WHITEPAPER

A Multi-Decentralized Interaction Protoco

ant block chain protocol

ABCP333@gmail.com

https://www.abcp.club/

Reinhad

August 3, 2014

“Laissez faire les propri´etaires.” — Pierre-Joseph Proudhon

Abstract

The popularization of Bitcoin, a decentralized crypto-currency has inspired the production of several alternative, or “alt”, currencies. Ethereum, CryptoNote, and Zerocash all represent unique contributions to the cryptocurrency space. Although most alt currencies harbor their own source of innovation, they have no means of adopting the innovations of other currencies which may succeed them. We aim to remedy the potential for atrophied evolution in the crypto-currency space by presenting ABCP, a generic and self-amending crypto-ledger. ABCP can instantiate any blockchain based protocol. Its seed protocol specifies a procedure for stakeholders to approve amendments to the protocol, including amendments to the amendment procedure itself. Upgrades to ABCP are staged through a testing environment to allow stakeholders to recall potentially problematic amendments. The philosophy of ABCP is inspired by Peter Suber’s Nomic[1], a game built around a fully introspective set of rules. In this paper, we hope to elucidate the potential benefits of ABCP, our choice to implement as a proof-of-stake system, and our choice to write it in OCaml.

Ants block chain protocol (abbreviated as ant coin: ABCP) is a multi-decentralized interactive protocol. It runs on the ant blockchain and has different forms of heterogeneous multi-centered interaction protocols, which is convenient for developers. Developed asset management applications; the contract layer uses the creation contract and control contract for asset issuance and management, and supports the extended utxo model butxo at the bottom.

We have proposed a system for electronic transactions without relying on trust. We started with the usual

framework of coins made from digital signatures, which provides strong control of ownership, but is incomplete without a way to prevent double-spending. To solve this, we proposed a peer-to-peer network

using proof-of-work to record a public history of transactions that quickly becomes computationally impractical for an attacker to change if honest nodes control a majority of CPU power. The network is robust in its unstructured simplicity. Nodes work all at once with little coordination. They do not need to be identified, since messages are not routed to any particular place and only need to be delivered on a best effort basis. Nodes can leave and rejoin the network at will, accepting the proof-of-work chain as proof of what

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

The A Multi-Decentralized Interaction Protocol protocol was originally conceived as an upgraded version of a cryptocurrency, providing advanced features such as on-blockchain escrow, withdrawal limits, financial contracts, gambling markets and the like via a highly generalized programming language. The A Multi-Decentralized Interaction Protocol protocol would not "support" any of the applications directly, but the existence of a Turing-complete programming language means that arbitrary contracts can theoretically be created for any transaction type or application. What is more interesting about A Multi-Decentralized Interaction Protocol, however, is that the A Multi-Decentralized Interaction Protocol protocol moves far beyond just currency. Protocols around decentralized file storage, decentralized computation and decentralized prediction markets, among dozens of other such concepts, have the potential to substantially increase the efficiency of the computational industry, and provide a massive boost to other peer-to-peer protocols by adding for the first time an economic layer. Finally, there is also a substantial array of applications that have nothing to do with money at all.

The concept of an arbitrary state transition function as implemented by the A Multi-Decentralized Interaction Protocol protocol provides for a platform with unique potential; rather than being a closed-ended,single-purpose protocol intended for a specific array of applications in data storage, gambling or finance, A Multi-Decentralized Interaction Protocol is open-ended by design, and we believe that it is extremely well-suited to serving as a foundational layer for a very large number of both financial and non-financial protocols in the years to come.

DISCLAIMER

THE PURPOSE OF THIS WHITEPAPER IS TO PRESENT THE PROJECT OF ABCP TO POTENTIAL TOKEN HOLDERS AND PLATFORM USERS IN CONNECTION WITH THE PROPOSED TOKEN LAUNCH. THE INFORMATION SET FORTH BELOW MAY NOT BE EXHAUSTIVE AND DOES NOT IMPLY ANY ELEMENTS OF A CONTRACTUAL RELATIONSHIP. ITS SOLE PURPOSE IS TO PROVIDE RELEVANT AND REASONABLE INFORMATION TO POTENTIAL TOKEN HOLDERS IN ORDER FOR THEM TO DETERMINE WHETHER TO UNDERTAKE A THOROUGH ANALYSIS OF THE COMPANY WITH THE INTENT OF ACQUIRING ABCP TOKENS. NOTHING IN THIS WHITEPAPER SHALL BE DEEMED TO CONSTITUTE A PROSPECTUS OF ANY SORT OR A SOLICITATION FOR INVESTMENT, NOR DOES IT IN ANY WAY PERTAIN TO AN OFFERING OR A SOLICITATION OF AN OFFER TO BUY ANY SECURITIES IN ANY JURISDICTION. THIS DOCUMENT IS NOT COMPOSED IN ACCORDANCE WITH, AND IS NOT SUBJECT TO, LAWS OR REGULATIONS OF ANY JURISDICTION WHICH ARE DESIGNED TO PROTECT INVESTORS. THE TOKENS OFFERED IN THE TOKENSALE DO NOT PROVIDE ANY VOTING RIGHTS, PROFIT SHARE, DIVIDENDS OR OTHER EXPECTATIONS ON CURRENT OR PROFITS. CERTAIN STATEMENTS, ESTIMATES, AND FINANCIAL INFORMATION CONTAINED IN THIS WHITEPAPER CONSTITUTE FORWARD-LOOKING STATEMENTS OR INFORMATION. SUCH FORWARD-LOOKING STATEMENTS OR INFORMATION INVOLVE KNOWN AND UNKNOWN RISKS AND UNCERTAINTIES WHICH MAY CAUSE ACTUAL EVENTS OR RESULTS TO DIFFER MATERIALLY FROM THE ESTIMATES OR THE RESULTS IMPLIED OR EXPRESSED IN SUCH FORWARD-LOOKING STATEMENTS. THIS ENGLISH LANGUAGE WHITEPAPER IS THE PRIMARY OFFICIAL SOURCE OF INFORMATION ABOUT THE ABCP TOKEN LAUNCH. THE INFORMATION CONTAINED HEREIN MAY FROM TIME TO TIME BE TRANSLATED INTO OTHER LANGUAGES OR USED IN THE COURSE OF WRITTEN OR VERBAL COMMUNICATIONS WITH EXISTING AND PROSPECTIVE CUSTOMERS, PARTNERS, ETC. IN THE COURSE OF SUCH TRANSLATION OR COMMUNICATION SOME OF THE INFORMATION CONTAINED HEREIN MAY BE LOST, CORRUPTED, OR MISREPRESENTED. THE

ACCURACY OF SUCH ALTERNATIVE COMMUNICATIONS CANNOT BE GUARANTEED. ABCP RESERVES THE RIGHT TO CHANGE THE TERMS AND CONDITIONS SET FORTH IN THIS WHITEPAPER, ITS APPENDICES AND THE GENERAL TERMS AND CONDITIONS AS PUBLISHED ON THE WEBSITE AT ANY TIME WITHOUT PRIOR NOTICE. IN THE EVENT OF ANY CONFLICTS OR INCONSISTENCIES BETWEEN SUCH TRANSLATIONS AND COMMUNICATIONS AND THIS OFFICIAL ENGLISH LANGUAGE WHITEPAPER, THE PROVISIONS OF THIS ENGLISH LANGUAGE ORIGINAL DOCUMENT SHALL PREVAIL. ABCP DOES NOT ALLOW USERS TO AVOID CAPITAL CONTROLS OF ANY KIND AND DOES NOT ALLOW THEM TO INVEST IN ANY FOREIGN ASSETS. 4 NO PERSON IS BOUND TO ENTER INTO ANY CONTRACT OR BINDING LEGAL COMMITMENT IN RELATION TO THE SALE AND PURCHASE OF ABCP TOKENS AND NO CRYPTOCURRENCY OR OTHER FORM OF PAYMENT IS TO BE ACCEPTED FOR ABCP PRODUCTS AND SERVICES BASED ON THIS WHITEPAPER.

Abstract

A purely peer-to-peer version of electronic cash would allow online payments to be sent directly from one party to another without going through a financial institution. Digital signatures provide part of the solution, but the main benefits are lost if a trusted third party is still required to prevent double-spending. We propose a solution to the double-spending problem using a peer-to-peer network. The network timestamps transactions by hashing them into an ongoing chain of hash-based proof-of-work, forming a record that cannot be changed without redoing the proof-of-work. The longest chain not only serves as proof of the sequence of events witnessed, but proof that it came from the largest pool of CPU power. As long as a majority of CPU power is controlled by nodes that are not cooperating to attack the network, they'll generate the longest chain and outpace attackers. The network itself requires minimal structure. Messages are broadcast on a best effort basis, and nodes can leave and rejoin the network at will, accepting the longest proof-of-work chain as proof of what happened while they were gone.

While several consensus algorithms exist for the Byzantine Generals Problem, specifically as it pertains to distributed payment systems, many suffer from high latency induced by the requirement that all nodes within the network communicate synchronously. In this work, we present a novel consensus algorithm that circumvents this requirement by utilizing collectively-trusted subnetworks within the larger network. We show that the "trust" required of these subnetworks is in fact minimal and can be further reduced with principled choice of the member nodes. In addition, we show that minimal connectivity is required to maintain agreement throughout the whole network. The result is a low-latency consensus algorithm which still maintains robustness in the face of Byzantine failures. We present this algorithm in its embodiment in the A Multi-Decentralized Interaction Protocol Protocol. The intent of A Multi-Decentralized Interaction Protocol is to create an alternative protocol for building decentralized applications, providing a different set of tradeoffs that we believe will be very useful for a large class of decentralized applications, with particular emphasis on situations where rapid development time, security for small and rarely used applications, and the ability of different applications to very efficiently 1 interact, are important. A Multi-Decentralized Interaction Protocol does this by building what is essentially the ultimate abstract foundational layer: a blockchain with a built-in Turing-complete programming language, allowing anyone to write smart contracts and decentralized applications where they can create their own arbitrary rules for ownership, transaction formats and state transition functions. A bare-bones version of Namecoin can be written in two lines of code, and other protocols like currencies and reputation systems can be

built in under twenty. Smart contracts, cryptographic "boxes" that contain value and only unlock it if certain conditions are met, can also be built on top of the platform, with vastly more power than that offered by Bitcoin scripting because of the added powers of Turing-completeness, value-awareness, blockchain-awareness and state. The bitcoin protocol can encompass the global financial transaction volume in all electronic payment systems today, without a single custodial third party holding funds or requiring participants to have anything more than a computer using a broadband connection. A decentralized system is proposed whereby transactions are sent over a network of micropayment channels (a.k.a. payment channels or transaction channels) whose transfer of value occurs off-blockchain. If Bitcoin transactions can be signed with a new sighash type that addresses malleability, these transfers may occur between untrusted parties along the transfer route by contracts which, in the event of uncooperative or hostile participants, are enforceable via broadcast over the bitcoin blockchain in the event of uncooperative or hostile participants, through a series of decrementing timelocks. Introduction Commerce on the Internet has come to rely almost exclusively on financial institutions serving as trusted third parties to process electronic payments. While the system works well enough for most transactions, it still suffers from the inherent weaknesses of the trust based model. Completely non-reversible transactions are not really possible, since financial institutions cannot avoid mediating disputes. The cost of mediation increases transaction costs, limiting the minimum practical transaction size and cutting off the possibility for small casual transactions, and there is a broader cost in the loss of ability to make non-reversible payments for nonreversible services. With the possibility of reversal, the need for trust spreads. Merchants must be wary of their customers, hassling them for more information than they would otherwise need. A certain percentage of fraud is accepted as unavoidable. These costs and payment uncertainties can be avoided in person by using physical currency, but no mechanism exists to make payments over a communications channel without a trusted party. What is needed is an electronic payment system based on cryptographic proof instead of trust, allowing any two willing parties to transact directly with each other without the need for a trusted third party. Transactions that are computationally impractical to reverse would protect sellers from fraud, and routine escrow mechanisms could easily be implemented to protect buyers. In this paper, we propose a solution to the double-spending problem using a peer-to-peer distributed timestamp server to generate computational proof of the chronological order of transactions. The system is secure as long as honest nodes collectively control more CPU power than any cooperating group of attacker nodes. Interest and research in distributed consensus systems has increased markedly in recent years, with a central focus being on distributed payment networks. Such networks allow for fast, low-cost transactions which are not controlled by a centralized source. While the economic benefits and drawbacks of such a system are worthy of much research in and of themselves, this work focuses on some of the technical challenges that all distributed payment systems must face. While these problems are varied, we group them into three main categories: correctness, agreement, and utility. By correctness, we mean that it is necessary for a distributed system to be able to discern the difference between a correct and fraudulent transaction. In traditional fiduciary settings, this is done through trust 2 between institutions and cryptographic signatures that guarantee a transaction is indeed coming from the institution that it claims to be coming from. In distributed systems, however, there is no such trust, as the identity of any and all members in the network may not even be known. Therefore, alternative methods for correctness must be utilized. Agreement refers to the problem of maintaining a single global truth in the face of a decentralized accounting system. While similar to the correctness problem, the difference lies in the fact that while a malicious user of the network may be unable to create a fraudulent transaction

(defying correctness), it may be able to create multiple correct transactions that are somehow unaware of each other, and thus combine to create a fraudulent act. For example, a malicious user may make two simultaneous purchases, with only enough funds in their account to cover each purchase individually, but not both together. Thus each transaction by itself is correct, but if executed simultaneously in such a way that the distributed network as a whole is unaware of both, a clear problem arises, commonly referred to as the "Double-Spend Problem." Thus the agreement problem can be summarized as the requirement that only one set of globally recognized transactions exist in the network. Utility is a slightly more abstract problem, which we define generally as the "usefulness" of a distributed payment system, but which in practice most often simplifies to the latency of the system. A distributed system that is both correct and in agreement but which requires one year to process a transaction, for example, is obviously an inviable payment system. Additional aspects of utility may include the level of computing power required to participate in the correctness and agreement processes or the technical proficiency required of an end user to avoid being defrauded in the network. Many of these issues have been explored long before the advent of modern distributed computer systems, via a problem known as the "Byzantine Generals Problem." In this problem, a group of generals each control a portion of an army and must coordinate an attack by sending messengers to each other. Because the generals are in unfamiliar and hostile territory, messengers may fail to reach their destination (just as nodes in a distributed network may fail, or send corrupted data instead of the intended message). An additional aspect of the problem is that some of the generals may be traitors, either individually, or conspiring together, and so messages may arrive which are intended to create a false plan that is doomed to failure for the loyal generals (just as malicious members of a distributed system may attempt to convince the system to accept fraudulent transactions, or multiple versions of the same truthful transaction that would result in a double-spend). Thus a distributed payment system must be robust both in the face of standard failures, and so-called "Byzantine" failures, which may be coordinated and originate from multiple sources in the network. In this work, we analyze one particular implementation of a distributed payment system: the A Multi-Decentralized Interaction Protocol Protocol. We focus on the algorithms utilized to achieve the above goals of correctness, agreement, and utility, and show that all are met (within necessary and predetermined tolerance thresholds, which are well-understood). In addition, we provide code that simulates the consensus process with parameterizable network size, number of malicious users, and message-sending latencies. Satoshi Nakamoto’s development of Bitcoin in 2009 has often been hailed as a radical development in money and currency, being the first example of a digital asset which simultaneously has no backing or "intrinsic value" and no centralized issuer or controller. However, another, arguably more important, part of the Bitcoin experiment is the underlying blockchain technology as a tool of distributed consensus, and attention is rapidly starting to shift to this other aspect of Bitcoin. Commonly cited alternative applications of blockchain technology include using on-blockchain digital assets to represent custom currencies and financial instruments ("colored coins"), the ownership of an underlying physical device ("smart property"), non-fungible assets such as domain names ("Namecoin"), as well as more complex applications involving having digital assets being directly controlled by a piece of code implementing arbitrary rules ("smart contracts") or even 3 blockchain-based "decentralized autonomous organizations" (DAOs). What A Multi-Decentralized Interaction Protocol intends to provide is a blockchain with a built-in fully fledged Turing-complete programming language that can be used to create "contracts" that can be used to encode arbitrary state transition functions, allowing users to create any of the systems described above, as well as many others that we have not yet imagined, simply by writing up

the logic in a few lines of code. "Bitcoin" has been a successful implementation of the concept of p2p electronic cash. Both professionals and the general public have come to appreciate the convenient combination of public transactions and proof-of-work as a trust model. Today, the user base of electronic cash is growing at a steady pace; customers are attracted to low fees and the anonymity provided by electronic cash and merchants value its predicted and decentralized emission. Bitcoin has effectively proved that electronic cash can be as simple as paper money and as convenient as credit cards. Unfortunately, Bitcoin suffers from several deficiencies. For example, the system's distributed nature is inflexible, preventing the implementation of new features until almost all of the network users update their clients. Some critical flaws that cannot be fixed rapidly deter Bitcoin's widespread propagation. In such inflexible models, it is more efficient to roll-out a new project rather than perpetually fix the original project. In this paper, we study and propose solutions to the main deficiencies of Bitcoin. We believe that a system taking into account the solutions we propose will lead to a healthy competition among different electronic cash systems. We also propose our own electronic cash, "A Multi-Decentralized Interaction Protocol", a name emphasizing the next breakthrough in electronic cash. Bitcoin is the first widely used financial system for which all the necessary data to validate the system status can be cryptographically verified by anyone. However, it accomplishes this feat by storing all transactions in a public database called "the blockchain" and someone who genuinely wishes to check this state must download the whole thing and basically replay each transaction, check each one as they go. Meanwhile, most of these transactions have not affected the actual final state (they create outputs that are destroyed a transaction later). At the time of this writing, there were nearly 150 million transactions committed in the blockchain, which must be replayed to produce a set of only 4 million unspent outputs. It would be better if an auditor needed only to check data on the outputs themselves, but this is impossible because they are valid if and only if the output is at the end of a chain of previous outputs, each signs the next. In other words, the whole blockchain must be validated to confirm the final state. Add to this that these transactions are cryptographically atomic, it is clear what outputs go into every transaction and what emerges. The "transaction graph" resulting reveals a lot of information and is subjected to analysis by many companies whose business model is to monitor and control the lower classes. This makes it very non-private and even dangerous for people to use. Some solutions to this have been proposed. Greg Maxwell discovered to encrypt the amounts, so that the graph of the transaction is faceless but still allow validation that the sums are correct. Dr Maxwell also produced CoinJoin, a system for Bitcoin users to combine interactively transactions, confusing the transaction graph. Nicolas van Saberhagen has developed a system to blind the transaction entries, goes much further to cloud the transaction graph (as well as not needed the user interaction). Later, Shen Noether combined the two approaches to obtain "confidential transactions" of Maxwell AND the darkening of van Saberhagen. These solutions are very good and would make Bitcoin very safe to use. But the problem of too much data is made even worse. Confidential transactions require multi-kilobyte proofs on every output, and van Saberhagen signatures require every output to be stored for ever, since it is not possible to tell when they are truly spent. Dr. Maxwell's CoinJoin has the problem of needing interactivity. Dr. Yuan Horas Mouton fixed this by 4 making transactions freely mergeable, but he needed to use pairing-based cryptography, which is potentially slower and more difficult to trust. He called this "one-way aggregate signatures" (OWAS). OWAS had the good idea to combine the transactions in blocks. Imagine that we can combine across blocks (perhaps with some glue data) so that when the outputs are created and destroyed, it is the same as if they never existed. Then, to validate the entire chain, users only need to know when money is entered into the system (new money in each block as in

Bitcoin or Monero or peg-ins for sidechains) and final unspent outputs, the rest can be removed and forgotten. Then we can have Confidential Transactions to hide the amounts and OWAS to blur the transaction graph, and use LESS space than Bitcoin to allow users to fully verify the blockchain. And also imagine that we must not pairing-based cryptography or new hypotheses, just regular discrete logarithms signatures like Bitcoin. Here is what I propose. I call my creation A Multi-Decentralized Interaction Protocol because it is used to prevent the blockchain from talking about all user's information. History The concept of decentralized digital currency, as well as alternative applications like property registries, has been around for decades. The anonymous e-cash protocols of the 1980s and the 1990s were mostly reliant on a cryptographic primitive known as Chaumian Blinding. Chaumian Blinding provided these new currencies with high degrees of privacy, but their underlying protocols largely failed to gain traction because of their reliance on a centralized intermediary. In 1998, Wei Dai's b-money became the first proposal to introduce the idea of creating money through solving computational puzzles as well as decentralized consensus, but the proposal was scant on details as to how decentralized consensus could actually be implemented. In 2005, Hal Finney introduced a concept of "reusable proofs of work", a system which uses ideas from b-money together with Adam Back's computationally difficult Hashcash puzzles to create a concept for a cryptocurrency, but once again fell short of the ideal by relying on trusted computing as a backend. In 2009, a decentralized currency was for the first time implemented in practice by Satoshi Nakamoto, combining established primitives for managing ownership through public key cryptography with a consensus algorithm for keeping track of who owns coins, known as "proof of work." The mechanism behind proof of work was a breakthrough because it simultaneously solved two problems. First, it provided a simple and moderately effective consensus algorithm, allowing nodes in the network to collectively agree on a set of updates to the state of the Bitcoin ledger. Second, it provided a mechanism for allowing free entry into the consensus process, solving the political problem of deciding who gets to influence the consensus, while simultaneously preventing Sybil attacks. It does this by substituting a formal barrier to participation, such as the requirement to be registered as a unique entity on a particular list, with an economic barrier - the weight of a single node in the consensus voting process is directly proportional to the computing power that the node brings. Since then, an alternative approach has been proposed called proof of stake, calculating the weight of a node as being proportional to its currency holdings and not its computational resources. The discussion concerning the relative merits of the two approaches is beyond the scope of this paper but it should be noted that both approaches can be used to serve as the backbone of a cryptocurrency.

EXECUTIVE SUMMARY

ANT BLOCK CHAIN PROTOCOL - BLOCKCHAIN INVESTMENTS SAFE AND EASY

ant block chain protocol is a decentralized investment platform that focuses on blue-chip blockchain assets with unique value proposition and high performance potential. Our products and services cover the whole lifecycle of blockchain innovation: From angel & venture capital investments (AC & VC) and initial coin offerings (ICO), to cryptocurrencies and asset-backed tokens.

A DECENTRALIZED INVESTMENT PLATFORM AND EXCHANGE By

tokenizing all our funds and assets, we simplify safe investments into selected blockchain assets. All major cryptocurrencies will be interchangeable on our platform. What's revolutionary about the platform is that it solves the problem of crypto spending and allows management of crypto assets in a safe and easy way.

ASSET-BACKED GOLD TOKEN

Gold has always been the world's best hedge against all sort of market turmoil. Our first product in this product range will therefore be the Ant Block Chain Protocol Gold Coin (ABCP), which refers to exactly one gram of segregated, unallocated 999.9 fine gold from LBMA-approved refineries. There is no comparable asset out on the market at given time.

TOKENIZED FUNDS

Our tokenized funds invest into the most valuable cryptocurrencies, ICOs and ABCP. Investing into ICOs and tokenized ABCP leverages the performance of our funds compared to regular crypto investments by far. Our fund management avails of profound experience from the traditional- , as well as from the hedge fund industry.

A CRYPTOCURRENCY CARD

Our crypto debit card allows coin and token spending in fiat currencies around the world. The prepaid debit card spares local exchange fees and makes digital currency spending a whole lot easier. Ant Block Chain Protocol enables cryptocurrency spending with instant liquidity.

AN ICO INCUBATOR, A BLOCKCHAIN START UP ANGELIST

Our expertise in traditional and hedgefund management qualifies us to accompany blockchain businesses successfully spreading their wings. We launch ICO campaigns with a dedicated fund, management support, legal advice and more. At the same time ant block chain protocol shares earned profits with its company token holders via a buyback program.

VISION

The idea of blockchain has opened a whole new chapter of value transfer. Which currencies will stay in place and proof themselves successful, at which price and on base of which business model or technology is yet unclear. What we do know for certain is that cryptocurrencies and crypto assets are definitely a part of the future.

They have revolutionized payments, transactions and many other markets and will soon become established as the new standard.

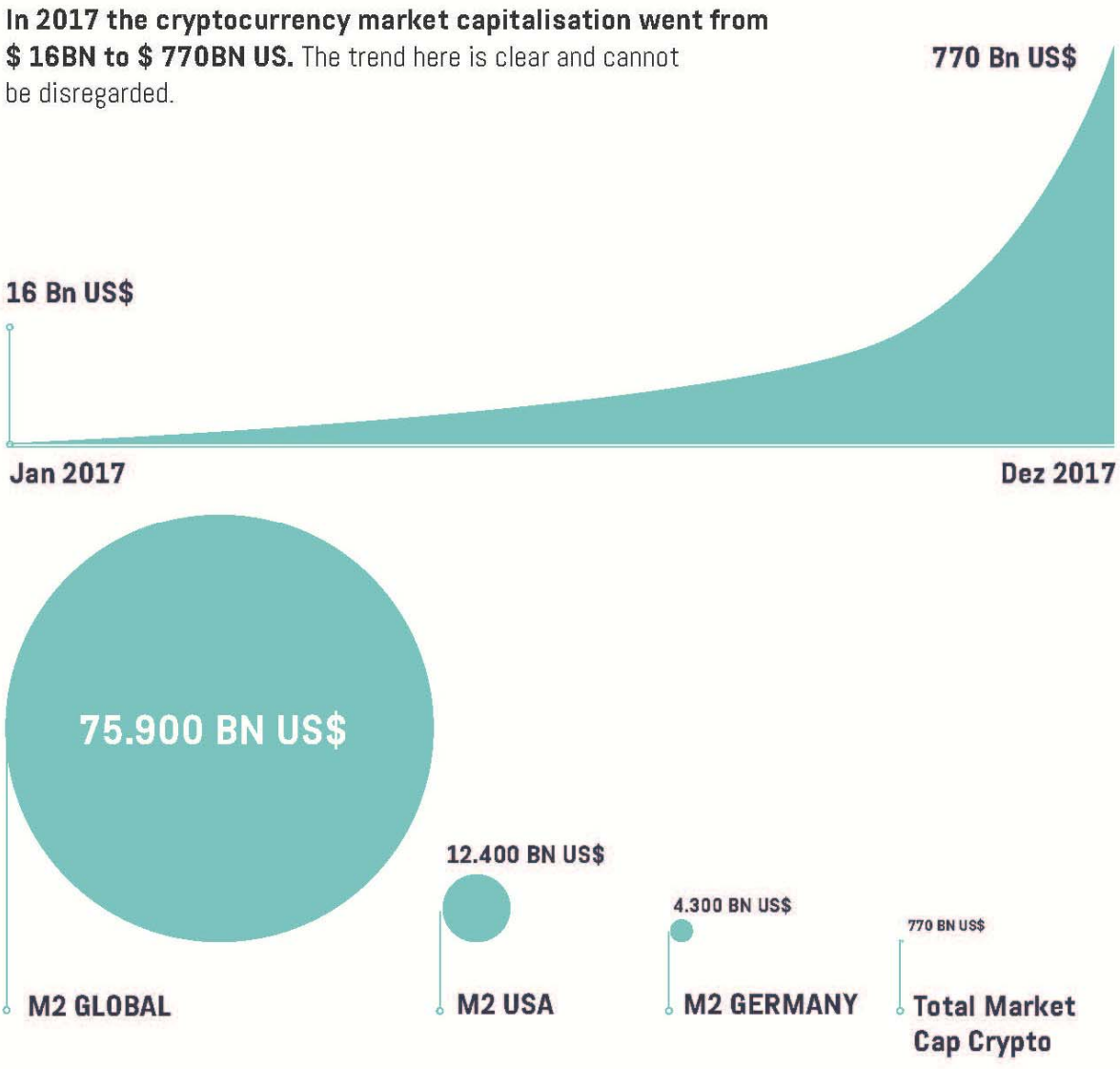
Ant Block Chain Protocol’ s mission is to open the cryptocurrency market to everyone with a variety of interesting products and services. As a trustworthy company, we encourage blockchain adoption and offer a safe and easy way to take part in the future of financial markets. We personally believe in the philosophical mission of Satoshi Nakamoto and created a safe and transparent investment solution with easy access for everyone.

