

Chronos: An open protocol for streaming money

Paul Berg

hello@paulrberg.com

Mark Milton

markbmilton@protonmail.com

August 17, 2018

DRAFT Version 0.2.2

Abstract




Chronos is a cryptoeconomic protocol that facilitates continuous value transfers between parties. It is designed as a trustless, decentralised and efficient system to "stream" funds by enabling transaction units per time units. The protocol is interoperable with all digital currencies in the Ethereum ecosystem including, but not limited to, ETH, DAI and all other ERC20 tokens. By adding a new, time-based dimension to money, consenting financial commitments become dramatically more versatile. Chronos leverages the power of Plasma child chains, the fluidity of state channels and versatility of Tendermint consensus, in order to create a low-latency protocol ready for commercial-scale decentralised applications.


Contents


1	Introduction and Rationale	4
2	Legend	5
2.1	Stream	5
3	Origins and Status-Quo	5
4	Limitations of Contracts	6
4.1	Ricardian Contracts	6
4.2	On-Chain Smart Contracts	7
4.3	Scenarios	7
5	Existing Work	8
5.1	Lightning Networks	9
5.2	Generalised State Channels	10
5.3	ERC948	11
5.4	Ethereum Alarm Clock	12
6	Architecture	13
6.1	Consensus	15
6.2	Application State Machine	15
6.3	Streams	16
6.4	Further Improvements	18
6.4.1	Plasma Implementation and Mechanics	18
6.4.2	Time Management	19
6.4.3	Unlimited Streaming	19
6.4.4	Spending Ongoing Streams	19
6.4.5	Nested Plasma Chains	20
6.4.6	Interoperability	20
6.4.7	WASM	20
7	Cryptoeconomics	20
7.1	Governance	21
7.2	Participants	22
7.2.1	Users	22
7.2.2	Nodes	22

8	Use Cases	22
8.1	Subscriptions	22
8.2	Salaries	22
8.3	Consultations	23
8.4	Trust Based Payments	23
8.5	Other use cases	23
9	Summary	24
10	Appendix	25
10.1	ERC20 Token	25
10.2	ERC721 Token	25
10.3	ERC948	25
10.4	Ethermint	26
11	References	27

1 Introduction and Rationale

The Internet is going through significant structural changes and power shifts. Centralised services, and their providers, are under threat of decentralisation. The emergence of a new class of disruptor, with no central management or trusted party, governed by algorithms and cryptoeconomics,  will free consumers from censorship, extortionate middleman fees, data sinkholes and privacy infringement.  

At the core of most centralised processes is the function to load-balance  money between peers (individuals, entities, machines), an industry that reaps \$2tn in revenue per annum[1]. Despite the shared vision of decentralisation, the blockchain ecosystem is tackling disintermediating payment processors in a disjointed way.

 Some blockchain payment gateways are merely upgrading elements of existing financial infrastructure with no goal of decentralisation[2]. Conversely, off-chain scaling solutions, whilst removing the middle-men from the equation, were designed for one-off payments, requiring the user to be online to sign transactions. Some of them impose the usage of specific cryptocurrencies which, in times of increased price fluctuations, are not feasible for medium-to-long term financial arrangements. There is a need for sound technical bridges to allow users to "stream" a plethora of digital assets, including stablecoins such as DAI[3] or Basis[4], removing the current volatility risk of cryptocurrencies. What would a blank-canvas solution look like?

Chronos is a protocol that challenges the assumption that money has to move in bulky chunks between parties. We aim to reduce the reliance on contracts to govern peer agreements and dispute resolution. As an alternative, the Chronos protocol embeds the promises of delayed reciprocal altruism in the continuous payment itself. The primary focus of Chronos is to create a standard for "streamed" value transfers.

2 Legend

2.1 Stream

A set of continuous value transfers over a given amount of time. Let's imagine Alice agrees to pay Bob 30 units u for a service Bob provides, but she instead pays in hourly payments of $0.0416u$. Alice can close the Stream at any time, but, naturally, Bob will simultaneously stop providing his service. This novel solution removes the necessity for parties to sign a contract in the first place, as the agreement is ingrained directly in the continuous transfers of value. It is not mandatory that Alice (1) has to be online around-the-clock and (2) she pays Bob in 1 hour time units; she can mandate 2 hours, 3 hours, 1 day or even quirky values such as 1729 minutes.

3 Origins and Status-Quo

Despite being predominantly defined as a medium¹ of exchange, money hardly behaves as such. It sits in containers, such as bank accounts and wallets, and is transacted in chunks, in both the centralised and decentralised world. As a byproduct of this, humankind developed a host of products and services dedicated to money management. Multi-billion dollar industries emerged to track the movement of chunks of money: financial tools, accounting procedures and anti-counterfeit heuristics to prevent tampering. Audits, for example, are simply a laborious reconciliation process costing companies 7-figure amounts.

