Plasm Project Draft.

Takumi Yamashita takumi@stake.co.jp, Sota Watanabe sota@stake.co.jp

Sep 28, 2019

Abstract

Reminder: This is the first draft. We will polish this draft to make it understandable and professional.

Our mission is to realize Web3.0 by providing a scalable and decentralized platform for all developers. In this document, we describe a purpose of Plasm project, technical details ,and how Plasm shapes the next-generation blockchain ecosystem. In addition to that, we explain the specification of three core products, "Plasm", "PlasmChain", and "Plasma as a Service."

1 Introduction

We realize Web3.0 by utilizing blockchain technologies. In the existing society, data and wealth are controlled by central authorities and history shows us that they made a monopoly structure and created the rules that justify themselves [1]. Even if the centralized authority builds a fair mechanism, this has not been verifiable because the system has no transparency. Traditionally, the protocol and application have been based on the trust of the party. On the other hand, public blockchain, a decentralized peer to peer network has decentralized governance to realize a trust-less system with high transparency and fairness without a single point of failure. In this sense, public blockchain has fault tolerance and a high tamper resistance on the system that anyone can view, verify, and operate [2].

An application is necessary when protocol developers deliver the benefits of blockchain because an application is an interface between a protocol and a user. Blockchain applications are generally called Decentralized Application (DApps). A lot of DApps have been developed and provided to users on various blockchains in the form of smart contracts and chain codes. However, the processing performance of Dapps is not high due to the distributed and redundant structure of blockchains. The transaction throughput of Ethereum[3], the largest smart contract platform, is 14 transactions per second[4]. On the other hand, VISA and Alipay, which have many users around the world, process 3,000 transactions[5] and 256,000 transactions per second[6]. It turns out, that the current performance is too inadequate for many users to get benefits from Dapps. Therefore, various scaling solutions were invented.

There are several blockchain scaling solutions. For example,

1. **SegWit** : Fixing transaction malleability by removing the signature information and storing it outside of the base transaction block [7].
2. **State channel** : Combining off-chain transactions among particular users and only the final state is committed to the main blockchain[8].
3. **Sharding :** Allowing many more transactions to be processed in parallel at the same time by making shards[9].
4. **Plasma** : Storing transactions in separate child chains and only the root hash is stored in the main chain[10].

And so on.

To Do: Clarify the sentences above.

Among all scaling solutions, we focus on layer2 solutions because public blockchain like Bitcoin and Ethereum are almost full[11]. We believe that blockchain will be used in a different way from the way we use today. The 1st layer will be a trust layer and a 2nd layer will be a transaction layer.

Among all layer2 solutions, the reason we focus on Plasma is that it is a scaling solution that is the least dependent on the processing performance of the main chain. In Plasma, an operator manages its side-chain without sacrificing decentralization. This means that many transactions can be handled in a centralized way that does not require a consensus process, but all participants on the side chain can safely exit by submitting fraud proofs. The scaling solutions used in the existing centralized system can be used as they are. Hence, it is possible to achieve high processing performance that is not feasible with a native distributed ledger. Plasma should be recognized as an indispensable technology in the future because it can dramatically improve processing performance for all distributed ledgers.

However, Plasma has several draw backs. First, there is a limitation on what can be done with Plasma Applications (Plapps). All the things that Plasma can is described with Predicate[12]. But, the Predicate is under development. **To Do: Add an explanation of Predicate.**

Second, it is more difficult to make a Plapps compared to making a traditional DApp because writing and deploying smart contracts are not enough to make a Plapps. Plasma is the complicated technical stack that consists of several components [13]. Precisely, Plasma application consists of 4 components, a smart contract on a parent chain, a child chain, an operator, and a user. We solve these two problems through **"Plasm Project"**.

"Plasm Project" has

- **Plasm** : a set of standard libraries that enable to write Predicates quickly.
- **PlaaS** : cloud service to deploy and manage the Plasma components.
- **Plasm chain** : a blockchain that is expected to be a parachain on Polkadot.

With these tools, Plasm developers can build their applications easily.

We are planning to build these systems around Polkadot[14]. Polkadot is a heterogeneous multi chain framework which empowers blockchain networks to work together under the protection of shared security. In addition, there is a framework to create blockchains called Substrate[15]. Currently, Polkadot itself and parachains are created with Substrate. In the future, we think blockchains will be paralleled simply because there is no single perfect blockchain which supports all governance models and customer's needs by itself. Hence, there are more than 900 public blockchains have been built and more and more blockchains are being created. Polkadot and Substrate empower this movement of creating the perfect custom blockchain based on the user's need. Plasma is one of the most promising domains on Polkadot as Dr.Gavin mentioned at Subzero Summit[16]. We decided to make Plasma solutions for Polkadot and Substrate.

2 Plasm Project

Plasm Project is a project which provides a method to build decentralized and scalable applications. Figure1 is the whole Plasma Project overview.

We make it happen through providing three products as follow.

2.1 Plasm Libraries

Plasm is a set of libraries to make an Plasma application.

- **Plasm**: Core functions written in Rust
- **Plasm-Client**: Client implementations written in Typescript

2.2 PlasmChain

Plasm chain is an original blockchain that is compatible with Polkadot. This means that the Plasm chain can be a parachain. It is the main chain where developer deploy Plasma applications.

2.3 PlaaS - Plasma as a Service -

PlaaS is a software that enable Plapps developers to deploy and manage their application quickly.

In the next section, we will explain one by one.

3 Plasm Libraries

Plasm libraries are a set of Substrate Runtime Module Libraries (SRML) that allow developers to add Plasma functions to their Substrate chain quickly and seamlessly. Plasm libraries have the following features.

