The pNetwork v3 white paper

pNetwork Community Association

January 24th, 2023

Release: DRAFT v0.3.4

Abstract

The pNetwork is a decentralized network of validators contributing to the verification of crypto asset swaps across blockchains.

As the cryptocurrency industry continues to evolve, the development of alternative financial platforms is on the rise. A critical component is assets' liquidity, which, in the decentralized scene, is currently spread across multiple independent blockchain protocols.

The pNetwork gives users the possibility to move and make use of their liquidity in different blockchains. In particular, after a token has been locked on its native blockchain, a set of bridges provides the issuance of fully collateralized tokens, called a pToken, on a selected host blockchain. These bridges are currently operated by a network of validators that verify the cross-chain asset swaps and aim to guarantee the 1:1 peg with the underlying asset.

From its inception on March 5th, 2020, pNetwork has evolved from pNetwork v1, integrating numerous tokens on different blockchains, to pNetwork v2, permitting users to transfer their liquidity along with user data and directly between host blockchains.

The pNetwork is currently based on a hybrid decentralized approach in which a limited group of permissioned nodes operate the bridges, and third parties may join the network albeit with limited functionalities. The purpose of this paper is to delineate a new architectural change to pNetwork that will lead to pNetwork v3 and the complete decentralization of its operations.

Contents

1 Introduction

The pNetwork aims to be a decentralized system facilitating the cross-chain movement of assets between blockchains. At the basis of pNetwork there are pTokens: they identify a token that aims to be provable, portable, and pegged. A pToken represents a one-to-one relationship with a specific native cryptocurrency and is issued and redeemed using pNetwork technology.

The first pTokens bridge (pBTC on ETH) had a successful launch on March 5th, 2020 [5].

Since its first inception, pNetwork has deployed various bridges supporting multiple tokens in different blockchains, e.g. EOS, Algorand, Ethereum Layer 2 protocols like Polygon and Arbitrum, and many others. The live decentralized application (dApp) gives a clear idea of how pNetwork permits users to cross-chain their liquidity.

The pNetwork project has also its governance token called PNT, and holders of this token can participate in the pNetwork decentralized autonomous organization (DAO) by submitting their votes to proposals inherent to the project development and organization. PNT holders can also stake their tokens to run network validator nodes and earn a share of the fees accrued from the cross-chain activity.

The protocol has at the time of writing reached a cross-chain cumulative volume of over 1 billion dollars, and the Total Value Locked (TVL) peaked at 280 million dollars[2].

Besides further integrations, pNetwork has evolved in its architecture as well. The recent release of pNetwork v2 introduced improvements aiming to increase system scalability; it is now possible to move pTokens directly from one host blockchain to another, without the need to convert to the original native asset (host-to-host swaps). Moreover, users may leverage the protocol to exchange custom data between blockchains.

Further, pNetwork is built upon a hybrid decentralized approach: currently, the bridges are operated by a limited set of permissioned nodes; users can join the network by running a full or light node by staking 200k PNT or 10 Yolo Parrot NFTs (each of which accounts for 400 PNT), thus earning from the fees collected by the protocol, however their role is limited in the current architecture.

This white paper introduces pNetwork v3, a protocol upgrade that aims to diffuse pNetwork control to all those users that want to join and participate in the network, thus achieving full decentralization.

2 A brief overview of pNetwork v2

This section aims to describe how the current pNetwork v2 works.

Definitions:

- Native Blockchain: it is the original blockchain or network on which a cryptocurrency or token is anchored.
- Host Blockchain: it is the destination blockchain or network hosting a non-native cryptocurrency or token, i.e. the pToken, the one-to-one representation of the native token.
- Deposit Address: an address typical of UTXO blockchains (such as Bitcoin), used to lock funds in a native blockchain (recall these blockchains do not support smart contracts). Eventually, locked funds may be released with a specific transaction.
- Vault: a smart contract living in a native blockchain, and used to lock funds in non-UTXO blockchains. Eventually, locked funds may be released with a specific transaction.
- Trusted Execution Environment (TEE): it is a computational environment that is isolated from the main operating system running on a given device. Such isolation is achieved via both software- and hardware-enforced mechanisms.
- pTokens Binary: a program that analyzes blocks from a blockchain watching for pTokens issuance or redemption requests, and acting upon them by signing the required transactions.
- Enclave: euphemism hereafter used to describe a pTokens binary running inside a trusted execution environment.
- Syncer: a module aimed to collect blocks from a blockchain and submit them to the enclave.
- Broadcaster: a module aimed to broadcast transactions signed by the enclave.

Since a pNetwork v2 bridge is a superset of a v1 bridge, the following will explain a v1 bridge: suppose there is a token T defined by a smart contract deployed in a blockchain N (the native blockchain). A pNetwork bridge would allow a user to lock tokens T in the native blockchain N, and use them in form of a pToken pT from a smart contract in a host blockchain H. This will be called a peg-in operation.