Similarly, accounting software providers enjoy gargantuan gross margins, but their role is to merely simplify the mess of periodic, discrete payments - profiting from services that should not be necessary, in many scenarios, in the first place.

There is a massive gap between the promised flexibility of digital money and the supporting infrastructure. Cryptocurrencies have hinted at the huge potential to improve the way we transact, whether it be peer-to-peer,

¹As per the Oxford Dictionary, "medium" is defined as the intervening substance through which sensory impressions are conveyed or physical forces are transmitted.

cross-border or instant financial reconciliation but, even with the latest technological advances, we are still limited by the old-world perception of payments as discrete transactions. With existing blockchain and cryptocurrency infrastructure, it would be prohibitively expensive for individuals, entities and devices to engage in transactions that were not large, discrete payments.

We propose an ecosystem that allows digital-money to move continuously between peers. The goal is to rethink existing financial infrastructure that inefficiently moves money in chunks. The protocol described in this paper provides a technically secure foundation upon which a plethora of applications can be built.

4 Limitations of Contracts

The core theory behind Chronos is that wet-signed and Ricardian[5] contracts are fundamentally flawed in governing peer-to-peer financial commitments over non-trivial periods of time. On-chain smart contracts slightly improve the status-quo, but they cannot utterly discard the trust factor.

4.1 Ricardian Contracts

Although physical, wet-signed contracts date back to ancient civilisations, they are still one of the largest mechanisms preventing counterparty default. The methodology is simple: two or more parties agree on a specific set of terms, they sign and timestamp the document, and the state assures that either party can sue the counter-party if they act fraudulently.

To retrofit the archaic, pen and paper contract model to the digital world, the Ricardian contract was proposed. Ricardian contracts define the interaction between two or more peers in a series of terms and conditions - designed to be human and machine readable and digitally signed. By means of cryptographic hashes, humans and machines can also reach an agreement on what is the correct version of the contract. Notably, the Ricardian contract, including all privacy policies, requires trust: counterparties, payment services, storage companies and courts.

4.2 On-Chain Smart Contracts

A huge leap forward from the Ricardian model is the smart contract[6] paradigm in which business and legal logic is embedded into computer code. Instead of relying on humans and courts to fulfil promises and resolve disputes, the smart contract code itself is designed to be self-enforcing.

However, designing an ongoing Stream on a canonical smart contract blockchain is cumbersome, due to the inability to calculate balances based on elapsed time. We shall discuss, in the following chapters, how Chronos overcomes current limitations of on-chain smart contracts to effectively enforce long-term financial commitments at scale.

4.3 Scenarios

In a decentralised economy, with no middleman or judicial system, there are a number of outcomes for peer-to-peer transactions or interactions. Using paid software as an example:

1. Bob supplies software to Alice and demands payment upfront. Alice is exposed to the risk of the software not being delivered or not working.
2. Bob first provides the software and is exposed to the risk of Alice not paying.
3. The arrangement is governed with self-enforcing agreements and escrow accounts (On-chain contracts).
4. Payments are made in instalments, reducing total-exposed risk.

In scenarios 1 and 2, both pre-paying and post-paying Ricardian clauses fail to satisfy the risk tolerance of both parties simultaneously. One party is always at risk of a default on the engagement: not providing the software or not paying.

Receiving software is not an objective exercise - Alice's utility is subjective, based on whether Bob is providing legitimate software. Encoding this situation into a smart contract would be impractical, as many disputes would be based around a subjective view of the service. Equally, a Ricardian contract would lack the self-enforceability required to remove third parties or escrow agents. Thus, scenario 3 stands so long as the parties believe the

agreement is mutually beneficial and not breached by either party. In order for a smart contract to enforce anything, both parties must stand to lose a deposit (asset/money), which is held in escrow. The smart contract must contain rules about who will get what under every condition, which proves to be severely limited.

Firstly, several use cases for smart contracts and Ricardian contracts are not feasible due to enforcement limitations. This, in part, is due to the ambiguity and vagueness of current law, making it highly unlikely that smart contracts can be developed that contain provisions for every possible outcome of the interaction between Alice and Bob. Code cannot be law all of the time.

Secondly, in order to understand where the contract has been breached, some propose the use of data oracles[7] as unbiased, third-party data sources. This both introduces a new third-party as well as removing the self-enforcing nature of the smart contract. It is also implausible to rely on third parties to accurately assess the subjective reaction of Alice when she consumes a service.

Thirdly, in the likely event of a dispute, a third-party (arbiter) can be introduced to pass final judgement. This is possible due to the ability of smart contracts to hold funds in escrow, but the arbitration process adds overhead costs and it is not invulnerable to collusion.