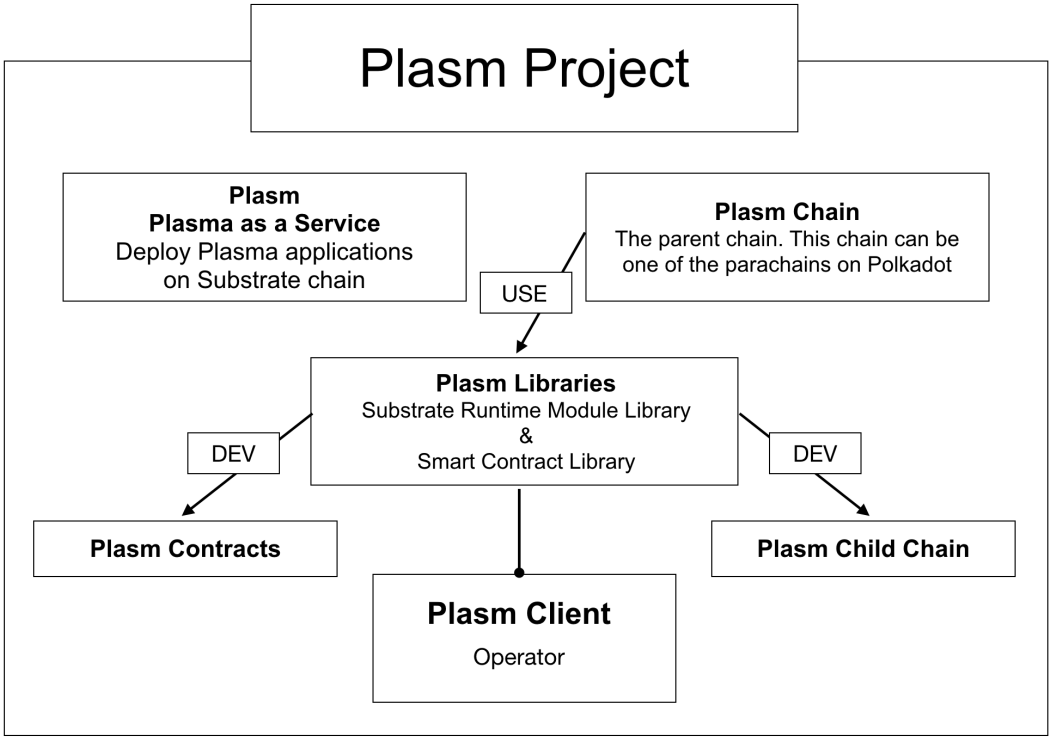

Figure 1: Plasm Project.

- The first Rust implementation of Plasma SRMLs
- Simple but versatile SRMLs and makes it easier for developers to make a Plasma chain with Substrate.
- Will Support Optimistic Virtual Machine, a virtual machine designed to support all layer 2 (L2) protocols.

Note that Substrate consists of 2 components, Substrate Core and Substrate Runtime Module Library. Developers can customize Substrate Core with SRML and make an original Substrate chain.

Currently, Plasma Group Specification (PGSpec) is implemented as a default specification of the Plasm libraries. However, PGSpec is expected to be implemented on Ethereum. Hence, the Plasm team optimized the specification for Substrate. Figure2 is the overall structure.

We implement Plasma specifications and default standards by using ink![18], a smart contract language on Substrate as a first step. We call it **Plasm Contract**. Plasm libraries also support a data base of a child chain. A data base of a child chain and the endpoint are implemented. We call the set of implementations **Plasm Childchain**. The client applications are also implemented to point out the end point. We call it **Plasm Client**.

3.1 Plasm Contract[19]

Plasm Contract is written by ink!, a smart contract language on Substrate. By using Plasm Contract libraries, developers can make a Plasma parent chain contract that meets a Plasma standard. Plasm Contract provides Trait which describes Plasma standard implementations, a generalized data structure, and a Plasma Cash[20] implementation which is created based on the Plasma Group Specification. Figure3 is the Plasm Contract architecture.

Each component has the following roles.

3.1.1 Commitment

This component has all block headers (Merkle Tree) of a child chain. It verifies the block creation of an operator on the child chain and the status. As a default standard, Merkle Interval Tree (Figure4) is implemented to reference ranges of state objects simultaneously.

3.1.2 Deposit

It manages the assets that were deposited on a Plasma child chain. The deposited assets are transferred on the child chain ,and just the legitimate owner can exit from the child chain to the parent chain. As a standard-setting, a Plasma exit game logic which does not rely on tokens is implemented. Figure5 is a transaction diagram of an exit game. The role of each checkpoint is to prove that all states before the pointed block number are correct.

To Do: Make these sentences understandable.

