrepreneur is only in possession of either their own [ENTREPRENEUR'S COIN], or the cryptocurrency funds provided by the dICO participants—but not both. The entrepreneur is never at risk of losing both their own funds and the funds of their audience, which is a strong advantage over today's ICO model.

Regarding human error, should one of the node's databases be corrupted by accident or hardware failure, only one node's coin supply is lost.

Since the coins are immediately available on the BarterDEX exchange for trading, the entrepreneur's audience has an immediate trading market. This stands in contrast to today's ICO model, where users often wait weeks or even months before liquidity for their ICO product arises in a centralized exchange.

Finally, through Jumblr technology, participants have the option of privacy when purchasing the dICO product. This enables them to support the crowdsourcing efforts of the entrepreneur within their inherent right to barter in private.

Upon conclusion of the distribution of the dICO coin supply the entrepreneur has successfully and immediately completed all the crowdsourcing-related steps that could have taken months in today's typical ICO model.

Komodo's dICO model is significantly easier, freer from manipulation, more flexible, and more secure.

Part III

Komodo’s Atomic-Swap Powered, Decentralized Exchange: BarterDEX

Abstract

Komodo's decentralized exchange, BarterDEX, allows people to trade cryptocurrency coins without a counterparty risk. The protocol is open-source and trading is available for any coin that any developers choose to connect to BarterDEX. The parent project, Komodo, freely provides BarterDEX technology through open-source philosophy. Our service fully realizes decentralized order matching, trade clearing, and settlement. The order-matching aspect uses a low-level pubkey-to-pubkey messaging protocol, and the final settlement is executed through an atomic cross-chain protocol. Like any exchange, our decentralized alternative requires liquidity, and we provide methods and incentives therein.

Introduction

The current, most practical method for cryptocurrency exchange requires the use of centralized exchange services. Such centralized solutions require vouchers to perform the exchange. Among many dangers present in this system, end-users are under the constant risk of their assets being stolen either by an inside theft or an outside hack. Furthermore, the operators of centralized exchanges can exhibit bias in how they facilitate trading among their users. They can also create fake levels of volume on their exchange. To eliminate such dangers and limitations requires the creation of a decentralized-exchange alternative.

Among all the centralized exchanges, trading tends to coalesce around a few of the most popular. There is a reason for this behavior. Trading via vouchers is fast; a central exchange can swap internal vouchers instantaneously, whereas trading actual cryptocurrencies through human-to-human coordination requires communication from both parties. It requires waiting for blockchain miners to calculate transaction confirmations.

The speed advantage of a centralized exchange, therefore, creates a compounding effect on the centralization of traders. The faster processing time of vouchers attracts more people: the increased presence of traders creates higher liquidity: with more liquidity, the exchange can feature better prices: the higher quality of prices in turn attracts a larger community, and the cycle. This is a classic Network Effect, and it is the reason that a few centralized exchanges dominate with high-volume trading, while smaller exchanges—both centralized and decentralized—suffer from a lack of liquidity.

The Beginnings and Travails of Decentralized Exchanges

In 2014 a project called The MultiGateway created one of the first decentralized resources for trading cryptocurrencies. The MultiGateway relied on a separate, though related, blockchain project called The NXT Asset Exchange. The latter facilitated the decentralized exchange of blockchain coins by using proxy tokens (as opposed to vouchers), and these proxy tokens represented external cryptocurrencies (such as Bitcoin).

The underlying technology of this solution is still in use by many blockchain platforms, but the proxy-token protocol is too limited to compete with centralized exchanges. Because trading by the means of proxy tokens requires trading on an actual blockchain, the trading process loses the speed of a centralized exchange. Also, a proxy-token decentralized exchange must still have a storage center to hold the external cryptocurrencies represented by the proxy tokens. At best, this storage center is only distributed, and therefore end-users are under the same counterparty risk that exists in centralized exchanges. Furthermore, the process of trading on proxy-token platforms requires using a set of gateways (i.e. “The MultiGateway”) to convert external native coins (such as Bitcoin) to and from the affiliated proxy tokens. Together, these many problems make the proxy-token method of decentralized trading an impractical solution.

Therefore, a decentralized exchange alternative that seeks to successfully remove the threats and limitations of centralized exchanges must feature the same speed, liquidity, and convenience of a centralized exchange. As of today, no decentralized exchange has successfully replaced any of their centralized counterparts.

BarterDEX: A Complete Solution

We now present a fully functional, new decentralized technology that makes a competitive decentralized exchange possible. We call our technology BarterDEX, and it allows people to freely and safely exchange cryptocurrency coins from one person to another.

The BarterDEX decentralized exchange creates a competitive method for bartering cryptocurrencies, combining three key components: order matching, trade clearing, and liquidity provision. These components are combined into a single integrated system that allows users to make a request to trade their coins, find a suitable trading partner, and complete the trade using an atomic cross-chain protocol. Additionally, BarterDEX provides a layer of privacy during the order-matching process, enabling two nodes to perform a peer-to-peer atomic swap without any direct IP contact.

The “order matching” component is the process of pairing an end-user’s offer to buy with another end-user’s offer to sell. This component is not the actual trade itself, but is only a digitally created promise between end-users stating that they will perform their parts of the trade.

The order-matching process is achieved by algorithms that define how the orders are paired, and in which order they are fulfilled.

After a successful order-matching execution, the next component is the “clearing” aspect of the trade, wherein end-users must fulfill their promises. This is the process wherein the assets are swapped between the trading parties. BarterDEX facilitates this process and assures the safety of the users therein.

Recall that in previous decentralized exchanges there lies a problem when an exchange has low liquidity. BarterDEX solves this problem by creating Liquidity Provider Nodes (LP nodes). LP’s are trading parties that act as market-makers, buying and selling assets. They provide liquidity to the exchange, and make their profit from the spread between bid and ask orders. LP’s bring price stability to the market, and facilitate end-users in making fast and efficient trades.

Improvements in Our Current BarterDEX Iteration

BarterDEX is the result of years of development and iterated versions, with each iteration adding the next layer of required functionality to achieve our eventual goal of large-scale adoption.

With this incarnation, BarterDEX adds support for SPV Electrum-based coins, as well as dozens of normal bitcoin-protocol coins running native-coin daemons. Internally, the “SPV” aspect of a coin is abstracted so that most of the API calls work transparently for these SPV-mode coins and native-coin daemons.

BarterDEX also enables a feature known as Liquidity Multiplication, a protocol that allows the same funds to be used in multiple requests on BarterDEX “orderbooks.” The first request to fill completes the trade, and all outstanding requests are immediately cancelled.

Liquidity Multiplication therefore allows an initial amount of funding to create an exponentially higher amount of liquidity on the exchange. This also provides a special advantage for traders that like to wait for below-market dumps. While this feature is something that any other exchange could implement, very few do; on BarterDEX, all orderbook entries are 100% backed by real funds, as opposed to a centralized exchange’s vouchers, which are not as reliable and therefore would present yet another danger for their end-users.

BarterDEX Technology

Before we get into details regarding the nature of atomic swaps, there are several aspects of BarterDEX that are critical to understand.

Order Matching

The first is the decentralized orderbook. The orderbook is the collection of bids and offers that end-users place on the network. To create our orderbook, BarterDEX creates a custom peer-to-peer network that employs two separate types of nodes: a full-relay node and a non-relay node.

Order Matching with Full-Relay and Non-Relay Nodes

The difference between a full-relay node and a non-relay node is that the former is typically a high-volume trader who provides liquidity to the network in exchange for being a trading hub on the network. This puts him in the position of being able to complete trades more quickly than his trading competitors. The latter type of node (non-relay) is the more common user, who engages with BarterDEX when trading one cryptocurrency for another, given the user’s daily motivations.

There are no requirements or payments necessary to become either type of node, and so anyone desiring to become a high-volume full-relay node will find no restrictions. To be successful as a full-relay node, however, one must be able to carry out transactions on the network with a competitive Internet connection and high-capacity bandwidth.

There are several incentives encouraging users to become full-relay nodes, as these types of nodes are necessary to build the backbone of the BarterDEX network. One incentive to run a full-relay node is that by being at the center of a wide network of non-relay nodes, the full-relay node has better connectivity and thus a higher chance of being the first to complete a trade.

A non-relay node has all the same available trading options. Non-relay nodes are only limited, naturally, in terms of the total number of connections they maintain to other users. We expect that most nodes joining the network will be non-relay nodes.

In theory, roughly 100 full-relay nodes should be able to support thousands (if not tens of thousands) of non-relay nodes, thus providing a large and high-volume network. We are in the process of achieving real-world implementation. As of the writing of this white paper, the public Komodo community has performed 21,000 atomic-swap trades on BarterDEX.

When limitations do arise in the scaling process, we have various contingencies in place, one of which is the creation of clusters. It is possible to create clusters of BarterDEX nodes that are separate from other clusters on the network. To achieve this, when one cluster approaches a level of user load that is overcapacity, users can opt to seed a new cluster by creating an independent set of seed nodes. This feature amplifies the scalability of the BarterDEX network, as it allows clusters of users to form in accordance with user desires. We assume that at large scales there will be sufficient inventory in the orderbooks for clusters to provide ample asset liquidity, especially if the act of partitioning into a new cluster is based on trading a coin that is overcrowded.

Furthermore, as we continue to develop this new technology, we may also create a protocol that will allow these separate clusters to share their order boards via bridge nodes, which in theory can act to cross-pollinate desired orders from one cluster to another.

To optimize the network load, we minimize the hierarchical transmission of the orderbooks and the fetching of data. There are also several different methods of obtaining data by which we can maximize the number of nodes that can fully connect to the BarterDEX network.

Jumblr Technology Adds Privacy

While BarterDEX does not require non-relaying nodes to publicly share their IP addresses, it is important to note that BarterDEX itself is not private. Instead, we use Jumblr, an accompanying Komodo technology, to provide privacy options.

Users should assume that if privacy is important for their given trading activity, they need to employ Komodo's additional privacy technology, Jumblr. On the surface, non-relaying nodes perform addressing via a <curve25519> pubkey, and the IP address of one non-relaying node is normally not directly shared with their accompanying non-relaying trading partner. However, full-relay nodes are capable of monitoring IP addresses at the lower levels of the network, and therefore a malicious actor would be able to link IP addresses of non-relay nodes to pubkeys, thus uncovering the most crucial aspects of their privacy.

Iguana Core Provides the Foundation for Our "Smart Address" Feature

BarterDEX itself is a fork of one our earliest codebase experiments, Iguana Core, which we briefly encounter in each part of this paper. All BarterDEX transactions that use the atomic-swap protocol are created and signed in a format that is managed by the Iguana Core codebase. This enables a powerful combination of features.

The following page is a high-level discussion of one method that Iguana Core supports the fluidity of the Komodo ecosystem. Newcomers to the cryptocurrency industry and those who are not familiar with developer language may find this section too challenging to understand. We encourage the reader to simply read the two warning below, and then to skip to the next section, if this material proves too challenging.