We argue that scenario 4 aligns the incentives of all participants in the most efficient way, setting recurring intervals of payment to significantly reduce the risk of default. The additional optimisation proposed by Chronos is to make the recurring intervals infinitesimally small, hence enabling the ability to "stream" money.

5 Existing Work

As of the writing of this paper, decentralised infrastructure is still in its infancy. It is not fit to cope with billions of interactions per day, removing hugely profitable centralised parties, nor overthrowing the daily

operations of the judiciary system. Ethereum, the most prominent platform for Turing-complete smart contracts, suffers from scalability issues². To alleviate this, "Layer 2" solutions like the Lightning Network[8] or Raiden[9] or Liquidity[10] have emerged, but they are not suited to processing native time-based payments.

By "Layer 2", we mostly refer to off-chain communication via state channels. Parties can transact and update the state of their balances outside the blockchain. In order to guarantee the safety of funds whilst transacting, sound economic mechanisms are mandated. That is, if a party acts maliciously and tries to cheat by broadcasting an old transaction when they had more money, the counterparty is able to cryptographically prove the misbehaviour and receive all the funds in the balance sheet.

State channel heuristics are instrumental in scaling decentralised platforms, as state updates can be processed extremely cheaply. As a result, noted sub-features of off-chain solutions include the ability to process micropayments and continuous payments unidirectionally and bidirectionally. However, as explored in this chapter, current Layer 2 solutions are not specifically designed as continuous payment protocols and come with a number of technical shortcomings that prevent the efficient and granular streaming of value transfers.

5.1 Lightning Networks

The Lightning Network offers a new perspective on the future of Bitcoin. It covers the aforementioned state channels and, in addition to that, the idea of "payment hubs": entities with a deluge of funds and active state channels that can wire payments for other participants, in exchange for small fees. Hubs are ideal for micropayments and unit-based transactions, but not so much for time-based subscriptions as there are several uncertainties regarding the liquidity available in the network. Concerns of money time-value can also be brought up.

²At the time of writing this paper, Ethereum can handle roughly 15 transactions per second

A list of compelling reasons against running a Stream on the Lightning Network:

1. Users are required to be online to sign transactions, making continuous payments unfeasible.
2. Users need to monitor the blockchain to prevent malicious counterparty behaviour.
3. Funds may be wired through a large network of nodes, limiting the liquidity and availability that is normally required to keep the Stream active.
4. Users are forced to use Bitcoin and its forks, which are exceptionally volatile. This volatility can be cumbersome to users, leading to a wild variance in value over long-term financial commitments.
5. Bitcoin is one of the least private currencies in the world, due to the public nature of its ledger. Digital forensics analysis can be performed, proving it is possible to identify individuals from their addresses.

Ethereum-based lightning networks including, but not limited to, Raiden and Liquidity, are confronted with similar issues. However, unlike the Bitcoin Lightning Network, Ethereum based solutions benefit from integrating ERC20 tokens.

Despite the great amount of interest in designing solutions capable of enabling micropayments, we posit that time-based subscriptions are superior, as they represent an accounting relief for end users compared to pay-as-you-consume Layer 2 solutions. The use of micropayments incurs a heavy mental accounting cost[11] to users due to reduced certainty about one's cashflow and non-trivial overhead costs in accurately gauging the attributes of a product.

5.2 Generalised State Channels

Several methodologies to create generalised state channels have been proposed, the most notable are Perun[12] and Counterfactual[13]. Their goal is to allow users to add new functionality to a state channel without paying a fee on the root chain.

Perun and Counterfactual create a modular structure within the state channel. On Perun the state of a multi-state channel is split into "substates" that can be updated and disputed in parallel. Likewise, Counterfactual splits Counterfactual states within a state channel into "Counterfactually Instantiated Contracts".

Both approaches are designed to allow Substates and Counterfactual Instantiation to be updated off-chain without an on-chain transaction - a paradigm which allows the user greater flexibility to install new applications at a lower cost. Thus, the user can create a multitude of Substates or Counterfactually Instantiated Contracts on the same state-channel without paying a fee on the root chain. For example, a chess game between Alice and Bob may be staking ETH, but both parties wish to add a go game to the same channel, with DAI staked, as well as opening an ongoing coffee tab. These sub-contracts are added to the state channel without opening a new channel between the parties and recording it on the main chain. Intuitively, it can be observed that generalised state channels work best for behaviour which drives dynamic, unpredictable state updates.