WE PERSONALLY BELIEVE IN THE PHILOSOPHICAL MISSION OF SATOSHI NAKAMOTO AND CREATED A SAFE AND TRANSPARENT INVESTMENT SOLUTION WITH EASY ACCESS FOR EVERYONE.

MARKET OPPORTUNITY

Since the introduction of blockchain technology, there has been a high demand for crypto assets on the markets with likeliness of further increase. Fintech is a completely new sector in finance without any institutions yet and financial service providers have not been able to meet rising demands for Bitcoin and/or blockchain investment solutions. Finance & Investments are still heavily underrepresented as a category, due to a lack of innovation and decentralization. It is just as of recent, that CBOE and CME group launched Bitcoin futures markets.

The dawn of a new investment era began starting with the incredibly successful Ethereum ICO in July 2014. Imagine having invested an amount of two thousand dollars in ETH at that time. Right now in 2017, in less than three years, those two thousand dollars would have achieved an amount worth more than 1 million US $ by now. The same applies for blockchain solution "Stratis": Only 12 month after its ICO, tokenholders have multiplied their initial investment by factor 600.

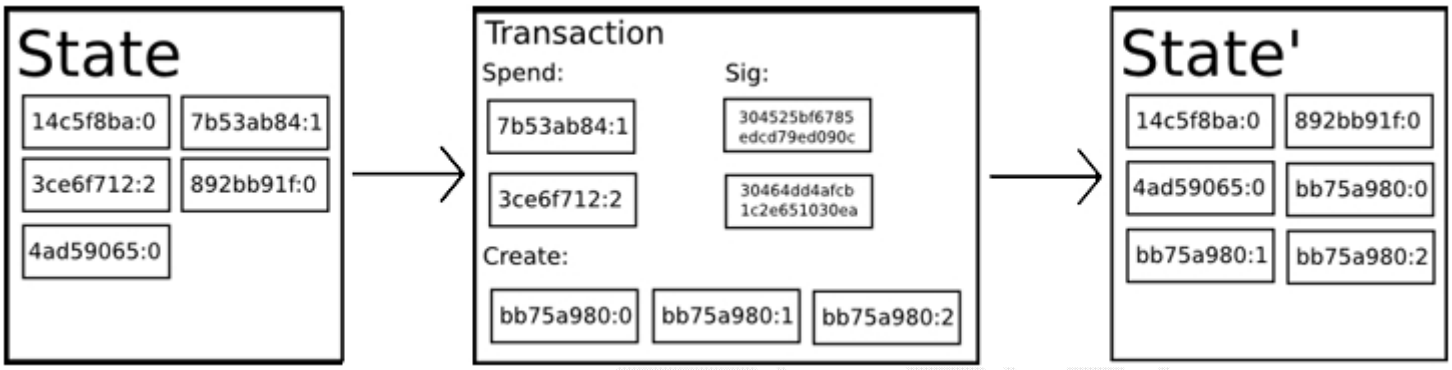


Nevertheless, there are a load of obstacles to be challenged before adapting cryptocurrency usage on a broad scale. It is hard for non-tech-savvy users to easily enter the crypto markets. There are no institutional guidelines or trustworthy partners to rely on. And of course as the cryptocurrency market is a very young and new sector in finance, it comes with rather high volatility.

MARKET OPPORTUNITY

- We provide transparency in a complex market
- We are constantly adapting and refining our sustainable business model
- We offer a functioning set of working products
- We provide highest safety standards through cold storage
- We offer all transactions via one platform
- We offer a scalable low cost platform solution
- We combine constant advancement with a lean cost structure
- We can rely on a strong community of more than 50.000 enthusiasts

Bitcoin As A State Transition System



From a technical standpoint, the ledger of a cryptocurrency such as Bitcoin can be thought of as a state transition system, where there is a "state" consisting of the ownership status of all existing bitcoins and a "state transition function" that takes a state and a transaction and outputs a new state which is the result. In a standard banking system, for example, the state is a balance sheet, a transaction is a request to move $X from A to B, and the state transition function reduces the value of A's account by $X and increases the value of B's account by $X. If A's account has less than $X in the first place, the state transition function returns an error. Hence, one can formally define:

```
APPLY(S,TX) -> S' or ERROR
```

In the banking system defined above:

```
APPLY({ Alice: $50, Bob: $50 },"send $20 from Alice to Bob") = { Alice: $30, Bob: $70 }
```

But:

```
APPLY({ Alice: $50, Bob: $50 },"send $70 from Alice to Bob") = ERROR
```

The "state" in Bitcoin is the collection of all coins (technically, "unspent transaction outputs" or UTXO) that have been minted and not yet spent, with each UTXO having a denomination and an owner (defined by a 20-byte address which is essentially a cryptographic public key). A transaction contains one or more inputs, with each input containing a reference to an existing UTXO and a cryptographic signature produced by the private key associated with the owner's address, and one or more outputs, with each output containing a new UTXO for addition to the state.

The state transition function APPLY(S,TX) -> S' can be defined roughly as follows:

1. For each input in TX: If the referenced UTXO is not in S, return an error. If the provided signature does not match the owner of the UTXO, return an error. 2. If the sum of the denominations of all input UTXO is less than the sum of the denominations of all output UTXO, return an error. 3. Return S' with all input UTXO removed and all output UTXO added. The first half of the first step

prevents transaction senders from spending coins that do not exist, the second 6 half of the first step prevents transaction senders from spending other people's coins, and the second step enforces conservation of value. In order to use this for payment, the protocol is as follows. Suppose Alice wants to send 11.7 BTC to Bob. First, Alice will look for a set of available UTXO that she owns that totals up to at least 11.7 BTC. Realistically, Alice will not be able to get exactly 11.7 BTC; say that the smallest she can get is 6+4+2=12. She then creates a transaction with those three inputs and two outputs. The first output will be 11.7 BTC with Bob's address as its owner, and the second output will be the remaining 0.3 BTC "change". If Alice does not claim this change by sending it to an address owned by herself, the miner will be able to claim it

Definitions, Formalization and Previous Work

2. We begin by defining the components of the A Multi-Decentralized Interaction Protocol Protocol. In order to prove correctness, agreement, and utility properties, we first formalize those properties into axioms. These properties, when grouped together, form the notion of consensus: the state in which nodes in the network reach correct agreement. We then highlight some previous results relating to consensus algorithms, and finally state the goals of consensus for the A Multi-Decentralized Interaction Protocol Protocol within our formalization framework.

A Multi-Decentralized Interaction Protocol Protocol Components

3. We begin our description of the A Multi-Decentralized Interaction Protocol network by defining the following terms:

Server: A server is any entity running the A Multi-Decentralized Interaction Protocol Server software (as opposed to the A Multi-Decentralized Interaction Protocol Client software which only lets a user send and receive funds), which participates in the consensus process.

Ledger: The ledger is a record of the amount of currency in each userâ€™s account and represents the ″ground truth″ of the network. The ledger is repeatedly updated with transactions that successfully pass through the consensus process.

Open Ledger: The open ledger is the current operating status of a node (each node maintains its own open ledger). Transactions initiated by end users of a 7 given server are applied to the open ledger of that server, but transactions are not considered final until they have passed

through the consensus process, at which point the open ledger becomes the last-closed ledger.

Unique Node List (UNL): Each server, s, main- tains a unique node list, which is a set of other servers that s queries when determining consensus. Only the votes of the other members of the UNL of s are considered when determining consensus (as opposed to every node on the network). Thus the UNL represents a subset of the network which when taken collectively, is "trusted" by s to not collude in an attempt to defraud the network. Note that this definition of "trust" does not require that each individual member of the UNL be trusted.

Proposer: Any server can broadcast transactions to be included in the consensus process, and every server attempts to include every valid transaction when a new consensus round starts. During the consensus process, however, only proposals from servers on the UNL of a server s are considered by s.

Formalization

4. We use the term nonfaulty to refer to nodes in the net- work that behave honestly and without error. Conversely, a faulty node is one which experiences errors which may be honest (due to data corruption, implementation errors, etc.), or malicious (Byzantine errors). We reduce the notion of validating a transaction to a simple binary decision problem: each node must decide from the information it has been given on the value 0 or 1.

5. As in Attiya, Dolev, and Gill, 1984, we define consensus according to the following three axioms:

(C1): Every nonfaulty node makes a decision infinite time

(C2): All nonfaulty nodes reach the same decision value

(C3): 0 and 1 are both possible values for all non-faulty nodes. (This removes the trivial solution in which all nodes decide 0 or 1 regardless of the information they have been presented).

Existing Consensus Algorithms

6. There has been much research done on algorithms that achieve consensus in the face of Byzantine errors. This previous work has included extensions to cases where all participants in the network are not known ahead of time, where the messages are sent asynchronously (there is no bound on the amount of time an individual node will take to reach a decision), and where there is a

delineation between the notion of strong and weak consensus.

7. One pertinent result of previous work on consensus algorithms is that of Fischer, Lynch, and Patterson, 1985, which proves that in the asynchronous case, non-termination is always a possibility for a consensus algorithm, even with just one faulty process. This introduces the necessity for time-based heuristics, to ensure convergence (or at least repeated iterations of non-convergence). We shall describe these heuristics for the A Multi-Decentralized Interaction Protocol Protocol later.

8. The strength of a consensus algorithm is usually measured in terms of the fraction of faulty processes it can tolerate. It is provable that no solution to the Byzantine Generals problem (which already assumes synchronicity, and known participants) can tolerate more than (n - 1)/3 byzantine faults, or 33% of the network acting maliciously. This solution does not, however, require verifiable authenticity of the messages delivered between nodes (digital signatures). If a guarantee on the unforgeability of messages is possible, algorithms exist with much higher fault tolerance in the synchronous case.

9. Several algorithms with greater complexity have been proposed for Byzantine consensus in the asynchronous case. FaB Paxos will tolerate (n - 1)/5 Byzantine failures in a network of n nodes, amounting to a tolerance of up to 20% of nodes in the network colluding maliciously. Attiya, Doyev, and Gill introduce a phase algorithm for the asynchronous case, which can tolerate (n - 1)/4 failures, or up to 25% of the network. Lastly, Alchieri et al., 2008 present BFT-CUP, which achieves Byzantine consensus in the asynchronous case even with unknown participants, with the maximal bound of a tolerance of (n - 1)/3 failures, but with additional restrictions on the connectivity of the underlying network.

Formal Consensus Goals

10. Our goal in this work is to show that the consensus algorithm utilized by the A Multi-Decentralized Interaction Protocol Protocol will achieve consensus at each ledger-close (even if consensus is the trivial consensus of all transactions being rejected), and that the trivial consensus will only be reached with a known probability, even in the face of Byzantine failures. Our goal in this work is to show that the consensus algorithm utilized by the A Multi-Decentralized Interaction Protocol Protocol will achieve consensus at each ledger-close (even if consensus is the trivial consensus of all transactions being rejected), and that the trivial consensus will only be reached with a known probability, even in the face of Byzantine failures.

11. Lastly we will show that the A Multi-Decentralized Interaction Protocol Protocol can achieve these goals in the face of (n-1)/5 failures, which is not the strongest result in the literature, but we will also show that the A Multi-Decentralized Interaction Protocol Protocol possesses several other desirable features that greatly enhance its utility.

Irregular emission

12. Bitcoin has a predetermined emission rate: each solved block produces a fixed amount of coins. Approximately every four years this reward is halved. The original intention was to create a limited smooth emission with exponential decay, but in fact we have a piecewise linear emission function whose breakpoints may cause problems to the Bitcoin infrastructure

13. When the breakpoint occurs, miners start to receive only half of the value of their previous reward. The absolute difference between 12.5 and 6.25 BTC (projected for the year 2020) may seem tolerable. However, 9 when examining the 50 to 25 BTC drop that took place on November 28 2012, felt inappropriate for a significant number of members of the mining community. Figure 1 shows a dramatic decrease in the network's hashrate in the end of November, exactly when the halving took place. This event could have been the perfect moment for the malevolent individual described in the proof-of-work function section to carry-out a double spending attack.

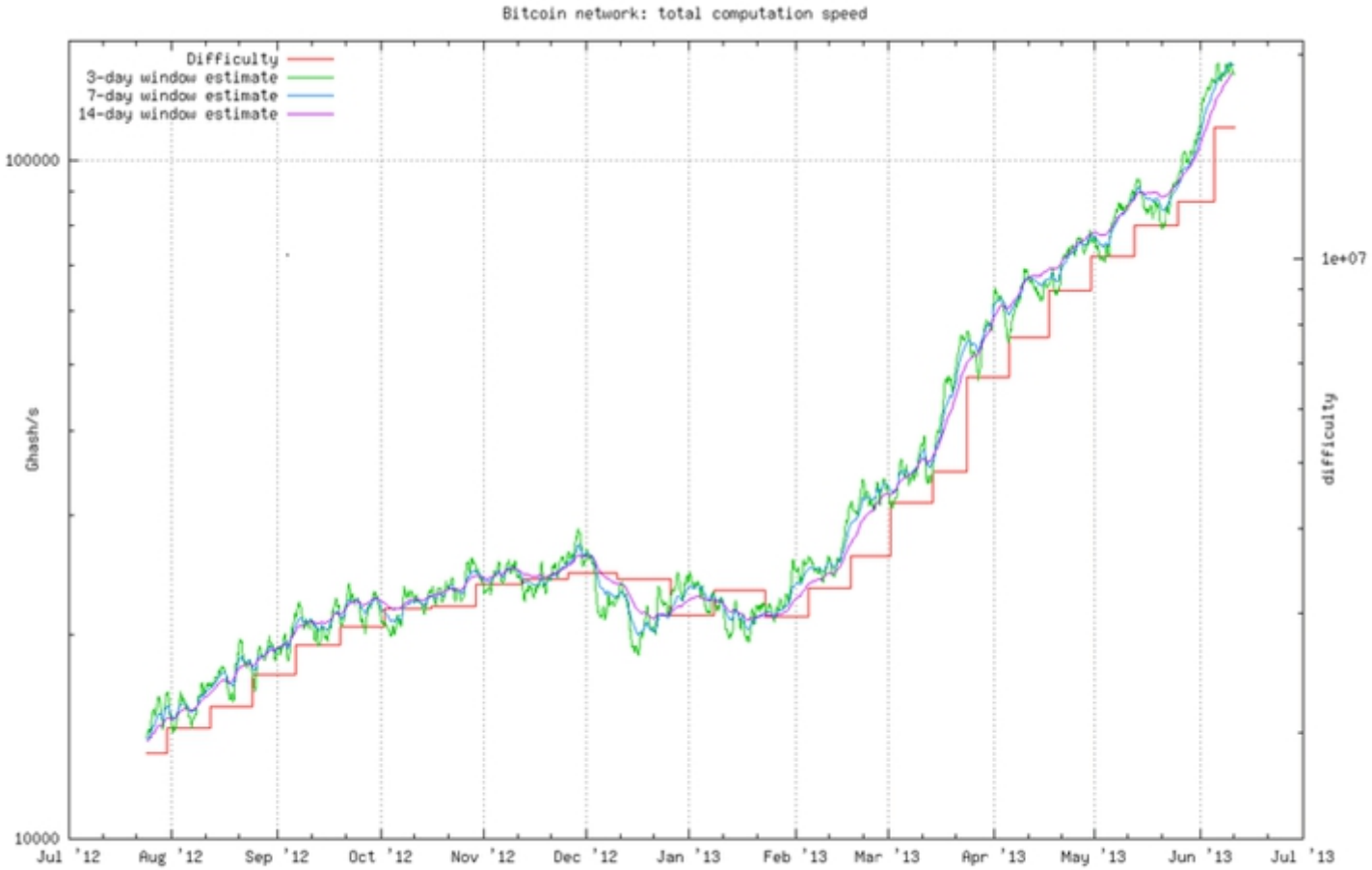


Fig. 1. Bitcoin hashrate chart
(source: http://bitcoin.sipa.be)

Hardcoded constants

14. Bitcoin has many hard-coded limits, where some are natural elements of the original design (e.g. block frequency, maximum amount of money supply, number of confirmations) whereas other seem to be artificial constraints. It is not so much the limits, as the inability of quickly changing them if necessary that causes the main drawbacks. Unfortunately, it is hard to predict when the constants may need to be changed and replacing them may lead to terrible consequences.

15. A good example of a hardcoded limit change leading to disastrous consequences is the block size limit set to 250kb1. This limit was sufficient to hold about 10000 standard transactions. In early 2013, this limit had almost been reached and an agreement was reached to increase the limit. The change was implemented in wallet version 0.8 and ended with a 24-blocks chain split and a successful double-spend attack. While the bug was not in the Bitcoin protocol, but rather in the database engine it could have been easily caught by a simple stress test if there was no artificially introduced block size limit.

16. Constants also act as a form of centralization point. Despite the peer-to-peer nature of Bitcoin, an 10 overwhelming majority of nodes use the official reference client developed by a small group of people. This group makes the decision to implement changes to the protocol and most people accept these changes irrespective of their "correctness". Some decisions caused heated discussions and even calls for boycott, which indicates that the community and the developers may disagree on some important points. It therefore seems logical to have a protocol with user-configurable and self-adjusting variables as a possible way to avoid these problems.

Bulky scripts

The scripting system in Bitcoin is a heavy and complex feature. It potentially allows one to create sophisticated transactions, but some of its features are disabled due to security concerns and some have never even been used. The script (including both senders' and receivers' parts) for the most popular transaction in Bitcoin looks like this:

```
<sig> <pubKey> OP DUP OP HASH160 <pubKeyHash> OP EQUALVERIFY OP CHECKSIG
```

The script is 164 bytes long whereas its only purpose is to check if the receiver possess the secret key required to verify his signature.

The A Multi-Decentralized Interaction Protocol Technology

Now that we have covered the limitations of the Bitcoin technology, we will concentrate on presenting the features of A Multi-Decentralized Interaction Protocol.

A Multi-Decentralized Interaction Protocol Consensus Algorithm

The A Multi-Decentralized Interaction Protocol Protocol consensus algorithm, is applied every few seconds by all nodes, in order to maintain the correctness and agreement of the network. Once consensus is reached, the current ledger is considered "closed" and becomes the last-closed ledger. Assuming that the consensus algorithm is successful, and that there is no fork in the network, the last-closed ledger maintained by all nodes in the network will be identical.

Elliptic curve parameters

As our base signature algorithm we chose to use the fast scheme EdDSA, which is developed and implemented by D.J. Bernstein et al. Like Bitcoin's ECDSA it is based on the elliptic curve discrete logarithm problem, so our scheme could also be applied to Bitcoin in future. Common parameters are:

q: a prime number; $q = 2^{255} - 19$;

d: an element of $\mathbb{F}_q$; $d = -121665/121666$;

E: an elliptic curve equation; $-x^2 + y^2 = 1 + dx^2y^2$;

G: a base point; $G = (x, -4/5)$;

l: a prime order of the base point; $l = 2^{252} + 27742317777372353535851937790883648493$;

$\mathcal{H}_s$: a cryptographic hash function $\{0,1\}^* \to \mathbb{F}_q$;

$\mathcal{H}_p$: a deterministic hash function $E(\mathbb{F}_q) \to E(\mathbb{F}_q)$.

Terminology

Enhanced privacy requires a new terminology which should not be confused with Bitcoin entities.

private ec-key is a standard elliptic curve private key: a number $a \in [1, l-1]$;

public ec-key is a standard elliptic curve public key: a point $A = aG$;

one-time keypair is a pair of private and public ec-keys;

private user key is a pair (a, b) of two different private ec-keys;

tracking key is a pair (a, B) of private and public ec-key (where $B = bG$ and $a \neq b$);

public user key is a pair (A, B) of two public ec-keys derived from (a, b);

standard address is a representation of a public user key given into human friendly string with error correction;

truncated address is a representation of the second half (point B) of a public user key given into human friendly string with error correction.

The transaction structure remains similar to the structure in Bitcoin: every user can choose several independent incoming payments (transactions outputs), sign them with the corresponding private keys and send them to different destinations. Contrary to Bitcoin's model, where a user possesses unique private and public key, in the proposed model a sender generates a one-time public key based on the recipient's address and some random data. In this sense, an incoming transaction for the same recipient is sent to a one-time public key (not directly to a unique address) and only the recipient can recover the corresponding private part to redeem his funds (using his unique private key). The recipient can spend the funds using a ring signature, keeping his ownership and actual spending anonymous. The details of the protocol are explained in the next subsections

Unlinkable payments

Classic Bitcoin addresses, once being published, become unambiguous identifier for incoming payments, linking them together and tying to the recipient's pseudonyms. If someone wants to receive an "untied" transaction, he should convey his address to the sender by a private channel. If he wants to receive different transactions which cannot be proven to belong to the same owner he should generate all the different addresses and never publish them in his own pseudonym.

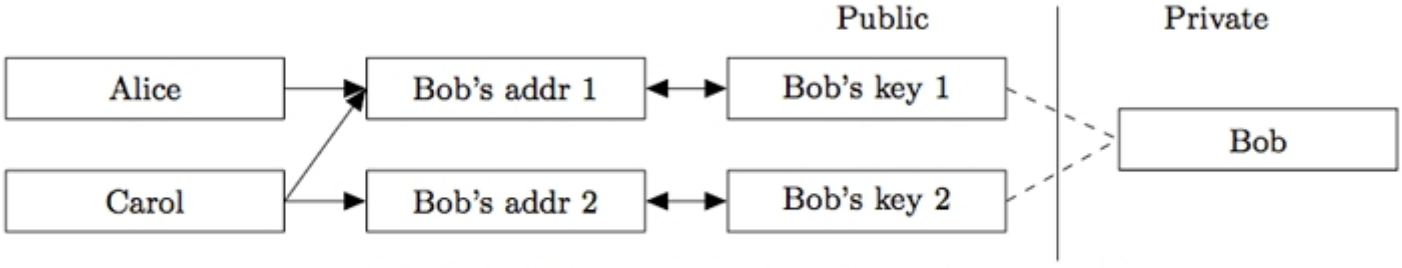


Fig. 2. Traditional Bitcoin keys/transactions model.

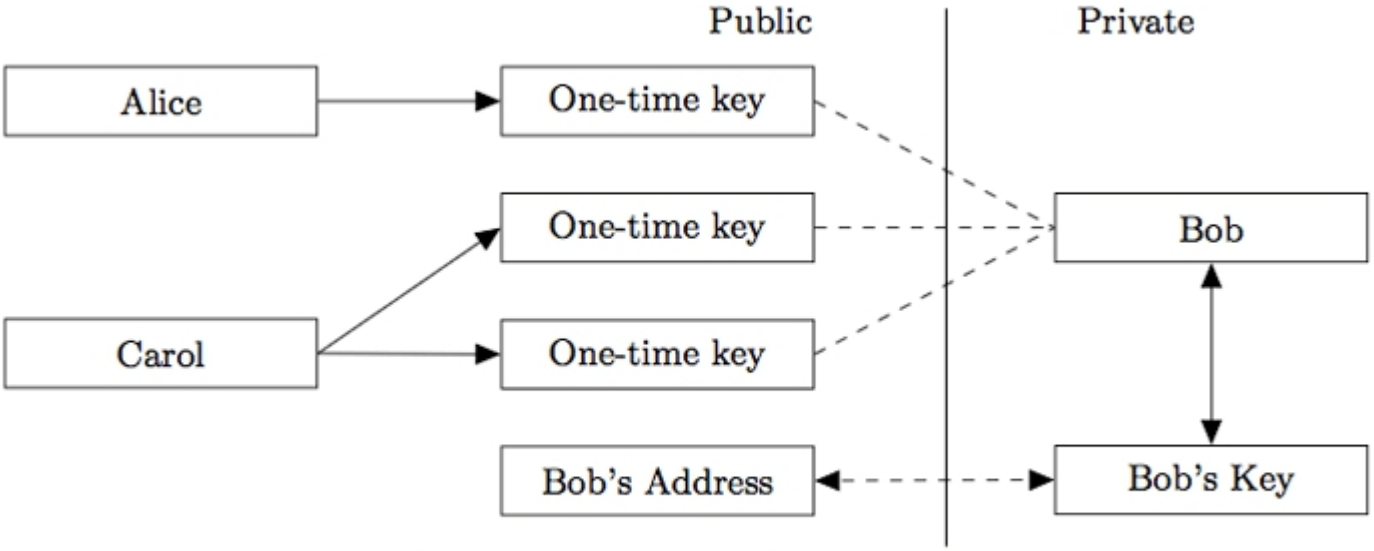


Fig. 3. CryptoNote keys/transactions model.

We propose a solution allowing a user to publish a single address and receive unconditional unlinkable payments. The destination of each A Multi-Decentralized Interaction Protocol output (by default) is a public key, derived from recipient's address and sender's random data. The main advantage against Bitcoin is that every destination key is unique by default (unless the sender uses the same data for each of his transactions to the same recipient). Hence, there is no such issue as "address reuse" by design and no observer can determine if any transactions were sent to a specific address or link two addresses together.

First, the sender performs a Diffie-Hellman exchange to get a shared secret from his data and half of the recipient's address. Then he computes a one-time destination key, using the shared secret and the second half of the address. Two different ec-keys are required from the recipient for these two steps, so a standard A Multi-Decentralized Interaction Protocol address is nearly twice as large as a Bitcoin wallet address. The receiver also performs a Diffie-Hellman exchange to recover the corresponding secret key.

A standard transaction sequence goes as follows:

1. Alice wants to send a payment to Bob, who has published his standard address. She unpacks the address and gets Bob's public key (A, B).
2. Alice generates a random $r \in [1, l-1]$ and computes a one-time public key $P = \mathcal{H}_s(rA)G + B$.
3. Alice uses P as a destination key for the output and also packs value $R = rG$ (as a part of the Diffie-Hellman exchange) somewhere into the transaction. Note that she can create other outputs with unique public keys: different recipients' keys (A_i, B_i) imply different P_i even with the same r.

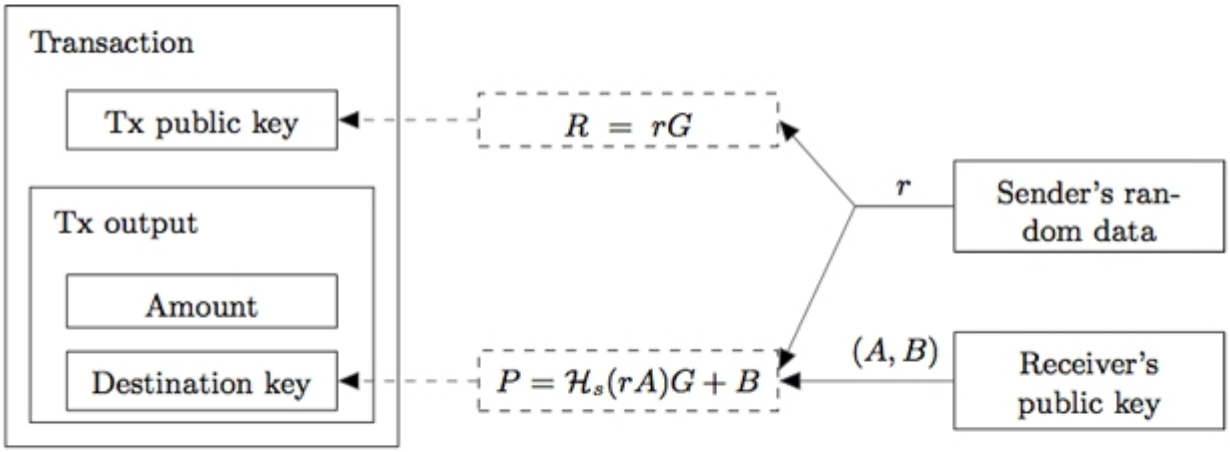


Fig. 4. Standard transaction structure.