First Warning: Some of the features that Iguana Core enables are highly advanced, and therefore users interacting with BarterDEX and other Iguana-compatible GUI software applications should always perform proper research and exercise caution.

Second Warning: The important thing for users to understand is that they should be careful not to spend the same funding in two different standalone apps. In other words, if they are trading with funds in a BarterDEX GUI, they should not also try to spend those funds in their Agama Wallet (or another Iguana-compatible wallet). Instead, they should wait for both apps to be in sync before moving forward.

One specific feature is a specialty wallet that can manage and trade among a multiplicity of different blockchain coins. To explain the significance of this multi-coin wallet feature, let us observe how a standalone GUI app formerly interacted with cryptocurrencies.

Previously, for a GUI software application to manage cryptocurrencies, the software application usually required the creation of a wallet.dat file, which is locally stored on the user's computer. This wallet.dat file held the privkeys—passwords that unlock funds on a blockchain—and other encryption-enabled protocols necessary for the user to manage funds. There are many limitations in the wallet.dat method. For instance, typically only one software application should access the wallet.dat file at a time, to prevent data conflict and corruption.

The Iguana Core codebase enables the user to interact with their funds on the blockchain(s) without requiring a wallet.dat file. Because the Iguana Core codebase works with raw transaction data, the codebase allows a user to first create and then manage a public blockchain "smart address" that can be accessed from anywhere, by any compatible standalone GUI, simply with a passphrase that unlocks their privkey.

To maintain control over their funds without requiring a wallet.dat file, users need only create a smart address and then retain a copy of the accompanying passphrase (typically a collection of 12 to 24 common dictionary words arranged in a specific order) that is provided at the moment of creation. By entering this passphrase into an Iguana Core compatible standalone GUI app, Iguana Core then activates their <privkey>, which then enables users to manage their funds.

Furthermore, the smart address created by Iguana Core can manage and maintain multiple types of coins and other blockchain assets. When a user sends any compatible coin to the smart address, Iguana Core stores those coins in a separate address that is compatible with the appropriate blockchain, and links this sub-address to the smart address of the user.

Therefore, in the underlying Iguana code, each of the unique coins gets an address that is compatible with its own blockchain, but the smart address enables the user to access these coins all at once. Funds deposited to this smart address are automatically eligible for trading, and therefore a BarterDEX GUI app can work with speed to enable users to trade between a multiplicity of coins.

One key function of the Iguana codebase that makes this possible is the <withdraw> command in the Iguana API. It is this command that allows individual GUI apps, such as a standalone BarterDEX GUI app, to work with the underlying funds in the user's addresses.

Notice several of the freedoms this provides to the user. All the funds are only spendable by the user with the passphrase, and because there is no need for a wallet.dat file to be stored locally, there is less danger (though users should exercise caution) of data corruption between different standalone software applications all accessing these funds.

Therefore, an end-user can have a standalone BarterDEX GUI app running on their local machine, which they use to trade, and can also have a separate standalone GUI wallet app that is managing their long-term cryptocurrency holdings.

This also allows standalone GUI applications that are Iguana Core compatible to support each other. For instance, while a BarterDEX GUI can function without any native-coin daemon process running in the background simply by relying on Iguana Core and public Electrum SPV servers (which remove the need to download blockchain data), the BarterDEX GUI can also work with a native wallet's coin daemon background process to coordinate blockchain synchronization.

For instance, a Komodo user may run the Komodo Agama wallet, which runs a native Komodo coin daemon (and has a local wallet.dat file), alongside a BarterDEX GUI app. Iguana Core can then enable the BarterDEX GUI to rely on the native coin daemon running in the background of the Komodo Agama wallet, which speeds up the trading process for an end-user, as they do not have to wait for the public Electrum servers to update. The native Komodo coin daemon is the software we encountered in Part II, Komodod.

The UTXO: An Elusive, Yet Fundamental Concept

BarterDEX relies heavily on a rarely understood technology called the "UTXO," short for Unspent Transaction, which was invented in the original Bitcoin protocol. This technology is fundamental to the operations of any blockchain project that utilizes the original Bitcoin protocol. However, even the most active of cryptocurrency users rarely know what UTXOs are or why they exist.

Because UTXOs play an important part in BarterDEX, and to provide a pleasant user experience, it is essential we adequately explain the UTXO concept. In the future, as the technology surrounding BarterDEX iterates, and as the cryptocurrency community continues to learn, we hope that the concept of UTXOs will be less taxing on a user's learning curve.

To begin our explanation of UTXOs, let us first examine the language of a common user when describing how much cryptocurrency money they have and how they perceive that money. We will therefore need to understand the concept of "satoshis," the way a blockchain handles the collection and distribution of funds, and how we utilize these core technologies when trading on BarterDEX.

Comparing the UTXO to Fiat Money

Let us assume a cryptocurrency user, whom we name Michael, has $10,000 in his physical wallet. Naturally, when Michael thinks about the amount of physical (or "fiat") money he has, he says to himself, "I have $10,000."

However, there is no such thing as a $10,000-dollar bill. Instead, Michael actually has a collection of smaller bills stacked together. For instance, he could have a stack of $100-dollar bills, the total of which equals $10,000 dollars.

If Michael goes to purchase an item that costs $1, and he only has $100-dollar bills in his wallet, to make his purchase he will take out a single $100-dollar bill and give it to the cashier. The cashier then breaks that $100-dollar bill down into a series of smaller bills. The cost for the item, $1, remains with the cashier, and the cashier then provides change—perhaps in the form of one $50-dollar bill, two $20-dollar bills, one $5-dollar bill, and four $1-dollar bills.

Michael now thinks to himself, “I have $9,999.” Specifically, however, he has ninety-nine $100-dollar bills, a $50-dollar bill, two $20-dollar bills, one $5-dollar bill, and four $1-dollar bills.

We emphasize that not only does he not have ten thousand $1-dollar bills, he also does not have one million pennies ($0.01). Furthermore, because pennies are the smallest divisible unit of value in Michael’s wallet, we could point out that each bill is a collection of its respective units of pennies. For instance, a $1-dollar bill in Michael’s wallet we could describe as, “a bill that represents a collection of one hundred pennies and their value.”

Understanding Cryptocurrencies and Their UTXOs

A Satoshi is The Smallest Divisible Unit of a Cryptocurrency

Continuing with our explanation of UTXOs, we next need to understand the concept of “satoshis.” The name “satoshi” is derived in honor of Satoshi Nakamoto, author of the original Bitcoin white paper. By convention in the cryptocurrency community, one satoshi is equal to one unit of a coin at the smallest divisible level. For instance, 1 satoshi of Bitcoin is equal to 0.00000001 BTC.

Let us suppose now that Michael has 9.99000999 BTC (Bitcoin) in his digital wallet. Assuming Michael correctly understands the concept of satoshis, Michael could say to himself, “I have nine hundred and ninety-nine million, nine hundred and ninety-nine satoshis of bitcoin.”

This is how Michael might mentally perceive the collection of money that exists in his digital wallet, like he perceives the $9,999 in his fiat wallet.

A UTXO is a Packet of Satoshis, just as a Fiat Dollar Bill is a Packet of Pennies

Recall now that with fiat money, Michael did not think about how his original $10,000 was comprised of smaller individual $100-dollar bills. Similarly, Michael also does not think about how his 9.99000999 BTC could be comprised of smaller collections of satoshis.

Furthermore, just as Michael did not carry around fiat money as a collection of pennies, he also is not carrying around a raft of satoshis. Were he to try to carry a million pennies in his physical wallet, the weight of the wallet would be unmanageable. Similarly, if the Bitcoin protocol were to attempt to manage nine hundred and ninety-nine million, nine-hundred and ninety-nine satoshis, the “data weight” would be so heavy, the Bitcoin protocol would be enormous and unmanageable.

To optimize “data weight,” the Bitcoin protocol therefore bundles up the satoshis into something that is like the example of dollar bills earlier, but with one important difference. In fact, here is where the Bitcoin protocol exercises a superiority over fiat money by deviating from the limitations fiat money must obey when bundling smaller values into larger values.

In fiat money, one hundred pennies are bundled into a one-dollar bill, which can then be bundled into a larger bill, and so on. All the sizes of fiat money are preset and predetermined by the issuer of the fiat money when they print their bills and coins.

The Bitcoin protocol, however, does not need to pre-plan the sizes of “bills” (i.e. the collections of satoshis) in the owner’s wallet. Bitcoin is freer in this sense; it can shift and change the sizes of its “bills” at will because there is no need to accommodate for the printing of physical coins and paper.

Instead, the Bitcoin protocol allows for the developer of digital wallets to write code that can optimize how bitcoin satoshis are packaged into “bills,” and thus the community of developers can work together to keep the data weight of the blockchain manageable. The better the digital-wallet developer, the more efficient the size of the “bills” (a.k.a. the packets of satoshis).

The Bitcoin protocol does have one limitation, however: It must keep track of how these satoshis are being collected into larger “bills” in everyone’s digital wallets. After all, the very idea of Bitcoin stands in the idea that everything happens under the public eye, where it can be verified. Because the Bitcoin blockchain must keep track of the sizes of these packets of satoshis, the only time the packets can be assembled or disassembled into larger and smaller sizes is at the moment when the user is spending money on the public blockchain. It is at this time that the user is under the public eye, and therefore his actions can be verified.

To compare this limitation to fiat money, consider the effect created were Michael to cut a $100-dollar bill into smaller pieces. The $100-dollar bill would no longer be respected as a valid form of currency.

Because Bitcoin is so flexible, and because the word “bill” does not well describe this flexibleness, we do not use the word “bill” in cryptocurrencies. In fact, we have yet to come up with a sonorous word for a packet of satoshis. For the time being, we simply use the word that the original bitcoin developers provided, UTXO, which stands for “Unspent Transaction.”

The packet can be any size, and the developer of the spender’s digital-wallet software decides on this process. Most importantly, and to reiterate, a UTXO can only be resized during the process of spending, as this is the moment when the user interacts with the public blockchain.

To further clarify this, let us return to Michael’s example with fiat money. Recall that when Michael went to purchase a $1-dollar item, he only had $100-dollar bills in his wallet. He had to give out one $100-dollar bill, and then receive a broken-down collection of dollar bills in return.

This is exactly how it works with UTXOs. Michael has a collection of UTXOs in his digital wallet, and when he goes to buy something, he will give out UTXOs until he surpasses how much he owes, and then the extra change from the last UTXO used will be broken down and returned to him.