Conversely, Chronos is focused primarily on deterministic, single-service, ongoing value transfers. Long-term financial commitments undergo logistics which are hard to replicate on the native EVM. The closest we can get to the concept of a Stream, as defined in this paper, is to use the ERC20 *approve*³ function. This heuristic withstands a significant systemic drawback: at any point in time, Bob can withdraw the whole amount of tokens Alice allowed him to withdraw. This places trust in Bob, a fiduciary circumstance which we aim to avoid by any means. We shall see in the following chapters how a Plasma child chain, with an appended custom EVM, eradicates the aforementioned issues by handling the interactions between Alice and Bob.

5.3 ERC948

ERC948 is an open discussion around implementing recurring subscription models directly onto the Ethereum main chain. Some notable projects

³See ERC20 in Appendix. "approve" is a function which allows an account on Ethereum to withdraw tokens from another account up to a specific amount.

pioneering this approach are 8x[14] and Groundhog, which focus on traditional monthly payments between customers and businesses.

Chronos, on the other hand, is a streaming protocol between peers; that is, continuous value transfers need not be tied precisely to monthly business subscriptions. There are certain limitations with the EVM, sparsely discussed up to this point, which led us to implement Chronos on an off-chain Plasma child chain.

Although ERC948 and Chronos share some common goals, we posit that infinitesimally small, recurring value transfers offer a new perspective on long-term financial commitments. Strictly referring to the customer-business relationship, individuals are now empowered with greater control and flexibility, while businesses can reach larger audiences (as the cost of the software is not contingent on fixed periods).

5.4 Ethereum Alarm Clock

The Ethereum Alarm Clock[15] (EAC) enables the scheduling of transactions on the canonical Ethereum blockchain. While this is definitely useful for some applications, such as event-constrained payment windows, the protocol is not scalable enough to achieve granular recurring payments due to fees becoming too high. That is, scheduling a payment worth k once per hour over the next week would require a payment of $168*k$ upfront.

EAC works by generating a contract for the user and including data that can be later picked up and executed by any server adhering to the protocol. The server receives the fee included in the data, and the user gets their transaction executed at a later point in time. However, two assumptions have to be made. Firstly, the average fee on Ethereum, at the time when the contract was created, has to be similar to the average fee at the moment when the transaction is picked up and, secondly, there are enough servers monitoring the blockchain for scheduled transactions.

6 Architecture

Leveraging the security of the canonical Ethereum blockchain, Chronos provides high throughput and low fees via a Plasma Cash[16][17] implementation and state channels. Using Tendermint[18] as the elected consensus mechanism, we inherit two prime technical components: a blockchain consensus engine and a generic application interface. We shall describe them in the ensuing sections.

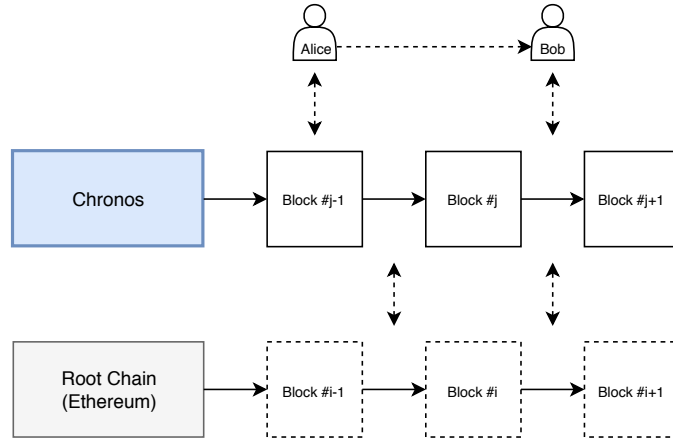


Figure 1: The overall architecture of Chronos, which periodically communicates with Ethereum via a Plasma smart contract.

Chronos allows the creation of unidirectional⁴ state channels. Similarly to how Raiden built its own communication blueprints, parties engaged in a Stream have a common building block upon which they can mutate the state and ensure that the chain will act as an adjudication court, given nefarious counterparties. The key observation is that the adjudication court here is Chronos itself.

⁴Funds can only flow from Alice to Bob, but never the other way around. Bob is able to propose updates though (price, duration et al).

Due to the flexibility of Plasma child chains, entities with sufficient resources can spawn their own private blockchain and publish updates to the Chronos "main" chain, rather than defaulting to a unidirectional state channel. It is likely that these blockchains would be forks of Chronos, allowing said entities to avoid spending fees, by aggregating their transactions. It would simultaneously afford increased privacy for their counterparties.