4. Alice sends the transaction.
5. Bob checks every passing transaction with his private key (a, b), and computes $P' = \mathcal{H}_s(aR)G + B$. If Alice's transaction for with Bob as the recipient was among them, then $aR = arG = rA$ and $P' = P$.
6. Bob can recover the corresponding one-time private key: $x = \mathcal{H}_s(aR) + b$, so as $P = xG$. He can spend this output at any time by signing a transaction with x.

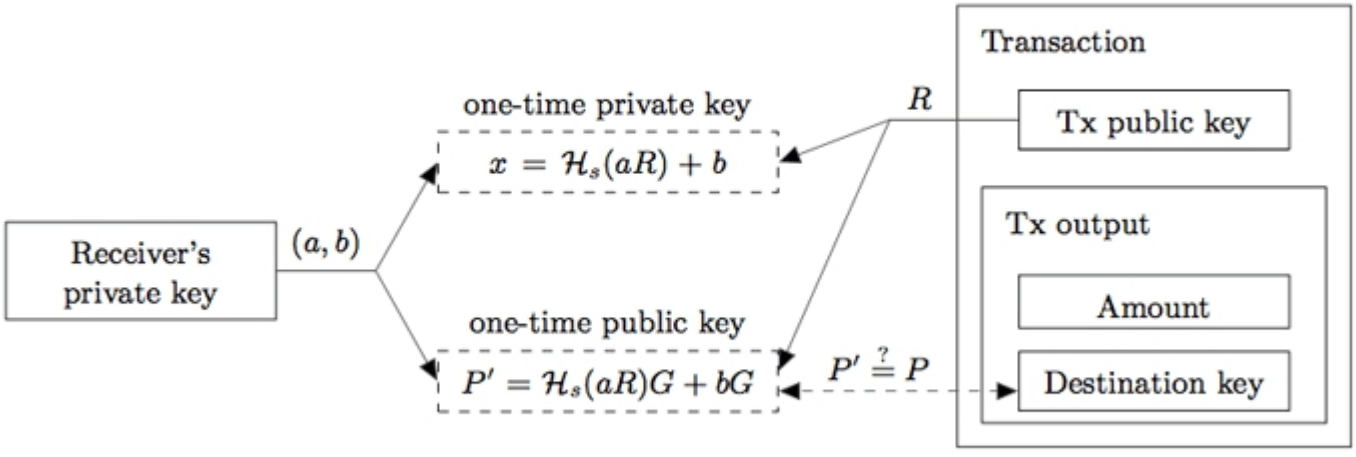


Fig. 5. Incoming transaction check.

As a result Bob gets incoming payments, associated with one-time public keys which are unlinkable for aspectator. Some additional notes:

When Bob "recognizes" his transactions (see step 5) he practically uses only half of his private information: (a, B). This pair, also known as the tracking key, can be passed to a third party (Carol). Bob can delegate her the processing of new transactions. Bob doesn't need to explicitly trust Carol, because she can't recover the one-time secret key p without Bob's full private key (a, b). This approach is useful when Bob lacks bandwidth or computation power (smartphones, hardware wallets etc.).

In case Alice wants to prove she sent a transaction to Bob's address she can either disclose r or use any kind of zero-knowledge protocol to prove she knows r (for example by signing the transaction with r).

If Bob wants to have an audit compatible address where all incoming transaction are linkable, he can either publish his tracking key or use a truncated address. That address represent only one public ec-key B, and the remaining part required by the protocol is derived from it as follows: a = Hs(B) and A = Hs(B)G. In both cases every person is able to "recognize" all of Bob's incoming transaction, but, of course, none can spend the funds enclosed within them without the secret key b.

One-time ring signatures

A protocol based on one-time ring signatures allows users to achieve unconditional unlinkability. Unfortunately, ordinary types of cryptographic signatures permit to trace transactions to their respective senders and receivers. Our solution to this deficiency lies in using a different signature type than those currently used in electronic cash systems. We will first provide a general description of our algorithm with no explicit reference to electronic cash.

A one-time ring signature contains four algorithms: (GEN, SIG, VER, LNK):

GEN: takes public parameters and outputs an ec-pair (P, x) and a public key I.

SIG: takes a message m, a set $\mathcal{S}'$ of public keys $\{P_i\}_{i \neq s}$, a pair (P_s, x_s) and outputs a signature σ and a set $\mathcal{S} = \mathcal{S}' \cup \{P_s\}$.

VER: takes a message m, a set $\mathcal{S}$, a signature σ and outputs "true" or "false".

LNK: takes a set $\mathcal{I} = \{I_i\}$, a signature σ and outputs "linked" or "indep".

The idea behind the protocol is fairly simple: a user produces a signature which can be checked by a set of public keys rather than a unique public key. The identity of the signer is indistinguishable from the other users whose public keys are in the set until the owner produces a second signature using the same keypair.

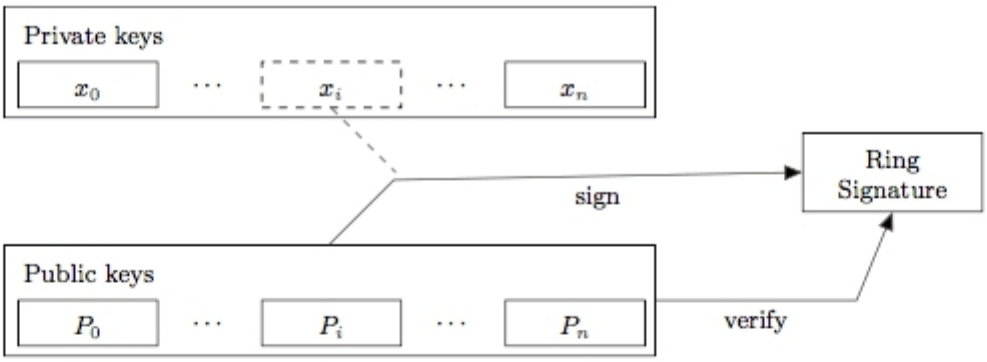


Fig. 6. Ring signature anonymity.

GEN: The signer picks a random secret key $x \in [1, l-1]$ and computes the corresponding public key $P = xG$. Additionally he computes another public key $I = x\mathcal{H}_p(P)$ which we will call the "key image".

SIG: The signer generates a one-time ring signature with a non-interactive zero-knowledge proof using the techniques from [21]. He selects a random subset $\mathcal{S}'$ of n from the other users' public keys P_i, his own keypair (x, P) and key image I. Let $0 \leq s \leq n$ be signer's secret index in $\mathcal{S}$ (so that his public key is P_s).

He picks a random $\{q_i \mid i = 0 \ldots n\}$ and $\{w_i \mid i = 0 \ldots n, i \neq s\}$ from $(1 \ldots l)$ and applies the following *transformations*:

$$L_i = \begin{cases} q_i G, & \text{if } i = s \\ q_i G + w_i P_i, & \text{if } i \neq s \end{cases}$$

$$R_i = \begin{cases} q_i \mathcal{H}_p(P_i), & \text{if } i = s \\ q_i \mathcal{H}_p(P_i) + w_i I, & \text{if } i \neq s \end{cases}$$

The next step is getting the non-interactive *challenge*:

$$c = \mathcal{H}_s(m, L_1, \ldots, L_n, R_1, \ldots, R_n)$$

Finally the signer computes the *response*:

$$c_i = \begin{cases} w_i, & \text{if } i \neq s \\ c - \sum_{i=0}^{n} c_i \mod l, & \text{if } i = s \end{cases}$$

$$r_i = \begin{cases} q_i, & \text{if } i \neq s \\ q_s - c_s x \mod l, & \text{if } i = s \end{cases}$$

The resulting signature is $\sigma = (I, c_1, \ldots, c_n, r_1, \ldots, r_n)$.

VER: The verifier checks the signature by applying the inverse transformations:

$$\begin{cases} L'_i = r_i G + c_i P_i \\ R'_i = r_i \mathcal{H}_p(P_i) + c_i I \end{cases}$$

Finally, the verifier checks if $\sum_{i=0}^{n} c_i \stackrel{?}{=} \mathcal{H}_s(m, L'_0, \ldots, L'_n, R'_0, \ldots, R'_n) \mod l$

If this equality is correct, the verifier runs the algorithm **LNK**. Otherwise the verifier rejects the signature.

LNK: The verifier checks if I has been used in past signatures (these values are stored in the set $\mathcal{I}$). Multiple uses imply that two signatures were produced under the same secret key.

The meaning of the protocol: by applying L-transformations the signer proves that he knows such x that at least one $P_i = xG$. To make this proof non-repeatable we introduce the key image as $I = x\mathcal{H}_p(P)$. The signer uses the same coefficients (r_i, c_i) to prove almost the same statement: he knows such x that at least one $\mathcal{H}_p(P_i) = I \cdot x^{-1}$.

If the mapping $x \to I$ is an injection:

1. Nobody can recover the public key from the key image and identify the signer;
2. The signer cannot make two signatures with different I's and the same x.

Standard A Multi-Decentralized Interaction Protocol transaction

By combining both methods (unlinkable public keys and untraceable ring signature) Bob achieves new level of privacy in comparison with the original Bitcoin scheme. It requires him to store only one private key (a, b) and publish (A, B) to start receiving and sending anonymous transactions.

While validating each transaction Bob additionally performs only two elliptic curve multiplications and one addition per output to check if a transaction belongs to him. For his every output Bob recovers a one-time keypair (pi,Pi) and stores it in his wallet. Any inputs can be circumstantially proved to have the same owner only if they appear in a single transaction. In fact this relationship is much harder to establish due to the one-time ring signature.

With a ring signature Bob can effectively hide every input among somebody else's; all possible spenders will be equiprobable, even the previous owner (Alice) has no more information than any observer.

When signing his transaction Bob specifies n foreign outputs with the same amount as his output, mixing all of them without the participation of other users. Bob himself (as well as anybody else) does not know if any of these payments have been spent: an output can be used in thousands of signatures as an ambiguity factor and never as a target of hiding. The double spend check occurs in the LNK phase when checking against the used key images set.

Bob can choose the ambiguity degree on his own: n = 1 means that the probability he has spent the output is 50% probability, n = 99 gives 1%. The size of the resulting signature increases linearly as O(n + 1), so the improved anonymity costs to Bob extra transaction fees. He also can set n = 0 and make his ring signature to consist of only one element, however this will instantly reveal him as a spender.

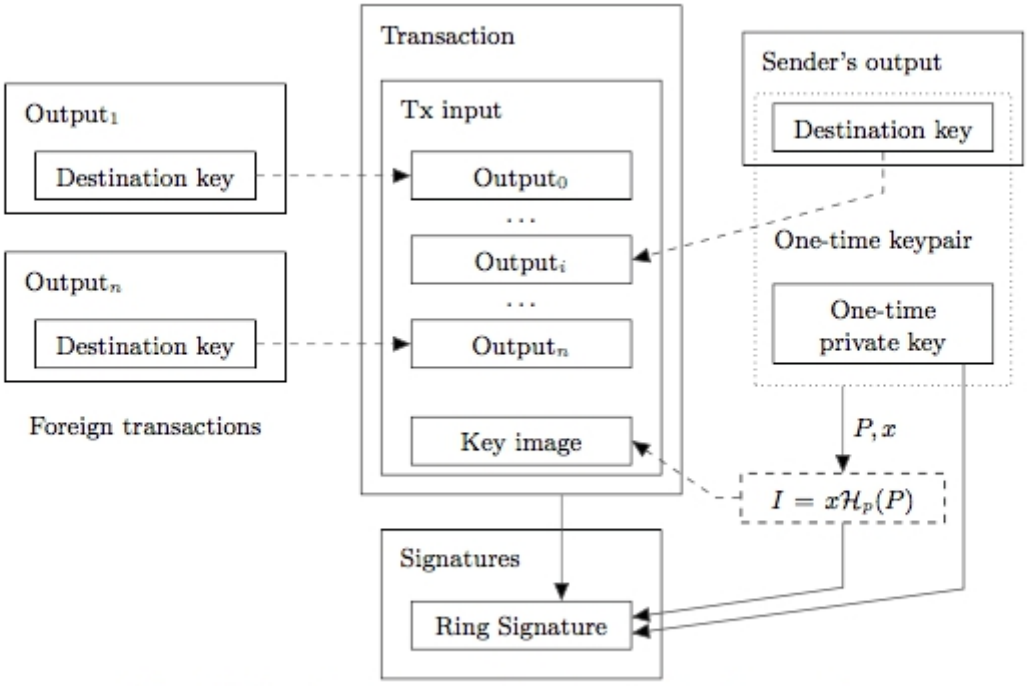


Fig. 7. Ring signature generation in a standard transaction.

Egalitarian Proof-of-work

In this section we propose and ground the new proof-of-work algorithm. Our primary goal is to close the gap between CPU (majority) and GPU/FPGA/ASIC (minority) miners. It is appropriate that some users can have a certain advantage over others, but their investments should grow at least linearly with the power. More generally, producing special-purpose devices has to be as less profitable as possible.

Definition

The RPCA proceeds in rounds. In each round:

Initially, each server takes all valid transactions it has seen prior to the beginning of the consensus round that have not already been applied (these may include new transactions initiated by endusers of the server, transactions held over from a previous consensus process, etc.), and makes them public in the form of a list known as the "candidate set".

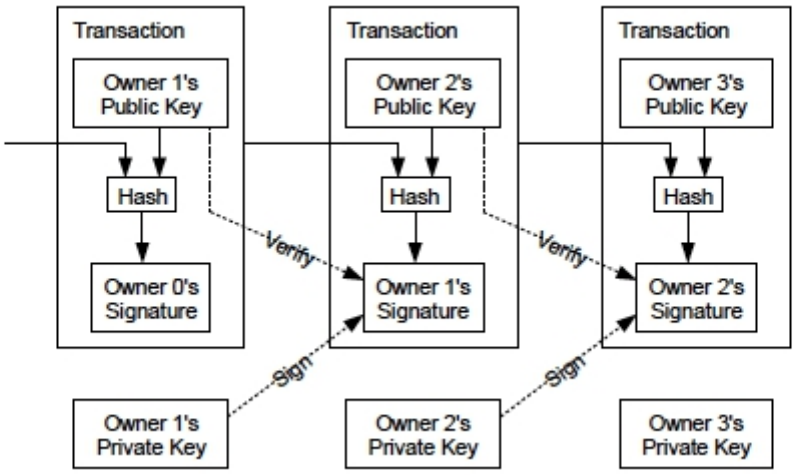
Each server then amalgamates the candidate sets of all servers on its UNL, and votes on the veracity of all transactions.

Transactions that receive more than a minimum percentage of "yes" votes are passed on to the next round, if there is one, while transactions that do not receive enough votes will either be discarded, or included in the candidate set for the beginning of the consensus process on the next ledger.

The final round of consensus requires a minimum percentage of 80% of a server's UNL agreeing on a transaction. All transactions that meet this requirement are applied to the ledger, and that ledger is closed, becoming the new last-closed ledger.

Transactions

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



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

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

Confidential Transactions and OWAS

The first thing we need to do is remove Bitcoin Script. This is sad, but it is too powerful so it is impossible to merge transactions using general scripts. We will demonstrate that confidential

transactions of Dr. Maxwell are enough (after some small modification) to authorize spending of outputs and also allows to make combined transactions without interaction. This is in fact identical to OWAS, and allows relaying nodes take some transaction fee or the recipient to change the transaction fees. These additional things Bitcoin can not do, we get for free. 19 We start by reminding the reader how confidential transactions work. First, the amounts are coded by the following equation:

```
C = r*G + v*H
```

where C is a Pedersen commitment, G and H are fixed nothing-up-my-sleeve elliptic curve group generators, v is the amount, and r is a secret random blinding key.

Attached to this output is a rangeproof which proves that v is in [0, 2^64], so that user cannot exploit the blinding to produce overflow attacks, etc.

To validate a transaction, the verifier will add commitments for all outputs, plus f*H (f here is the transaction fee which is given explicitly) and subtracts all input commitments. The result must be 0, which proves that no amount was created or destroyed overall.

We note that to create such a transaction, the user must know the sum of all the values of r for commitments entries. Therefore, the r-values (and their sums) act as secret keys. If we can make the r output values known only to the recipient, then we have an authentication system! Unfortunately, if we keep the rule that commits all add to 0, this is impossible, because the sender knows the sum of all _his_ r values, and therefore knows the receipient's r values sum to the negative of that. So instead, we allow the transaction to sum to a nonzero value k*G, and require a signature of an empty string with this as key, to prove its amount component is zero. We let transactions have as many k*G values as they want, each with a signature, and sum them during verification.

To create transactions sender and recipient do following ritual:

1.Sender and recipient agree on amount to be sent. Call this b.

2.Sender creates transaction with all inputs and change output(s), and gives recipient the total blinding factor (r-value of change minus r-values of inputs) along with this transaction. So the commitments sum to r*G - b*H.

3. Recipient chooses random r-values for his outputs, and values that sum to b minus fee, and adds these to transaction (including range proof). Now the commitments sum to k*G - fee*H for some k that only recipient knows.

4. Recipient attaches signature with k to the transaction, and the explicit fee. It has done.

Now, creating transactions in this manner supports OWAS already. To show this, suppose we have two transactions that have a surplus k1*G and k2*G, and the attached signatures with these. Then you can combine the lists of inputs and outputs of the two transactions, with both k1*G and k2*G to

the mix, and voilÃ¡! is again a valid transaction. From the combination, it is impossible to say which outputs or inputs are from which original transaction.

Because of this, we change our block format from Bitcoin to this information:

1. Explicit amounts for new money (block subsidy or sidechain peg-ins) with whatever else data this needs. For a sidechain peg-in maybe it references a Bitcoin transaction that commits to a specific excess k*G value?

2. Inputs of all transactions

3. Outputs of all transactions

4. Excess k*G values for all transactions

Each of these are grouped together because it do not matter what the transaction boundaries are originally. In addition, Lists 2 3 and 4 should be required to be coded in alphabetical order, since it is quick to check and prevents the block creator of leaking any information about the original transactions.

Note that the outputs are now identified by their hash, and not by their position in a transaction that could easily change. Therefore, it should be banned to have two unspent outputs are equal at the same time, to avoid confusion.

Traceability of Transactions

Privacy and anonymity are the most important aspects of electronic cash. Peer-to-peer payments seek to be concealed from third party's view, a distinct difference when compared with traditional banking. In particular, T. Okamoto and K. Ohta described six criteria of ideal electronic cash, which included "privacy: relationship between the user and his purchases must be untraceable by anyone." From their description, we derived two properties which a fully anonymous electronic cash model must satisfy in order to comply with the requirements outlined by Okamoto and Ohta:

Untraceability: for each incoming transaction all possible senders are equiprobable.

Unlinkability: for any two outgoing transactions it is impossible to prove they were sent to the same person.

Unfortunately, Bitcoin does not satisfy the untraceability requirement. Since all the transactions that take place between the network's participants are public, any transaction can be unambiguously traced to a unique origin and final recipient. Even if two participants exchange funds in an indirect way, a properly engineered path-finding method will reveal the origin and final recipient

It is also suspected that Bitcoin does not satisfy the second property. Some researchers stated that a careful blockchain analysis may reveal a connection between the users of the Bitcoin network and

their transactions. Although a number of methods are disputed, it is suspected that a lot of hidden personal information can be extracted from the public database.

Bitcoin's failure to satisfy the two properties outlined above leads us to conclude that it is not an anonymous 21 but a pseudo-anonymous electronic cash system. Users were quick to develop solutions to circumvent this shortcoming. Two direct solutions were "laundering services" and the development of distributed methods. Both solutions are based on the idea of mixing several public transactions and sending them through some intermediary address; which in turn suffers the drawback of requiring a trusted third party.

Recently, a more creative scheme was proposed by I. Miers et al.: "Zerocoin". Zerocoin utilizes a cryptographic one-way accumulators and zero-knoweldge proofs which permit users to "convert" bitcoins to zerocoins and spend them using anonymous proof of ownership instead of explicit public-key based digital signatures. However, such knowledge proofs have a constant but inconvenient size - about 30kb (based on today's Bitcoin limits), which makes the proposal impractical. Authors admit that the protocol is unlikely to ever be accepted by the majority of Bitcoin users.

Untraceable Transactions

In this section we propose a scheme of fully anonymous transactions satisfying both untraceability and unlinkability conditions. An important feature of our solution is its autonomy: the sender is not required to cooperate with other users or a trusted third party to make his transactions; hence each participant produces a cover traffic independently.

Merging Transactions Across Blocks

Now, we have used Dr. Maxwell's Confidential Transactions to create a noninteractive version of Dr. Maxwell's CoinJoin, but we have not seen the last of marvelous Dr. Maxwell! We need another idea, transaction cut-through, he described in. Again, we create a noninteractive version of this, and to show how it is used with several blocks.

We can imagine now each block as one large transaction. To validate it, we add all the output commitments together, then subtracts all input commitments, k*G values, and all explicit input amounts times H. We find that we could combine transactions from two blocks, as we combined transactions to form a single block, and the result is again a valid transaction. Except now, some output commitments have an input commitment exactly equal to it, where the first block's output was spent in the second block. We could remove both commitments and still have a valid transaction. In fact, there is not even need to check the rangeproof of the deleted output.

The extension of this idea all the way from the genesis block to the latest block, we see that EVERY nonexplicit input is deleted along with its referenced output. What remains are only the unspent outputs, explicit input amounts and every k*G value. And this whole mess can be validated as if it were one transaction: add all unspent commitments output, subtract the values k*G, validate explicit input amounts (if there is anything to validate) then subtract them times H. If the sum is 0, the entire

chain is good.

What is this mean? When a user starts up and downloads the chain he needs the following data from each block:

Explicit amounts for new money (block subsidy or sidechain peg-ins) with whatever else data this needs.

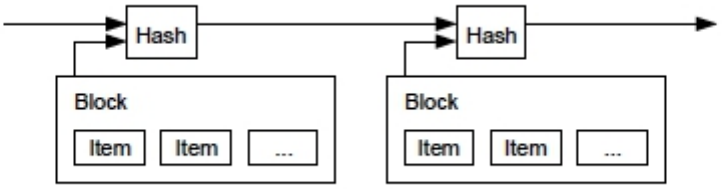
Unspent outputs of all transactions, along with a merkle proof that each output appeared in the original block.

Excess k*G values for all transactions

Bitcoin today there are about 423000 blocks, totaling 80GB or so of data on the hard drive to validate everything. These data are about 150 million transactions and 5 million unspent nonconfidential outputs. Estimate how much space the number of transactions take on a A Multi-Decentralized Interaction Protocol chain. Each unspent output is around 3Kb for rangeproof and Merkle proof. Each transaction also adds about 100 bytes: a k*G value and a signature. The block headers and explicit amounts are negligible. Add this together and get 30Gb -- with a confidential transaction and obscured transaction graph!

Timestamp Server

The solution we propose begins with a timestamp server. A timestamp server works by taking a hash of a block of items to be timestamped and widely publishing the hash, such as in a newspaper or Usenet post. The timestamp proves that the data must have existed at the time, obviously, in order to get into the hash. Each timestamp includes the previous timestamp in its hash, forming a chain, with each additional timestamp reinforcing the ones before it.

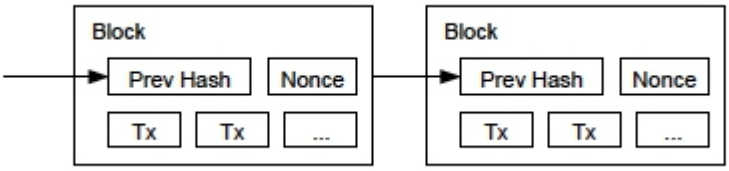


Proof-of-Work

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

For our timestamp network, we implement the proof-of-work by incrementing a nonce in the block

until a value is found that gives the block's hash the required zero bits. Once the CPU effort has been expended to make it satisfy the proof-of-work, the block cannot be changed without redoing the work. As later blocks are chained after it, the work to change the block would include redoing all the blocks after it.



The proof-of-work also solves the problem of determining representation in majority decision making. If the majority were based on one-IP-address-one-vote, it could be subverted by anyone able to allocate many IPs. Proof-of-work is essentially one-CPU-one-vote. The majority decision is represented by the longest chain, which has the greatest proof-of-work effort invested in it. If a majority of CPU power is controlled by honest 23 nodes, the honest chain will grow the fastest and outpace any competing chains. To modify a past block, an attacker would have to redo the proof-of-work of the block and all blocks after it and then catch up with and surpass the work of the honest nodes. We will show later that the probability of a slower attacker catching up diminishes exponentially as subsequent blocks are added.

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

Bitcoin creator Satoshi Nakamoto described the majority decision making algorithm as "one-CPU-one-vote" and used a CPU-bound pricing function (double SHA-256) for his proof-of-work scheme. Since users vote for the single history of transactions order, the reasonableness and consistency of this process are critical conditions for the whole system.

The security of this model suffers from two drawbacks. First, it requires 51% of the network's mining power to be under the control of honest users. Secondly, the system's progress (bug fixes, security fixes, etc...) require the overwhelming majority of users to support and agree to the changes (this occurs when the users update their wallet software).Finally this same voting mechanism is also used for collective polls about implementation of some features.

This permits us to conjecture the properties that must be satisfied by the proof-of-work pricing function. Such function must not enable a network participant to have a significant advantage over another participant; it requires a parity between common hardware and high cost of custom devices. From recent examples, we can see that the SHA-256 function used in the Bitcoin architecture does not posses this property as mining becomes more efficient on GPUs and ASIC devices when compared to high-end CPUs.

Therefore, Bitcoin creates favourable conditions for a large gap between the voting power of participants as it violates the "one-CPU-one-vote" principle since GPU and ASIC owners posses a much larger voting power when compared with CPU owners. It is a classical example of the Pareto principle where 20% of a system's participants control more than 80% of the votes.

One could argue that such inequality is not relevant to the network's security since it is not the small number of participants controlling the majority of the votes but the honesty of these participants that matters. However, such argument is somewhat flawed since it is rather the possibility of cheap specialized hardware appearing rather than the participants' honesty which poses a threat. To demonstrate this, let us take the following example. Suppose a malevolent individual gains significant mining power by creating his own mining farm through the cheap hardware described previously. Suppose that the global hashrate decreases significantly, even for a moment, he can now use his mining power to fork the chain and double-spend. As we shall see later in this article, it is not unlikely for the previously described event to take place.

Network

The steps to run the network are as follows:

1)New transactions are broadcast to all nodes.

2)Each node collects new transactions into a block.

3)Each node works on finding a difficult proof-of-work for its block.

4)When a node finds a proof-of-work, it broadcasts the block to all nodes.

5)Nodes accept the block only if all transactions in it are valid and not already spent.

6) Nodes express their acceptance of the block by working on creating the next block in the chain, using the hash of the accepted block as the previous hash.

Nodes always consider the longest chain to be the correct one and will keep working on extending it. If two nodes broadcast different versions of the next block simultaneously, some nodes may receive one or the other first. In that case, they work on the first one they received, but save the other branch in case it becomes longer. The tie will be broken when the next proof-of-work is found and one branch becomes longer; the nodes that were working on the other branch will then switch to the longer one.