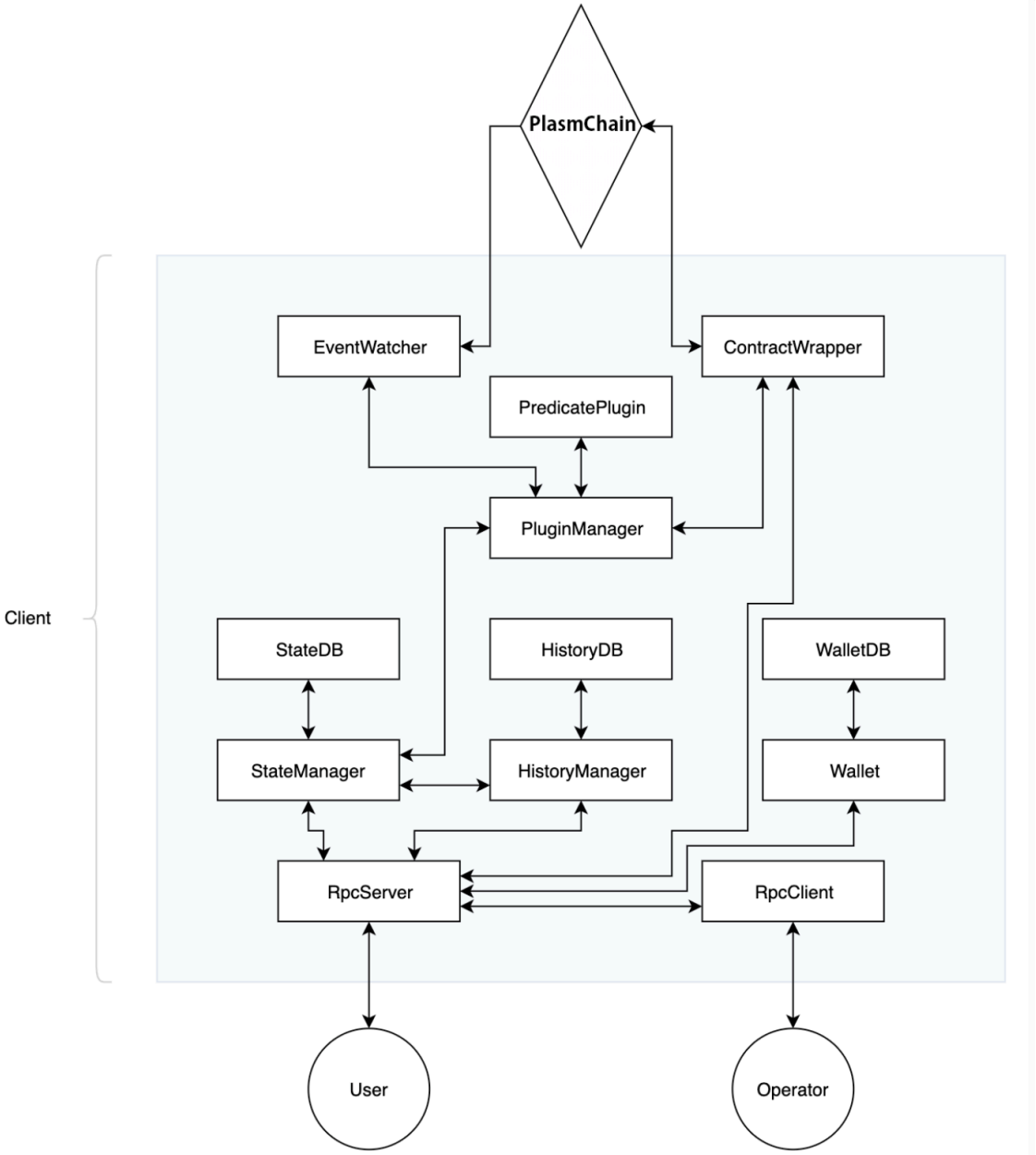

Figure 2: Plasma Group Spec optimized for Substrate[17]

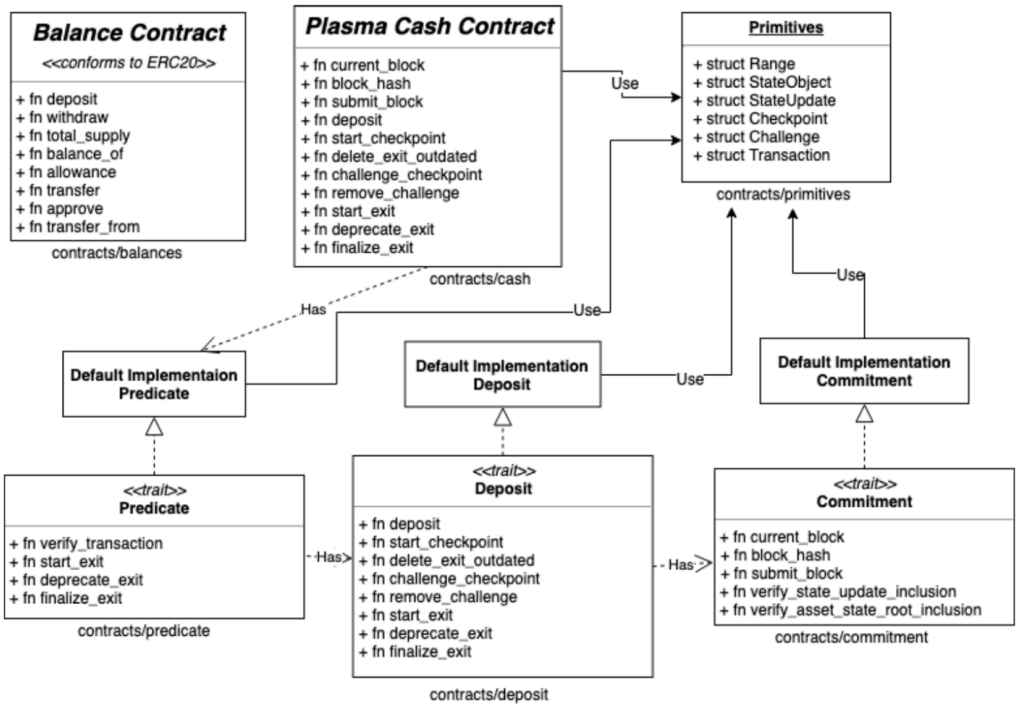

Figure 3: Plasm Contract Archtecture.

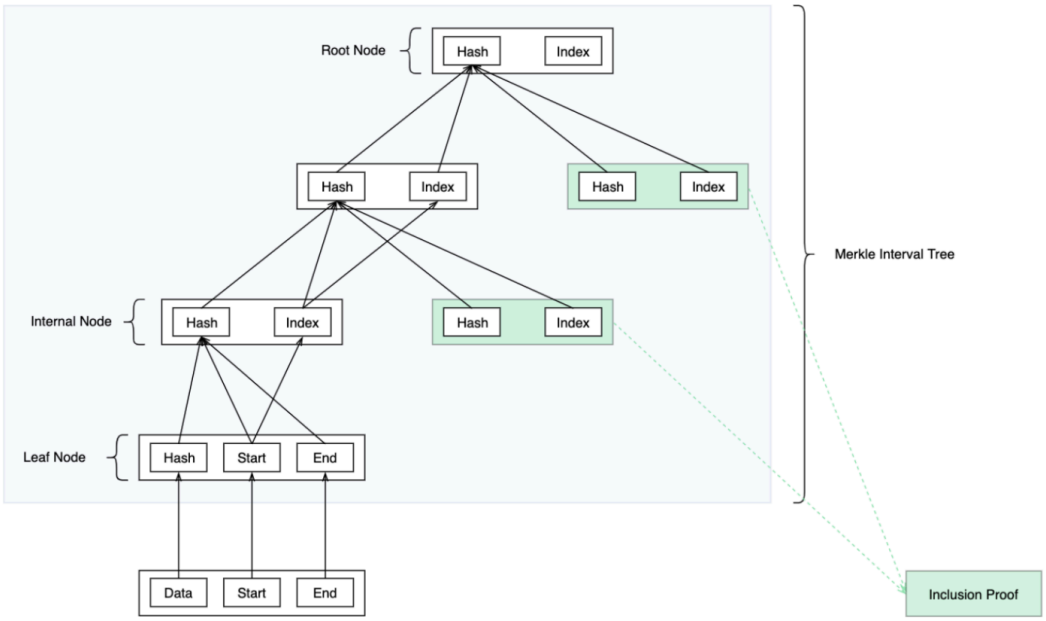

Figure 4: Merkle Interval Tree figure from Plasma Group[21]