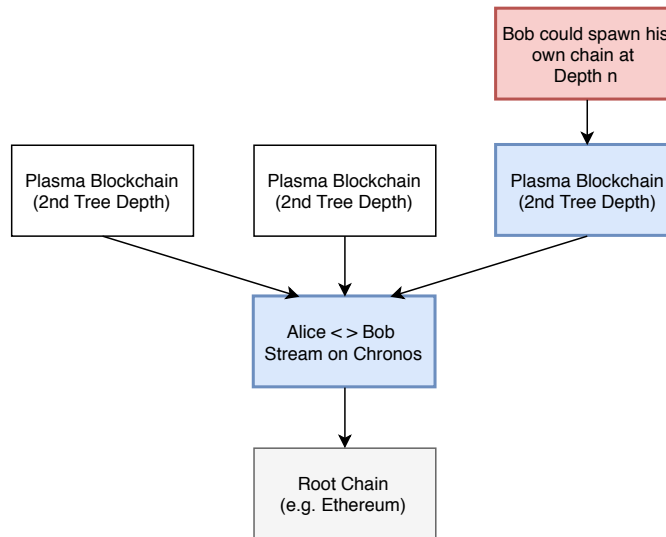


Figure 2: Spawning child chains on Plasma. Chronos is a Layer 2 solution on Plasma and can be intertwined with unlimited spawned child chains. Entities can spawn private blockchains at arbitrary depths.

Although there is no limit, theoretically, to the depth at which blockchains can be spawned, Chronos will not prioritise nested chains in the first versions of the protocol.

6.1 Consensus

Chronos is a secure and consistent state replication machine, with blocks achieving fast finality in seconds. It is Byzantine Fault Tolerant (BFT), thus it stays functional if less than a third of the nodes fail in arbitrary ways. Proof of Stake (PoS) is used as a Sybil control mechanism. In order to run on a Validator node, one needs to own CHR tokens and hold them as collateral against misbehaviour.

If the protocol fails to prevent Byzantine nodes from controlling more than a third of the network, users are always able to default (i.e. get their funds back) on the main Ethereum chain. The exit is accomplished by publishing the last two transactions in the coin's ownership history.

6.2 Application State Machine

The Application Blockchain Interface (ABCI)⁵ empowers developers to forge a state machine in any programming language. Due to its indisputable large-scale adoption, we chose to build a flavour of the Ethereum Virtual Machine (EVM), which is optimised to have the minimum requirements to perform streamed value transfers. That is, less OPCODES and less overhead computations for nodes. Performance and fees are similar to Ethermint, a fork of Ethereum with its consensus engine replaced by Tendermint and PoS.

⁵The ancillary technical component of Tendermint.

6.3 Streams

Streams can be viewed as cryptographic timestamps that Chronos understands, to the extent that, at any arbitrary point in time, balances are not truly reflected by the ERC 721s held in the ledger. As they only reflect the balances at the time of initiating the Stream, an "hourglass" effect can be observed. The mandated mechanics underlie a simple mathematical formula based on time, which designates the specific amount of funds the recipient should receive at the end of the Stream. Block height correctness is assumed to validate data and state mutations.

Table 1: Streams are composed of five fundamental parameters.

Symbol	Name	Data Type	Description
tid	tokenId	uin256	Id of the Plasma coin
p	price	uint256	Price per time unit
i	interval	uint32	Recurring payment interval in block height distance
s	startTime	uint32	Stream starting block height
c	closeTime	uint32	Stream closing block height

Below is the general sequence of steps used for initiating a Stream between Alice, the payer, and Bob, the payee. Funds can only flow from Alice to Bob, but Bob can also propose state updates.

1. At time s , Alice and Bob agree on a price p , payment interval i and stream closing time c
2. Alice deposits funds worth:

$$p * ((c - s) / i) \quad (1)$$

3. Alice now controls a coin cid representing her funds on the root chain
4. Alice and Bob initiate the Stream by creating a pseudo-escrow account
5. Alice and Bob can optionally mutate the state to p' , i' and c' and later hand over a cryptographic proof which redeems:

$$p' * ((c' - s) / i') \quad (2)$$

6. Stream is closed by Alice, before or when block c is generated, or by Bob, only when or after block c block is generated

The initial transaction, outlining the chosen parameters of the agreement, is logged on the Chronos ledger, incurring a small fee to the users. As to which party incurs this fee, it depends on the nature of the transaction (e.g. is the fee absorbed by Bob as part of the contract, is it split between peers in a peer-to-peer agreement or it is incurred by Alice). This is not something that is stipulated by the protocol.

It is important to note that the result of equation (1) cannot be bigger than equation (2)'s. That is, changes in the price, interval and time-to-live can be made but, when plugged in the formula, the total price paid must be lower than the initial value. The reason for this is simple: Alice only has the Plasma coin put in pseudo-escrow and she cannot go beyond that.

If Alice closes the Stream before time c , the protocol performs an atomic swap by making Bob give Alice her chargeback in the form of a Plasma coin (it's assumed Bob has additional funding on the root chain if he doesn't have an exact denomination on the side chain). Otherwise, the stream is closed normally and Bob receives *cid* without any further obligations. Here's a visual representation.