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

Incentive

By convention, the first transaction in a block is a special transaction that starts a new coin owned by the creator of the block. This adds an incentive for nodes to support the network, and provides a way to initially distribute coins into circulation, since there is no central authority to issue them. The

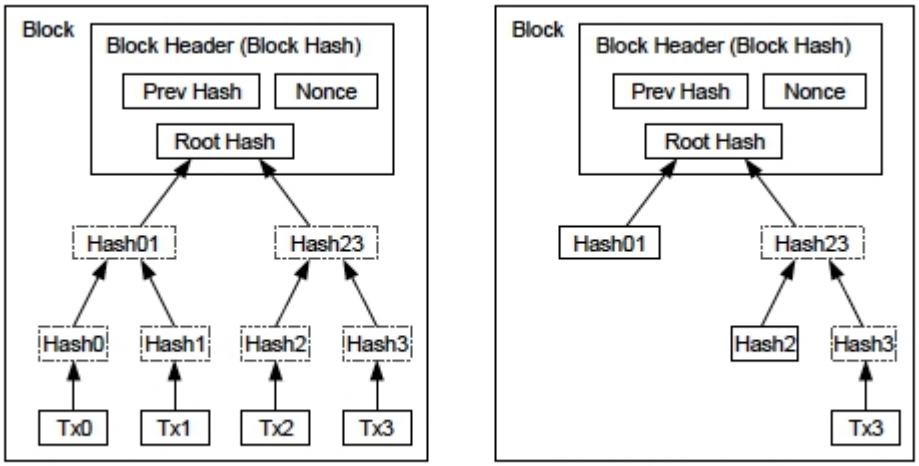
steady addition of a constant of amount of new coins is analogous to gold miners expending resources to add gold to circulation. In our case, it is CPU time and electricity that is expended.

resources to add gold to circulation. In our case, it is CPU time and electricity that is expended. The incentive can also be funded with transaction fees. If the output value of a transaction is less than its input value, the difference is a transaction fee that is added to the incentive value of the block containing the transaction. Once a predetermined number of coins have entered circulation, the incentive can transition entirely to transaction fees and be completely inflation free.

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

Reclaiming Disk Space

Once the latest transaction in a coin is buried under enough blocks, the spent transactions before it can be discarded to save disk space. To facilitate this without breaking the block's hash, transactions are hashed in a Merkle Tree, with only the root included in the block's hash. Old blocks can then be compacted by stubbing off branches of the tree. The interior hashes do not need to be stored.



Transactions Hashed in a Merkle Tree

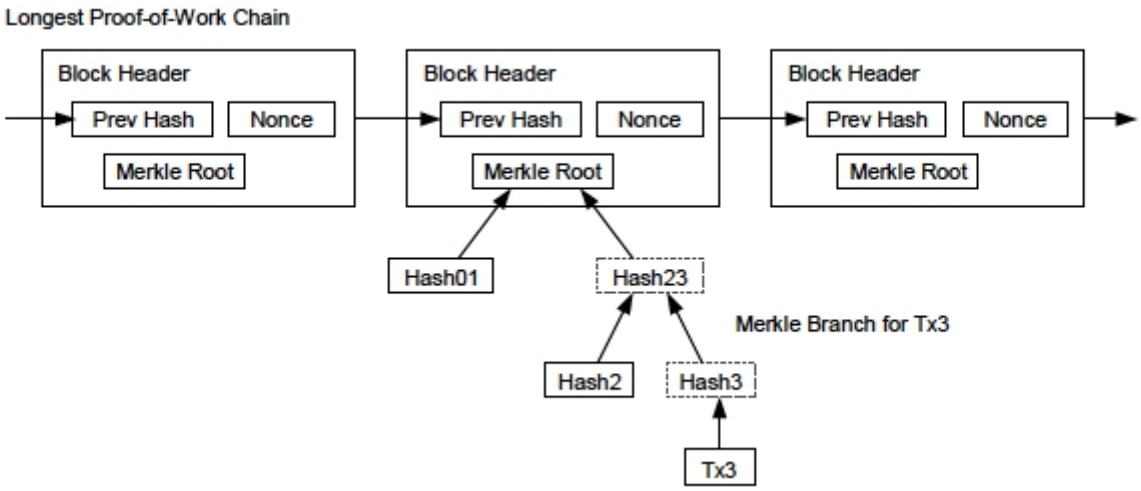
After Pruning Tx0-2 from the Block

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

Simplified Payment Verification

It is possible to verify payments without running a full network node. A user only needs to keep a copy of the block headers of the longest proof-of-work chain, which he can get by querying network

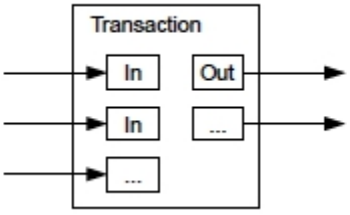
nodes until he's convinced he has the longest chain, and obtain the Merkle branch linking the transaction to the block it's timestamped in. He can't check the transaction for himself, but by linking it to a place in the chain, he can see that a network node has accepted it, and blocks added after it further confirm the network has accepted it.



As such, the verification is reliable as long as honest nodes control the network, but is more vulnerable if the network is overpowered by an attacker. While network nodes can verify transactions for themselves, the simplified method can be fooled by an attacker's fabricated transactions for as long as the attacker can continue to overpower the network. One strategy to protect against this would be to accept alerts from network nodes when they detect an invalid block, prompting the user's software to download the full block and alerted transactions to confirm the inconsistency. Businesses that receive frequent payments will probably still want to run their own nodes for more independent security and quicker verification.

Combining and Splitting Value

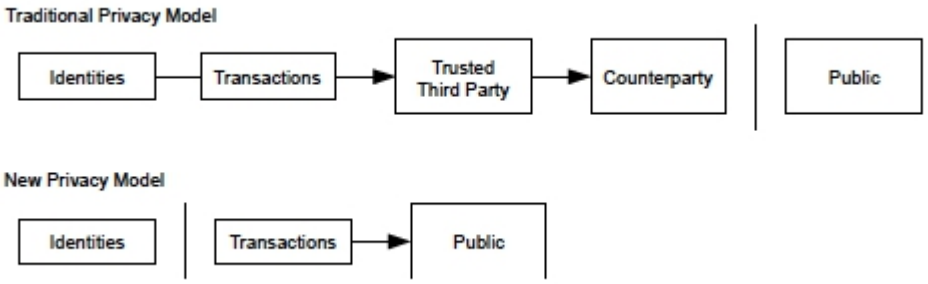
Although it would be possible to handle coins individually, it would be unwieldy to make a separate transaction for every cent in a transfer. To allow value to be split and combined, transactions contain multiple inputs and outputs. Normally there will be either a single input from a larger previous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



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

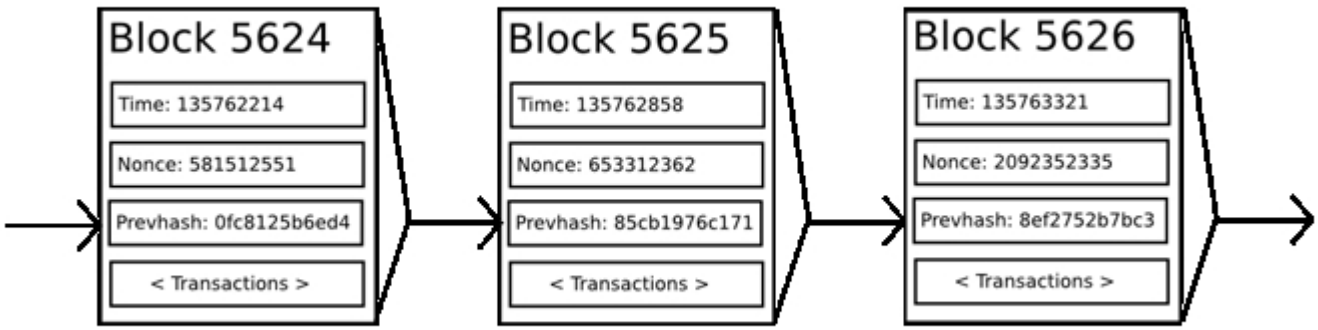
Privacy

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



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

Mining



If we had access to a trustworthy centralized service, this system would be trivial to implement; it could be coded exactly as described, using a centralized server's hard drive to keep track of the state. However, with Bitcoin we are trying to build a decentralized currency system, so we will need to combine the state transition system with a consensus system in order to ensure that everyone agrees on the order of transactions. Bitcoin's decentralized consensus process requires nodes in the network

to continuously attempt to produce packages of transactions called "blocks". The network is intended to create one block approximately every ten minutes, with each block containing a timestamp, a nonce, a reference to (i.e., hash of) the previous block and a list of all of the transactions that have taken place since the previous block. Over time, this creates a persistent, ever-growing, "blockchain" that continually updates to represent the latest state of the Bitcoin ledger.

The algorithm for checking if a block is valid, expressed in this paradigm, is as follows:

1.Check if the previous block referenced by the block exists and is valid.

2.Check that the timestamp of the block is greater than that of the previous block and less than 2 hours into the future

3.Check that the proof of work on the block is valid.

4.Let S[0] be the state at the end of the previous block.

5.Suppose TX is the block's transaction list with n transactions. For all i in 0...n-1, set S[i+1] = APPLY(S[i],TX[i]) If any application returns an error, exit and return false.

6.Return true, and register S[n] as the state at the end of this block.

Essentially, each transaction in the block must provide a valid state transition from what was the canonical state before the transaction was executed to some new state. Note that the state is not encoded in the block in any way; it is purely an abstraction to be remembered by the validating node and can only be (securely) computed for any block by starting from the genesis state and sequentially applying every transaction in every block. Additionally, note that the order in which the miner includes transactions into the block matters; if there are two transactions A and B in a block such that B spends a UTXO created by A, then the block will be valid if A comes before B but not otherwise.

The one validity condition present in the above list that is not found in other systems is the requirement for "proof of work". The precise condition is that the double-SHA256 hash of every block, treated as a 256-bit number, must be less than a dynamically adjusted target, which as of the time of this writing is approximately 2^187. The purpose of this is to make block creation computationally "hard", thereby preventing Sybil attackers from remaking the entire blockchain in their favor. Because SHA256 is designed to be a completely unpredictable pseudorandom function, the only way to create a valid block is simply trial and error, repeatedly 28 incrementing the nonce and seeing if the new hash matches.

At the current target of ~2^187, the network must make an average of ~2^69 tries before a valid block is found; in general, the target is recalibrated by the network every 2016 blocks so that on average a new block is produced by some node in the network every ten minutes. In order to compensate miners for this computational work, the miner of every block is entitled to include a

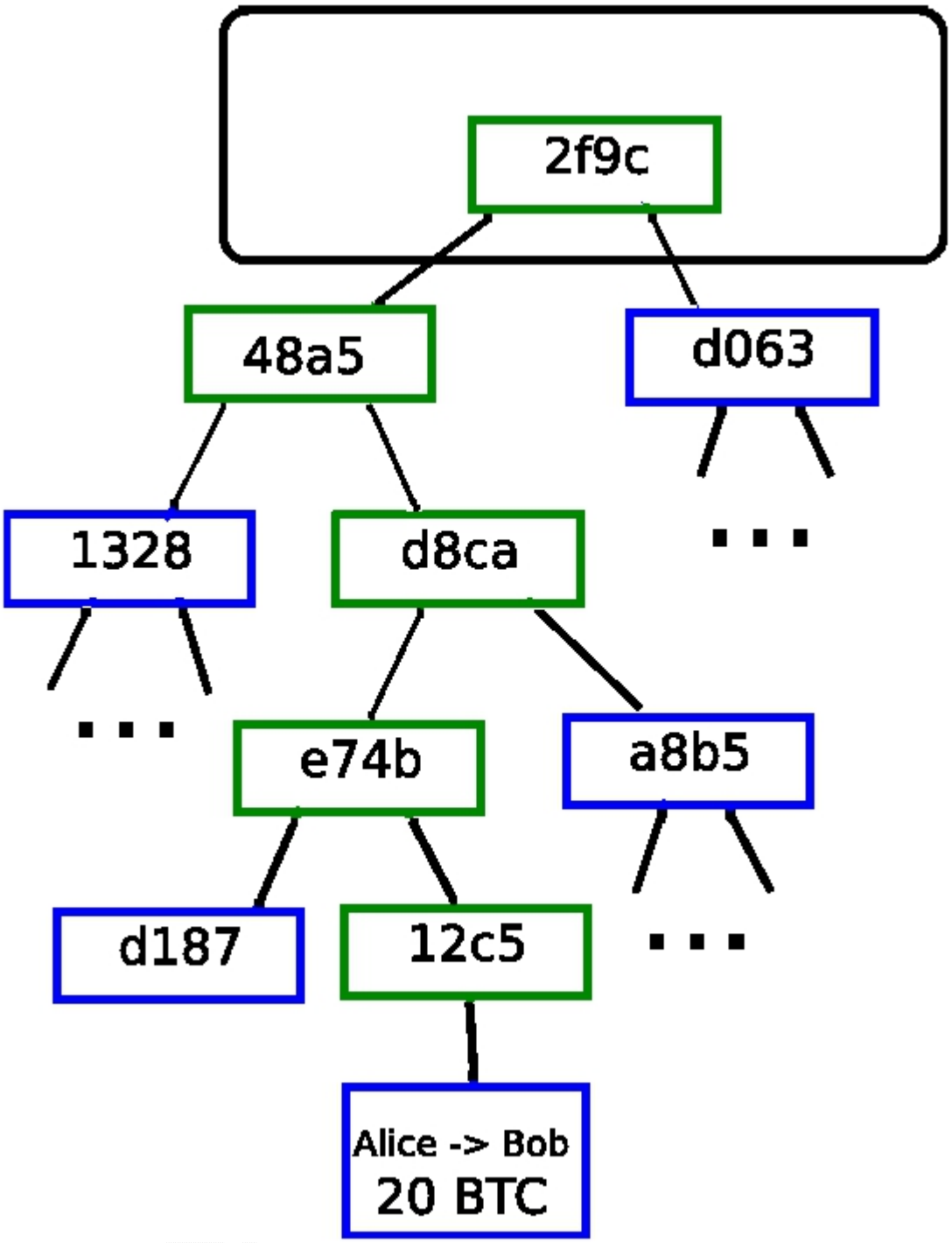
transaction giving themselves 25 BTC out of nowhere. Additionally, if any transaction has a higher total denomination in its inputs than in its outputs, the difference also goes to the miner as a "transaction fee". Incidentally, this is also the only mechanism by which BTC are issued; the genesis state contained no coins at all.

In order to better understand the purpose of mining, let us examine what happens in the event of a malicious attacker. Since Bitcoin's underlying cryptography is known to be secure, the attacker will target the one part of the Bitcoin system that is not protected by cryptography directly: the order of transactions. The attacker's strategy is simple:

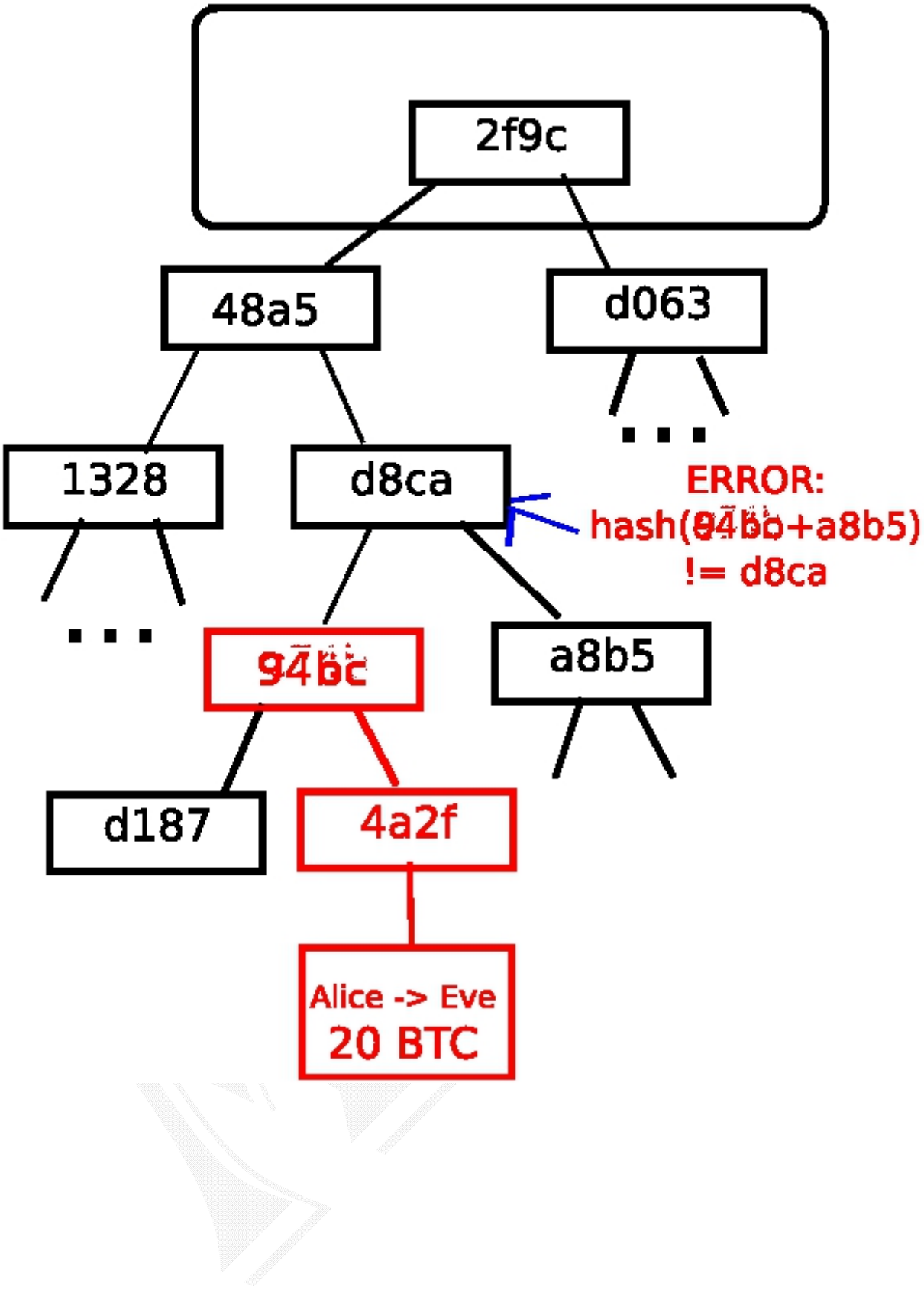
1.Send 100 BTC to a merchant in exchange for some product (preferably a rapid-delivery digital good)
2. Wait for the delivery of the product
3. Produce another transaction sending the same 100 BTC to himself Try to convince the network that his transaction to himself was the one that came first.

Once step (1) has taken place, after a few minutes some miner will include the transaction in a block, say block number 270000. After about one hour, five more blocks will have been added to the chain after that block, with each of those blocks indirectly pointing to the transaction and thus "confirming" it. At this point, the merchant will accept the payment as finalized and deliver the product; since we are assuming this is a digital good, delivery is instant. Now, the attacker creates another transaction sending the 100 BTC to himself. If the attacker simply releases it into the wild, the transaction will not be processed; miners will attempt to run APPLY(S,TX) and notice that TX consumes a UTXO which is no longer in the state. So instead, the attacker creates a "fork" of the blockchain, starting by mining another version of block 270000 pointing to the same block 269999 as a parent but with the new transaction in place of the old one. Because the block data is different, this requires redoing the proof of work. Furthermore, the attacker's new version of block 270000 has a different hash, so the original blocks 270001 to 270005 do not "point" to it; thus, the original chain and the attacker's new chain are completely separate. The rule is that in a fork the longest blockchain is taken to be the truth, and so legitimate miners will work on the 270005 chain while the attacker alone is working on the 270000 chain. In order for the attacker to make his blockchain the longest, he would need to have more computational power than the rest of the network combined in order to catch up (hence, "51% attack").

Merkle Trees

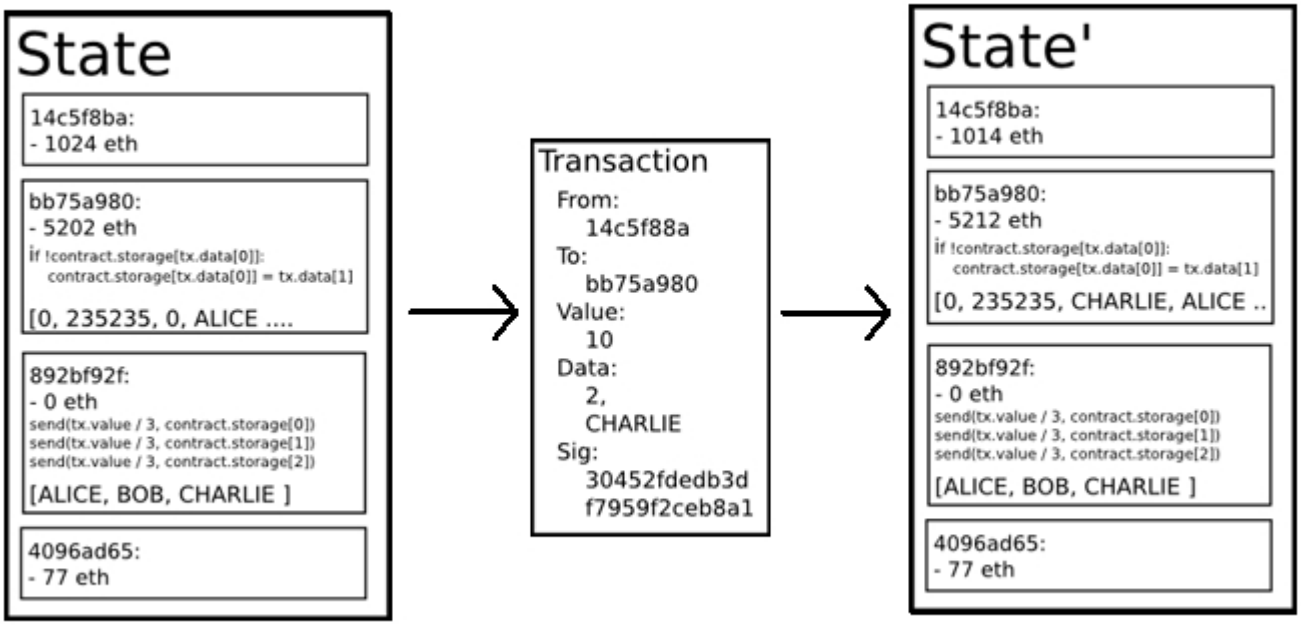


01: it suffices to present only a small number of nodes in a Merkle tree to give a proof of the validity of a branch.



02: any attempt to change any part of the Merkle tree will eventually lead to an inconsistency somewhere up the chain.

A Multi-Decentralized Interaction Protocol State Transition Function



The A Multi-Decentralized Interaction Protocol state transition function, APPLY(S,TX) -> S' can be defined as follows:

1.Check if the transaction is well-formed (ie. has the right number of values), the signature is valid, and the nonce matches the nonce in the sender's account. If not, return an error.

2.Calculate the transaction fee as STARTGAS * GASPRICE, and determine the sending address from the signature. Subtract the fee from the sender's account balance and increment the sender's nonce. If there is not enough balance to spend, return an error.

3.Initialize GAS = STARTGAS, and take off a certain quantity of gas per byte to pay for the bytes in the transaction.

4.Transfer the transaction value from the sender's account to the receiving account. If the receiving account does not yet exist, create it. If the receiving account is a contract, run the contract's code either to completion or until the execution runs out of gas.

5.If the value transfer failed because the sender did not have enough money, or the code execution ran out of gas, revert all state changes except the payment of the fees, and add the fees to the miner's account.

6.Otherwise, refund the fees for all remaining gas to the sender, and send the fees paid for gas consumed to the miner.

For example, suppose that the contract's code is:

```
if !self.storage[calldataload(0)]: self.storage[calldataload(0)] = calldataload(32)
```

Note that in reality the contract code is written in the low-level EVM code; this example is written in Serpent, one of our high-level languages, for clarity, and can be compiled down to EVM code. Suppose that the contract's storage starts off empty, and a transaction is sent with 10 ABCP value, 2000 gas, 0.001 ABCP gasprice, and 64 bytes of data, with bytes 0-31 representing the number 2 and bytes 32-63 representing the string CHARLIE. The process for the state transition function in this case is as follows:

1.Check that the transaction is valid and well formed.

2.Check that the transaction sender has at least 2000 * 0.001 = 2 ABCP. If it is, then subtract 2 ABCP from the sender's account.

3.Initialize gas = 2000; assuming the transaction is 170 bytes long and the byte-fee is 5, subtract 850 so that there is 1150 gas left.

4.Subtract 10 more ABCP from the sender's account, and add it to the contract's account.

5.Run the code. In this case, this is simple: it checks if the contract's storage at index 2 is used, notices that it is not, and so it sets the storage at index 2 to the value CHARLIE. Suppose this takes 187 gas, so the remaining amount of gas is 1150 - 187 = 963

6.Add 963 * 0.001 = 0.963 ABCP back to the sender's account, and return the resulting state.

If there was no contract at the receiving end of the transaction, then the total transaction fee would simply be equal to the provided GASPRICE multiplied by the length of the transaction in bytes, and the data sent alongside the transaction would be irrelevant.

Note that messages work equivalently to transactions in terms of reverts: if a message execution runs out of gas, then that message's execution, and all other executions triggered by that execution, revert, but parent executions do not need to revert. This means that it is "safe" for a contract to call another contract, as if A calls B with G gas then A's execution is guaranteed to lose at most G gas. Finally, note that there is an opcode, CREATE, that creates a contract; its execution mechanics are generally similar to CALL, with the exception that the output of the execution determines the code of a newly created contract.

COMPARISON

Ant Block Chain Protocol is a public gateway to the cryptomarket: It is a decentralized investment platform simplifying blockchain investments, backed by a transparent and trustworthy company.

Our platform is a one-stop-shop and solves the problem of crypto spending with one simple dashboard. It allows converting from and to cryptocurrencies as well as investing into valuable assets like the commodity-backed gold token or Ant Block Chain Protocol ‘s blue-chip crypto fund (ABCP)

Besides that, Ant Block Chain Protocol is the only blockchain investment platform with a free exchange and a cryptocurrency card. Another strong pillar of Ant Block Chain Protocol´s unique position in the market: Our users gain customer lifetime value as we are not a product centric seller or exchange, but a care taker with active management and active customer support.