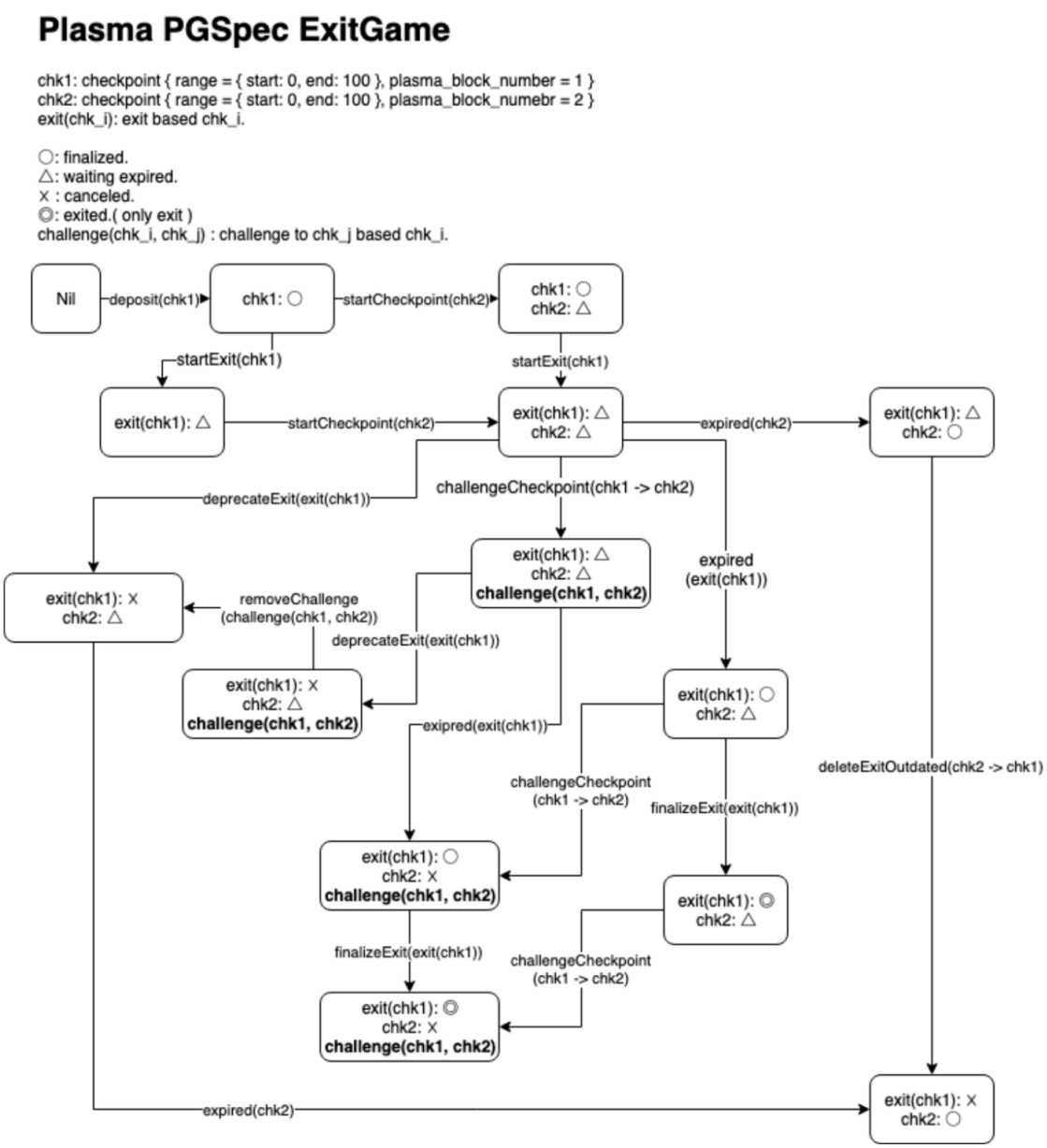

Figure 5: ExitGame state machine figure

3.1.3 Predicate

Predicate is a smart contract which defines an original logic of an exit game. For example, the most fundamental logic is DeprecateExit which makes an outdated exit invalid.

Predicates can be used as fully customized extensions to the base plasma cash exit game. In this sense, Predicate is a generalized Plasma Cash. In other words, Plasma Cash can only handle currencies and Predicate extends the ability to assets.

As a standard design, Ownership Predicate is provided. The contract manages the information of token holders. A digital signature of the owner is required when the state is changed.

3.1.4 Primitive

Common Plasma contract structures and the features are provided as a set of Primitive libraries.

3.1.5 Balance Contract

Every asset on a Plasma contract needs to be confirmed to ERC20 standard. The Plasm team provides customized balance contracts to use ERC20 on Substrate.

3.1.6 Plasma Cash Contract

Plasma contract which has equivalent functions to Plasma Cash is implemented. More specifically, it consists of Ownership Predicate, Deposit, and Commitment on the figure3.

To do: More explanation is needed.

3.2 Plasm Childchain

Plasm child chain provides a data base and a corresponding endpoint for Plasma client (Figure1). Plasm child chain will be implemented with Plasma Rust Framework[22] provided by Cryptoeconomics Lab[23].