For example, let us suppose that Michael’s 9.99000999 BTC is comprised of three UTXOs worth the following values:

UTXOs in Michael’s Wallet

UTXO #1:	0.50000000 BTC
UTXO #2:	0.49000999 BTC
UTXO #3:	9.00000000 BTC
Total:	9.99000999 BTC

Michael now desires to purchase an item that costs 0.60000000 BTC. He will have to hand out enough UTXOs from his wallet until he covers the costs of this transaction, just as he would if he were using fiat money. The Bitcoin protocol calculates the change from the transaction and then returns his change to him.

Remember that there is a fee when spending money on a blockchain. Since we are using Bitcoin in this example, the fee would be paid to cryptocurrency miners. Let us imagine that the fee the miners charge Michael is 999 satoshis.

We begin by looking at how Michael would see the process of making the purchase, assuming he does not understand the concept of UTXOs. For now, Michael only understands how much is in his wallet at the satoshi level as he conducts his transaction:

9.99000999 BTC - The amount Michael initially owns

- 0.60000000 BTC - The amount Michael sends to the digital cashier for his purchase

<u>- 0.00000999 BTC</u> - The network fee paid to miners

9.39000000 BTC - The amount left in his wallet

This deduction for his purchase all appears very simple to Michael—a testament to the Bitcoin protocol's effective design.

In the background, however, the digital wallet handles the UTXOs and the change process in a manner as determined by the programmer. In Michael's example, let us assume that it proceeds this way:

The wallet first brings out UTXO #1, which is worth 0.50000000 BTC:

0.60000999 BTC - The total amount that Michael owes to the cashier and network

<u>- 0.50000000 BTC</u> - The wallet sends the full value of **UTXO #1** to the digital cashier

0.10000999 BTC - This is the remaining total amount that Michael still owes

The wallet now brings out UTXO #2, which is worth 0.49000999 BTC:

This UTXO is broken down or shattered into smaller pieces.

0.49000999 BTC - The size of Michael's **UTXO #2,** now in the process of change

- 0.10000000 BTC - This shatter of UTXO #2 goes to the cashier (payment fulfilled)

<u>- 0.00000999 BTC</u> - This shatter of UTXO #2 pays the network fee to the miners

0.39000000 BTC - This last shatter now returns to Michael's wallet as a new UTXO

Michael now has one new UTXO in his wallet, and it is worth 0.39000000 BTC:

Michael's New Wallet State:

UTXO #3:	9.00000000 BTC
UTXO #4:	0.39000000 BTC
Total:	9.39000000 BTC

If Michael wants to buy something later, these UTXOs will have to be broken up once more, according to the costs and programming of the digital wallet. Again, whatever is left over from his last UTXO comes back to his own wallet as a new UTXO.

Now let us suppose that Michael receives 0.4 BTC from someone else. In Michael's wallet, he will see a total of 9.79 BTC. However, in his wallet there are now actually three UTXOs:

Michael's New Wallet State:

UTXO #3:	9.00000000 BTC
UTXO #4:	0.39000000 BTC
UTXO #5:	0.40000000 BTC
Total:	9.79000000 BTC

As a result, the number and sizes of UTXOs in Michael's wallet will vary over time. He may have many smaller UTXOs that make up his full balance, or sometimes he might just have one large UTXO that comprises all of it. For Michael, it is normally possible to ignore this since the wallet developer could handle everything automatically.

However, understanding the nature of BarterDEX currently encourages users to understand UTXOs, as the process relies on their UTXO inventory during trading, as explained below.

Trading on BarterDEX

From our point of view as developers, the most difficult aspect of creating BarterDEX was in matching the inventory of UTXOs between trading partners.

To illustrate this complexity, let us briefly return to the example of Michael and fiat currency. If Michael had only a $50-dollar bill in his wallet, and wanted to spend $35 dollars at a video arcade. He needs to trade $35 for the equivalent number of video-game tokens. However, he can only work with the bill that is in his wallet to trade for the tokens.

In a typical arcade, this process is simple. There are just two currencies—his dollars and the video game tokens—and he will have a human cashier available to manage the trade. He gives the $50-dollar bill to the human cashier, and the cashier returns $15 dollars in dollar bills, and $35 dollars' worth of video-game tokens.

In creating BarterDEX, however, our goal is to decentralize all points of control. (The "cashier," in this sense, is a centralized authority who could be corrupted or could commit human error). That means that we cannot have a human cashier present in BarterDEX to trade Michael's three UTXOs into their appropriate sizes when he wants to swap for other currencies.

A further challenge lies in the number of currencies. For BarterDEX there are not just two coins, but many coins, with myriad users, each having a variety of unique UTXO sizes in their wallets. In addition, the trading happens in real-time, through automation, on a decentralized peer-to-peer network, supporting a countless number of separate blockchain projects, while providing a speed and (eventually) liquidity comparable to that of a centralized exchange. All of this must be accomplished while maintaining a level of security and safety that only decentralization can provide.

Finally, imagine if there were no cashier to break down Michael's $50-dollar bill. What if instead, he had to approach other arcade customers to barter for their tokens? This would create a difficult scenario for Michael.

In its current iteration (continuing the use of the $50-dollar metaphor as applied to UTXOs), we limit BarterDEX's capability to only perform a trade for Michael's $50-dollar bill in exchange for the currency that another customer holds. BarterDEX does not provide a service whereby Michael can break down his $50-dollar bill into a convenient set of $10-dollar and $5-dollar bills for trading. He must give up his full $50-dollar bill for whatever he wants in return.

The process of breaking down UTXO inventory, therefore, is both in the hands of the user and in those of the developers creating the standalone GUI apps. We are working with our community to simplify this process. Naturally, it is complex and will take time. Therefore, we recommend that users who engage with BarterDEX have a basic understanding of their UTXO inventory and how they are bartering with other users before using it.

How BarterDEX Deals with Order Offers and UTXOs

When a BarterDEX user offers a trade to the network, the BarterDEX protocol itself does not prioritize the total number of satoshis that the user offers. Instead, BarterDEX simply looks through the user's inventory for the largest-sized UTXO that is below the amount the user offered.

For example, let us suppose that Michael has 100.01287001 KMD (Komodo coin) in his wallet. It is comprised of three UTXOs:

Michael's Initial Wallet State:

UTXO #1:	90.00000000 KMD

UTXO #2:	00.01287001 KMD
UTXO #3:	10.00000000 KMD
Total:	100.01287001 KMD

Michael wants to trade 50.00 KMD on the BarterDEX network. He puts out an order for an alternate cryptocurrency called MNZ (Monaize), and he wants to exchange in a 1:1 ratio.

BarterDEX itself will not attempt to manage for Michael's misunderstanding of his UTXO inventory. (The developer of Michael's standalone software could try to help him, but that is a separate matter.) Rather, BarterDEX will simply look through his inventory for the largest UTXO that is below the total amount he offered. In this example, BarterDEX will select his UTXO #3, worth 10 KMD. BarterDEX will then calculate the necessary fee, which so happens to be exactly equal to the amount of UTXO #2: 0.01287001 KMD.

BarterDEX can then take these two UTXOs and facilitate a trade for MNZ in a 1:1 price ratio. Michael's final wallet appears as so:

Michael's Final Wallet State:

UTXO #1:	90.00000000 KMD
UTXO #3:	10.00000000 MNZ
Total:	90.00000000 KMD
	10.00000000 MNZ

It is up to Michael, or to the creators of any standalone GUI wallet, to manage the UTXOs. BarterDEX only manages the matching of the UTXOs once they are created.

Detailed Explanations of the BarterDEX Process

With an understanding of the specifics of what BarterDEX is actually trading, we can now approach an explanation of how the trading procedure occurs.

Atomic Swaps on The Komodo BarterDEX

To facilitate trading among users, BarterDEX implements a variation of the atomic-swap protocol, as described by Tier Nolan on BitcoinTalk.org. The original concept provided by Tier Nolan can be said to be "ahead of its time," as it is both complex and relies conceptually on technology that yet does not exist. Therefore, to create our variation of the atomic-swap protocol, we adapted for the current technology. A thorough study of Nolan's original exposition can provide a solid background into the tradeoffs that we made as we selected our final version of our atomic-swap protocol.

We emphasize to the reader that the key aspect that we maintained from the original concept is that at each step there are both incentives to proceed to the next step in the proper manner, and disincentives to avoid abandoning the procedure. With this structure in place, regardless of where the protocol stops, each party receives their proper reward. If a party attempts to deviate from the proper path, their funding is penalized to the point of eliminating any potential rewards a user could gain by acting maliciously. These incentives and disincentives create the foundation for the requisite trustless nature of our atomic-swap protocol.

Meet Alice and Bob

To understand why the atomic-swap protocol is necessary, it is first important to recall that computer code is executed in linear fashion. Even if we were to assume that both parties in a trade may be honest, on a computer the process of taking money from each digital wallet and pulling the money into the open must happen one wallet at a time. Therefore, one person must send out their money first. The atomic-swap protocol protects that person from vulnerability. Without the atomic swap, any malicious party involved (whether it be a full-relay node, trading partner, or other external agent) would be able to destroy the fairness of the trade.

There are two parties in an atomic swap: the liquidity provider and the liquidity receiver. Once the process of an atomic swap begins, the behavior of each party's public trading profile is recorded and added to their reputation on the BarterDEX network.

The process of an atomic swap begins with the person who makes the initial request—this is the liquidity receiver. By convention, we call this person, "Alice." Alice will need two UTXOs to perform her swap. One UTXO will cover the protocol fee, which is roughly 1/777th the size of her desired order. We call this fee the <dexfee>, and its primary purpose is to serve as a disincentive to Alice from spamming the network with rapid requests.

The second UTXO required of Alice is the actual amount she intends to swap. BarterDEX first verifies that she has these funds, but for the moment she retains these funds in the safety of her own digital wallet.

On the other side of the atomic swap, we have the liquidity provider—we call this person, "Bob." Bob sees the request on the network for Alice's atomic swap and decides to accept the trade. Now his part of the process begins.

To complete the trade, he must also have two UTXOs, but with one important difference: the first UTXO is equal to 112.5% of the amount that Alice requested; the second UTXO is exactly equal to the amount that Alice intends to swap. In other words, Bob must provide liquidity of 212.5% of the total amount of the currency that Alice requests.

The first UTXO (112.5%) Bob now sends out as a security deposit, placed on the BarterDEX network. The network's encryption holds the deposit safely in view, but untouchable. We call this UTXO, <bobdeposit>. It will remain there until his side of the bargain completes in full, or until Alice's request for a swap times out. Assuming Bob keeps his promises and stays alert, these funds will be automatically returned to him at the appropriate moment.

The second UTXO (100%) he retains within the safety of his own wallet for the moment.

Performing a successful connection between Bob and Alice, and verifying their requisite UTXOs, is the most complex and difficult aspect of creating the BarterDEX network. Myriad factors are involved in a successful attempt for Bob and Alice to connect: human motivation; the experience level of the users; economics; connection technology; user hardware setups; normal variations within Internet connections; etc.