MARKET PLAYERS	ABCP	Blackmoon crypto	Enigma	Grayscale	TenX	Monaco	Xapo
Invest in BTC, ETH & Altcoins	✓	✓	✓	✗	✓	✗	✗
Invest in Altcoins	✓	✓	✓	✗	✓	✗	✗
Invest in ICOs	✓	✓	✓	✗	✗	✗	✗
Invest in AC & VC	✓	✓	✓	✗	✗	✗	✗
CC debit card	✓	✗	✗	✗	✓	✓	✓
Only decentralized investment	✓	✗	✓	✓	✓	✗	✓
Active management	✓	✓	✗	✓	✗	✗	✗
Save / spend All-in-one-solutiontion	✓	✗	✗	✗	✗	✗	✗
Asset-backed tokens	✗	✗	✗	✗	✗	✗	✗

HOW PECUNIO WORKS

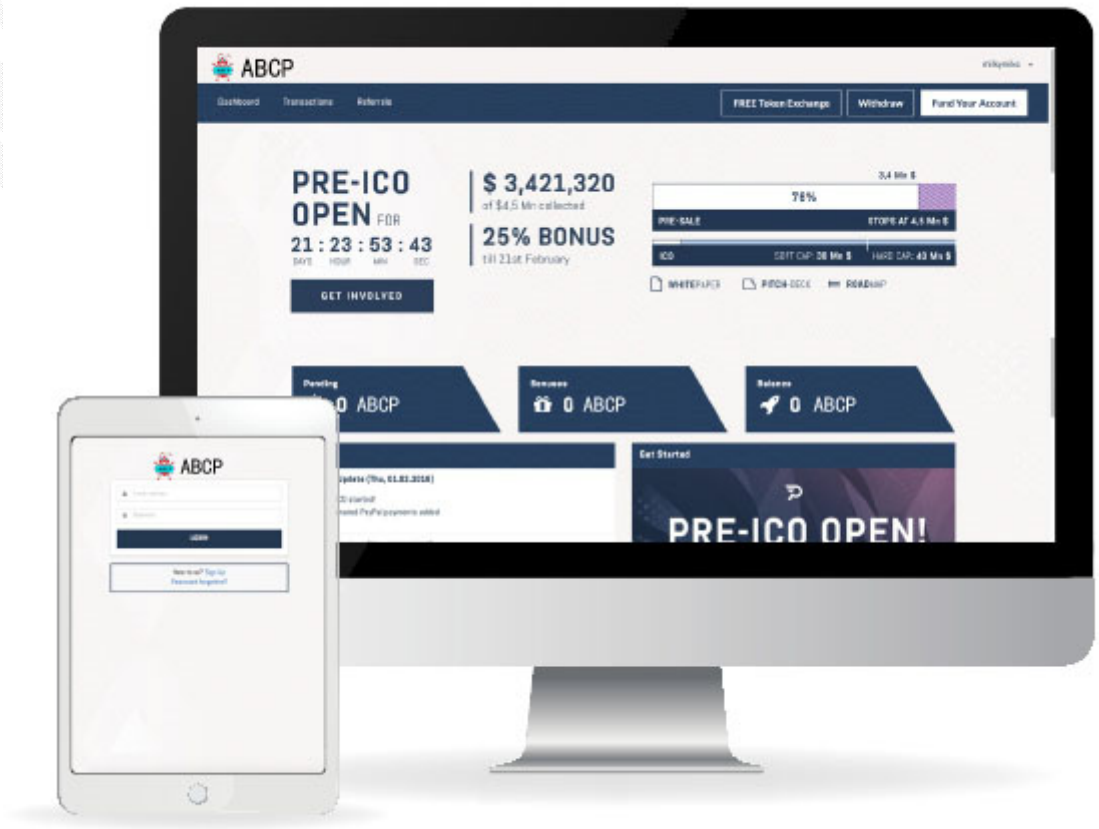
SIGN UP

Sign up is as easy as creating a Facebook Account

The user signs up to create an account on Ant Block Chain Protocol's platform by providing an email address, a user name and a password. Upon confirmation via email, a personal wallet is created and he is automatically directed to the user dashboard where he finds an overview of his personal account and holdings.

After successfully transferring cryptocurrencies into his account, it is now the user's choice whether to spend it via the crypto card, exchange it or invest it into Ant Block Chain Protocol's products.

Every platform user has the opportunity to purchase into different types of products: The assetbacked token (PGC), various fund tokens (PCC, PICO, PAV), as well as the Ant Block Chain Protocol utility token (ABCP). All platform related products and services can only be accessed with the ABCP.



WHY BLOCKCHAIN WHY FUND MANAGEMENT

WHY BLOCKCHAIN

"We have elected to put our money and faith in a mathematical framework that is free of politics and human error. "

Tyler Winklevoss, Co-creator of Facebook, top investor in Bitcoin.

The Blockchain is a foundational technology like the Internet itself — a big system enabling applications to be built on it and value to be stored and transferred safely via a transparent public ledger. The idea of a decentralized, global payment network with low cost and settlement time has profound implications across the world-wide financial system.

Abstracting away from solely the financial sector, the transparency and integrity of data on the Blockchain may transform broad industries and industry standards in areas like record keeping, logistics, law, identity management, securities settlement, application development and many more. Furthermore corporate formation & governance, asset representation, contract & agreement structuring, and the traditional thought processes associated with; each are rapidly being challenged, questioned, re-thought, re-approached, and re-structured because of the spark ignited by the concept of the blockchain. We now find ourselves at the forefront of a huge wave of innovation. Approximately 80% of banks are developing their own blockchain technology today. This is one of the main reasons the world's biggest firms are now investing into this promising technology and/or adopting it as well.

The real value of the blockchain is that it renders intermediaries completely obsolete. Middlemen acting as third party establishing "trust" between unknown parties are no longer inevitable. The blockchain basically replaces these middlemen

A series of industries will become seriously disrupted, first and foremost:

Financial Services (Banks, Clearing houses, Money Transfer)

Banks are essentially secure storehouses and transfer hubs for money. Blockchain's secure, decentralized, and tamper-proof ledger addresses this function—at a fraction of the cost. A company called "Thought Machine" has already created a "blockchain bank." Clearing houses and stockbrokers are also in the firing line for the same reason. Companies like Western Union and MoneyGram have always dominated money transfer services. But blockchain start-ups are trying to create a competitive scenario by offering faster, cheaper and versatile forms of money transfer.

Real Estate

When most people think of buying and selling property, they think of copious amounts of paperwork, long, administrative processes, and high agent fees. Using blockchain, anyone can manage, track, and transfer land titles and property deeds—no need for intermediaries. A company called Ubiquity is providing this service already.

Music Streaming

Music streaming is great—well, maybe not for the musicians. It's estimated that artists lose up to 86% of the proceeds of their music because of illegal downloading. The blockchain makes it possible for artists to earn royalties on their music without going through a record label.
Grammy Award Winner Imogen Heap has created a blockchain-based streaming platform called MYCELIA that is, besides others, facilitating this issue.

WHY FUND MANAGEMENT

Despite Bitcoin’s impressive annualized returns since its inception in 2009, the reality is that blockchain technology is still in its infancy and no one knows whether it will become globally accepted.

This uncertainty surrounding blockchain and Bitcoin’s future is reflected in the price volatility of cryptocurrencies. When it comes to investing in cryptocurrencies and ICOs, there are a few risks investors need to be aware of and mitigate:

Regulatory Risk

Probably the biggest risk to the future success of Bitcoin, as both a currency and as an investment class, is regulatory risk. If China, for example, decides to ban its citizens from holding Bitcoin, the price of the digital currency would crash. China is by far the largest market for Bitcoin trading, with over 90 percent of trading occurring in the People’s Republic. Hence, any negative regulatory changes

would have a direct impact on the world’s Bitcoin investments.
The same goes for leading Bitcoin start-up hubs like the U.S. and the U.K. Should any large economy ban Bitcoin, the price will collapse and struggle to recover.

ICO Scams

Many ICOs/token launches lack legitimacy for one of the following reasons:
1. Some ICOs are outright scams. Their founders simply cash in on the absurd amount of “dumb money” that people throw into token launches. (The “investors” can often be people who missed the boat on Bitcoin and/or Ether, who truly don’t understand the differences in between the various cryptocurrencies, particularly when it comes to value.)

2. Other ICOs are illegitimate because they are essentially a solution without a problem — not everything needs a decentralized application with its own currency. Because of these risks, a significant set of due diligence questionnaire has to be implied before buying into a potential ICO.

It is asked too much, for an average individual with a day job to occupy oneself with all the details of a rapidly changing market. In order to react to these dynamic systems it needs experts and due diligence; both are pillars of our performance policy, which is the reason why we preferred active management of our funds in order to adapt to fluctuated markets. Additionally portfolio diversification mitigates market volatility, fraud and operational risks while as being a hedge fund renders us profitable even during falling markets.

OUR PRODUCTS AND SERVICES

ASSET BACKED TOKENS

Gold has always been the perfect hedge against volatility. Blockchain technology with its high speed transfer of value, security and convenience of operation enables this next step in the future of finance:
It allows us to reintroduce gold in digital form as a global currency on the blockchain. Each Ant Block Chain Protocol
Gold Coin represents exactly 1 gram of segregated, unallocated 999.9 fine gold from LBMA-approved

refineries. We will be the first FinTech company to exchange physical gold for tokens worldwide.
Ant Block Chain Protocol and its partners will store physical gold with the intent, that its gold reserves constantly

meet or exceed the amount of PGC in circulation. We do not deliver a promise; we deliver gold via a cryptocurrency.

IT IS THE MOST PRESTIGIOUS AND SAFEST WAY TO PARTICIPATE IN THE CRYPTOCURRENCY MARKET.

OUR PRODUCTS AND SERVICES

The ABCP Gold Coin is:

- 999.9 fine gold in tokenized form (PGC)
- The world’s best hedge against volatility as a token
- A payment option for goods and services (gold backed cryptocurrency)
- The easiest way to handle and export physical gold (world wide branches for direct pick up)
- The safest and most transparent way of transacting goldASSET BACKED TOKENS

STABILITY
GOLD BACKED

PHYSICALLY
AVAILABLE ASSES

WORLD WIDE AVAILABLE
COOPERATION WITH GVS

ABSOLUTE SAFE
STRIKT PROTOCOLLS

Future distribution

EUROPE

Stadt	Staat	Website
Vienna	Austria	goldvorsorge.at
Salzburg	Austria	goldvorsorge.at
Graz	Austria	goldvorsorge.at
Frankfurt	Germany	silbervorsorge.de
Budapest	Hungary	aranykereskedes.hu
Tallinn	Estonia	gvs.ee
-	Spain	orofino.es
-	France	orfin.fr

NORTH AMERICA

Stadt	Staat	Website
Boston	USA	goldvorsorge.at
New York	USA	goldvorsorge.at
Delaware	USA	goldvorsorge.at
Toronto	Canada	goldvorsorge.at

ASIA

Stadt	Staat	Website
Dubai	United Arab Emirates	goldvorsorge.at
Singapore	Singapore	goldvorsorge.at
Hong Kong	China	goldvorsorge.at

OUR PRODUCTS AND SERVICES

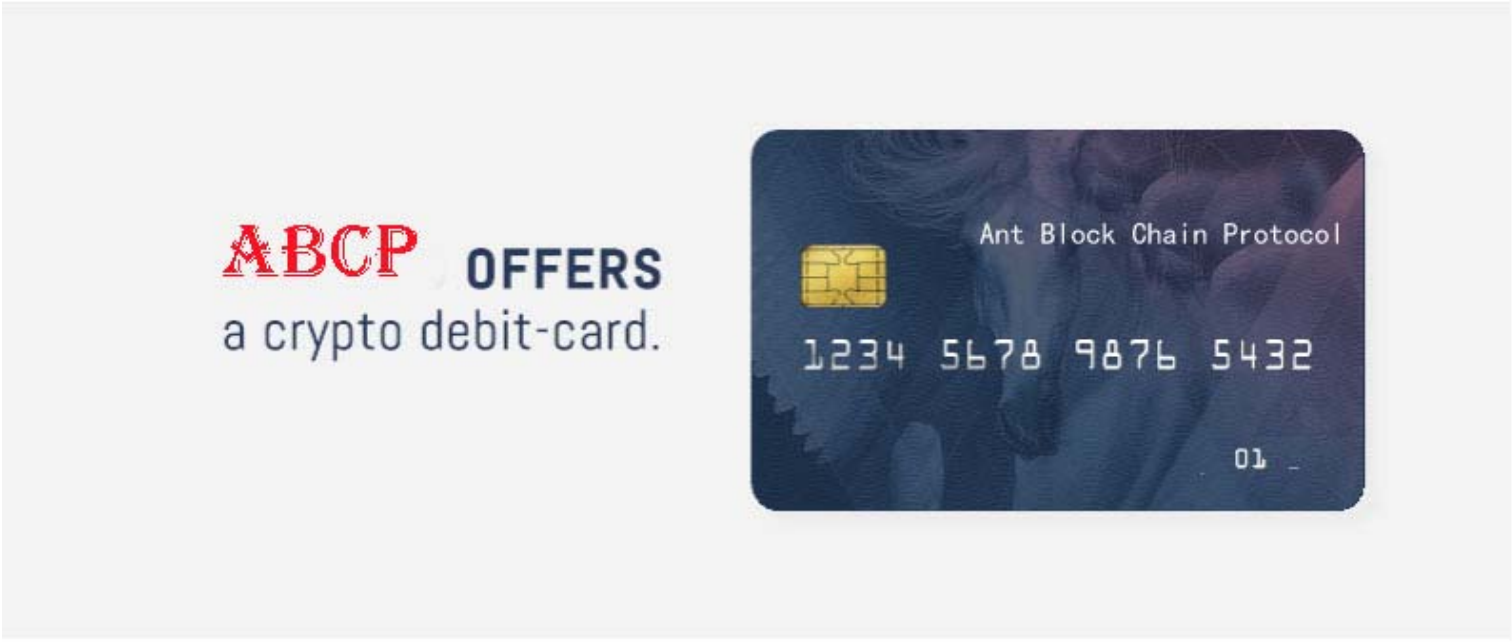
FREE EXCHANGE

We accept and exchange all major cryptocurrencies, whereas we do not charge any fees on behalf of exchanging currencies on our platform. The Exchange will be directly connected to the user wallets and therefore only charges fees (incl. regular network fee) when cryptocurrencies are removed from the platform. Full functionality of the exchange will be implemented after the ICO.

CRYPTOCURRENCY CARD

Ant Block Chain Protocol’ s cryptocurrency card is a prepaid multi-asset debit card, enabling coin and token spending
in all fiat currencies around the world without exchange fees. The user will be able to manage assets via the platform, charging the card with Bitcoin, Ethereum, Litecoin, Monero, Dash, Ant Block Chain Protocol and many other cryptocurrencies.

The debit cards come with free choice of name on card and will be valid for 36 months. No matter where in the world you live, you can be a Ant Block Chain Protocol cryptocurrency card holder and easily manage your assets via our user-friendly dashboard.



PLATFORM TOKENS

All of our tokens are ERC-20 standard and based on Ethereum; a global, secure decentralized

verification network, that enables tokenizing our blockchain funds. The PCO tokens are utility tokens that act as a purchasing voucher for our products.

‘Revocable Commitment Transactions

By combining the ascribing of blame as well as the revocable transaction, one is able to determine when a party is not abiding by the terms of the contract, and enforce penalties without trusting the counterparty.

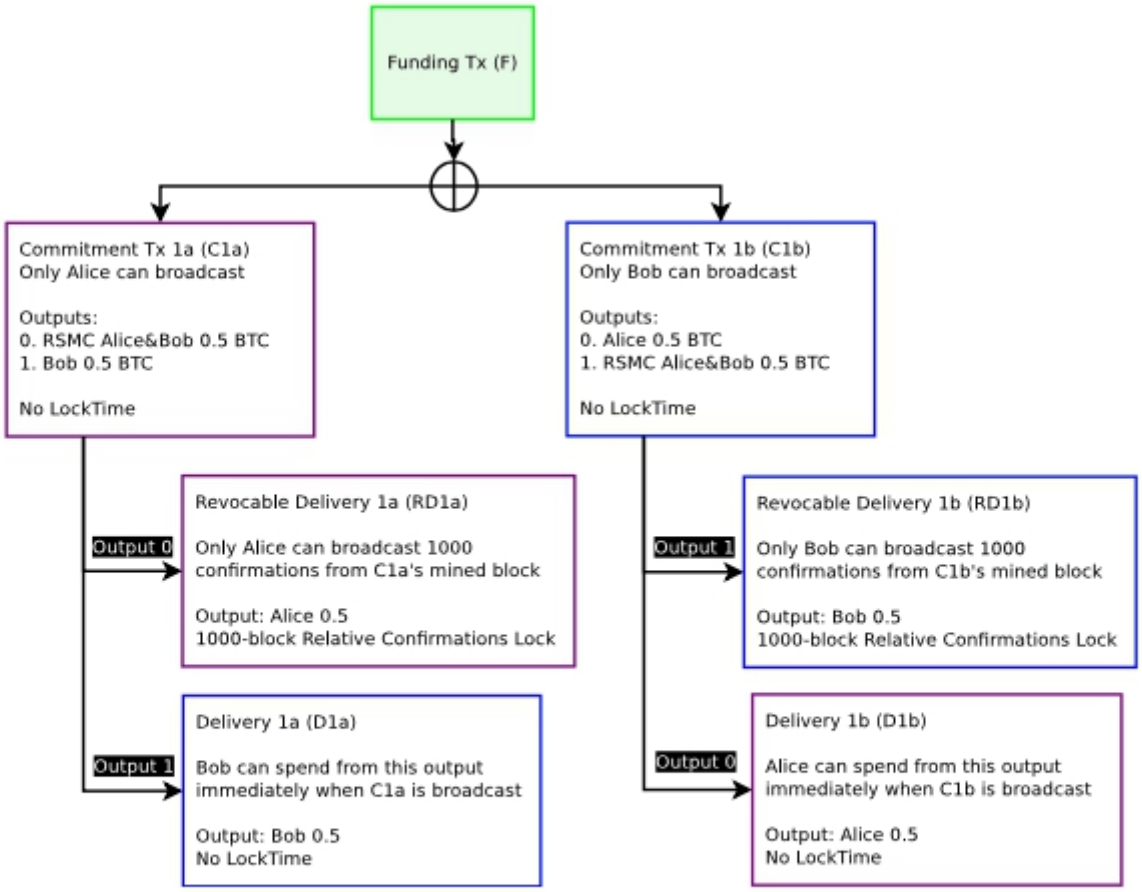


Figure 4: The Funding Transaction F, designated in green, is broadcast on the blockchain after all other transactions are signed. All transactions which only Alice can broadcast are in purple. All transactions which only Bob can broadcast is are blue. Only the Funding Transaction is broadcast on the blockchain at this time.

The intent of creating a new Commitment Transaction is to invalidate all old Commitment Transactions when updating the new balance with a new Commitment Transaction. Invalidation of old transactions can happen by making an output be a Revocable Sequence Maturity Contract (RSMC). To invalidate a transaction, a superseding transaction will be signed and exchanged by both parties that gives all funds to the counterparty in the event an older transaction is incorrectly broadcast. The incorrect broadcast is identified by creating two different Commitment Transactions with the same final balance outputs, however the payment to oneself is encumbered by an RSMC.

In effect, there are two Commitment Transactions from a single Funding Transaction 2-of-2 outputs.

Of these two Commitment Transactions, only one can enter into the blockchain. Each party within a

channel has one version of this contract. So if this is the first Commitment Transaction pair, Alice's Commitment Transaction is defined as C1a, and Bob's Commitment Transaction is defined as C1b. By broadcasting a Commitment Transaction, one is requesting for the channel to close out and end. The first two outputs for the Commitment Transaction include a Delivery Transaction (payout) of the present unallocated balance to the channel counterparties. If Alice broadcasts C1a, one of the output is spendable by D1a, which sends funds to Bob. For Bob, C1b is spendable by D1b, which sends funds to Alice. The Delivery Transaction (D1a/D1b) is immediately redeemable and is not encumbered in any way in the event the Commitment Transaction is broadcast.

For each party's Commitment Transaction, they are attesting that they are broadcasting the most recent Commitment Transaction which they own. Since they are attesting that this is the current balance, the balance 64 paid to the counterparty is assumed to be true, since one has no direct benefit by paying some funds to the counterparty as a penalty.

The balance paid to the person who broadcast the Commitment Transaction, however, is unverified. The participants on the blockchain have no idea if the Commitment Transaction is the most recent or not. If they do not broadcast their most recent version, they will be penalized by taking all the funds in the channel and giving it to the counterparty. Since their own funds are encumbered in their own RSMC, they will only be able to claim their funds after some set number of confirmations after the Commitment Transaction has been included in a block (in our example, 1000 confirmations). If they do broadcast their most recent Commitment Transaction, there should be no revocation transaction superseding the revocable transaction, so they will be able to receive their funds after some set amount of time (1000 confirmations).

By knowing who broadcast the Commitment Transaction and encumbering one's own payouts to be locked up for a predefined period of time, both parties will be able to revoke the Commitment Transaction in the future.

Redeeming Funds from the Channel: Cooperative Counterparties

Either party may redeem the funds from the channel. However, the party that broadcasts the Commitment Transaction must wait for the predefined number of confirmations described in the RSMC. The counterparty which did not broadcast the Commitment Transaction may redeem the funds immediately.

For example, if the Funding Transaction is committed with 1 BTC (half to each counterparty) and Bob broadcasts the most recent Commitment Transaction, C1b, he must wait 1000 confirmations to receive his 0.5 BTC, while Alice can spend 0.5 BTC. For Alice, this transaction is fully closed if Alice agrees that Bob broadcast the correct Commitment Transaction (C1b).

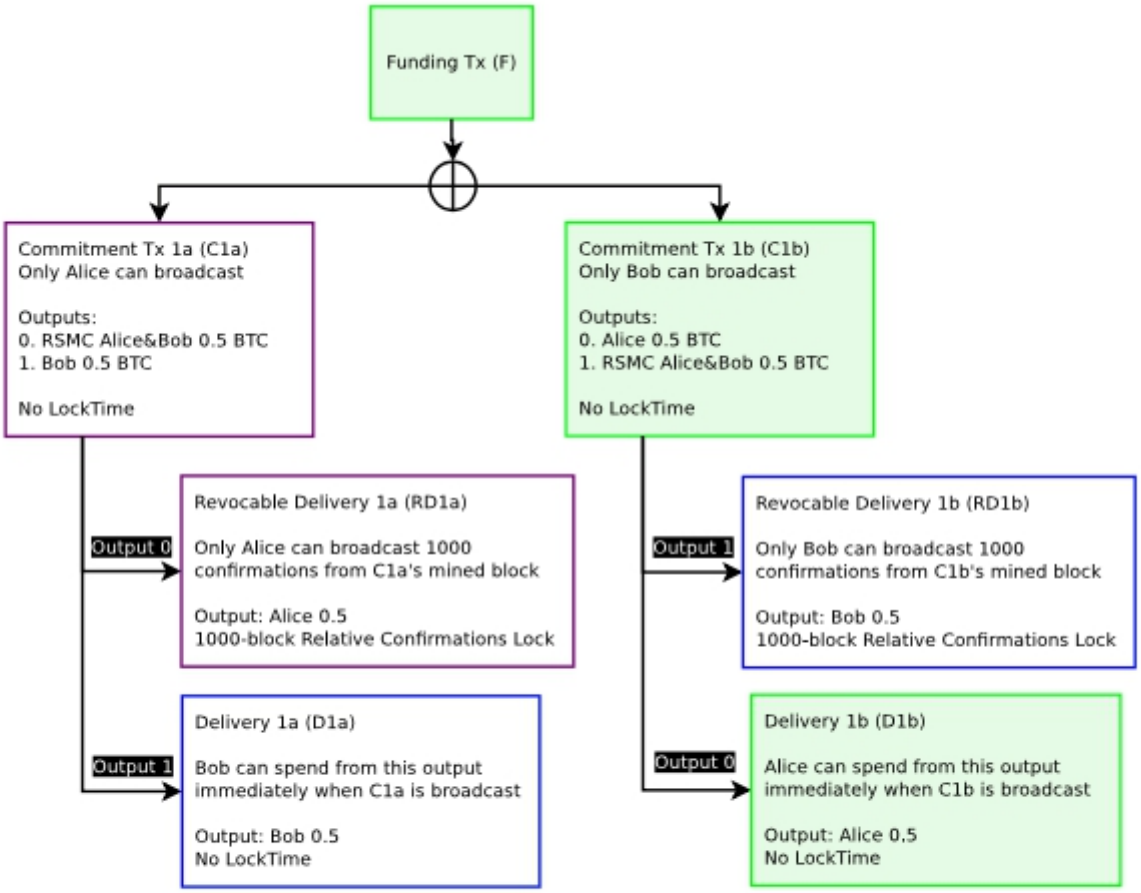


Figure 5: When Bob broadcasts C1b, Alice can immediately redeem her portion. Bob must wait 1000 confirmations. When the block is immediately broadcast, it is in this state. Transactions in green are transactions which are committed into the blockchain.

After the Commitment Transaction has been in the blockchain for 1000 blocks, Bob can then broadcast the Revocable Delivery transaction. He must wait 1000 blocks to prove he has not revoked this Commitment Transaction (C1b). After 1000 blocks, the Revocable Delivery transaction will be able to be included in a block. If a party attempt to include the Revocable Delivery transaction in a block before 1000 confirmations, the transaction will be invalid up until after 1000 confirmations have passed (at which point it will become valid if the output has not yet been redeemed).

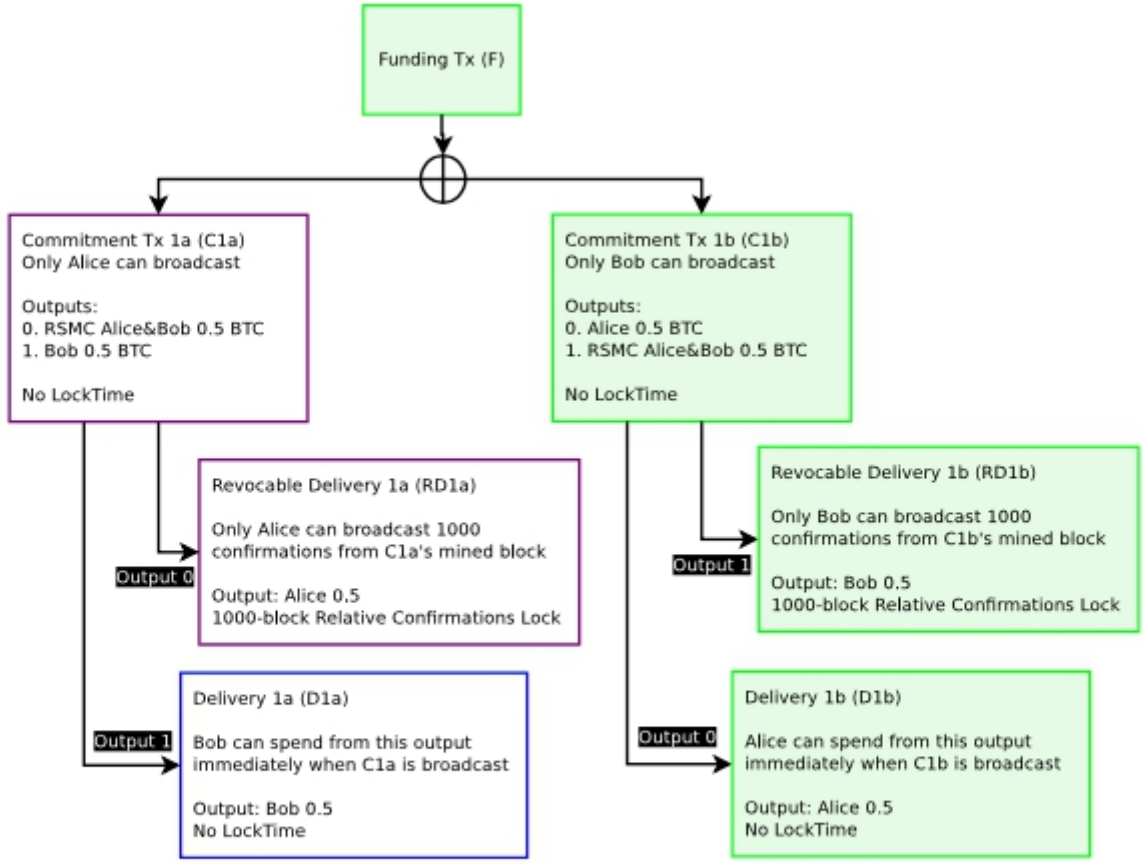


Figure 6: Alice agrees that Bob broadcast the correct Commitment Transaction and 1000 confirmations have passed. Bob then is able to broadcast the Revocable Delivery (RD1b) transaction on the blockchain.

After Bob broadcasts the Revocable Delivery transaction, the channel is fully closed for both Alice and Bob, everyone has received the funds which they both agree are the current balance they each own in the channel. If it was instead Alice who broadcast the Commitment Transaction (C1a), she is the one who must wait 1000 confirmations instead of Bob.

Creating a new Commitment Transaction and Revoking Prior Commitments

While each party may close out the most recent Commitment Transaction at any time, they may also elect to create a new Commitment Transaction and invalidate the old one.

Suppose Alice and Bob now want to update their current balances from 0.5 BTC each refunded to 0.6 BTC for Bob and 0.4 BTC for Alice. When they both agree to do so, they generate a new pair of

Commitment Transactions.