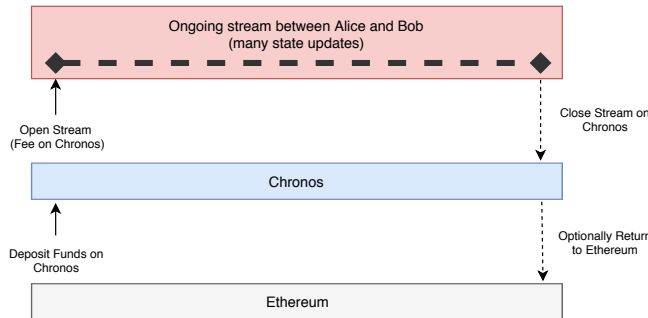


Figure 3: A visual representation of a Stream between Alice and Bob. If Alice does not already have funds on Chronos, she can deposit using her Ethereum account. An unlimited number of state updates can take place between the two parties. Importantly, the parties are under no obligation to close the stream on the root chain.

All the processes described up until now take place directly on Chronos, inside its consensus engine and application state machine. Although they are able to, participants do not necessarily need to return to Ethereum

when the Stream ends. Prolonging the stay on Chronos means lower fees when opening future Streams.

In the event of chain halting, Stream states are lost. Users can mass exit to the root chain, but balances are distributed according to the each Stream’s state at either initialisation or most recent mutation time. Alice and Bob can reach an agreement to fairly distribute funds.

6.4 Further Improvements

The current version of this paper provides a non-exhaustive introduction to Chronos.

6.4.1 Plasma Implementation and Mechanics

We chose Plasma Cash as the underlying design pattern due to its efficiency in managing off-chain states and exits. However, several other proposals⁶ are an ongoing focus of research and we might redraft some heuristics described in chapter 6. Plasma, whilst a cleverly designed scalability solution, requires active coordination across blockchain communities to forge the best implementation.

As it concerns the Stream mechanics, alternatives were considered. Instead of embedding a formula paradigm in the consensus engine, one could pre-sign all transactions for the desired streaming period. That is, we have a *txs* array whose length l is given by:

$$f(i) = (c - s)/i \tag{3}$$

Each of these transactions has a start and close time block height. The outputs in each transaction are incrementally larger: transaction $i + 1$ has an output greater than transaction i by $p * i$. The reasoning is that transaction i is obsolete at the time transaction $i + 1$ can be broadcasted.

Although this methodology will lower the risks involved with chain halting, it poses a few drawbacks:

⁶Notable examples would be Plasma Debit or Xt

1. Added friction in mutating the state
2. Transactions become a storage burden:

$$\lim_{i \rightarrow 0} f(i) = \infty \quad (4)$$

6.4.2 Time Management

The protocol assumes block height correctness to validate data and state mutations. As Chronos is a fast finality blockchain, payment intervals can be granular - as low as a few seconds. However, to ensure stability and a steady growth, the first iterations of the protocol will only allow a minimum of 24 hours or similar.

It is crucial for Chronos to prevent Stream payees from colluding with Validators. Game-theoretical slashing mechanisms are in place but further research on stricter consensus mechanisms, such as Casper⁷, has to be conducted to ensure maximum safety.

6.4.3 Unlimited Streaming

Thus far, we described a protocol that allows continuous value transfers over a defined period of time. What if the user wishes to stream money indefinitely?

If we define "indefinitely" as simply a large period of time, e.g. a year, the solution is to deposit more on Chronos, potentially splitting the funds into multiple Plasma coin. It is still a fixed time-to-live, with users only topping up once per desired streaming period. We posit that this scenario suffices for most users. Truly indefinite Streams, that is, Bob receives payments from Alice for the whole duration of her life (and even beyond that), are not really sound. Annual subscription revisions do not represent an accounting burden to the user.

6.4.4 Spending Ongoing Streams

Chronos does not currently allow the spending of incoming payments, as redeeming a Stream will automatically cancel it. However, provided that

⁷The Ethereum PoS implementation.

both parties are online, they are able to atomically cancel and initiate a new Stream. Solutions to securely spend ongoing Streams are contingent on the Plasma implementation, but we envision heuristics that allow Bob to publish a proof and redeem a coin which is subtracted from the streaming balance. Liquidity providers could provide this service and receive a fee for it.

6.4.5 Nested Plasma Chains

Plasma allows the creation of blockchains which have parent-child relationships. Although there is no theoretical limit on the depth at which these blockchains can be spawned, the Chronos protocol will not provide any SDK or tools for creating such blockchains. Not in the foreseeable future, at least. The bulk of the effort will go towards designing and securing the Plasma implementation.