We emphasize to users here that the process of performing these actions over a peer-to-peer network has almost an artistic element to it. An attempt to successfully connect Bob and Alice can be thought of more like fishing, where we must simply cast and recast our line until we successfully connect with our target. If a user attempts a trade and no response returns from the network, the user should slightly adjust the parameters of their offer and try again. As BarterDEX continues to iterate and improve, and as the number of users increases, we expect any required effort to lessen for users, the network, and the BarterDEX GUI apps.

Alice and Bob Make a Deal

Assuming Alice and Bob have now successfully connected, the process from this point forward becomes quite simple:

(Note: in some cases, it is possible to perform an atomic swap with fewer steps, but for the sake of brevity we will focus only on this scenario.)

A summary of the procedure:

1. Alice requests a swap and sends the <dexfee> to the BarterDEX full-relay nodes
    a. The full-relay nodes receive her request and publish it to the network
2. Bob sees the request on the network, accepts it, and sends out <bobdeposit>
    a. <bobdeposit> enters a state of limbo on the BarterDEX network, held safely by encryption, awaiting either Alice to proceed, or for the swap to time out
        i. If the latter occurs, <bobdeposit> is automatically refunded to Bob via the BarterDEX protocol
3. Alice now sends her <alicepayment> to Bob
    a. She does not send the payment to Bob directly, but rather into a temporary holding wallet on the BarterDEX exchange, which is encrypted and protected by his private keys
        i. Only Bob has access to this wallet, via the set of privkeys that only he owns
        ii. However, the BarterDEX code does not yet allow Bob to unlock this temporary holding wallet; he must continue his end of the bargain first
        iii. The <alicepayment> will remain in Bob's temporary holding wallet for a limited amount of time, giving him the opportunity to proceed

4. Bob now sends his <bobpayment> to Alice
    a. Again, this is not sent to Alice directly, but rather into yet another temporary holding wallet
    b. Likewise, only Alice has access to the necessary privkeys for this wallet
    c. The <bobpayment> will automatically be refunded if she does not complete her part of the process
5. Alice now "spends" the <bobpayment>
    a. By the word "spends," we simply mean that she activates her privkeys and moves all the funds to another wallet—most likely to her smart address
    b. BarterDEX registers that Alice's temporary holding wallet successfully "spent" the funds
6. Bob "spends" the <alicepayment>
    a. Likewise, Bob simply moves the entirety of the <alicepayment> into a wallet of his own—again, it will most typically be his own smart address
    b. BarterDEX now knows that Bob also successfully received his money
7. Seeing both temporary holding wallets now empty, the BarterDEX protocol recognizes that the atomic swap was a complete success
    a. BarterDEX now refunds <bobdeposit> back to Bob and the process is complete.

While it may seem inefficient to have seven transactions for a swap that could be done with two, the complexity of this process provides us with the requisite "trustless-ness" to maintain user safety.

Incentives and Disincentives to Maintain Good Behavior

As we will now explain, at every step along the way there are incentives for each side to proceed, and there are various financial protections in place should one side fail. Also, because payments are sent to these "temporary holding wallets" that exist within the BarterDEX protocol, the protocol itself can assist in the process of moving money at the appropriate steps. Let us now examine what is happening at each step.

1 - Alice Sends <dexfee>

If Bob accepts the offer to trade, but does not send <bobdeposit>, Alice only stands to lose her <dexfee> UTXO. This is only 1/777th of the entire transaction amount, so she loses very little. Bob, on the other hand, stands to lose more. Since Bob did not follow through with his end of the bargain, the BarterDEX network indicates on his public BarterDEX trading profile that he failed in a commitment, thus decreasing his profile's reputation. If Bob continues this behavior as a habit, he may find it difficult to discover trading partners.

So long as the frequency of “Bobs” failing is low, the occasional extra <dexfee> paid by an Alice is a minor issue. However, if there is a sudden spike in misbehavior, the BarterDEX code has in-built contingency plans which can provide refunds to Alice(s), should a particular Alice node(s) experience a large loss via <dexfee>’s

2 - Bob Successfully Sends <bobdeposit>

If Alice does not follow with her next step, the <alicepayment>, then Alice loses not only the <dexfee>, but she also receives a mark on her public BarterDEX profile. She gains nothing, and Bob has no reason to fear as <bobdeposit> will automatically return to him via the BarterDEX protocol.

3 - Alice Successfully Sends <alicepayment>

If Bob does not proceed with his next step, the <bobpayment>, then after 4 hours Alice can simply activate a BarterDEX protocol that will allow her to claim <bobdeposit>. Recall that <bobdeposit> is 112.5% of the original intended trade; Bob has every incentive therefore to continue with his end of the bargain, and Alice has nothing to fear should Bob fail. She even stands to gain a 12.5% bonus, at Bob’s expense.

4 - Bob Sends <bobpayment>

Now, if Alice does not follow by “spending” the <bobpayment> (i.e. taking the money out of the temporary holding wallet and into her own smart address), then after 2 hours Bob can activate a BarterDEX protocol that allows him to reclaim his <bobpayment> immediately. Furthermore, four hours later Bob may activate a refund of <bobdeposit>; Bob is safe from Alice, should she fail. For Alice, the BarterDEX protocol allows Alice to reclaim her <alicepayment> after Bob reclaims both of his payments.

Everything herein is recorded to the respective users’ BarterDEX trading profiles, ensuring their reputations are on the line. Recall also that the BarterDEX protocol requires each step to be performed in the proper order, thus ensuring that neither party can take any funds before the users’ appropriate moment.

Thus, at this integral stage of the process, every step of the path is intricately interconnected and maintains various levels of protection.

5 - Alice Spends <bobpayment>

At this point, Alice is entirely through with any risk to her reputation, her <dexfee> payment, or of the loss of her time.

If Bob does not follow by also “spending” the <alicepayment>, it is of no concern to Alice because she has already received her funds. If Bob is simply sleeping and forgets to spend the <alicepayment>, he can only hurt himself.

Naturally, for Bob this is slightly dangerous. Bob’s best course of action is to remain alert and spend the <alicepayment> once it is received.

If after four hours, Bob is still sleeping, Alice can still activate the protocol that allows her to claim <bobdeposit>. In this scenario, she receives both the <bobpayment> and <bobdeposit>, at only the costs of the <alicepayment> and <dexfee>.

Bob can still make a later claim for the <alicepayment> when he regains his awareness.

6 - Bob Spends <alicepayment>

Assuming all has gone according to plan, and having spent the <alicepayment>, Bob may now reclaim <bobdeposit>. Just as before, if Bob does not refund his own deposit, it is his loss; in four hours Alice will be able to activate a claim on <bobdeposit>.

7 - Bob Reclaims <bobdeposit>

The process is complete. Alice received the <bobpayment>: Bob received the <alicepayment>: Bob has <bobdeposit> back in his own possession. The entire process only cost Alice the original <dexfee>. At each step along the way, the side that needs to take the next step is motivated to do so, with greater and greater urgency until the process is complete.

Additional BarterDEX Atomic Swap Details

The BarterDEX implements the above series of commands in a cross-platform manner, enabling users to atomic-swap trade with hundreds of coins of many types, including both native coin daemon's and those that run on SPV Electrum servers. A swap that is not completed immediately can carry on as long as the time has not expired within the BarterDEX protocol.

Naturally, users must understand that outside forces can disable the process and thereby damage one of the users. For instance, an Internet outage for Bob could be particularly dangerous. Therefore, users are advised only to trade manageable sums that they are willing to put at risk, and only with nodes that have reliable reputations.

This atomic-swap protocol, with all its cryptographic validations and intricate key exchanges, is less than half of the difficulty Komodo experienced in creating BarterDEX. Relatively speaking, it is "easy" to do an atomic swap in isolation between two test nodes, using UTXOs that are carefully prepared for the test.

It is an entirely different matter to open this up to the public at large, including the enabling of our orderbooks and order-matching features. Due to the peer-to-peer nature of The BarterDEX, on a live network it is impossible to guarantee that a user that indicates they would like to begin a swap will receive a successful reply.

For instance, a Bob may see a potential swap that he would like to make, but by the time his attempt to accept the swap crosses the expanse of the Internet, someone else could have already accepted the swap, thus leaving Bob in his original position. There are legion scenarios wherein the initial connection can fail. Once the connection is made, however, the rest of the process maintains reliability and user safety.

Failed attempts at establishing a connection only result in the loss of a few seconds of the user’s time, and there is no cost associated with the failure. The <dexfee> paid by an Alice never occurs, and BarterDEX disregards Bob’s attempt to send <bobdeposit>.

Therefore, while we cannot guarantee that BarterDEX will always form a valid connection for each attempt at a trade, we can offer comfort in knowing that the users’ losses in these scenarios are insubstantial.

A More Detailed Explanation of the Atomic-Swap Connection Process

The following is a brief explanation of the complex process by which BarterDEX establishes a connection between Alice and Bob.

For BarterDEX to accept a request to begin an atomic swap, the code first needs to register and create all the necessary backend elements for the <dexfee>, <AlicePayment>, <BobDeposit>, and <BobPayment>. All four must be specified before BarterDEX can indicate to Alice and Bob that it can successfully support this atomic swap.

This is more complicated than it appears. As we explained earlier, most users do not understand the true nature of how funds operate in a cryptocurrency. Rather, most simply view their balance as a single conglomerate of coins that they can spend at the “satoshi level.” This misperception is important to correct to understand how BarterDEX performs an atomic swap.

Naturally, because users have varying sizes of UTXOs in their wallets, the true challenge in creating BarterDEX was to create a method of maintaining a network that would coordinate each user’s list of UTXOs in their wallets, and to allow them to match with other users in trading pairs. In addition, BarterDEX also automatically calculates the appropriate mining and transaction fees for the blockchains involved, according to a speed that maintains an optimized atomic-swap process.

As we created the necessary code to make the atomic swap possible for the public, we found that it is not practical to have the user specify which UTXO pair they have sitting in their wallet when choosing to make a swap. This would also not be intuitive for the user. Furthermore, we did not even want to code a way for an Alice to know the UTXOs a Bob has available at the moment of negotiating a trade.

Instead, here is how BarterDEX deals with the complexity of matching these unbroken and mismatching UTXOs to process an atomic swap. It is important to note that users are not required to have a sophisticated understanding of the backend UTXO process, and may simply trade using either a minimal understanding of UTXO inventories, or at least rely on the support of a cleverly coded standalone BarterDEX GUI app.

Assuming Alice has already indicated she desires to perform an atomic swap, BarterDEX calculates out the proper divisions of her UTXOs, defines how they will be appropriated during the process, and sends an “Alice Request" to Bob with information regarding her pair of UTXOs (which are the <dexfee> and the <AlicePayment>). Also, BarterDEX verifies her desired price and volume.