3.3 Plasm Client[24]

Plasm client provides client applications and a common set of libraries for an operator and a user.

4 Plasm Chain

Plasm chain is supposed to be the first scaling parachain on Polkadot. In general, Plasm chain will be a default root chain for application developers to connect the

application. In addition to that, Plasm chain has an innovative token mechanism that we will describe below.

4.1 Plasm Chain Token Design

In this chapter, we describe the Plasm token economic design. The Plasm leverages a token model called Lockdrop[25] which is invented by Edgeware[26]. To participate in the lockdrop, individuals need to lock ETH (or DOT) for their selected duration of the lockdrop. After that, the same amount of ETH (or DOT) will be returned. Based on the amount they locked, they can get PLM tokens because they collateralized the opportunity costs. We extend this token model as described below.

4.1.1 Lockdrop Overview

The Plasm has the first lockdrop by collateralizing the opportunity cost of ETH. Hence, this explanation is based on the lockdrop on Ethereum. However, the lockdrop should be possible on any blockchains that have a TimeLock function. Figure6 is the lockdrop overview.

The First Lockdrop Processes

1. Ethereum token holders are able to lock their ETH on a LockContract for their selected duration. (e.g. 3 months) The longer the timelock is, the more Plasm tokens they get.

2. Plasm tokens are distributed on the genesis block to the participants who locked their ETH. A participant gets `Locked ETH` $\times$ `Locked duration` $\times$ α Plasm tokens.

3. The genesis block of Plasm chain distributes 10% (`The total amount of distributed tokens` $\times$ `1/9`) Plasm tokens to the Plasm team.

4. After the selected locked time, the locker receives exactly the same number of ETH.

Ethereum token holder has opportunity costs depending on the locked amount and the locked duration. Plasm tokens are issued relying on their values. We don't decide the total supply of Plasm at the first point because we intend to do two lockdrops (in total three lockdrops) after the genesis and distribute tokens more equally. More specifically, the token distribution model is written below.

4.1.2 Multi-Lockdrops

The total supply of Plasm tokens is not decided at the genesis. Technically speaking, it is determined at the 3rd lockdrop. There are two reasons why we do lockdrops three times. First, if the Plasm distributes all the remaining tokens at once and there are not enough participants, it may be possible for malicious attacker to get a large portion of the Plasm network and eventually

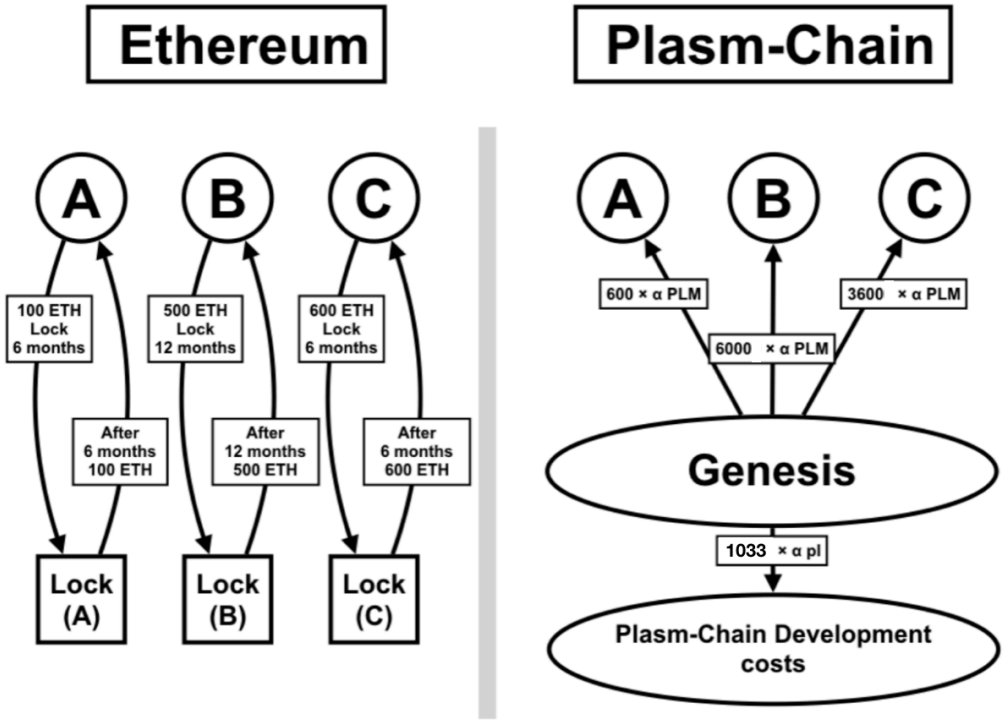

Figure 6: About Lockdrop

attacks it. Second, we intend to expand the Plasm network step by step by maintaining fairness. At first, the Plasm is manageable chain to a certain extent to become a scalable and secured public blockchain in the end. The strong security of a public blockchain is realized thanks to a large number of validators and widespread token holders. It is challenging to get such a vast ecosystem from scratch. Therefore, the Plasm is organized by authorized parties in the beginning. Then, becomes more decentralized and permissionless through other two lockdrops.

In every lockdrop, the system decides a fixed number α to calculate Plasm tokens that participants can receive. There are some conditions:

- The round of a lockdrop: `i = {1, 2, 3}`
- α `(i)` : α in the round `i`
- α `(i)` $>$ α `(i+1)`
- `n(i)` : The number of participants in the round `i`
- `j` : j-th participant
- `LockedETH (i, j)` : `j` 's locked ETH in the round `i`
- `LockedTime (i, j)` : `j` 's locked duration in the round `i`

The total amount of issued Plasm tokens in the round `i`

$$\sum_{j=1}^{n_i}(LockedEth_{i,j} \times LockedTime_{i,j}) \times \alpha_i$$

The Plasm has 3 lockdrops. The total supply is calculated as follows: (Mining rewards are excluded)

$$\sum_{i=1}^{3}\sum_{j=1}^{n_i}(LockedEth_{i,j} \times LockedTime_{i,j}) \times \alpha_i \times \frac{10}{9}$$

Given that we fixed the total supply at the genesis, we would need to decide the percentage of each lockdrop. (e.g. 1st round: 30%, 2nd round: 30%, 3rd round: 30%, Plasm team: 10%)

Assuming the number of participants in the round `i` is more than the one in the round `i - 1`, the first mover advantage could be more exorbitant than it actually is. This is not the thing which all contributors want. Therefore, the total supply is determined after the 3rd lockdrop.