6.4.6 Interoperability

We are monitoring the evolution of Cosmos[19] and Polkadot[20] because interoperability is an important stepping stone for distributed ledgers. In relation to Chronos, more intertwined blockchains means a diverse set of digital currencies available to stream. MakerDao is a successful project, but it is fairly experimental. Adding more stablecoins would bring an extra layer of security.

6.4.7 WASM

Chronos uses a flavour of the EVM that has the consensus engine replaced by Tendermint with PoS. It is reasonably similar to Ethermint. That said, Web Assembly presents itself as an interesting prospect which could offer a host of benefits to a blockchain not yet launched.

7 Cryptoeconomics

The protocol's native currency, the CHR token, aligns the incentives[21] of all rational economic agents involved the ecosystem: users, validators and foundation members. Although it is a Plasma child chain, Chronos behaves

as a cryptoeconomic protocol, as it maintains its own virtual machine and runs a BFT network of validators.

Without strong economic fundamentals, it is not possible to avoid centralisation, as founders would be the only participants incentivised to keep validating transactions. Parties may want to transact in different cryptocurrencies, use incompatible wallets and be physically located in separate jurisdictions, hence the motivation to create an ubiquitous token to coordinate all parties involved in streaming money.

Fees for interacting with the Chronos ledger, which includes Stream instantiations, are paid in CHR, but the protocol does not require users to manually acquire the token. Nodes will abstract away the conversion, as they collect fees in CHR and simultaneously hold ETH to post updates on Ethereum.

7.1 Governance

In addition to paying fees, CHR grants its owner voting weight in governing Chronos. Designing the governance structure of the Chronos protocol is an ongoing priority. Inputs to the decision making process will be multifaceted, including consensus amongst the core development team, weighted token holder voting, user polls and deviation from the stated roadmap and vision taken into account.

It is quixotic to assume there will not be a need for backward-incompatible upgrades, hard forks et al. Development trade-offs of this nature highlight the need for the token to coordinate decisions among participants with skin in the game[22].

The Chronos core team will spearhead the technical development in the early stages, but our aim is to become a decentralised autonomous organisation(DAO) as soon as we have a fully functioning main-net and a formalised, robust governance structure.

7.2 Participants

7.2.1 Users

Users will hardly interact directly with Chronos or any software developed by our foundation, as the goal is to integrate the protocol in as many digital wallets as possible. Thanks to the compatibility with Ethereum, this will be a trivial task.

7.2.2 Nodes

A node, or a Validator, is a participator in the consensus protocol which synchronises the ledger and broadcasts both cryptographic signatures and blocks containing transactions.

8 Use Cases

The Chronos protocol is designed to be agnostic to any industry or application and is suitable to current business models and economies as well as the economies of the future (decentralised, subscriptions, IoT, cross-border). Below are a few practical use cases that our team have researched.

8.1 Subscriptions

In both the centralised and decentralised economy, Chronos gives users the ability to consent to ongoing financial commitments on a regular time-schedule without the burden of maintaining an online presence to facilitate transactions. The ability to cancel subscriptions at any point, makes lock-in and cancellation less pertinent pain points for consumers.

8.2 Salaries

Companies will be able to use the Chronos protocol to create a continuous flow of money between their payroll function and their employees. With the Chronos protocol, employees can set their own rate of receiving their salary, from monthly payments down to each minute or second.

8.3 Consultations

Centralised matching platforms have been created to mitigate counterparty risk, but they enforce fees into the double-digits. Using Chronos, consultants, freelancers, computer scientists, lawyers or doctors, can start Streams with customers, which will begin draining a few monetary units from their account at an agreed rate. Streaming will remove counterparty risk and the payer can end the Stream at any time with fair value already exchanged.

8.4 Trust Based Payments

One-off payments to charities, local governments or fundraising projects risk funds being utilised in a non-preferred way. With Chronos, Streams are activated and closed based on your trust that the other party is upholding their commitment to a cause. For example, a friend running a marathon for charity may receive a Stream dependent on their training regime. Local government may receive a Stream whilst they are focusing on reducing homelessness.

8.5 Other use cases

There is no hard and fast rule as to what could be built using the Chronos protocol. We anticipate third parties developing a wide range of solutions that enable the continuous flow of money between two or more parties. We equally expect existing entities, both centralised, decentralised, fiat and crypto, to integrate Chronos as a new payment mechanism.

9 Summary

Chronos is a protocol that facilitates continuous value transfers called Streams. The core focus of the protocol is to:

- Standardise continuous payments over non-trivial periods of time
- Enable the usage of all digital currencies compatible with Ethereum, with a clear focus on stable coins
- Implement Plasmatic modularity and efficacy
- Reduce the usage of discrete payments for long-term financial commitments

10 Appendix

10.1 ERC20 Token

[ERC20](#) establishes a standard building block for tokens in the Ethereum ecosystem and has become the de facto representation for all fungible digital assets. ERC20 tokens share the same contract interface, simplifying integration with external applications and contracts.