Bob, the human user (or an artificial intelligence bot acting on his behalf), indicates that he is willing to accept the trade. The automation of the BarterDEX Bob-side protocol now takes over in the background. It validates the "Alice Request" to make sure the UTXO pair is valid, and then the Bob-side protocol scans through Bob’s UTXO inventory for the most efficient way to create both the <BobPayment> and <BobDeposit> UTXOs.

The Bob-side protocol understands that the UTXOs will not perfectly match, and it will therefore calculate the most efficient method of making any “spare change” UTXOs as needed. An additional constraint the protocol needs to consider is that the result must match the price and volume Alice wants to pay. Finally, it accounts for the requirement that <bobdeposit> be at least 12.5% bigger than the “Alice Request.” (Note that BarterDEX is directly involved with managing Bob’s UTXOs, but is not involved with managing Alice’s UTXO offers.)

Once BarterDEX verifies all these conditions, the Bob-side protocol sends back a data packet, labeled “reserved,” to the Alice-side protocol to indicate that all is in order. All of this is optimized and conducted in a manner that prevents the human Bob from having his funds frozen in an unnecessary deposit duty, should the human Alice find another “Bob” in the interim.

Next, the Alice-side protocol validates the "reserved" packet from the Bob-side protocol, making sure all the UTXOs are valid, and the protocol verifies that the price and volumes are acceptable according to the original intent.

Assuming everything successfully validates, the Alice-side protocol sends a "connect" packet back to the Bob-side protocol with the same parameters, indicating that her funds are now "reserved” as well.

Between the "request" being sent and the "reserved" packet being received there is a 10-second timeout which prevents Alice from making further trade requests. This gives BarterDEX the time necessary to perform all the calculations.

Note: This 10-second timeout also provides a contribution to what we call “whale resistance” during the Komodo dICO process. Whale resistance is a way Komodo and BarterDEX resist “whales” from purchasing an entire coin supply and thus forcing an artificial market scarcity.

The Bob-side protocol now validates Alice’s "connect" packet and, assuming everything is in order, the protocol starts a new Bob-side thread of code, thus beginning the actual atomic swap. The Alice-side protocol also receives the "connect" packet, verifies, and then starts an Alice-side thread of code.

There is one more "negotiation" step that is needed between the Alice-side and Bob-side protocols: in the event the two sides to the protocol do not achieve consensus, the entire atomic swap aborts without any payments sent from either party (i.e. “no harm, no foul”).

(This final negotiation could have been included earlier, but due to the way the atomic swap organically developed during our creation process, it ended up inside the atomic-swap protocol itself.)

The Alice-side and Bob-side protocols have now properly performed their duties, and thus completed the most challenging aspect of the atomic-swap protocol. The BarterDEX returns control to the humans (or bots acting on their behalf) to send their respective payments.

The DEX Fee: <dexfee>

People will notice that there is a small <dexfee> required as part of the BarterDEX protocol. This is 1/777 of the transaction amount and it is calibrated to make spam attacks impractical. By forcing a would-be attacker to spend real money, attacking the network becomes costly. Without this spam prevention, the BarterDEX could otherwise be attacked at the protocol level by any person performing a plethora of trade requests.

The 1/777 fee ends up being equal to 0.1287% of the <alicepayment>; this is already far less than the fees paid on an average centralized exchange. Also, centralized exchanges charge both sides of the trade, so even if they charge you 0.2%, they are actually harvesting 0.4% in total fees between both parties.

Furthermore, they often have fees and limitations for withdrawing funds, as well as a lengthy, challenging, or invasive registration process. BarterDEX has none of these things. Users need only record the passphrase they create when first entering the BarterDEX software, and they are prepared to trade.

It is possible that some atomic swaps can initiate, and then fail to complete, which raises questions about what happens to the <dexfee>. The <dexfee> is the first charge in the protocol; in this sense, there is a <dexfee> charged for these failed atomic swaps.

However, this failure should not be looked upon in isolation. The BarterDEX protocol is based on statistics. Statistically speaking, there will be some percentage of atomic swaps that start and will not complete. Let us suppose a 15% failure rate at this stage of the atomic swap (15% is three times higher than the rate of failure we currently observe in our testing). Even in this scenario, the effective <dexfee> cost is still only 0.15% to all Alice-side requests across the entire network.

Therefore, if you experience the loss of a <dexfee> transaction for an atomic swap that fails to complete (which would be due to a failure to receive a response from Bob), know that this is all part of the statistical process. If you find yourself paying more than 0.15% of your completed trades in <dexfee>'s, please let us know. This would be a highly unusual statistical outlier, and we will therefore want to find and fix the cause.

As an organization, when speaking generally to our audience online, we typically state that the <dexfee> is just 0.15%. In this manner, we hope to create the expectation that 0.15% is normal; if the network performs perfectly, on the other hand, users will get a blessing in the form of a lower fee, 0.1287%.

Dealing with Confirmations

Since BarterDEX is trading permanently on blockchains (as opposed to updating an internal database of vouchers, or managing a proxy-token account balance), both humans need to wait and watch as miners on the respective blockchains calculate transaction confirmations.

Because the payments that occur on one blockchain will proceed regardless of the actions on the other blockchain (i.e. a confirmation failure on one chain will not stop with the other blockchain performing its duties as normal), it is therefore important that the BarterDEX protocol observe and adjust as necessary. Each side of the BarterDEX protocol (Bob-side and Alice-side) watches and attempts to provide a level of protection for the human users.

BarterDEX achieves this protection by an array of <setconfirms> API calls, which gives each side the option to specify how many confirmations they expect before the automated process should be satisfied on behalf of the human users' interests. The setting for the <setconfirms> feature must be decided before the atomic swap begins, as the number of confirmations the users choose will persist until the process completes. If the users have differing preferences for the total <numconfirms> they prefer, the BarterDEX protocol automatically sets the larger of the two preferences as the requirement for both parties. Furthermore, this feature also includes a <maxconfirms> value to prevent one side from specifying an unreasonable or malicious number of required confirmations.

Zero Confirmations

The BarterDEX also supports a high-speed trading mode. Using this feature, a user can activate an extremely fast mode of trading: <zeroconf>. This initiates a form of atomic-swap trading that does not wait for any confirmations at all. When using this feature, atomic swaps can be completed in as little as 3 seconds. This is a high-risk endeavor, naturally, and users should exercise extreme caution when implementing it.

One potential application for the <zeroconf> feature is to allow groups of individuals to form their own organizations where they decide personal trust levels, and work together to correct any mistakes that are made in their accounting endeavors.

BarterDEX also features a special Trust API that users can enable for themselves and groups that they form to indicate how much they trust different traders. By default, the Trust API is set to neutral for all users. A group of users can form their own organization and develop a trusted network for trading, using the Trust API to set each other's trader profile to Trust = Positive. In such cases, if a user, or a group of users, tells the Trust API to set another trader profile to Trust = Negative, that trader's <pubkey> is blacklisted for any of the participating individuals or groups.

High-Speed Mode: An Experimental Feature Using Time-Locked Deposits

Using the <zeroconf> protocol, we developed a new feature for the BarterDEX network that is functional, but still experimental. It is called "Speed Mode," and it adds one additional step to the Alice and Bob process.

Alice places a one-time security deposit of an amount equal to or greater than the amount she would like to actively trade. This security deposit is sent to a conditional p2sh wallet address (currently controlled by the Komodo team). Alice indicates within her security-deposit transaction the amount of time the deposit should remain in the wallet. The p2sh wallet will lock the funds from Alice's end until the completion of the expiration date, though the wallet will allow the Komodo team to access the funds if necessary. This is called a "time-locked deposit." After her chosen date of expiration, she can reclaim her security at any time. Note that her KMD funds continue to accrue interest at the normal 5% APR.

This enables Alice to participate in our experimental High-Speed Mode feature, a fully automated protocol that tracks users' trading activities and monitors their unconfirmed swaps against their time-locked deposits. While using High-Speed Mode, Alice can trade funds of amounts smaller than her time-locked deposit. (The Bob that accepts her request must also be willing to engage in the Speed Mode feature.)

Her trading partners dynamically decrease her trust level as she trades, monitoring the amount of her unconfirmed transactions against her total security deposit. Should she reach an unconfirmed trading capacity that is roughly equal to the amount in her deposit, the protocol blocks her from participating in the Speed Mode feature until her funds obtain more clearance through notarization on their respective networks.

Should Alice attempt to cheat during any period of zero confirmations, the Komodo team can activate the p2sh wallet security deposit and deduct the amount of her offense, and a penalty fee, from her security deposit to compensate the affected parties. The remainder will be available for her to reclaim at the date of the original expiration, at the latest.

With the security deposit in place, Alice can use the Speed Mode feature to complete a trade in as little as three to five seconds. Note that this feature is new, highly experimental, and we recommend users exercise extreme caution when participating. If a user cannot activate Speed Mode, BarterDEX defaults to the normal, non-<zeroconf> atomic-swap trading method.

Realtime Metrics

Nodes on BarterDEX use Realtime metrics (RTmetrics) to filter the possible candidates for atomic-swap matching. All nodes track global stats via a <stats.log> file. This log file allows each node to self-update the list of pending swaps on the network. By nature, the BarterDEX protocol has filters that give less priority to nodes that are already occupied. Additionally, the Alice-side protocol gives less preference to Bob-side protocols that do not have enough UTXO sizes visible in the orderbook. This is a new feature, and we expect to optimize and enhance it in future iterations.

Orderbook Propagation

When considering how prices compare between two cryptocurrencies, BarterDEX uses the convention of “base/rel,” which can be translated as “base currency to relevant currency.” The price is calculated by determining (base currency)/(relevant currency). The relevant currency is the cryptocurrency Alice is using to make the initial purchase, and the base is the currency Alice intends to buy.

To construct a public orderbook, a node needs to have price information. Since BarterDEX communicates primarily by means of pubkeys, the price for each currency must naturally be obtained from a pubkey. In the long run, for orderbook performance, we will need a specific <txid>/<vout> for each node, as each individual node could have hundreds of UTXOs. Currently, propagating all this information globally would use an excessive amount of bandwidth, so we therefore use a different solution. BarterDEX instead uses a hierarchical orderbook, where the skeleton of the orderbook is simply the (pubkey)/(price) for any (base)/(relevant) pair.

Note: this means that purchasing a cryptocurrency at the (base)/(relevant) price is directly comparable to selling the cryptocurrency using (relevant)/(base) at a ratio of 1/(price).

Using the (pubkey)/(price) pairing, all that is needed to populate the orderbook skeleton is for nodes to broadcast their pubkey and price for any (base)/(rel) pair. Nodes that are running a local coin daemon therefore broadcast their lists of UTXOs, which helps to propagate the orderbook. All of this is done in the background, on-demand.

Critical information is broadcast with fully signed encryption to prevent spoofing; thus all nodes can verify the smart address associated with a pubkey. In this way, nodes can validate the price broadcasted. (The electrum SPV coins have their own specific SPV-validation process for all UTXOs before they can be approved for trading on BarterDEX.)