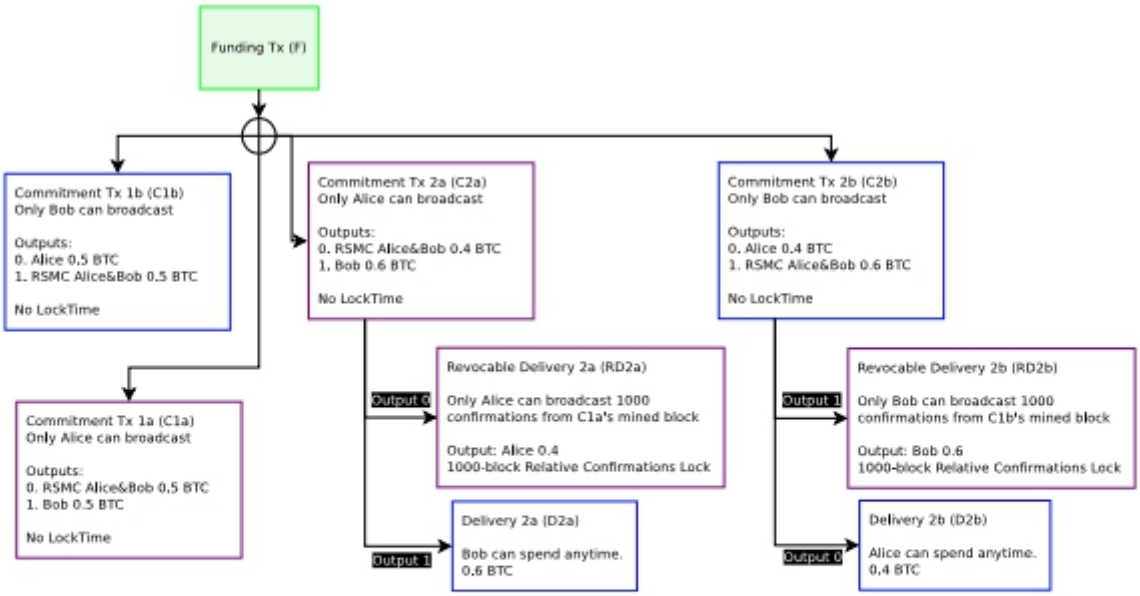


Figure 7: Four possible transactions can exist, a pair with the old commitments, and another pair with the new commitments. Each party inside the channel can only broadcast half of the total commitments (two each). There is no explicit enforcement preventing any particular Commitment being broadcast other than penalty spends, as they are all valid unbroadcasted spends. The Revocable Commitment still exists with the C1a/C1b pair, but are not displayed for brevity.

When a new pair of Commitment Transactions (C2a/C2b) is agreed upon, both parties will sign and exchange signatures for the new Commitment Transaction, then invalidate the old Commitment Transaction. This invalidation occurs by having both parties sign a Breach Remedy Transaction (BR1), which supersedes the Revocable Delivery Transaction (RD1). Each party hands to the other a half-signed revocation (BR1) from their own Revocable Delivery (RD1), which is a spend from the Commitment Transaction. The Breach Remedy Transaction will send all coins to the counterparty within the current balance of the channel. For example, if Alice and Bob both generate a new pair of Commitment Transactions (C2a/C2b) and invalidate prior commitments (C1a/C1b), and later Bob incorrectly broadcasts C1b on the blockchain, Alice can take all of Bob's money from the channel. Alice can do this because Bob has proved to Alice via penalty that he will never broadcast C1b, since the moment he broadcasts C1b, Alice is able to take all of Bob's money in the channel. In effect, by constructing a Breach Remedy transaction for the counterparty, one has attested that one will not be broadcasting any prior commitments. The counterparty can accept this, because they will get all the money in the channel when this agreement is violated

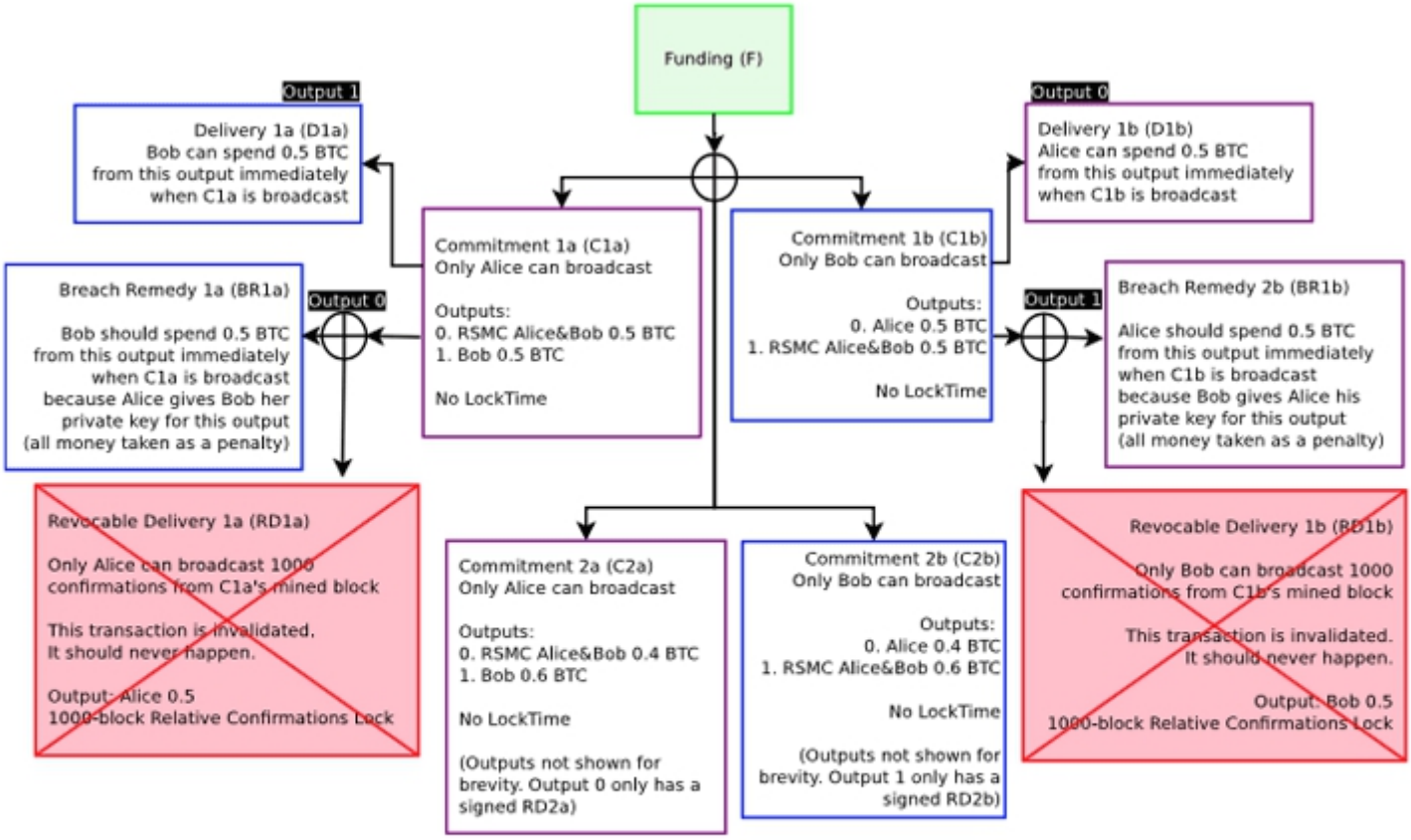


Figure 8: When C2a and C2b exist, both parties exchange Breach Remedy transactions. Both parties now have explicit economic incentive to avoid broadcasting old Commitment Transactions (C1a/C1b). If either party wishes to close out the channel, they will only use C2a (Alice) or C2b (Bob). If Alice broadcasts C1a, all her money will go to Bob. If Bob broadcasts C1b, all his money will go to Alice. See previous figure for C2a/C2b outputs.

Due to this fact, one will likely delete all prior Commitment Transactions when a Breach Remedy Transaction has been passed to the counterparty. If one broadcasts an incorrect (deprecated and invalidated Commitment Transaction), all the money will go to one's counterparty. For example, if Bob broadcasts C1b, so long as Alice watches the blockchain within the predefined number of blocks (in this case, 1000 blocks), Alice will be able to take all the money in this channel by broadcasting RD1b. Even if the present balance of the Commitment state (C2a/C2b) is 0.4 BTC to Alice and 0.6 BTC to Bob, because Bob violated the terms of the contract, all the money goes to Alice as a penalty. Functionally, the Revocable Transaction acts as a proof to the blockchain that Bob has violated the terms in the channel and this is programatically adjudicated by the blockchain.

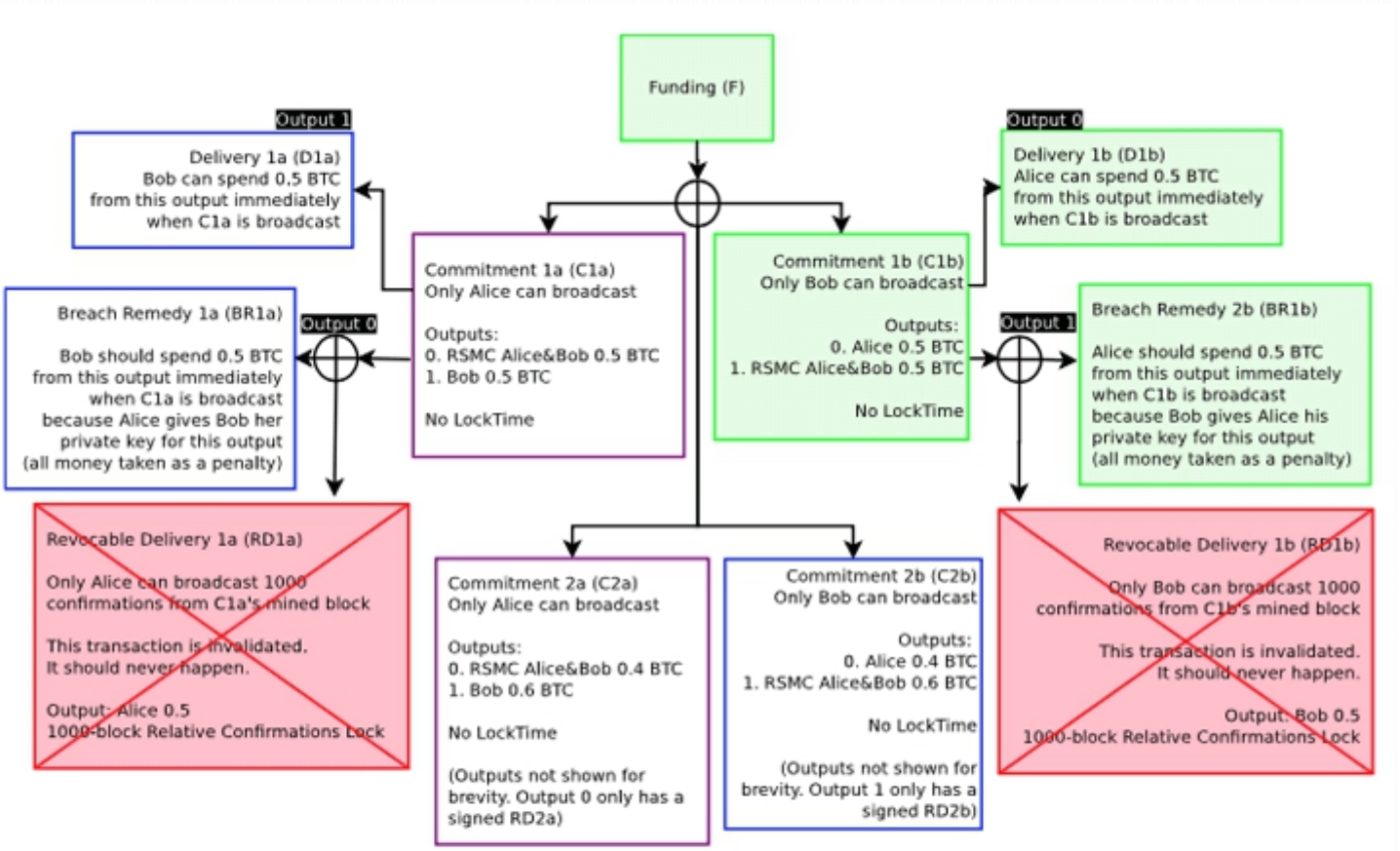


Figure 9: Transactions in green are committed to the blockchain. Bob incorrectly broadcasts C1b (only Bob is able to broadcast C1b/C2b). Because both agreed that the current state is the C2a/C2b Commitment pair, and have attested to each party that old commitments are invalidated via Breach Remedy Transactions, Alice is able to broadcast BR1b and take all the money in the channel, provided she does it within 1000 blocks after C1b is broadcast.

However, if Alice does not broadcast BR1b within 1000 blocks, Bob may be able to steal some money, since his Revocable Delivery Transaction (RD1b) becomes valid after 1000 blocks. When an incorrect Commitment Transaction is broadcast, only the Breach Remedy Transaction can be broadcast for 1000 blocks (or whatever number of confirmations both parties agree to). After 1000 block confirmations, both the Breach Remedy (BR1b) and Revocable Delivery Transactions (RD1b) are able to be broadcast at any time. Breach Remedy transactions only have exclusivity within this predefined time period, and any time after of that is functionally an expiration of the statute of limitations -according to Bitcoin blockchain consensus, the time for dispute has ended.

For this reason, one should periodically monitor the blockchain to see if one's counterparty has

broadcast an invalidated Commitment Transaction, or delegate a third party to do so. A third party can be delegated by only giving the Breach Remedy transaction to this third party. They can be incentivized to watch the blockchain broadcast such a transaction in the event of counterparty maliciousness by giving these third parties some fee in the output. Since the third party is only able to take action when the counterparty is acting maliciously, this third party does not have any power to force close of the channel.

Process for Creating Revocable Commitment Transactions

To create revocable Commitment Transactions, it requires proper construction of the channel from the beginning, and only signing transactions which may be broadcast at any time in the future, while ensuring that 70 one will not lose out due to uncooperative or malicious counterparties. This requires determining which public key to use for new commitments, as using SIGHASH NOINPUT requires using unique keys for each Commitment Transaction RSMC (and HTLC) output. We use P to designate pubkeys and K to designate the corresponding private key used to sign.

When generating the first Commitment Transaction, Alice and Bob agree to create a multisig output from a Funding Transaction with a single multisig(PAliceF , PBobF) output, funded with 0.5 BTC from Alice and Bob for a total of 1 BTC. This output is a Pay to Script Hash transaction, which requires both Alice and Bob to both agree to spend from the Funding Transaction. They do not yet make the Funding Transaction (F) spendable. Additionally, PAliceF and PBobF are only used for the Funding Transaction, they are not used for anything else.

Since the Delivery transaction is just a P2PKH output (bitcoin addresses beginning with 1) or P2SH transaction (commonly recognized as addresses beginning with the 3) which the counterparties designate beforehand, this can be generated as an output of PAliceD and PBobD. For simplicity, these output addresses will remain the same throughout the channel, since its funds are fully controlled by its designated recipient after the Commitment Transaction enters the blockchain. If desired, but not necessary, both parties may update and change PAliceD and PBobD for future Commitment Transactions.

Both parties exchange pubkeys they intend to use for the RSMC (and HTLC described in future sections) for the Commitment Transaction. Each set of Commitment Transactions use their own public keys and are not ever reused. Both parties may already know all future pubkeys by using a BIP 0032[17] HD Wallet construction by exchanging Master Public Keys during channel construction. If they wish to generate a new Commitment Transaction pair C2a/C2b, they use multisig(PAliceRSMC2, PBobRSMC2) for the RSMC output.

After both parties know the output values from the Commitment Transactions, both parties create the pair of Commitment Transactions, e.g. C2a/C2b, but do not exchange signatures for the Commitment Transactions. They both sign the Revocable Delivery transaction (RD2a/RD2b) and exchange the signatures. Bob signs RD1a and gives it to Alice (using KBobRSMC2), while Alice signs RD1b and gives it to Bob (using KAliceRSMC2).

When both parties have the Revocable Delivery transaction, they exchange signatures for the

Commitment Transactions. Bob signs C1a using KBobF and gives it to Alice, and Alice signs C1b using KAliceF and gives it to Bob

At this point, the prior Commitment Transaction as well as the new Commitment Transaction can be broadcast; both C1a/C1b and C2a/C2b are valid. (Note that Commitments older than the prior Commitment are invalidated via penalties.) In order to invalidate C1a and C1b, both parties exchange Breach Remedy Transaction (BR1a/BR1b) signatures for the prior commitment C1a/C1b. Alice sends BR1a to Bob using KAliceRSMC1, and Bob sends BR1b to Alice using KBobRSMC1. When both Breach Remedy signatures have been exchanged, the channel state is now at the current Commitment C2a/C2b and the balances are now committed.

However, instead of disclosing the BR1a/BR1b signatures, it's also possible to just disclose the private keys to the counterparty. This is more effective as described later in the key storage section. One can disclose the private keys used in one's own Commitment Transaction. For example, if Bob wishes to invalidate C1b, he sends his private keys used in C1b to Alice (he does NOT disclose his keys used in C1a, as that would permit coin theft). Similarly, Alice discloses all her private key outputs in C1a to Bob to invalidate C1a

If Bob incorrectly broadcasts C1b, then because Alice has all the private keys used in the outputs of C1b, she can take the money. However, only Bob is able to broadcast C1b. To prevent this coin theft risk, Bob should destroy all old Commitment Transactions

Cooperatively Closing Out a Channel

Both parties are able to send as many payments to their counterparty as they wish, as long as they have funds available in the channel, knowing that in the event of disagreements they can broadcast to the blockchain the current state at any time.

In the vast majority of cases, all the outputs from the Funding Transaction will never be broadcast on the blockchain. They are just there in case the other party is non-cooperative, much like how a contract is rarely enforced in the courts. A proven ability for the contract to be enforced in a deterministic manner is sufficient incentive for both parties to act honestly.

When either party wishes to close out a channel cooperatively, they will be able to do so by contacting the other party and spending from the Funding Transaction with an output of the most current Commitment Transaction directly with no script encumbering conditions. No further payments may occur in the channel.

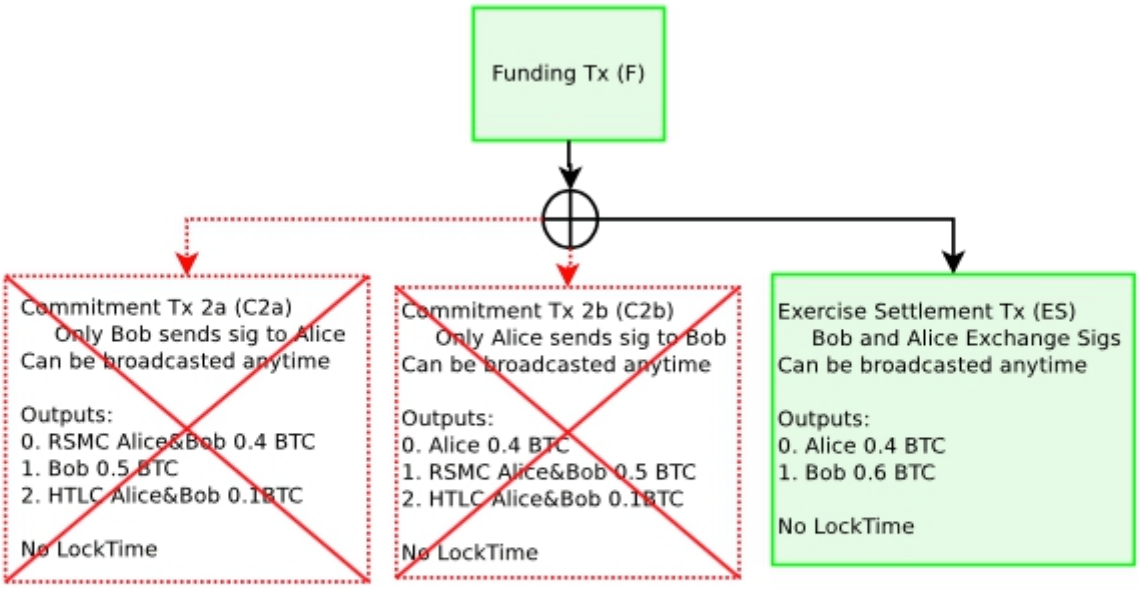


Figure 10: If both counterparties are cooperative, they take the balances in the current Commitment Transaction and spend from the Funding Transaction with a Exercise Settlement Transaction (ES). If the most recent Commitment Transaction gets broadcast instead, the payout (less fees) will be the same.

The purpose of closing out cooperatively is to reduce the number of transactions that occur on the blockchain and both parties will be able to receive their funds immediately (instead of one party waiting for the Revocation Delivery transaction to become valid).

Channels may remain in perpetuity until they decide to cooperatively close out the transaction, or when one party does not cooperate with another and the channel gets closed out and enforced on the blockchain.

Bidirectional Channel Implications and Summary

By ensuring channels can update only with the consent of both parties, it is possible to construct channels which perpetually exist in the blockchain. Both parties can update the balance inside the channel with whatever output balances they wish, so long as it's equal or less than the total funds committed inside the Funding Transaction; balances can move in both directions. If one party becomes malicious, either party may immediately close out the channel and broadcast the most current state to the blockchain. By using a fidelity bond construction (Revocable Delivery Transactions), if a party violates the terms of the channel, the funds will be sent to the counterparty, provided the proof of violation (Breach Remedy Transaction) is entered into the blockchain in a timely manner. If both parties are cooperative, the channel can remain open indefinitely, possibly for many years.

This type of construction is only possible because adjudication occurs programatically over the

blockchain as part of the Bitcoin consensus, so one does not need to trust the other party. As a result, one's channel counterparty does not possess full custody or control of the funds.

Hashed Timelock Contract (HTLC)

A bidirectional payment channel only permits secure transfer of funds inside a channel. To be able to construct secure transfers using a network of channels across multiple hops to the final destination requires an additional construction, a Hashed Timelock Contract (HTLC).

The purpose of an HTLC is to allow for global state across multiple nodes via hashes. This global state is ensured by time commitments and time-based unencumbering of resources via disclosure of preimages. Transactional "locking" occurs globally via commitments, at any point in time a single participant is responsible for disclosing to the next participant whether they have knowledge of the preimage R. This construction does not require custodial trust in one's channel counterparty, nor any other participant in the network.

In order to achieve this, an HTLC must be able to create certain transactions which are only valid after a certain date, using nLockTime, as well as information disclosure to one's channel counterparty. Additionally, this data must be revocable, as one must be able to undo an HTLC.

An HTLC is also a channel contract with one's counterparty which is enforcible via the blockchain. The counterparties in a channel agree to the following terms for a Hashed Timelock Contract:

1. If Bob can produce to Alice an unknown 20-byte random input data R from a known hash H, within three days, then Alice will settle the contract by paying Bob 0.1 BTC.

2. If three days have elapsed, then the above clause is null and void and the clearing process is invalidated,
both parties must not attempt to settle and claim payment after three days.

3. Either party may (and should) pay out according to the terms of this contract in any method of the participants choosing and close out this contract early so long as both participants in this contract agree.

4. Violation of the above terms will incur a maximum penalty of the funds locked up in this contract, to be
paid to the non-violating counterparty as a fidelity bond.

For clarity of examples, we use days for HTLCs and block height for RSMCs. In reality, the HTLC should
also be defined as a block height (e.g. 3 days is equivalent to 432 blocks).
In effect, one desires to construct a payment which is contingent upon knowledge of R by the recipient within
a certain timeframe. After this timeframe, the funds are refunded back to the sender.
Similar to RSMCs, these contract terms are programatically enforced on the Bitoin blockchain and do

not
require trust in the counterparty to adhere to the contract terms, as all violations are penalized via unilaterally
enforced fidelity bonds, which are constructed using penalty transactions spending from commitment states. If
Bob knows R within three days, then he can redeem the funds by broadcasting a transaction; Alice is unable to
withhold the funds in any way, because the script returns as valid when the transaction is spent on the Bitcoin
blockchain.
An HTLC is an additional output in a Commitment Transaction with a unique output script:

```
OP IF
OP ELSE
OP HASH160 <Hash160 (R)> OP EQUALVERIFY 2 <Alice2> <Bob2> OP CHECKMULTISIG
 2 <Alice1> <Bob1> OP CHECKMULTISIG OP ENDIF
```

Conceptually, this script has two possible paths spending from a single HTLC output. The first path (defined in the OP IF) sends funds to Bob if Bob can produce R. The second path is redeemed using a 3-day timelocked refund to Alice. The 3-day timelock is enforced using nLockTime from the spending transaction.

Non-revocable HTLC Construction

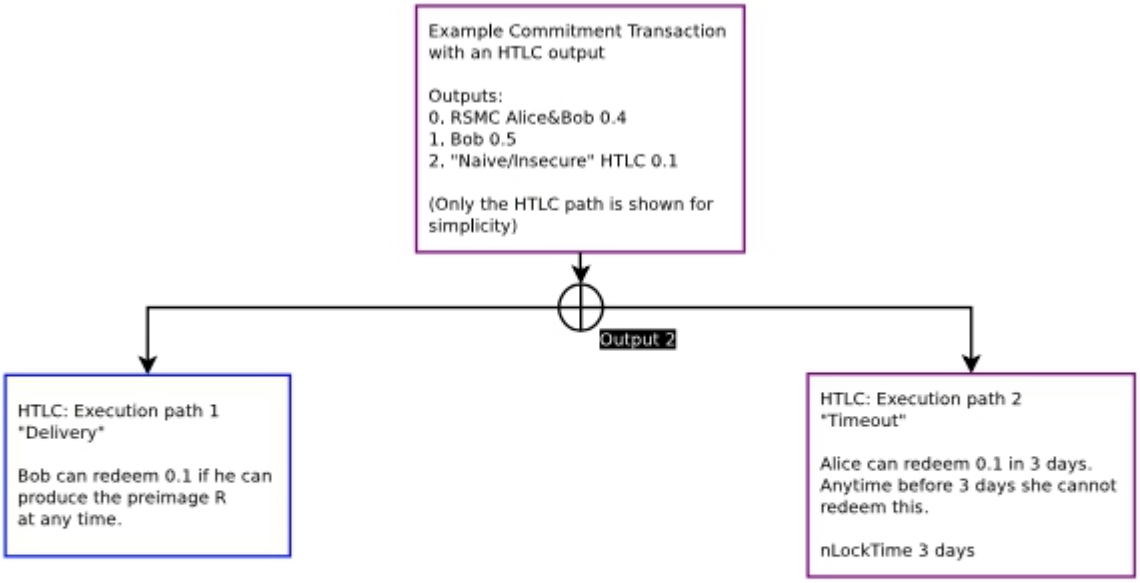


Figure 11: This is a non-functional naive implementation of an HTLC. Only the HTLC path from the Commitment Transaction is displayed. Note that there are two possible spends from an HTLC output. If Bob can produce the preimage R within 3 days and he can redeem path 1. After three days, Alice is able to broadcast path 2. When 3 days have elapsed either is valid. This model, however, doesn't work with multiple Commitment Transactions.

If R is produced within 3 days, then Bob can redeem the funds by broadcasting the "Delivery" transaction. A requirement for the "Delivery" transaction to be valid requires R to be included with the transaction. If R is not included, then the "Delivery" transaction is invalid. However, if 3 days have elapsed, the funds can be sent back to Alice by broadcasting transaction "Timeout". When 3 days have elapsed and R has been disclosed,either transaction may be valid.

It is within both parties individual responsibility to ensure that they can get their transaction into the blockchain in order to ensure the balances are correct. For Bob, in order to receive the funds, he must either broadcast the "Delivery" transaction on the Bitcoin blockchain, or otherwise settle with Alice (while cancelling the HTLC). For Alice, she must broadcast the "Timeout" 3 days from now to receive the refund, or cancel the HTLC entirely with Bob.