Core ERC20 functions include:

- `transfer(to, value)`
- `balanceOf(owner)`
- `approve(spender, value)`
- `allowance(owner, spender)`
- `transferFrom(from, to, value)`

10.2 ERC948

[ERC948](#) is an open discussion and thought debate on implementing recurring subscription models directly onto the Ethereum blockchain. It is focused on customer-business relationships. As of the writing of this paper, the standard implementation has not yet been drafted.

10.3 Ethermint

[Ethermint](#) is an implementation of the Ethereum Virtual Machine that leverages the Tendermint consensus and uses Proof of Stake as the elected Sybil control mechanism. It was designed to be a part of Cosmos.

Primary Ethermint goals are:

1. Hard Spoons, i.e. the ability to port a token from the Ethereum canonical blockchain and shift the account balances over to another network
2. Web3 compatibility for easier integration with many existing dApps

11 References

- [1] McKinsey. Global Payments: Strong Fundamentals Despite Uncertain Times. 2016.
- [2] David Schwartz, Noah Youngs, Arthur Britto. The Ripple Protocol Consensus Algorithm. https://ripple.com/files/ripple_consensus_whitepaper.pdf.
- [3] Dai: A Decentralised Stablecoin. <https://makerdao.com/whitepaper/DaiDec17WP.pdf>.
- [4] Nader Al-Naji, Josh Chen, Lawrence Diao. . https://www.basis.io/basis_whitepaper_en.pdf.
- [5] Ian Griggs. The Ricardian Contract. http://iang.org/papers/ricardian_contract.html.
- [6] Nick Szabo. Smart Contracts. <http://www.fon.hum.uva.nl/rob/Courses/InformationInSpeech/CDROM/Literature/LOTwinterschool2006/szabo.best.vwh.net/smart.contracts.html>.
- [7] Jack Peterson, Joseph Krug, Micah Zoltu, Austin K. Williams, and Stephanie Alexander. Augur: a Decentralized Oracle and Prediction Market Platform. <https://www.augur.net/whitepaper.pdf>.
- [8] Joseph Poon, Thaddeus Dryja. The Bitcoin Lightning Network: Scalable Off-Chain Instant Payments. <http://lightning.network/lightning-network-paper.pdf>.
- [9] Raiden: What is the Raiden Network? <https://raiden.network/101.html/>.
- [10] Liquidity Network: Blockchain Payments for Everyone. https://liquidity.network/whitepaper_Liquidity_Network.pdf.
- [11] Nick Szabo. Micropayments and Mental Transaction Costs. <https://nakamotoinstitute.org/static/docs/micropayments-and-mental-transaction-costs.pdf>.

- [12] Stefan Dziembowski and Sebastian Faust and Kristina Hostakova. Foundations of State Channel Networks. Cryptology ePrint Archive, Report 2018/320, 2018. <https://eprint.iacr.org/2018/320>.
- [13] Jeff Coleman, Liam Horne, Li Xuanji. Counterfactual: Generalized State Channels. <https://l4.ventures/papers/statechannels.pdf>.
- [14] Kerman Kohli, Kevin Zheng. 8x Protocol: Decentralised recurring payments on the Ethereum blockchain. <https://rawcdn.githack.com/8xprotocol/whitepaper/master/latest.pdf>.
- [15] Piper Merriam. Ethereum Alarm Clock Documentation. <https://media.readthedocs.org/pdf/ethereum-alarm-clock-service/latest/ethereum-alarm-clock-service.pdf>.
- [16] Joseph Poon, Vitalik Buterin. Plasma: Scalable Autonomous Smart Contracts. <https://plasma.io/plasma.pdf/>.
- [17] Vitalik Buterin, Karl Floersch. Plasma Cash: Plasma with much less per-user data checking. <https://ethresear.ch/t/plasma-cash-plasma-with-much-less-per-user-data-checking/1298>.
- [18] Jae Kwon. Tendermint: Consensus without Mining. <https://tendermint.readthedocs.io/projects/tools/en/master/introduction.html>.
- [19] Jae Kwon, Ethan Buchman. Cosmos: A Network of Distributed Ledgers. <https://github.com/cosmos/cosmos/blob/master/WHITEPAPER.md>.
- [20] Dr. Gavin Wood. Polkadot: Vision for a Heterogenous Multi-Chain Framework. <https://polkadot.network/PolkaDotPaper.pdf>.
- [21] Fred Ehrsam. Value of the Token Model. <https://medium.com/@FEhrsam/value-of-the-token-model-6c65f09bcba8>.
- [22] Nassim Nicholas Taleb. *Skin in the Game*. Allen Lane, 2018.