4.1.3 Advantages

- No ICO. We DON'T keep your assets.
- The participants get Plasm tokens by collateralizing the opportunity cost of ETH (or DOT). Though they can't withdraw their locked tokens during a lockdrop, the participants will have access to two utility tokens, ETH and PLM at the end of the lockdrop. This is not a problem for long term ETH (or DOT) holders.
- After the locking duration, ETH (or DOT) will have been returned to the original holder.
  Even when Plasm chain got a malicious attack or exploited, the participants would still retain their ETH (or DOT).
- Every ETH (or DOT) holder can participate in our lockdrop through any hardware accounts (e.g. Ledger, Trezor) or software accounts (e.g. Metamask). All participant have to do is just send ETH (or DOT) to a lockdrop contract and receive PLM.

4.2 Consensus Algorithm

Plasm chain uses NPoS which is used on the Polkadot relay chain. This consensus algorithm consists of 3 steps.

1. A Nominator selects a validator (NPoS) [27]
2. A validator verifies transactions and make a new block (BABE) [28]
3. Finalize the block that was delivered in the network (GRANDPA) [29]

The block reward is distributed to the validator who created the block and his Nominator. In addition to that, the reward is also paid to the Plasm chain contributors as follows.

4.3 Plasm Treasury

50% of the block reward on the Plasm chain is distributed to DApps developers who increase the value of the network. PLM holders can stake not only on a validator but also on a Plasma application operator (Plapp operator), the person who manages a child chain and an application on that chain.

Plapps nominator stakes Plasm tokens on Plasma application operator. Figure7 is the overview of rewards for operators and nominators.

Definition

- BlockReward: The block reward per a block
- `n`: The number of Plapp operators
- `m_i`: The number of Plapps nominators for i-th Plapp operator

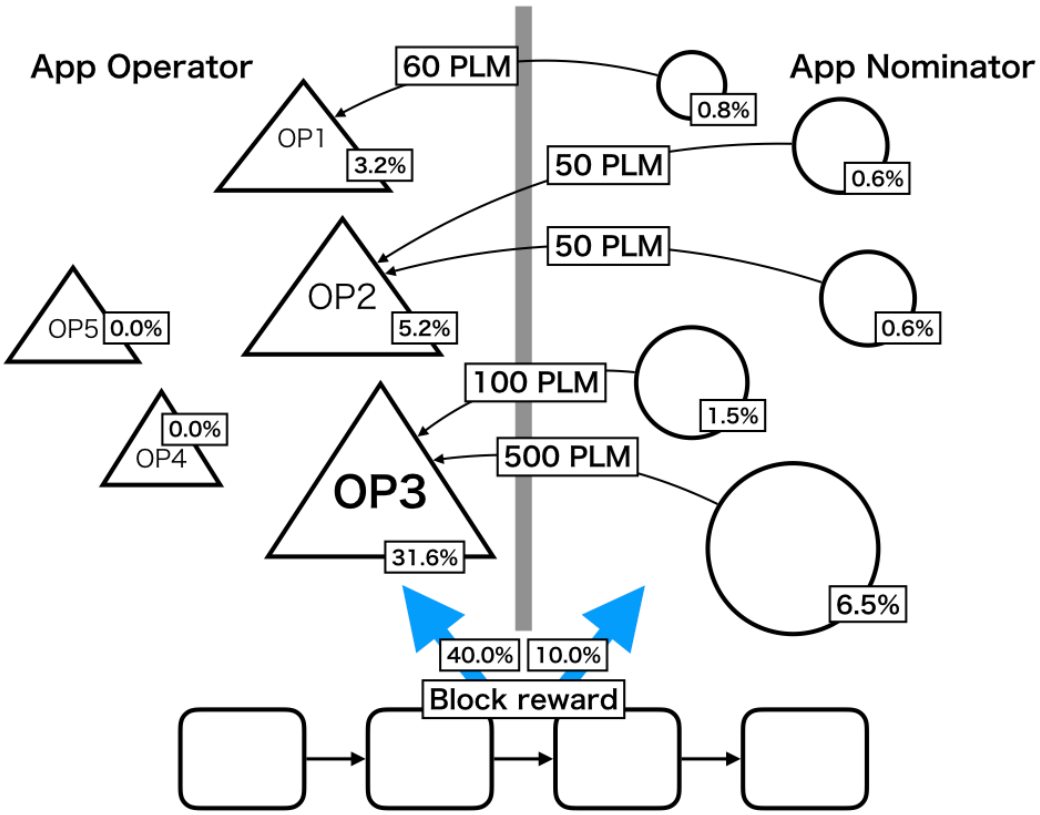

Figure 7: plasm application reward

- `stake_{i,j}`: The total staking amount of j-th Plapps Nominator for the i-th Plasma operator
- `Plapps Nominator_{i,j}`: j-th Plapps nominator who stakes on i-th Plapp operator

In this case, `Plapps Nominator_{i,j}` gets the following block reward.
Note: `Plapps Nominator_{i,j} = Plapps Nominator_{k,l} (i ≠ k)` is possible.

$$BlockReward \times \frac{stake_{i,j}}{\sum_i^n \sum_j^{m_i} stake_{i,j}} \times 0.1$$

A staked Plapp operator can get the following block reward

$$BlockReward \times \frac{\sum_j^{m_i} stake_{i,j}}{\sum_i^n \sum_j^{m_i} stake_{i,j}} \times 0.4$$

The reward of Plapps nominator is in proportion to the staking amount of the Plapps operator regardless of the selected operator. On the other hand, a Plapp operator can get rewards in proportion to the staked amount and in inverse proportion to the total staked amount. Plapps nominator tends to stake his tokens on an application which enhances the value of the Plasm network because the price of PLM is increased by the valuable application. In addition to that, Plapps operators are able to get semipermanent block rewards through this incentive design. In general, traditional DApps creators are definitely contributing to the ecosystem, but what they are actually doing is paying costs like gas and get few rewards. We solve this problem by fixing basic low layer functions.