Yet this kind of simplistic construction has similar problems as an incorrect bidirectional payment channel construction. When an old Commitment Transaction gets broadcast, either party may attempt to steal funds as both paths may be valid after the fact. For example, if R gets disclosed 1 year later, and an incorrect Commitment Transaction gets broadcast, both paths are valid and are redeemable by either party; the contract is not yet enforcible on the blockchain. Closing out the HTLC is absolutely necessary, because in order for Alice to get her refund, she must terminate the contract and receive her refund. Otherwise, when Bob discovers R after 3 days have elapsed, he may be able to steal the funds which should be going to Alice. With uncooperative counterparties it's not possible to terminate an HTLC without broadcasting it to the bitcoin blockchain as the uncooperative party is unwilling to create a new Commitment Transaction.

Off-chain Revocable HTLC

To be able to terminate this contract off-chain without a broadcast to the Bitcoin blockchain requires embedding RSMCs in the output, which will have a similar construction to the bidirectional channel.

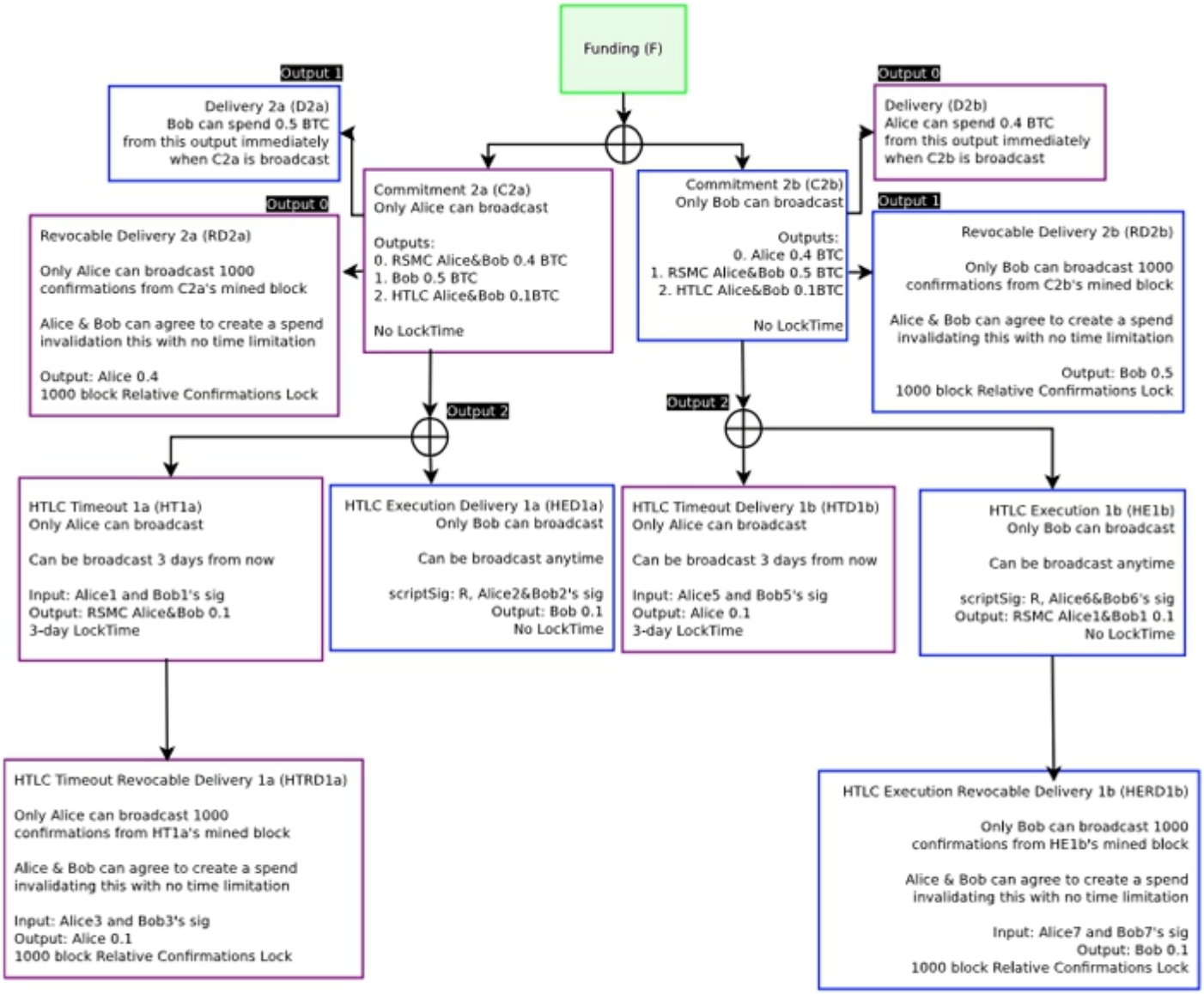


Figure 12: If Alice broadcasts C2a, then the left half will execute. If Bob broadcasts C2b, then the right half will execute. Either party may broadcast their Commitment transaction at any time. HTLC Timeout is only valid after 3 days. HTLC Executions can only be broadcast if the preimage to the hash R is known. Prior Commitments (and their dependent transactions) are not displayed for brevity.

Presume Alice and Bob wish to update their balance in the channel at Commitment 1 with a balance of 0.5 to 75 Alice and 0.5 to Bob.

Alice wishes to send 0.1 to Bob contingent upon knowledge of R within 3 days, after 3 days she wants her money back if Bob does not produce R.

The new Commitment Transaction will have a full refund of the current balance to Alice and Bob (Outputs 0 and 1), with output 2 being the HTLC, which describes the funds in transit. As 0.1 will be encumbered in an HTLC, Alice's balance is reduced to 0.4 and Bob's remains the same at 0.5.

This new Commitment Transaction (C2a/C2b) will have an HTLC output with two possible spends. Each spend is different depending on each counterparty's version of the Commitment Transaction. Similar to the bidirectional payment channel, when one party broadcasts their Commitment, payments to the counterparty will be assumed to be valid and not invalidated. This can occur because when one broadcasts a Commitment Transaction, one is attesting this is the most recent Commitment Transaction. If it is the most recent, then one is also attesting that the HTLC exists and was not invalidated before, so potential payments to one's counterparty should be valid.

Note that HTLC transaction names (beginning with the letter H) will begin with the number 1, whose values do not correlate with Commitment Transactions. This is simply the first HTLC transaction. HTLC transactions may persist between Commitment Transactions. Each HTLC has 4 keys per side of the transaction (C2a and C2b) for a total of 8 keys per counterparty.

The HTLC output in the Commitment Transaction has two sets of keys per counterparty in the output. For Alice's Commitment Transaction (C2a), the HTLC output script requires multisig(PAlice2,PBob2) encumbered by disclosure of R, as well as multisig(PAlice1, PBob1) with no encumbering

For Bob's Commitment Transaction (C2b), the HTLC output script requires multisig(PAlice6,PBob6) encumbered by disclosure of R, as well as multisig(PAlice5, PBob5) with no encumbering. The HTLC output states are different depending upon which Commitment Transaction is broadcast.

HTLC when the Sender Broadcasts the Commitment Transaction

For the sender (Alice), the "Delivery" transaction is sent as an HTLC Execution Delivery transaction (HED1a), which is not encumbered in an RSMC. It assumes that this HTLC has never been terminated off-chain, as Alice is attesting that the broadcasted Commitment Transaction is the most recent. If Bob can produce the preimage R, he will be able to redeem funds from the HTLC after the Commitment Transaction is broadcast on the blockchain. This transaction consumes multisig(PAlice2,PBob2) if Alice broadcasts her Commitment C2a. Only Bob can broadcast HED1a since only Alice gave her signature for HED1a to Bob.

However, if 3 days have elapsed since forming the HTLC, then Alice will be able broadcast a "Timeout" transaction, the HTLC Timeout transaction (HT1a). This transaction is an RSMC. It consumes the output multisig(PAlice1,PBob1) without requiring disclosure of R if Alice broadcasts C2a. This transaction cannot enter into the blockchain until 3 days have elapsed. The output for this transaction is an RSMC with multisig(PAlice3,PBob3) with relative maturity of 1000 blocks, and

multisig(PAlice4,PBob4) with no requirement for confirmation maturity. Only Alice can broadcast HT1a since only Bob gave his signature for HT1a to Alice.

After HT1a enters into the blockchain and 1000 block confirmations occur, an HTLC Timeout Revocable Delivery transaction (HTRD1a) may be broadcast by Alice which consumes multisig(PAlice3,PBob3). Only Alice can broadcast HTRD1a 1000 blocks after HT1a is broadcast since only Bob gave his signature for HTRD1a to Alice. This transaction can be revocable when another transaction supersedes HTRD1a using multisig(PAlice4,PBob4) which does not have any block maturity requirements.

HTLC when the Receiver Broadcasts the Commitment Transaction

For the potential receiver (Bob), the "Timeout" of receipt is refunded as an HTLC Timeout Delivery transaction (HTD1b). This transaction directly refunds the funds to the original sender (Alice) and is not encumbered in an RSMC. It assumes that this HTLC has never been terminated off-chain, as Bob is attesting that the broadcasted Commitment Transaction (C2b) is the most recent. If 3 days have elapsed, Alice can broadcast HTD1b and take the refund. This transaction consumes multisig(PAlice5,PAlice5) if Bob broadcasts C2b. Only Alice can broadcast HTD1b since Bob gave his signature for HTD1b to Alice.

However, if HTD1b is not broadcast (3 days have not elapsed) and Bob knows the preimage R, then Bob will be able to broadcast the HTLC Execution transaction (HE1b) if he can produce R. This transaction is an RSMC. It consumes the output multisig(PAlice6,PBob6) and requires disclosure of R if Bob broadcasts C2b. The output for this transaction is an RSMC with multisig(PAlice7,PBob7) with relative maturity of 1000 blocks, and multisig(PAlice8, PBob8) which does not have any block maturity requirements. Only Bob can broadcast HE1b since only Alice gave her signature for HE1b to Bob.

After HE1b enters into the blockchain and 1000 block confirmations occur, an HTLC Execution Revocable Delivery transaction (HERD1b) may be broadcast by Bob which consumes multisig(PAlice7,PBob7). Only Bob can broadcast HERD1b 1000 blocks after HE1b is broadcast since only Alice gave her signature for HERD1b to Bob. This transaction can be revocable when another transaction supersedes HERD1b using multisig(PAlice8,PBob8) which does not have any block maturity requirements.

HTLC Off-chain Termination

After an HTLC is constructed, to terminate an HTLC off-chain requires both parties to agree on the state of the channel. If the recipient can prove knowledge of R to the counterparty, the recipient is proving that they are able to immediately close out the channel on the Bitcoin blockchain and receive the funds. At this point, if both parties wish to keep the channel open, they should terminate the HTLC off-chain and create a new Commitment Transaction reflecting the new balance.

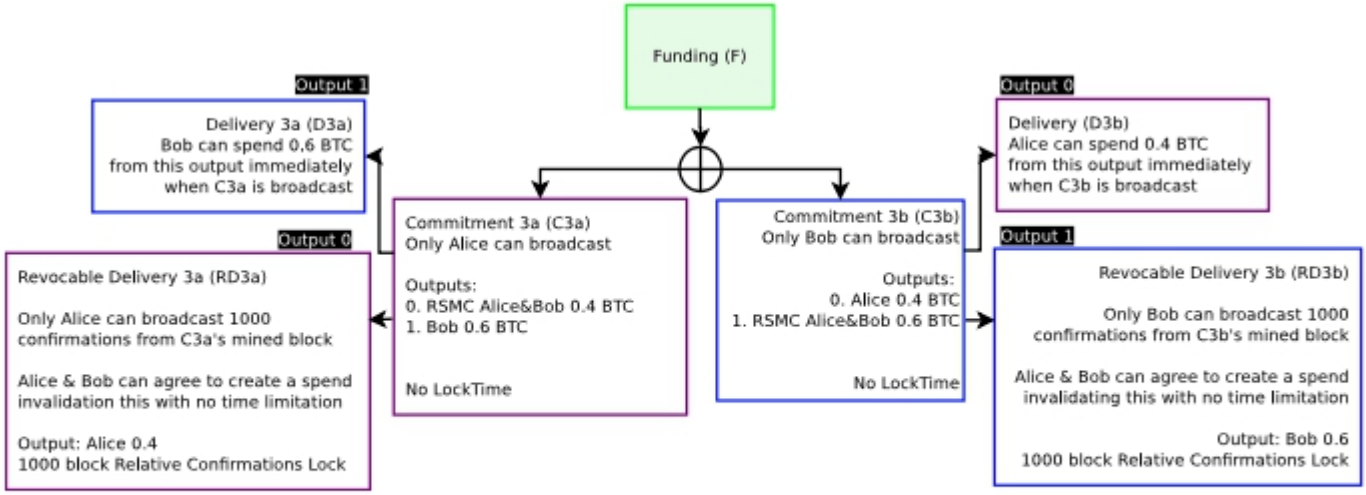


Figure 13: Since Bob proved to Alice he knows R by telling Alice R, Alice is willing to update the balance with a new Commitment Transaction. The payout will be the same whether C2 or C3 is broadcast at this time.

Similarly, if the recipient is not able to prove knowledge of R by disclosing R, both parties should agree to terminate the HTLC and create a new Commitment Transaction with the balance in the HTLC refunded to the sender

If the counterparties cannot come to an agreement or become otherwise unresponsive, they should close out the channel by broadcasting the necessary channel transactions on the Bitcoin blockchain.

However, if they are cooperative, they can do so by first generating a new Commitment Transaction with the new balances, then invalidate the prior Commitment by exchanging Breach Remedy transactions (BR2a/BR2b). Additionally, if they are terminating a particular HTLC, they should also exchange some of their own private keys used in the HTLC transactions.

For example, Alice wishes to terminate the HTLC, Alice will disclose KAlice1 and KAlice4 to Bob. Correspondingly if Bob wishes to terminate the HTLC, Bob will disclose KBob6 and KBob8 to Alice. After the private keys are disclosed to the counterparty, if Alice broadcasts C2a, Bob will be able to take all the funds from the HTLC immediately. If Bob broadcasts C2b, Alice will be able to take all funds from the HTLC immediately. Note that when an HTLC is terminated, the older Commitment Transaction must be revoked as well.

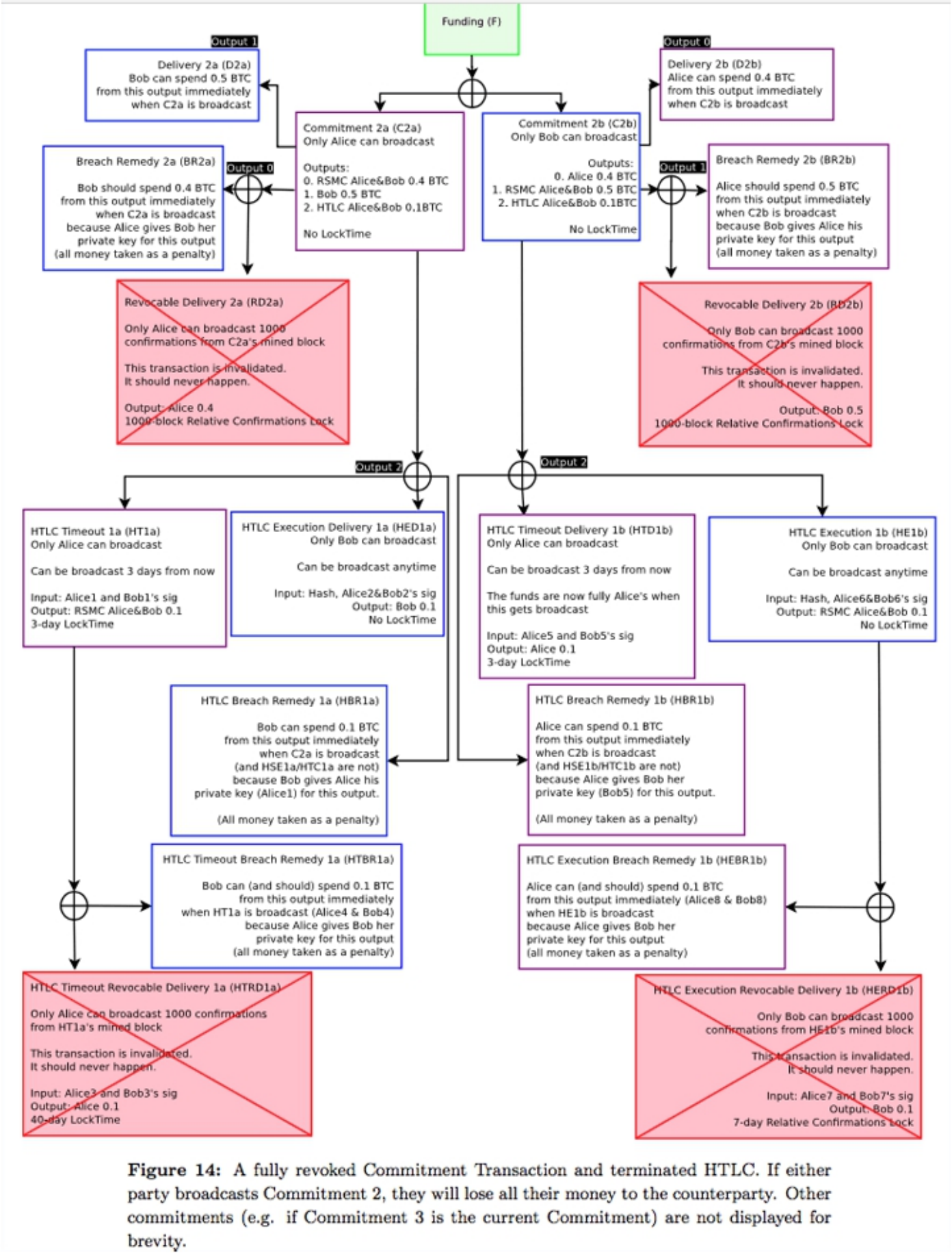


Figure 14: A fully revoked Commitment Transaction and terminated HTLC. If either party broadcasts Commitment 2, they will lose all their money to the counterparty. Other commitments (e.g. if Commitment 3 is the current Commitment) are not displayed for brevity.

Since both parties are able to prove the current state to each other, they can come to agreement on the current 79 balance inside the channel. Since they may broadcast the current state on the blockchain, they are able to come to agreement on netting out and terminating the HTLC with a new

Commitment Transaction.

TOKENIZED FUNDS

Our fund management avails of profound experience from the traditional- , as well as from the hedge fund industry. Yielding a good return is our aim as we work on making sizeable commissions and crypto earnings going forward. By choosing a tokenized investment vehicle, we bring liquidity to the angel- and venture capital market, which is otherwise a highly illiquid environment. Token holders can sell and exchange at any given time, making that alone the most significant innovation in AC & VC markets over recent decades.

Investors will benefit from managerial expertise, broad diversification, careful pre-selection and reallocation of included coins and tokens and ongoing risk management.

PCC

Consisting of a set of the 30 most valuable cryptocurrencies at time

PICO

A tokenized fund investing in the most promising ICOs

PAV

Atokenized fund investing in the most

promisingFinTechstart-ups

A simple comparison:

Ø BTC price (Bitfinex) October 4th 2017: $ 4309.60 US, January 10th 2018: $ 14435.00 US = 334.99% increase

Ø ETH price (Bitfinex) October 4th 2017: $ 291.40 US, January 10th 2018: $ 1284.00 US = 428.28%increase

Ø PCC on October 4th 2017: $ 100 US, January 10th 2018: $ 433.15 US = 433.15% increase

Ant Block Chain Protocol's Cryptocurrency fund clearly has the makings to outperform current market developments.

Risk Management

Investing into ICOs and tokenized ABCP leverages the performance of our funds compared to regular cryptocurrency investments by far. Typically, higher return means higher risk (of loss). We approach to lower the overall risk by carefully diversifying our fund's assets, maintaining high performance attributes. Selected due diligence parameters and a planned set of steps according to our risk analysis are the frameworks for our success in risk management. Nevertheless, it has to be stated, that a total loss of capital is not inevitable.

(For those interested in a deeper understanding of asset risk management, we recommend reading Quantitative Risk Management: Concepts, Techniques and Tools by Alexander J. McNeil, R ü diger Frey and Paul Embrechts, published by Princeton University Press in 2015; revised 2nd edition, 1st edition 2005)

ICO INCUBATOR, A BLOCKCHAIN ANGELIST

Selected start ups will receive management support, legal advice and financial aid to help spread (spreading) their wings as they get ready for /to take off. It is our business venture to spot, target and support potential top performers in this hyper-dynamic market. We help in developing their business ideas/ proposition and product portfolio. We support our clients using our analytical and social media marketing tools, and also by establishing communities and connecting with the press and the media. Delicately our clients are guided through crucial steps to circumvent unnecessary future red tape barriers which often impede on any positive achievement.

Our goal is to increase the success rate of highly potential ideas and future market leading developments.

PLATFORM / EXCHANGE / WALLET / MOBILE APP

The platform with its user-friendly dashboard enables quick access to a variety of features. On one hand, there is a wallet displaying the account balance, a cryptocurrency exchange and the opportunity to invest into Ant Block Chain Protocol ‘s tokenized funds and products. On the other hand there is a referral system, allowing the user to profit from his own ambition and satisfaction using peer-to-peer networking. Our easy-to-use platform will be available via web-interface and via wallets on both Android and iOS according to our roadmap timeline. Our Mobile Apps will use cutting edge native frameworks to provide a fast and natural user experience.

TOKEN SALE PROCEDURE

To fund the project and the platform, we conduct a token sale. (ICO/ITO) The ticker symbol for our company token will be ABCP. Token sale proceeds as follows.

HARD FACTS

Our crowdsale will be performed in two stages:

- Pre-ICO: November 1st – 21st 2019
- ICO: December 1st – 21st 2019

Available in ICO: 630,000 ABCP
Full supply: 2,100,000 ABCP
Hard Cap: $ 66,150 US
Accepted currencies: BTC, ETH, USDT(ERC20), LTC, BCH, EOS
Exchange Rate: 1 ABCP = $ 0.105 US

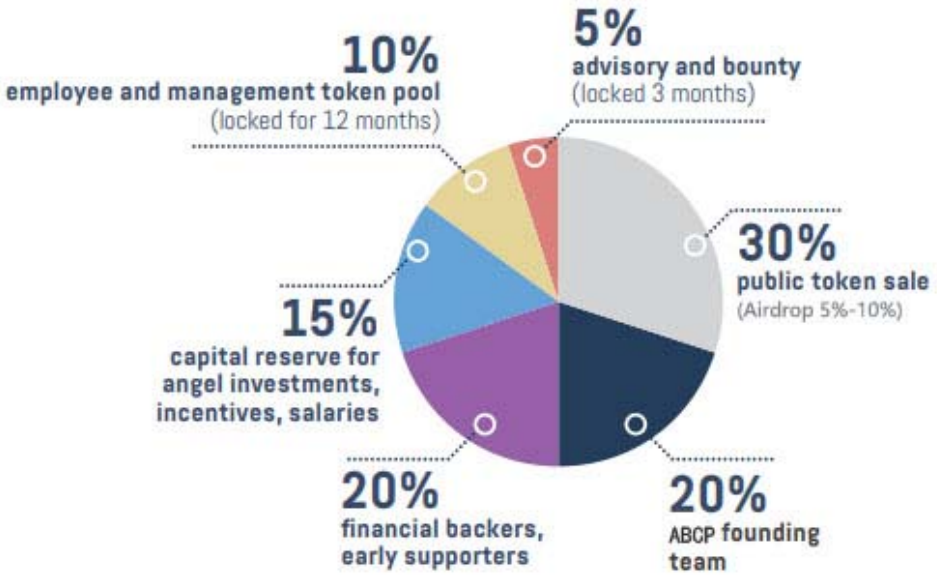
Token Utility Function: ABCPs are primarily a means of payment, „a digital coupon “, for all the services Ant Block Chain Protocol provides, especially ICO Services and Platform related fees. No other means of payment will be accepted. ABCP allows buying of funds and native tokens on platform and acts as value transfer.

No Expectation of Profits: Due to the utility nature of the ABCP tokens , it does not provide dividends, profit shares or voting rights to its holders. No guarantees or assurances are provided, theABCP token shall not be considered an investment.

Architecture: ABCP Tokens are Ethereum ERC-20 tokens

Acceptence of Tokens: ABCP Tokens are accepted on the Ant Block Chain Protocol platform solely for the time being. Since the tokens will be released after the Token Generation Event, secondary market trading is possible. Token buyers have to be aware that due the utility nature of the token a rise in value is possible however unlikely.

Thus ABCP tokens shall not be regarded as an investment. NO MORE ABCP WILL BE CREATED AFTER THIS TGE EVENT.



Allocation of tokens

• 420.000 of ABCP created during the creation event will be allocated to Ant Block Chain Protocol founders

• 420.000 ABCP will be allocated to financial backers and early stage supporters

• 210.000 of ABCP created during the creation event will be allocated to the company and utilized as a future employee and management token pool, to strengthen our ability to attract & retail top talent and contractual bonusses; these tokens will be locked for 12 months and are subject to a leak-out clause

• 1.050.000 of ABCP created during the creation event will be created for and granted to

advisors and bounty supporters; these tokens will be locked for 3 months and are subject to a leak-out clause

Capital Reserve

• 315.000.000 of ABCP will be credited to the reserve, but not issued. These ABCP´s will be available as an additional source of funding, but may never be issued, depending on circumstances in the future.

TOKEN VALUE

• Ant Block Chain Protocol’ s ABCP token is a utility token. It acts as a means of payment, a „digital coupon “, for all the services Ant Block Chain Protocol provides, especially ICO Services and Platform related fees. ABCP allows the purchase of fund tokens and native tokens on platform. The tokens shall not be confused with company shares, which they are not. They are digital goods, bearing no intrinsic value by themselves,but the value of the assets, services and/or rights which can be purchased by these tokens.

TECHNOLOGY

The asset tokens will be represented as smart contracts on the Ethereum blockchain. Coding and customizing work will be done in Solidity, a high level programming language similar to JavaScript, targeting the Ethereum Virtual Machine (EVM), which is the runtime environment for smart contracts in Ethereum.