While all nodes could broadcast their UTXO lists constantly to keep them updated, this would result in the network rapidly being overrun with congestion. To eliminate this issue, BarterDEX simply relies on the (pubkey)/(prices) as this is all that is necessary to maintain useful orderbooks.

Since there are N*N possible orderbooks (given *N* currencies), it is not practical to have BarterDEX configured to update all possible orderbooks constantly. Instead, orderbooks are created on the user end when requested from the raw public data. During orderbook creation, if the top entries in the orderbook do not possess any listunspent data, a request is made to the network to gather this information.

This process ensures that by the time a trade completes, there is already a request for an orderbook, which in turn requests the listunspent data for the most likely pubkeys. The actual order-matching process then iterates through the orderbook, scanning all the locally known UTXOs to find a high-probability counterparty to whom BarterDEX can then propose a "request" offer. In practice, early users on BarterDEX can currently experience nearly instantaneous responses, assuming all the parameters are properly met.

The BarterDEX API

We created an API model that is the same for all coins—with the obvious exceptions of the electrum-API call itself, and within some of the returned JSON files that have different calls, such as “listunspent.”

Furthermore, the underlying technology of BarterDEX enables the API to treat all bitcoin-protocol compatible coins with a universal-coin model. Therefore, when working with the BarterDEX API, an independent developer working to feature their coin on BarterDEX need only use the API “coin” symbol to receive the full set of BarterDEX features.

There are several feature requirements in the core code of the blockchain coin, and if these features are not included in the core there may be some limitations. For example, a coin that is not built on the Bitcoin-protocol Check Lock-Time Verify (CLTV) feature can still take advantage of the liquidity-taker side of the BarterDEX API. For a coin to work in native mode, it must also have a <gettxout> RPC call.

If the coin has the CLTV OP_CODE, it can be both the liquidity provider and the liquidity taker. For coins using SPV, BarterDEX only supports the liquidity-taking side (for overall network-performance reasons). Also, we assume that any trader with ambitions of being a serious liquidity provider should also be serious enough to install the coin daemon for the coins they are trading, as this will increase their speed of processing.

A Brief Discussion on the Future of BarterDEX

This concludes a high-level summary of the BarterDEX protocol as created by the Komodo organization. It is now fully functioning and live, and with the support of our community, we have successfully completed thousands of atomic swaps.

We are now preparing for our upcoming launch of the Monaize dICO. The MNZ token is an e-banking solution, built with our assistance by our partners, Monaize. When the Monaize dICO officially begins, it will be a trial by fire. Potentially, many atomic swaps will be invoked simultaneously on BarterDEX. Current stress testing indicates that the expected load is manageable. However, if we receive higher volume levels than we are expecting, it is possible we may experience a crash of the BarterDEX network. For this reason, we encourage all participants to proceed with caution. Every element of the Komodo ecosystem is still considered to be highly experimental. We provide no investment advice, nor any guarantees of any funds utilized on our network. Use our products only at your own risk.

Looking past the Monaize dICO, BarterDEX will continue to evolve. The current iteration has already identified several areas of improvement for the next iteration. We are currently implementing features that will enable BarterDEX to communicate with ERC20-based coins (a popular technology on the Ethereum network). Several different GUI systems are also under construction by various community members, all of which are utilizing the BarterDEX 1.0 API. As we develop the BarterDEX API, we are making sure that future iterations are backwards compatible for developer ease-of-use.

Part IV

Komodo’s Native Privacy Feature: Jumblr

Abstract

Jumblr is a Komodo technology that enables users to anonymize their cryptocurrencies. At its foundational level, Jumblr takes non-private funds from a transparent (non-private) address, moves the funds through a series of private and non-traceable zk-SNARK addresses—which disconnects the currency trail and anonymizes the funds—and then returns the funds to a new transparent address of the user's choosing. Through a connected Komodo technology, BarterDEX, Jumblr can provide this service not only for Komodo's native coin, KMD, but also for any cryptocurrency connected to the Komodo ecosystem.

Introduction

The Option of Privacy is Essential to the Komodo Ecosystem

One primary goal of the Komodo ecosystem is to provide our users with the highest levels of security. The option to enable oneself with privacy is an inherent part of a strong security system. Privacy empowers users with the ability to make choices without being directly controlled or observed by a third-party actor.

Many of humanity's most meaningful advancements in art, technology, and other human endeavors began in situations where the creator had the security of privacy in which to explore, to discover, to make mistakes, and to learn thereby.

The roots of the Komodo ecosystem stem from the seminal work of Satoshi Nakamoto and his Bitcoin protocol[10]. One of the key challenges in this technology is that the original protocol does not make any account for privacy. Therefore, in advancing blockchain technology, we created Jumblr—a privacy feature—to empower Komodo-ecosystem members with this necessary security.

Challenges for Privacy-centric Systems and the Komodo Solution

Current pathways to obtain privacy in the blockchain industry have many problems.

One of the most popular methods to obtain privacy is the use of a centralized mixing service. In this process, users send their cryptocurrencies to service providers, who then mix all the participants' coins together, and return the coins according to the relevant contributions. With this method, the most dangerous issue, among many, is that for the duration of the mixing period users lose control over their currency. The funds, therefore, are subject to theft and human error.

Other decentralized coin-mixing methods, such as the coin shuffle[11], require coordinating with other human parties. This also introduces the potential for the same issues of theft and human error, and adds yet another risk: the coordination between human parties can result in the disclosure of a user's privacy.

Some cryptocurrencies support mixing as a part of the normal transaction process out of a desire to provide constant anonymization. Varying methods for randomizing these transaction-mixing patterns

[10] https://bitcoin.org/bitcoin.pdf

[11] https://bitcoinmagazine.com/articles/shuffling-coins-to-protect-privacy-and-fungibility-a-new-take-on-traditional-mixing-1465934826/

exist among the many different brands of such cryptocurrencies. The most popular of these coins is Monero.

There is a problem underlying these mixing patterns: regardless of the amount of mixing, those who use those cryptocurrencies leave a data trail in the public domain for computers to analyze later. As computer-processing power grows, transactions that were formerly private can become transparent once computer power surpasses the necessary threshold. Therefore, this method of privacy suffers from a lack of permanence.

The Komodo Solution

An Introduction to Jumblr

Our Jumblr technology solves these issues through a two-layered approach, relying on connected technologies in the Komodo ecosystem—BarterDEX and our native Komodo coin (KMD). The process is managed locally on the user's machine and requires no third parties, human coordination, or other mixing services.

A Brief Explanation of The Two Foundational Technologies

Komodo Coin (KMD)

KMD is a cryptocurrency that enables users to conduct both transparent and private transactions. In developing the Komodo ecosystem, we use KMD as the native cryptocurrency for many connecting technologies. KMD thereby continually gains usefulness as more Komodo tools are built upon it, including Jumblr.

KMD Began as a Fork of Zcash

This coin began as a fork of the popular privacy coin, Zcash[12]. As such, KMD retains the same inherent privacy features. Notable among these features are the Zcash parameters and zk-SNARK technology. These enable users to move funds on a public blockchain without leaving a data trail for later analysis. This is one of the most powerful forms of blockchain privacy in existence, as the provided privacy is effectively permanent. The Zcash parameters and zk-SNARK technology provide the initial foundation for users to take transparent KMD funding and make it anonymous (with the assistance of Komodo's Jumblr technology).

BarterDEX

BarterDEX is an open-source protocol designed and pioneered by the Komodo team. It allows people to trade cryptocurrency coins without a counterparty risk. The protocol is open-source and trading is available for any coin that developers choose to connect to BarterDEX.

An in-depth discussion of BarterDEX is provided in the previous section of this paper.

[12] https://z.cash/

Iguana Core

A core Komodo technology called Iguana Core is fundamental to the overall functionality of the Komodo ecosystem. It is at the center of nearly all Komodo projects, and Jumblr is no exception. For more information on Iguana Core, please see our Komodo GitHub repository. There is also more detail provided in the BarterDEX section of this whitepaper.

Komodod

Komodod is the name of the background software (also called a daemon) that runs behind the scenes of essentially all Komodo-related software. There is more information provided on Komodod in the dICO part of this paper.

The Jumblr Process

Jumblr enables users to anonymize their funds. The Jumblr process is rooted in our native Komodo coin (KMD), and the privacy features can extend thereby to any blockchain project connected to the Komodo ecosystem.

Anonymizing Native Komodo Coin (KMD)

At its most simple level, Jumblr takes non-private KMD funds from a transparent (non-private) address, moves the funds through a series of private and non-traceable zk-SNARK addresses—which disconnects the currency trail and anonymizes the funds—and then returns the funds to a new transparent address of the user's choosing.

The entirety of the anonymization process is conducted through the user's local machine(s), with one exception—that of sending the data to the network for mining. Therefore, Jumblr eliminates many dangers, including the issues of theft, human error, the disclosure of user privacy through human coordination, and the unraveling of privacy by increasing computer processing power.

User Actions

The commands that initiate Jumblr exist within Komodo's foundational program on the user's local machine, Komodod. This program is included in a typical Komodo installation, and, under normal circumstances, Komodod is natively connected to the same KMD addresses accessed by the user.

Therefore, users in the Komodo ecosystem have access to Jumblr's privacy technology without any further effort. Developers of standalone GUI applications for the Komodo ecosystem can integrate Jumblr commands into user interfaces in any desired manner.

There are two main commands, or API calls, available:

- jumblr_deposit <KMDaddress>
- jumblr_secret <secretKMDaddress>

jumblr_deposit <KMDaddress>

This command initiates the anonymization of KMD.

Before executing the command, the user prepares the funds by placing them within the chosen <KMDaddress>. So long as Komodod has access to the private keys of the <KMDaddress>, nothing further is required. The user simply executes the command “jumblr_deposit <KMDaddress>” and Jumblr begins watching for and processing any funds in the <KMDaddress>.

Note: We call a transparent address a “T address.” These are fully accessible to the user, and they are the means of conducting normal transactions. All currency entering and leaving a T address is fully visible to the network.

On the other hand, we call a privacy-enabled address a “Z address,” as they utilize the Zcash parameters and zk-SNARK technology. Z addresses are internal to the Jumblr process and a user typically does not directly interact with them.

The first step Jumblr takes is to move the user’s funds from a T address to a Z address.

<u>The First Step of the Jumblr Anonymization Process</u>

Moving the funds from a transparent address to a privacy-enabled address.

T→Z

Naturally, as the T address is fully public, an outside observer can see the funds as they leave for the respective Z address. Therefore, to fully disconnect the currency trail, Jumblr then moves the funds from the initial Z address to yet another Z address.

Jumblr creates a new Z address for each individual lot.