An operator can make some malicious attacks, but fortunately, Plasma has a method to prevent such operator's behaviors. When an operator commits a fraud, any user on the plasma chain can submit the fraud proof as evidence and the operator's block reward will be slashed as well as the Plapps nominator's staked tokens once the evidence is successfully approved. In other words, every Plapps nominator has a risk to lose it's token when his operator commits a fraud. A stakeholder who owns a large portion of the network stakes tokens on his Plapps operator, the operator will be slashed.

Assuming that there is the slashing mechanism, a nominator tends not to stake on an unstable operator. If all nominators stake their tokens on stable operators, the network will be centralized. As a solution, the Plasm has an alternative incentive mechanism. Every nominator who stakes PLM on an operator who operates his application within three months prior to the birth has a right to get an additional bonus. This right can be executed for three months if the operator runs his application for more than a year.

`x^{old}` is the rate of `x` when a nominator stakes PLM. The nominator will get the reward calculated below.

$$BlockReward \times \frac{\sum_j^{m_i} stake_{i,j}}{\sum_i^n \sum_j^{m_i} stake_{i,j}} \times \frac{stake_{i,j}^{old}}{\sum_i^{n^{old}} \sum_j^{m_i^{old}} stake_{i,j}^{old}} \times 0.4$$

This means that the nominator can get bonus in proportion to the operator's rewards.

4.4 Operator Trading

Operator trading is a mechanism to buy and sell the right to be a Plasma operator. Since the right is tradable, this is similar to M&A. A Plasma operator gets constant basic income by creating Plapps owning to Plasm treasury. If the Plapp is meaningful for the community, the creator receives higher revenue. As a result, other players may want the right to operate the Plapp. The successor will receive the reward instead of a previous operator. The important thing is that the buyer does not need to take over the operation itself though the buyer has ownership. Through this mechanism, we assume that the new off-chain market will be created.

5 PlaaS

Plasm as a Service(PlaaS) is a tool to deploy and manage Plasma child chain easily under the Plasm parachain on Polkadot. Figure8 is the main screen of Plasm as a Service.

"Plasm" is a set of Substrate Runtime Module Libraries (SRML) which makes it easier to deal with Plasma functions on Substrate. By using Plasm, the Plasm team makes a Plasma oriented parachain where any developers can make a child chain and deploy a Plasma application. Plasma application takes a different approach from a traditional approach, making a smart contract on the 1st layer blockchain. The latter approach has a limited capability so that many DApps are not practical. What Plasma application is doing is that making a set of Plasma components mainly on "off-chain" and connect them to the main chain. Plasma application is the next generation technology which gives developers new possibilities.

However, Deploying and managing a Plasma application is not easy.

As you can see from the figure2, Plasma consists of several components. It is difficult and takes time to implement these components from scratch. Plasma specific knowledge and high technical skills are also needed. To solve these problems, the Plasm team provides a GUI tool which makes it easier to deploy and manage Plapps.

5.1 Functions

PlaaS makes it easy to deploy and manage Plasma applications. More specifically, a user creates an application by the following steps.

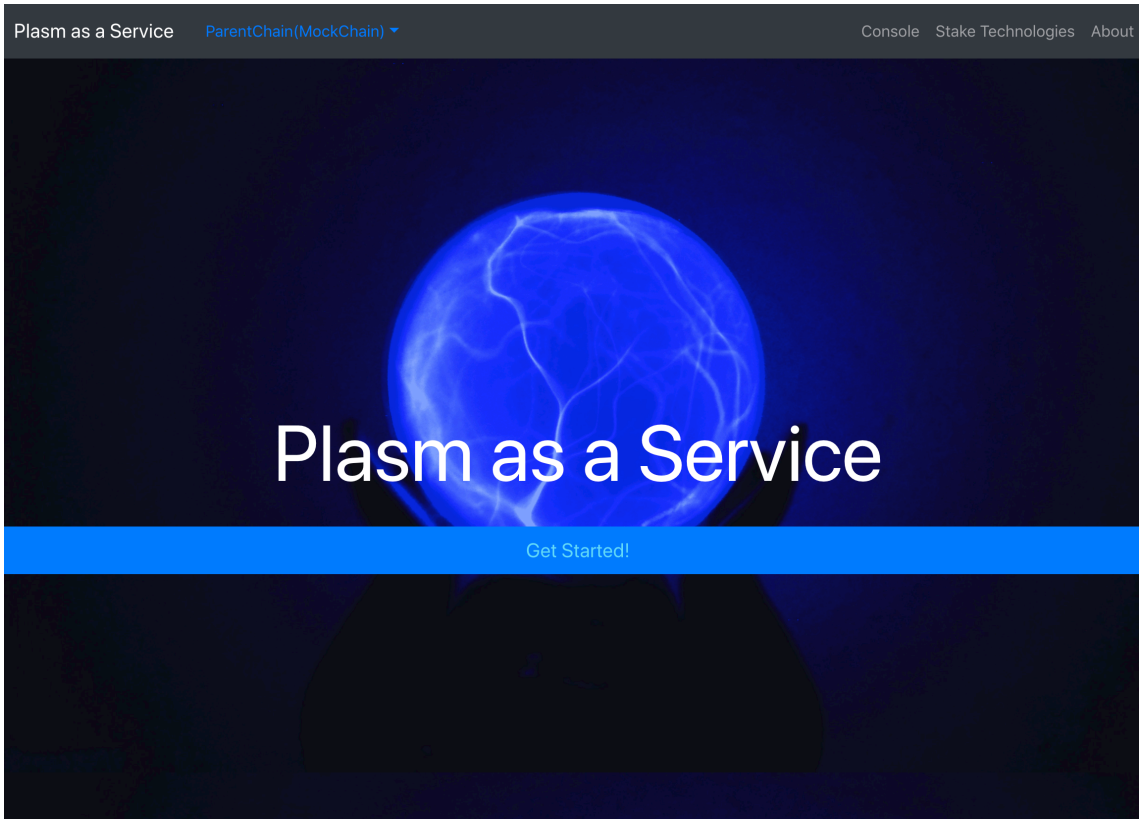

Figure 8: Plasma as a Service.