Ant Block Chain Protocol is an ERC-20 TOKEN

• Security and predictability (as opposed to having to run an independent blockchain network);

• Use of robust and well- supported clients (Ethereum based tokens can be managed with official Ethereum clients):

• High liquidity (interchangeable with other Ethereum based tokens or Ether);

• Easier listing on exchanges with infrastructure already in place;

• Ethereum smart contracts enable a transparent and secure way of value transfer

Exchange

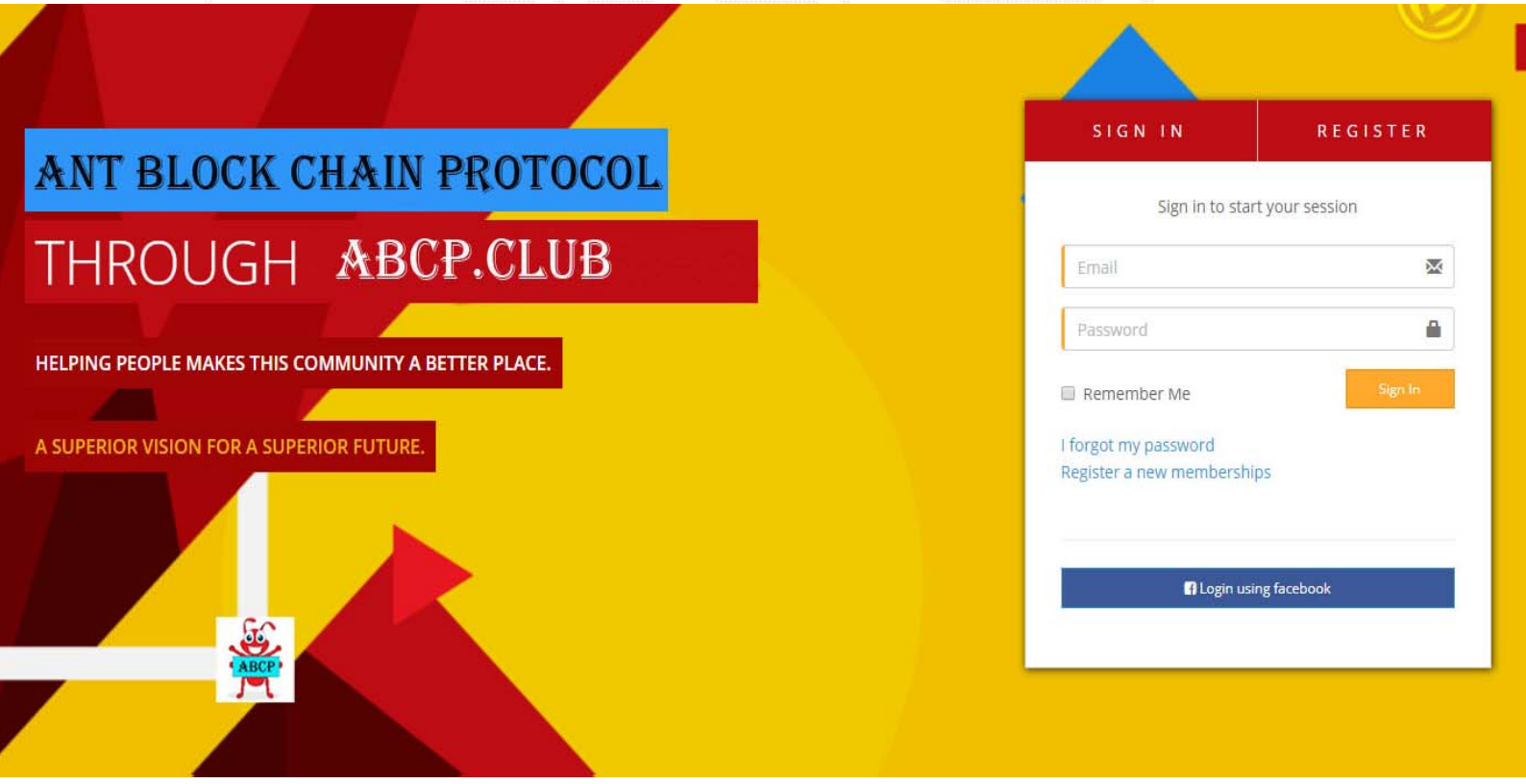
The exchange works via a matching engine through asset proxies. The interface is

therefore linked to external exchanges and addresses via the asset proxy smart contract, which complies with ERC-20 standards. This makes the exchange manipulation-safe and enables true transparency.

The software will be written in a hardware-friendly programming language to maintain speed for fast transactions. As for now, for safety reasons, we decided to keep development in a closed environment - with consideration of future decentralization.

Wallets

Ant Block Chain Protocol’s multi-wallets will accept multiple currencies and will be written in JavaScript, among other Bitcore framework. Furthermore, there will be dedicated apps for iOS and Android which assure the same functionality as the web interface. Mobile Apps will use cutting edge native frameworks to provide a fast and natural user experience.



ROADMAP

Here are some of our recent progressions towards full functionality of our platform, contract development and roll out of our campaigns.

- Created the platform with working accounts
- Gathered a 50,000 user strong member-base
- Constantly increased our product and service range
- Created our first crypto currency fund (ABCP), which outperformed all other comparable blockchain funds so far.

The future is bright and there is more to come, so please take a look on how we aim to achieve what we state.

OUR TEAM AND OUR VALUES

WE ACT WITH INTEGRITY AND SHOW RESPECT

We demonstrate a commitment to integrity and ethics. Show respect for and value all individuals for their diverse backgrounds, experiences, styles, approaches and ideas. Our brand is built on principles of trust and honesty.

WE VALUE CLARITY AND SIMPLICITY

We strive to make our products simple to understand and easy to use. We take extra care to think and communicate with absolute clarity, be it with our customers, our partners or internally. No fine print, clear words, clear actions.

WE DON’T FOLLOW WE LEAD

We push the envelope in blockchain investment products, doing things that have never been done before, in order to win the hearts and minds of our customers.

The blockchain revolution presents tremendous opportunities for the fast and the smart (see above). We will capture these opportunities for our tokenholders’ benefit for a simple reason: We are the prime mover in our field. As former securities and commercial lawyers, hedgefund managers, service industry pioneers, creative directors, and tech project managers, our big idea is to completely re-invent high-tech start-up investment – by introducing a truly decentralized investment platform: Ant Block Chain Protocol.

WE LOVE SUCCESS

We achieve results and celebrate when we do. We help people to be their best by providing coaching and feedback. We make people feel valued and appreciated. We communicate open and frequently.

Ant Block Chain Protocol team offers a unique combination of extensive business experience, software engineering skills, international perspective and hands-on attitude. We blend deep technological skills with artistic creativity and academic excellence with executive experience.

REINHAD BERGER

CHIEF EXECUTIVE OFFICER /CHIEF FINANCIAL OFFICER

Reinhad is an original FinTech veteran. Reinhad's career began as a business architect for global consulting firm Accenture contracted to redefine the global banking system for clients such as Credit Suisse, UBS and Invesco. He then served as a principal for French tech giant Capgemini. In 2003, Reinhad cofounded Alternative Invest Finance AG, an independent hedge fund operator, where he managed alternative assets in excess of $250M. Reinhard received a MSc in Computer Science, an M.A in law, as well as an MBA from Danube University.

CHRISTIAN THURNER

CHIEF OPERATING OFFICER

With more than 35 years in business, Christian is a highly experienced business organizer. As Head of Purchase, logistics and distribution for several VIVATIS HOLDING (RAIFFEISEN GROUP) companies, he acted for a variety of private banks and tax auditors. In 2015 he diversified into the field of "Consulting & Advisory" and accompanied some of the big shots within financial market, leading reputable companies to success.

DOMINIK BERGER

CHIEF MARKETING OFFICER

A young and passionate entrepreneur, founder and CEO of „ myDigital-Marketing Solutions ". Besides that he is a social media expert and strategist, coached by Tai Lopez (top social influencer and investor), and Gary Vaynerchuck. Dominik Berger knows how to effectively utilize social media and online marketing and is accountable elevating companies to the next level.

OAMA RICHSON

CONTENT CREATION & MEDIA RESEARCH

as Public relations and sciene-media communication are the strengths of this young creative mind. With a scientific background from the University of Vienna, his career led him to companies like Pfizer, ORF well as Theater an der Wien. He gathered experience as a team leader in PR-campaigns for the St.Johns Ambulance and organized several events and concerts. A cooltempered professional who is able to handle a crisis with confidence.

HEINZ LAUBERT

ANGEL INVESTOR

Serial entrepreneur in the service industry, with more than 15 years' experience. Strong focus on leadership and organization. Staged several high caliber events with 1000+ attendees. Dedicated leadership personality

MICHAEL NEUHOFER

CHIEF TECHNICAL OFFICER

a Michael is our expert in developing and securing web applications. Co-Founder and CTO of myDigital - Marketing Solutions. With bachelor's degree in software engineering and an ongoing study in cyber security he is the man for developing secure web applications. Revealing and cracking application weaknesses are his specialty

MELANIE SCHAUER

SOCIAL MEDIA RESEARCH

Melanie focuses on social media research and communication. She oversees our branding facilities and acts as quality assurance.

REINHARD WIDMEYER

SOFTWARE ENGINEERING

Reinhard‘s area of interest includes software engineering, online campaigning, user board communication, solidity (/smart contract coding).

FLORIAN ENGLMAIER

IT, PROGRAMMER

Florian is a dedicated white-hat hacker. He secures systems and assets against vulnerabilities and external intrusions. Florian has extensive experience in penetration testing, including DDoS attacks, SQL injection attacks, session hijacking and man-in-the-middle attacks.

NIMO ZIMMERHACKL

GRAPHIC ARTIST & PHOTOGRAPHER

ADVISORS & PARTNERS

JAMES VINCE

SECURITY & DIGITAL MARKETING EXPERT

James started his academic career at Cambridge University (Network Security class of 2006). In 2008, he graduated Suma Cum Laude from M.I.T. James is founder of the M.I.T Linux Society. He has over ten years of experience in finance, including experience at some of the world‘s best known investment banks. These include JPMorgan (New York, London), BNP Paribas (Paris), Societe Generale (Paris) and XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX(Tokyo). James is one of the world‘s leading crypto researchers. Whist at M.I.T., James cracked RSA 1024 in 30 minutes and offered several improvements on the algorithm. James has held security clearance level 4 (top secret level) in the UK whilst working for DEFRA and level 3 (secret level) in the US working for The Census Bureau.

THE BUSINESS CASE

OUR PRIME REVENUE STREAMS

In order to maintain a stable and growing product and service platform to provide purchasing power for the ABCP tokens, Ant Block Chain Protocol needs revenues.Ant Block Chain Protocol’s revenues come from different sources:

MAIN INCOME SOURCE:

We project the management fee to be 4%, the performance fee to be set at between 15% and 30%. This fee is perfectly on par with alternative investment manager’s industry standards.

- Fee: 4% of all assets under active management
- Estimated assets under management (2018): 500 Million US$
- Estimated revenue from asset management (2018): 20 Million US$

Gold-token-associated transaction and storage fee:

We are currently observing the markets in order to assure a competitive pricing for transaction of tokens and demurrage of physical gold. This will be a major income source due to the possible use of the token as a B2B payment token.

SECONDARY INCOME SOURCE:

Card fee:

- Fee: 4% of all assets under active management Transaction fee:
- Fee 1% of all transactions: approx. $ 600.000 US p.a.

ADDITIONAL REVENUES

Ant Block Chain Protocol intends to add additional recurring revenue streams within the next 12-18 months, including transaction and consultancy fees.

Revenues from ICO fees

ICO fees are calculated as a 25% cut of the ICO transaction volume, taken on average. We have also taken into account currency and token market growth

COMMUNICATION AND PARTNERSHIPS

Partnerships

Additionally, we will partner with innovative FinTech companies around the globe to provide our customers with the best digital investment.

We intend to be a leading digital investment platform and generate revenues with selected "best in class" partners in other financial categories (VC, hedge funds, savings, credit & loans, insurance) with an emphasis on blockchain empowered partners.

Strategic partnerships with investment funds, private banks and insurance companies will follow.

On top of that, we are proud to be chosen as one of the 12 DAA managers of Iconomi.

Live Events

• We plan on hosting different live events in various venues with regard to our core markets. These events include:

• Investor Days: Attracting potential investors at distinguished events with our products and services will add new opportunities and new relationships to Ant Block Chain Protocol

• Road shows: With the purpose of strengthening brand awareness and brand value we will chose different locations with the goal of increasing Ant Block Chain Protocol's reach in public.

• Live speeches: It is hard to compare what could be more convincing than, a good speech backed by a great product. Public events are a great way to communicate to a broad audience.

• Blockchain events: We will not miss out our chance to be represented at blockchain meet-ups all over the world in order to educate on our solutions in the world of blockchain.

• Financial events: We are in finance. In order to attract customers and clients from the old economy and open the gates to new possibilities it is of importance to connect, collaborate and co-operate.

Global Marketing:

Multi Channel marketing campaigns to be organized, designed and implemented by a team of senior professionals include

- ONLINE marketing
- DIRECT marketing
- SOCIAL marketing
- PR reach increase

SCENARIOS

Ant Block Chain Protocol‘s innovative business model is profound and elaborate. It is impossible to evaluate prospects and future financial results with absolute precision, which is why we offer three possible scenarios to help understand different levers and possible out-comes for our project:

AuM growth rate assumptions

We will grow the overall Assets under Management (AuM) progressively over the next 5 years, with a growth rate starting from 70% (year 2019), decelerating to 40% (year 2022). We expect the AuMs to be between 135 Mio US $ and 320 Mio US $, for our three scenarios. This would put us at par with a typical mid-sized hedge fund. Growing to about 1 Bn US $ at the end of the 5 year period, we would match the size criterion for a large-size hedge fund.

Market growth rates

Since the beginning of 2017, the Token market has grown at a value-weighted average of about 300 % annualized. The market will continue to grow, however at a more moderate pace.

We have therefore taken a range of growth rates from 50% to 80% per year as our yardstick. For each of our three scenarios, it is expected that the Blockchain economy grows by around 3% (Conservative), 5% (Realistic) and 7% (Optimistic) – per month for the next 5 years.

This equates to around a 40% and 80% annual growth in total assets, fueling PCO fee inflows. The ICO fees will be set at 25%, on average.

AuM Growth Rates (2018 to 2022) and ICO fees

Year	2018	2019	2020	2021	2022
Fund growth rates @	n/a	70%	60%	50%	40%
ICO fees @	25%	25%	25%	25%	25%

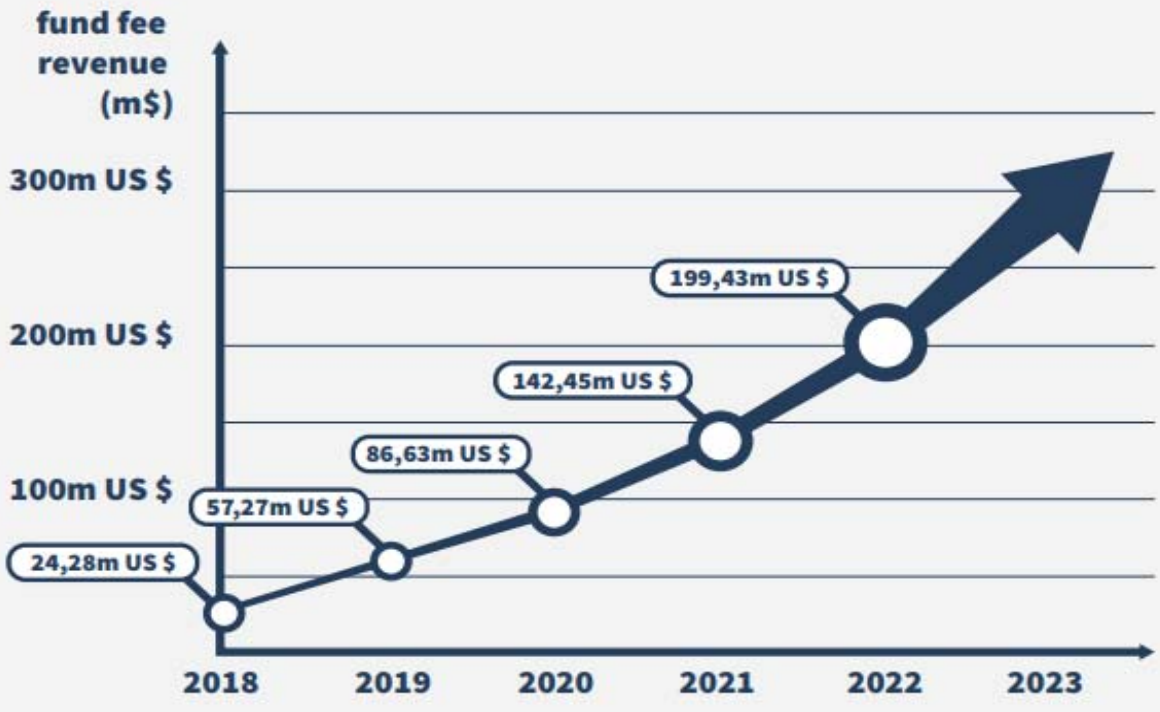
Conservative Scenario

In the conservative scenario, we modestly grow our asset base and revenue streams. We will operate in a highly competitive market space. However, we strengthen our position as the world’ s first fully decentralized investment platform by adding new fund features valued by the investment community. In this scenario, we expect AuM to be around 135 Mio US $, by the end of 2018.

	Assets under Management (AuM)			First full year fee revenues (2018E)		
	2017E	2018M	2018E	Performance fee (m$)	Management fee (m$)	Total Fund fees (m$)
Angel&Venture Fund	1	4	10	0.68	0.14	0.81
ICO Fund	3	10	25	0.66	0.22	0.88
Crypto Fund	10	40	100	1.69	0.90	2.59
Total Fund AuM	14	54	135	3.02	1.26	4.28

In the conservative scenario, we forecast only a modest increase of market and fund growth rates. The total fee revenues for the 5-year period are expected to be around 510 Mio US $

Fund fees (m$)	2018	2019	2020	2021	2022	Fund fees (m$)
	0.81	1.38	2.20	3.30	4.63	12.32
	0.88	1.50	2.39	3.59	5.03	13.39
	2.59	440	7.04	10.56	14.78	39.36
total fund fee revenue	4.28	7.27	11.63	1745	2443	65.07
# ICOs supported	4	10	15	25	35	89
avg ICO funding result	20.00	20.00	20.00	20.00	20.00	100.00
cut	5.00	5.00	5.00	5.00	5.00	25.00
total ICO revenue	20.00	50.00	75.00	125.00	175.00	445.00
total revenue (fund & ICO fees)	24.28	57.27	86.63	142.45	199.43	510.07



We estimate the Cost of Goods Sold (the variable cost) at around 20% of revenues. The sales, general & administrative cost will be around 2 Mio US $ plus 10% of revenues.

Net present value of revenues	455.44	(@ 3% discount factor)
- Cost of goods sold	91.09	(@ 20% cost factor as % of revenue)
- SG&A	47.54	(@2M fixed + 10% cost as % of revenue)
Net income	316.81	
Net present value per PCO token	3.17	(@ 100.000.000 tokens fixed supply)

Realistic Scenario

In the realistic scenario, we grow our asset base and revenue streams more dynamically. We will operate in a competitive, but expanding market space. We strengthen our position as the world’s first fully decentralized investment platform by adding new fund feature, growing the client base and capitalizing on market gains. In this scenario, we expect AuM to be around 215 Mio US $, by the end of 2018.

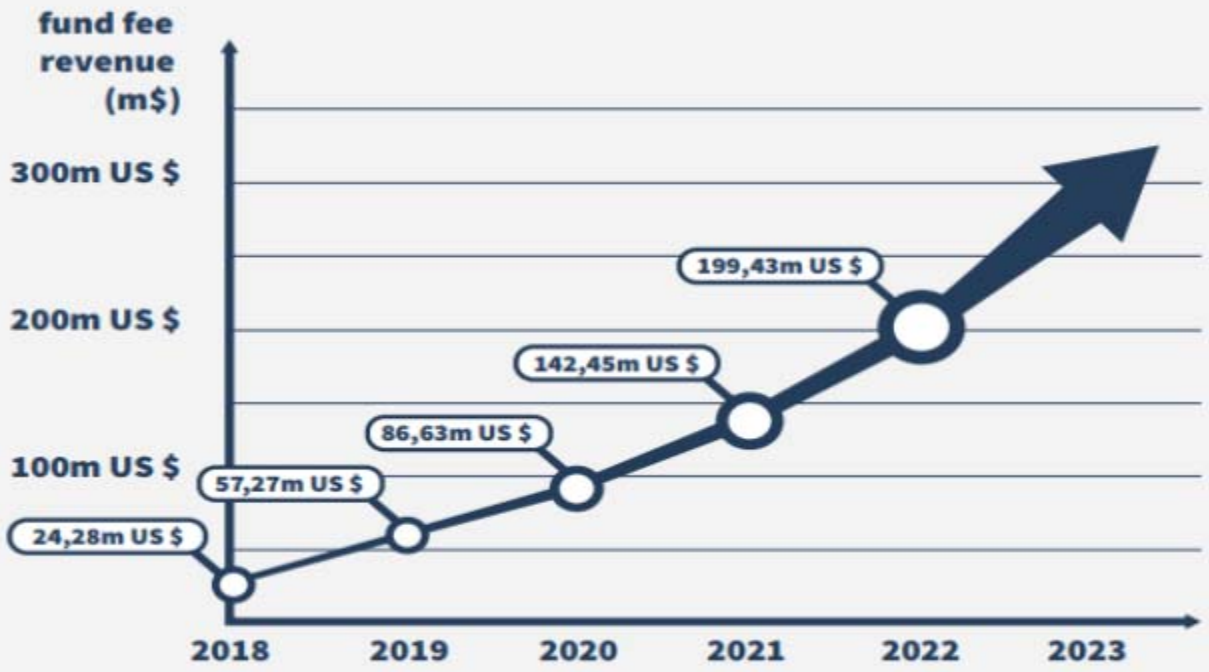
	Assets under Management (AuM)			First full year fee revenues (2018E)		
	2017E	2018M	2018E	performance fee (m$)	management fee (m$)	total fund fees (m$)
angel&venture fund	2	8	25	1.73	0.35	2.07
ICO fund	7	20	40	0.99	0.33	1.32
crypto fund	20	80	150	244	1.30	3.74
total fund AuM	29	108	215	5.15	1.98	7.13

n the realistic scenario, we forecast a progressive increase of market and fund growth rates. The total fee revenues for the 5-year period are expected to be around 733 Mio US $.

	Fund fees (m$)	Fund fees (m$)	Fund fees (m$)	Fund fees (m$)	Fund fees (m$)	
	2.07	3.52	5.63	845	11.82	3149
	1.32	2.24	3.59	5.39	7.54	20.08
	3.74	6.35	10.17	15.25	21.35	56.85
total fund fee revenue	7.13	12.12	19.39	29.08	40.71	10842

# ICOs supported	4	12	24	35	50	125
avg ICO funding result	20.00	20.00	20.00	20.00	20.00	100.00
cut	5.00	5.00	5.00	5.00	5.00	25.00
total ICO revenue	20.00	60.00	120.00	175.00	250.00	625.00

total revenue (fund & ICO fees)	27.13	72.12	139.39	204.08	290.71	733.42



We estimate the Cost of Goods Sold (the variable cost) at around 20% of revenues. The sales, general & administrative cost will be around 2 Mio US $ plus 10% of revenues.

Net present value of revenues	653.97	(@ 3% discount factor)
- Cost of goods sold	130.79	(@ 20% cost factor as % of revenue)
- SG&A	67.40	(@2M fixed + 10% cost as % of revenue)
Net income	455.78	
Net present value per PCO token	4.56	(@ 100.000.000 tokens fixed supply)

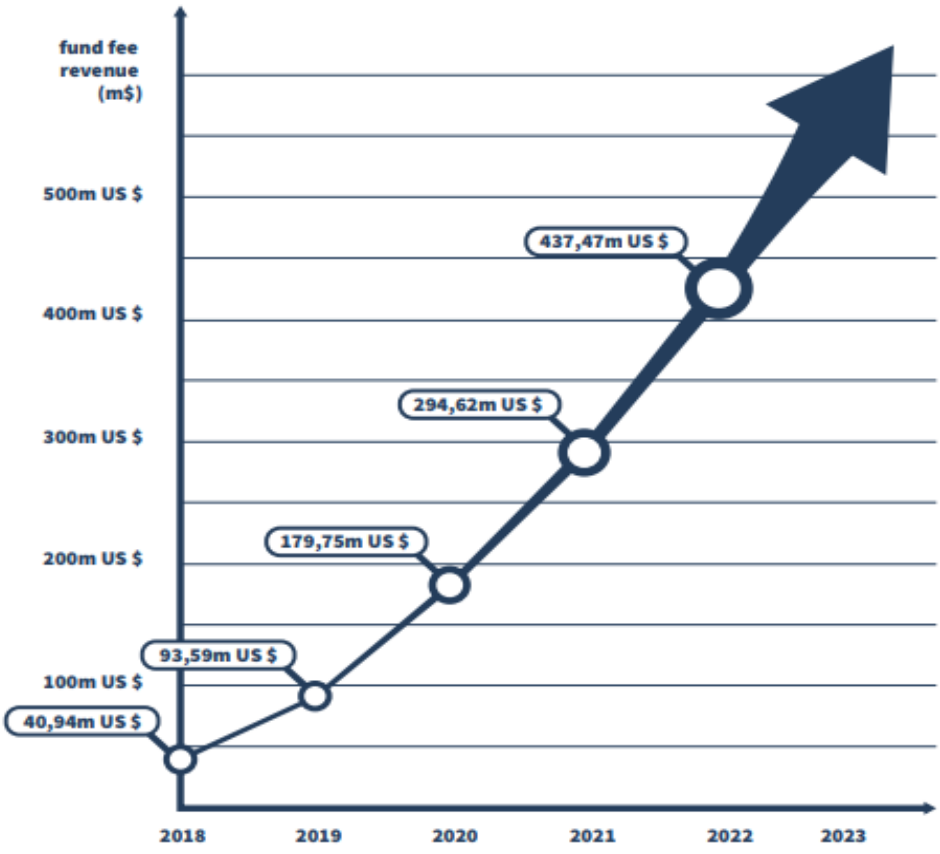
OPTIMISTIC SCENARIO

In the optimistic scenario, we grow our asset base and revenue streams still more dynamically. We will operate in a competitive, though more or less unrestricted growth market. The growth rates will resemble the dot.com market of the late 1990ies. We strengthen our position as the world’s first and leading fully decentralized investment platform by progressively growing the client base and capitalizing on significant market gains. In this scenario, we expect AuM to be around 320 Mio US $, by the end of 2018.

	Assets under Management (AuM)			First full year fee revenues (2018E)		
	2017E	2018M	2018E	Performance fee (m$)	Management fee (m$)	Total Fund fees (m$)
Angel&Venture Fund	3	12	40	2.78	0.56	3.33
ICO Fund	10	30	60	1.50	0.50	2.00
Crypto Fund	25	120	220	3.66	1.95	5.61
Total Fund AuM	38	162	320	7.93	3.01	10.94

In the optimistic scenario, we forecast a progressive increase of market and fund growth rates. The total fee revenues for the 5-year period are expected to be around 1.046 Mio US $.

	Fund fees (m$)	Fund fees (m$)	Fund fees (m$)	Fund fees (m$)	Fund fees (m$)	
	3.33	5.66	9.06	13.59	19.02	50.66
	2.00	3.40	5.44	8.16	11.42	30.42
	5.61	9.53	15.25	22.87	32.02	85.28
total fund fee revenue	10.94	18.59	29.75	44.62	62.47	166.36
# ICOs supported	6	15	30	50	75	176
avg ICO funding result	20.00	20.00	20.00	20.00	20.00	100.00
cut	5.00	5.00	5.00	5.00	5.00	25.00
total ICO revenue	30.00	75.00	150.00	250.00	375.00	880.00
total revenue (fund & ICO fees)	40.94	93.59	179.75	294.62	437.47	1,046.36



We estimate the Cost of Goods Sold (the variable cost) at around 20% of revenues. The sales, general & administrative (SG&A) cost will be around 2 Mio US $ plus 10% of revenues.

Net present value of revenues	931.59	(@ 3% discount factor)
- Cost of goods sold	186.32	(@ 20% cost factor as % of revenue)
- SG&A	95.16	(@2M fixed + 10% cost as % of revenue)
Net income	650.11	
Net present value per PCO token	6.50	(@ 100.000.000 tokens fixed supply)

CONTACT

Please don‘t hesitate to contact us if you have any further questions .
We will be happy to serve you.

Ant Block Chain Protocol INVESTMENTS - SAFE AND EASY

https://abcp.godaddysites.com/
https://www.abcp.club
https://medium.com/@abcp333
https://twitter.com/ABCP04285133
https://www.linkedin.com/in/reinhad-berger-3b597b158/

SEND US AN E-MAIL

abcp333@gmail.com
Ant Block Chain Protocol Technologies FZE
Umm Al Quwain License NO 4822
Dubai-United Arab Emirates

Office:

Business Center,
Al Shmookh Building, UAQ FTZ
Umm Al Quwain, United Arab Emirates