<u>The Second Step of the Jumblr Anonymization Process</u>

Moving the funds from one unique and untraceable Z address to another

Z→Z

Through the technology of the Zcash parameters, zk-SNARKs, and Jumblr, the specific whereabouts of the funds are known only to the user. The user does not need to follow the movements of T→Z and Z→Z. However, for the advanced user, there are Jumblr commands available that allow for more active interaction at these stages (see the Komodo wiki for further details). One command to mention here is z_gettotalbalance. This reveals to the user the total balance they hold within all their Z addresses.

Upon executing the command [jumblr_deposit <KMDaddress>], Jumblr begins continually observing the <KMDaddress>. Should the user send more funds into their <KMDaddress> while Jumblr is already processing the previous amount, Jumblr will simply take these new funds into account, perform any necessary actions to properly adopt them into the process, and continue its course.

Jumblr includes two subcommands that allow the user to pause Jumblr manually: <jumblr_pause> and <jumblr_resume>. The user can also halt Jumblr by shutting down Komodod (and any relevant standalone GUI applications).

Once the funds have reached their final Z address(es), they lay dormant, awaiting the user's next command.

jumblr_secret <secretKMDaddress>

The user executes this command to complete the Jumblr process. Jumblr will extract all the user's hidden currency from each Z address and place the funds in a new T address, which we call the <secretKMDaddress>. This makes the funds spendable again.

The Third and Final Step of the Jumblr Anonymization Process

Moving the funds from the final Z address to the final T address

Z→T

We recommend that you keep these private addresses primarily for storage. You should never share with anyone any information regarding your <secretKMDaddress>'s. Treat all relevant information like a password.

When you are prepared to spend from your private funds, we recommend that you repeat the Jumblr process again on the amount that you desire to spend. This will keep the bulk of your stored funds within a privacy "air gap," as it were. For maximum privacy, we also suggest that after emptying the public node of all funds, the user delete and destroy the wallet.dat file in which the initial privacy-creation process took place. This destroys the last remnants of the cryptocurrency trail.

Additional Security Layers

Jumblr's Process of Breaking Down Funds

The method by which Jumblr breaks down and processes the funds provides yet another layer of privacy. Jumblr begins by taking the total amount in the <KMDaddress> and, if necessary, splitting it until the largest quantities are all equal to ~7770 KMD. It then breaks down the remainder into quantities of ~100 KMD, and then the remainder thereafter into quantities of ~10 KMD. Any final remainder (which would be anything less than ~10 KMD) is ignored.

Note that Jumblr also automatically extracts its 0.3% overall fee during the Jumblr process.

Therefore, the total amount is broken down into lot sizes of ~7770 KMD, ~100 KMD, and ~10 KMD.

Jumblr's Process of Moving the Individual Lots into a Private Address

Jumblr does not immediately move each lot into a Z address. Instead, it performs its actions in a randomized pattern to optimize anonymity, using the collective of all Jumblr users in the Komodo ecosystem to blend the transactions of the crowd together.

First, all Jumblr actions throughout the ecosystem are programmed to cluster around block numbers that are a multiple of ten (i.e. blockchain height = XXXXX0). This gathers all Jumblr requests from all users for the given time into one large group, clustered together every ten minutes (a single block generates every minute, and therefore the tenth block occurs every tenth minute).

At the moment of activity, Jumblr does one of two things: it either performs the next action in the process of anonymization, or it chooses to do nothing.

Option 1: Jumblr performs the next action

When Jumblr looks at the next action, it can perform one of three possible steps:

- T→Z
    a. If the lot has yet to be moved out of the <KMDaddress>, Jumblr can move it from the first T address to the first Z address.
- Z→Z
    a. Assuming the lot is now in the first Z address, Jumblr can move it to the final Z address.
- Z→T
    a. Assuming the <jumblr_secret> API call is activated, Jumblr can move the lot from the final Z address to the final T address: <secretKMDaddress>.

Option 2: Jumblr does nothing

- At each turn, instead of performing any of the above steps, Jumblr can simply abstain from any action. This happens approximately half of the time.

Through these actions, Jumblr adds a layer of obfuscation on top of the Zcash parameters and zk-SNARK technology by adding privacy to the timing and movements of each step for each user.

Additional Privacy Considerations

Although the KMD anonymization process provides a measure of privacy and may appear to be sufficient, there are still more precautions a user must take. Two main attacks are available to a would-be sleuth.

The Timing Attack

In this attack, the sleuth simply studies the time the funds disappear from the <KMDaddress> and looks for funds to appear in a T address soon thereafter. If the privacy-user persistently chooses predictable timing for initiating and completing the Jumblr commands, a determined sleuth might deduce a user's <secretKMDaddress>.

The aforementioned process of grouping and randomizing the timing of movements provides one layer of security against The Timing Attack. Users thus blend the timing of their movements together, using the power of the collective to obscure their transactions from the sleuth.

However, The Timing Attack remains an issue if the user is the only person employing Jumblr for the duration of the anonymization of their funds. In this event, effectively no anonymization takes place. The sleuth can clearly see the funds leave from the <KMDaddress> and return to the <secretKMDaddress> later. Therefore, to be effective, Jumblr requires more than one user and gains strength with higher levels of adoption. Given the growing size of the Komodo community, we anticipate that users will easily be able to overcome The Timing Attack.

The Knapsack Attack

The Knapsack Attack is somewhat like The Timing Attack, but as applied to amounts. For example, if there is only one KMD address that entered ~1000000 KMD into Jumblr, and ~1000000 KMD later emerges elsewhere, the sleuth can easily discern the user's <secretKMDaddress>.

The process of breaking down the total amount into three equal sized lots (~7770, ~100, ~10 KMD) for all users provides one layer of security against The Knapsack Attack. Users again can blend their transactions together, using the power of the collective to obfuscate their movements.

Jumblr has another feature, Multiple Secret Addresses, that also protects against this attack. This feature is explained in the following section.

Further Security Enhancements to Combat The Timing and Knapsack Attacks

More Defense Against The Knapsack Attack: Multiple Secret Addresses

As another layer of security, users can create multiple secret KMD addresses (<secretKMDaddress>'s) and actively use them in the Jumblr process.

When using multiple <secretKMDaddress>'s, whenever Jumblr reaches the stage of Z→T for any given lot of KMD, Jumblr will randomly choose one of the <secretKMDaddress>'s for this lot's final T address. This enables the user to split their initial funding into many different <secretKMDaddress>'s, thus providing another layer of security against The Knapsack Attack.

Jumblr manages up to 777 <secretKMDaddress>'s at one time.

Further Enhancements Against The Timing Attack

The simplest and strongest defense against The Timing Attack is in the hands of the users. Recall that a user chooses the times they execute the commands <jumblr_deposit> and <jumblr_secret>. The longer a user maintains their currency within the shielded Z address(es), the more security they have against The Timing Attack. This is because the Jumblr actions of other users during the interim obfuscate the trail. We therefore encourage users who are mindful for protection against this attack to delay the period of execution between the two commands.

We also developed Jumblr to have additional inherent protections against The Timing Attack for cases where users desire a more immediate transfer. Assuming Jumblr is activated on the user's local computer, as soon as Jumblr detects a new deposit in the <KMDaddress>, it can begin the anonymization process. However, Jumblr deliberately delays its own progress to provide a layer of security against The Timing Attack.

Recall that all user actions are clustered around block numbers that are multiples of ten, and half the time, Jumblr decides to do nothing. Therefore, in statistical terms, although the Jumblr background process may be constantly running in Komodod, Jumblr only activates to check for pending tasks every tenth minute, and only performs tasks every twentieth minute. Thus, each hour has roughly three different moments when Jumblr will perform one of the three available actions: T→Z, Z→Z, and Z→T. This program randomizes the amount of time it takes to complete the Jumblr process.

Assuming during a given period of activity Jumblr decides to perform the action of T→Z, it begins by working through the different sizes of lots from largest to smallest—thus beginning with a ~7770-KMD lot until they are all allocated, then to the ~100-KMD lots, and finally to the ~10-KMD lots. During any individual period of activity, Jumblr will perform the T→Z movement for no more than a single lot, and then stop.

However, when Jumblr performs either of the other two actions (Z→Z and Z→T) it will make the transfers for all lots that are in play.

Through these additional securities, therefore, Jumblr defeats The Timing Attack and The Knapsack Attack, relying on the power of the Zcash parameters and zk-SNARK technology. The more participants in Jumblr, the more privacy users gain. For those who use Jumblr on a consistent basis, the 0.3% cost of utilizing Jumblr is offset by the 5% APR inherent in the Komodo coin (KMD). Thus, for a small fee, Jumblr users can provide both themselves and their community with privacy.

Offering Privacy to Other Cryptocurrencies

Jumblr can provide privacy to any cryptocurrency that is connected to the Komodo ecosystem, as BarterDEX is natively integrated. Currently, the user is required to perform the first and final steps of trading in the Jumblr process of non-KMD cryptocurrencies. In the long term, however, Jumblr is capable of fully automating the process. We await larger adoption to complete the non-KMD automation features.

The Current Jumblr Process: Manual non-KMD to KMD Trading on BarterDEX

Overall, to provide privacy to a non-KMD cryptocurrency in the Komodo ecosystem, that currency must first be traded on BarterDEX into KMD. Once the underlying value is held as KMD in a <KMDaddress>, Jumblr can complete its work. Upon completion, the anonymized KMD is then exchanged on BarterDEX again for the relevant non-KMD cryptocurrency, and returned to a secret address of the user's choosing.

At present, while BarterDEX is in its early stages, we are focusing our energies on increasing overall BarterDEX usability.

Future Capabilities: Jumblr Automates the BarterDEX Trading Process for the User

In the future, Jumblr will simply be a client of the BarterDEX service when providing privacy to non-KMD cryptocurrencies.

When a user activates Jumblr for a non-KMD coin, Jumblr will instruct BarterDEX to trade the non-KMD coin into transparent KMD according to the current prices. The underlying value now being in KMD, the Jumblr protocol performs the entirety of the process previously described. With the underlying value made private, Jumblr will direct BarterDEX to exchange the value back to the user's chosen cryptocurrency. Finally, Jumblr will return the final sum to a new cryptocurrency address, provided by the user at the outset of the process.

Due to market fluctuations, depending on liquidity, it is possible that a user will experience slippage in the underlying value of their non-KMD cryptocurrency. While it would be possible to prearrange the trade on BarterDEX (thereby eliminating any slippage), there is no available method to make such an arrangement without leaking privacy information. The party performing the second half of the trade on

BarterDEX would be a central point of failure. Therefore, the most private method for non-KMD privacy creation is to simply rely on the active BarterDEX liquidity providers.

A Word on Risks Inherent in Jumblr and the Komodo Ecosystem

The Komodo coin (KMD), and therefore Jumblr by association, both rely on the Zcash parameters as put forth by the Zcash team. The Zcash parameters are a “zero-knowledge” form of technology. This is a powerful form of privacy, and arguably superior to other forms as it is effectively permanent. Relying on the Zcash parameters allows us to turn our creative resources to other blockchain-technology challenges, while still empowering members of the Komodo ecosystem with the option of privacy.