1. **GetStart**: Move to the main page to make a Plasma application.
2. **Generate Plasma App**: Choose a template to make a Plasma application. As a default setting, Plasma Cash, one of the Plasma standards for payments is prepared. Other templates like Plasma Prime and Plasma ZK-SNARKs fw
3. **Settings**: Fill out application information which is required to run a template.
4. **Deploy a Contract and a Child Chain:** Deploy an application and a child chain that were created at the step 3. And,
    1. Deploy a smart contract on the parent chain.
    2. Deploy a server which has the DB of the child chain.
    3. Deploy the operator's account.
    4. Generate the genesis account.
5. **Console**: Manage the status of the Plapps. Specifically,
    1. The status on the parent chain
    2. The status of DB on a child chain
    3. Operator's account
    4. Registered accounts including the genesis account

    are controllable.

Through the functions listed above, developers can deploy and manage Plasma applications easily.

5.2 Demo

You can try a simple PlaaS application.
(※ This demo doesn't deploy an application)https://mock.d3dg769h9ndpcf.amplifyapp.com

6 Conclusion

Through the Plasm Project, we provide all developers with new methods to make decentralized and scalable applications, especially on Polkadot. Since Polkadot relay chain does not support smart contracts, scalable smart contract platforms are necessary to build applications on the top of it. Therefore, Plasma makes Polkadot network more valuable.

In addition to that, since it is complicated to make Plasma applications, we provide Plasma as a Service to make it much easier. All developers will be able

to make customized Plasma applications by using Plasm libraries depending on their needs. In terms of Plasm chain, the participant can buy and sell a particular application and its ecosystem like M&A. The individuals are creating not only a Plasma application but also an economy itself. Hence, Plasma application is more than an application. We are very excited to see the future of decentralized Plasma ecosystem.

7 Acknowledgment

Many thanks go out to Web3 Foundation, the creator of Polkadot and Parity Technologies, the creator of Substrate for making innovative protocols. Thanks to Cryptoeconomics Lab for reviewing and cooperating with us.

References

[1] Robert B. Reich, "Saving Capitalism: For the Many, Not the Few" https://www.jstage.jst.go.jp/article/lecgsa/15/0/15_139/_pdf/-char/en

[2] Mastering BitCoin : https://github.com/bitcoinbook/bitcoinbook

[3] Ethereum : https://www.ethereum.org/

[4] Ethereum Transaction Throughput : https://www.coindesk.com/information/will-ethereum-scale

[5] Visa Transaction Throughput: https://hackernoon.com/the-blockchain-scalability-problem-the-race-for-visa-like-transaction-speed-5cce48f9d44

[6] ALIPAY Transaction Throughput : https://www.barrons.com/articles/alibaba-records-25-3-billion-in-singles-day-sales-1510538618

[7] Segwit : https://github.com/bitcoin/bips/blob/master/bip-0141.mediawiki

[8] StateChannel : Jeff Coleman, Liam Horne, and Li Xuanji "Counterfactual: Generalized State Channels", June 12, 2018, https://l4.ventures/papers/statechannels.pdf

[9] Sharding : Loi Luu, Viswesh Narayanan, Chaodong Zheng, Kunal Baweja, Seth Gilbert, Prateek Saxena, "A Secure Sharding Protocol For Open Blockchains", https://www.bubifans.com/ueditor/php/upload/file/20181015/1539597837236127.pdf

[10] Plasma : Joseph Poon, Vitalik Buterin, "Plasma: Scalable Autonomous Smart Contracts" August 11, 2017https://plasma.io/plasma.pdf

[11] Ethereum "almost full" as controversial coin gobbles up capacity : https://www.bloomberg.com/news/articles/2019-08-26/ethereum-almost-full-as-controversial-coin-gobbles-up-capacity

[12] About Predicate : https://docs.plasma.group/projects/spec/en/latest/src/02-contracts/predicate-contract.html

[13] Plasma Components : https://docs.plasma.group/projects/spec/en/latest/src/05-client-architecture/introduction.html

[14] Polkadot : DR. GAVIN WOOD, "POLKADOT: VISION FOR A HETEROGENEOUS MULTI-CHAIN FRAMEWORK" https://polkadot.network/PolkaDotPaper.pdf

[15] Substrate : https://www.parity.io/substrate/

[16] Subzero Summit : https://youtu.be/05H4YsyPA-U?t=198

[17] PGSpec https://docs.plasma.group/projects/spec/en/latest/src/05-client-architecture/introduction.html.

[18] ink! : https://github.com/paritytech/ink/wiki

[19] Plasm Contract : https://github.com/stakedtechnologies/Plasm/tree/master/contracts

[20] Plasma Cash https://ethresear.ch/t/plasma-cash-plasma-with-much-less-per-user-data-checking/1298

[21] MerkleIntervalTree https://docs.plasma.group/projects/spec/en/latest/src/01-core/merkle-interval-tree.html

[22] Plasma Rust Framework https://github.com/cryptoeconomicslab/plasma-rust-framework

[23] Cryptoeconomics Lab https://www.cryptoeconomicslab.com/

[24] Plasm Clirnt : https://github.com/stakedtechnologies/plasm-client

[25] Lockdrop https://blog.edgewa.re/full-details-on-the-edgeware-lockdrop

[26] Edgeware : https://edgewa.re/

[27] NPoS https://research.web3.foundation/en/latest/polkadot/NPoS/

[28] BABE https://research.web3.foundation/en/latest/polkadot/BABE/Babe/

[29] GRANDPA https://research.web3.foundation/en/latest/polkadot/GRANDPA/