To create the Zcash parameters, the original Zcash developers had to create a series of keys that, when combined, created a master key that could unlock and lock the parameters. After using the master key to create the parameters, the team destroyed every individual key. The team conducted this endeavor in a public manner. We encourage interested readers to view the “Zcash Ceremony” explanation, and to search for other viewpoints as well.

To briefly summarize the security measures, the Zcash team used several layers of protection including: multi-party computation, air-gapped compute nodes, hard-copy evidence trails, a uniquely crafted distribution of the Linux operating system, and the physical destruction of each piece of hardware that held an individual key. The resulting layers of defense would be of the highest level of difficulty for an outsider to penetrate. Furthermore, the method of creation and destruction ensured that the internal security of the project was faultless, so long as at least one member of the entire Zcash team was honest.

By our observation, the team performed this endeavor with sufficient competence and due diligence. Furthermore, given the nature of the project, the longstanding reputation of the Zcash developers, and the modus operandi of their lives’ work, we believe they were properly motivated to perform the creation and destruction in a capable and honest manner.

Nevertheless, there are privacy advocates in the cryptocurrency industry who maintain a degree of suspicion over any project that requires an element of human trust. This suspicion extends to the Zcash parameters. These observers continually scrutinize the Zcash project, searching for more and more processes by which the creation ceremony could have failed. Yet, while various theories have been put forth, no actual failure in the Zcash parameters has been discovered.

In adopting the Zcash parameters, we receive frequent questions regarding how they affect the Komodo coin. The answer is that the privacy in the Komodo ecosystem is permanent, regardless of any potential fault by the Zcash team. Furthermore, we can adopt any updates the Zcash team releases to the parameters.

In the unlikely event that someone was able to retain a complete copy of the master key, the only power the holder would have, would be the ability to create new private money in our system. This holder could then trade that for transparent, spendable money. This could negatively impact the Komodo coin, and we would be required to adapt our platform. If a fault in the Zcash parameters were to be discovered, the Komodo team has various contingency methods at our disposal to remove the Zcash parameters and replace them with a new set of parameters.

Though in Komodo we do not see this as a realistic threat, we nevertheless include the information here in our white paper to provide complete transparency for any user who seeks to invest their resources in the Komodo project.

Jumblr Provides the Komodo Ecosystem with Privacy

For the Komodo ecosystem to reach its full potential, the option of enhanced privacy must be available to Komodo users. Jumblr fills this demand.

Jumblr relies on BarterDEX, KMD, and Iguana Core to connect to the Komodo ecosystem. The foundational privacy it offers is built upon the KMD coin, the Zcash parameters, and zk-SNARK technology. Additional enhancements are built into the Jumblr process to maximize user privacy, including protections against The Timing Attack and The Knapsack Attack. Through BarterDEX and Iguana Core, these privacy features extend to any cryptocurrency connected to the Komodo ecosystem.

As more users become a part of the Komodo ecosystem, they can work together to enhance both their own privacy and the privacy of fellow ecosystem members. As the ecosystem continues to grow, there are various levels of growth the Komodo team can offer to Jumblr, including automating the non-KMD Jumblr process. We look forward to receiving your feedback on this privacy-enhancing technology.

Part V

Additional Information Regarding the Komodo Ecosystem

Final Notes Regarding the Komodo Project

There are few final miscellaneous topics to discuss. These include our strategy for fiat-pegged cryptocurrencies (PAX) and our outlook for smart-contract technology.

Fiat-Pegged Cryptocurrencies

Our strategy towards fiat-pegged cryptocurrencies (PAX) has recently changed.

Previously, we featured on our website a white paper that outlined a PAX strategy. That former strategy was created before it was clear whether governments of the world would embrace blockchain technology.

Today, it seems that governments are updating their philosophies and preparing for blockchain adoption. Governments appear to be considering a need to create blockchain-based cryptocurrencies that can be exchanged for their existing fiat currencies.

In many cases, we may be able to directly integrate these government-sponsored fiat-to-blockchain cryptocurrencies natively in BarterDEX. Blockchain projects that properly utilize the core security features of the Bitcoin protocol are capable of properly performing atomic swaps.

As it is possible that government-sponsored cryptocurrencies may natively integrate with BarterDEX, it appears that creating our own PAX technology may be unnecessary. We are putting all PAX endeavors on hold at this time.

Smart Contracts on the Komodo Platform

Asset chains in the Komodo ecosystem can use the smart-contract capabilities native to the Bitcoin protocol. Various vendors and developers in the open-source community provide resources to make this easier, though we make no specific endorsements of any product. One example of smart-contract technology native to the Bitcoin protocol is a Conditional Time-Locked Deposit, which our BarterDEX technology utilizes in the trading process.

In the long term, we also intend to release our own smart-contract technology that will greatly enhance the coding experience. Our intention is to make our smart-contract technology language-agnostic, meaning that any language (JavaScript, Ruby, C#, Python, etc.) will be able to execute smart contracts in the Komodo ecosystem. This will empower both asset chains as well as the main chain. We intend to begin creating this smart-contract technology later this year (2018).

Details Regarding the Primary Chain of the Komodo Ecosystem: KMD

Circulating Coin Supply:	~100000000
Total Coin Supply (yr. 2030):	~200000000

The foundational coin of the Komodo ecosystem is named after the ecosystem itself, Komodo (KMD).

It is the most versatile coin we are building. Whenever we create new technologies for our ecosystem, we seek to establish a relationship between the functionality of the technology and the usefulness of KMD. For instance, KMD is the native cryptocurrency for Jumblr. All other cryptocurrencies in the Komodo ecosystem that seek to utilize Jumblr's privacy must first be traded on BarterDEX for KMD.

After the privacy process is complete, the users then exchange KMD on BarterDEX for their desired cryptocurrency. In the future, we also intend for KMD to be the fuel for our smart-contract technology. Those are but a few examples of Komodo's usefulness. Readers may discover many more by discussing KMD with members of our community.

Furthermore, those who hold KMD can receive a reward in the form of compound interest. Any wallet address that holds at least 10 KMD will automatically earn interest at 5% APR. This reward is built into the core code of Komodo. It comes from an opportunity provided by our unique security system, dPoW. The nature of the reward derives from rewards that are typically given to miners on a normal PoW chain.

Normally, in PoW blockchains there is a high reward paid to the miners as an incentive for providing the security of their hash power. This reward is given in the form newly minted coins on the blockchain. For instance, in Bitcoin, ~12.5 BTC are rewarded to a miner for each block mined. (This reward comes in addition to any fees that are paid for transactions on the network.) Each ~12.5 BTC reward increases the total coin supply on the Bitcoin blockchain. In Bitcoin, this reward will continue until the total supply reaches 21M BTC, at which point the reward and the new-coin minting process will cease. Miners will only be rewarded out of transaction fees thereafter.

Because KMD relies on the hash power provided by another PoW network for security (such as Bitcoin), KMD itself does not need the same level of hash power. Therefore, KMD does not need to provide the same incentive to miners. The Komodo team designed KMD to instead distribute the reward of newly minted coins not to the miners, but instead to those who hold at least 10 KMD in a wallet address.

To maintain a constant 5% APR, users should move their money on the blockchain at least once per year. The interest is calculated as a part of the UTXO transfer process (see Part III of this paper for details on UTXOs). The KMD code only calculates interest for UTXOs up to one full year, and then stops growing. By simply sending the full balance of a wallet to the same receiving address, a user can generate a new UTXO. This will continue the rewards for at least one more full year.

The reward will continue for a period of approximately twelve to fourteen years. When Komodo's overall coin supply reaches ~200M, this reward will also discontinue. Specifically, the reward will cease when the KMD chain reaches a block height of 7777777.

It is important to note that no one is forced into using KMD in our ecosystem. We are often asked why we chose this route, as the free nature of the Komodo ecosystem can be in direct contrast to the philosophies of many other ecosystems and exchanges. Other ecosystems often force users into using the developer's coin.

The reason why we follow a more open practice is that we strive to adhere to the guiding principles of decentralization and open-source technology. We want to create a blockchain platform where people are free to use whatever is most useful for them in their entrepreneurial endeavors. Keeping KMD as an optional element empowers the members of the Komodo ecosystem with freedom.

Conclusion

This concludes a thorough explanation of the foundational technologies of the Komodo ecosystem. We are working diligently to improve the user experience. While some may say that the cryptocurrency industry is but a bubble, at Komodo we believe we have not yet begun the fight. We hope that the innovations we provide will be a meaningful contribution to the remarkable advent of decentralization and open-source technology.

Acknowledgements

References

BarterDEX – A Practical Native DEX (https://github.com/SuperNETorg/komodo/wiki/BarterDEX-%E2%80%93-A-Practical-Native-DEX)

Nakamoto Satoshi (2008): Bitcoin: A peer-to-peer electronic cash system. (http://www.bitcoin.org/bitcoin.pdf)

Mtchl (2014): The math of Nxt forging (https://www.docdroid.net/ahms/forging0-4-1.pdf.html)

King Sunny, Nadal Scott (2012): PPCoin: Peer-to-Peer Crypto-Currency with Proof-of-Stake (https://peercoin.net/assets/paper/peercoin-paper.pdf)

Delegated Proof-of-Stake Consensus (https://bitshares.org/technology/delegated-proof-of-stake-consensus/)

Miers Ian, Garman Christina, Green Matthew, Rubin Aviel: Zerocoin: Anonymous Distributed E-Cash from Bitcoin (https://isi.jhu.edu/~mgreen/ZerocoinOakland.pdf)

Ben-Sasson Eli, Chiesa Alessandro, Garman Christina, Green Matthew, Miers Ian, Troer Eran, Virza Madars (2014): Zerocash: Decentralized Anonymous Payments from Bitcoin (http://zerocash-project.org/media/pdf/zerocash-extended-20140518.pdf)

Ben-Sasson Eli, Chiesa Alessandro, Green Matthew, Tromer Eran, Virza Madars (2015): Secure Sampling of Public Parameters for Succinct Zero Knowledge Proofs (www.diyhpl.us/~bryan/papers2/bitcoin/snarks/Secure%20sampling%20of%20public%20parameters%20for%20succinct%20zero%20knowledge%20proofs.pdf)

NXT Community: NXT White paper (http://wiki.nxtcrypto.org/wiki/White paper:Nxt)

Larimer Daniel, Scott Ned, Zavgorodnev Valentine, Johnson Benjamin, Calfee James, Vandeberg

Michael (March 2016): Steem, An incentivized, blockchain-based social media platform.(https://steem.io/SteemWhite paper.pdf)

BitFury Group (Sep 13, 2015): Proof of Stake versus Proof of Work White Paper (http://bitfury.com/content/5-white-papers-research/pos-vs-pow-1.0.2.pdf)