The same user could decide to redeem his pTokens pT and release what he pegged-in in the native blockchain. This operation will be referred to as a peg-out operation.

The bridge operations are made possible by a pNetwork full node. It comprises the following components:

- A TEE in which the pToken binary runs
- A native syncer, i.e. a syncer collecting blocks from the native blockchain N; blocks will be submitted to the pTokens binary;
- A host syncer, i.e. a syncer collecting blocks from the host blockchain H; blocks will be submitted to the pTokens binary;
- A native broadcaster, i.e. a broadcaster that pushes signed transactions on the native blockchain;
- A host broadcaster, i.e. a broadcaster that pushes signed transactions on the host blockchain.

In particular, upon block submission, the pTokens binary:

- Checks for deposits in the vault (or in deposit addresses) and, if legitimate, signs issuance transactions of pT in the host blockchain H;
- Checks for peg-out transactions in the host blockchain and, if legitimate, signs transactions to unlock native tokens T from the vault (or deposit addresses) in the native blockchain N

Generated transactions will be pushed on-chain by the respective broadcasters.

The number of pNetwork bridges has increased over time as the community required more and more tokens to be bridged across different blockchains. However, this v1 architecture does not scale well: a separate bridge is required for different tokens, even if they are from the same native network. Additionally, the same pToken cannot be swapped between different host blockchains without first redeeming to the native blockchain.

This scalability issue led to an architectural change that goes under the name of pNetwork v2.

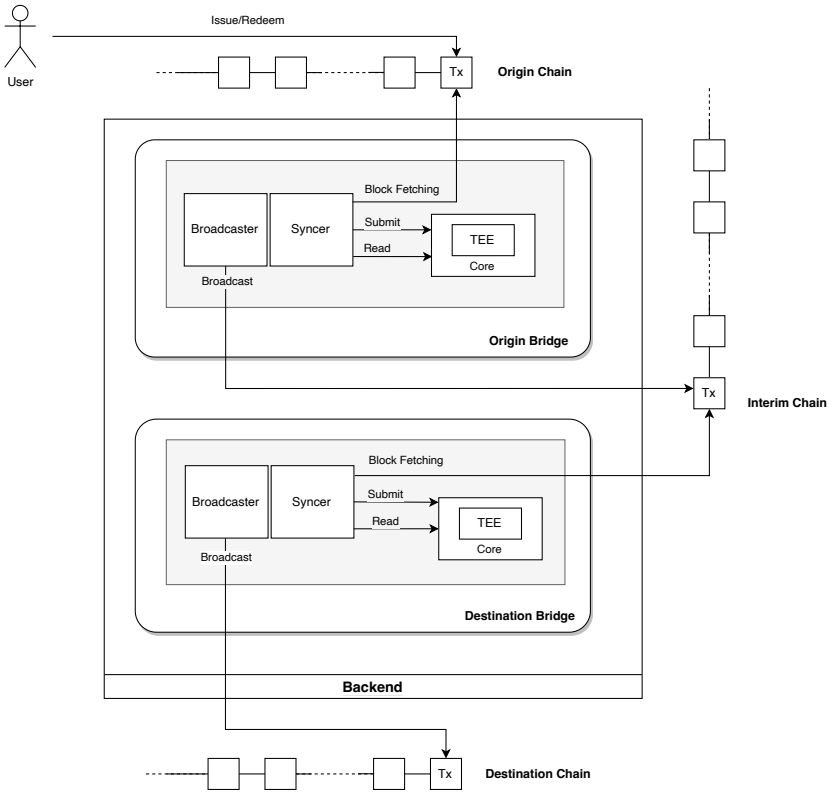

Figure 1: pNetwork v2 architecture.

Figure 1 shows the basic architecture of the v2 technology. Notice the two v1 bridges with the outlined components.

Fundamental to pNetwork v2 is the interim chain I, a blockchain that lies *in the middle* of the native and the host blockchains. Bridging token T from native blockchain N to pT in the host blockchain H involves two v1 bridges: one from the native blockchain N to the interim blockchain I, and one from the interim blockchain I to the host blockchain H. The interim chain will work with a collateralized interim token for T, and a set of interim contracts and vaults for locking those interim tokens that will be bridged to host blockchains, H at least. A router contract moves the issued interim tokens to the correct interim vault based on the host blockchain the user specified when he locked T in the native vault.

The interaction with vaults and pTokens smart contracts for unlocking and issuance operations is restricted to a limited set of permissioned nodes. This motivates the need for a new architectural upgrade to fully decentralize the protocol, thus permitting external third parties to actively participate in creating pToken issuance and redemption transactions within the network.

3 How to fully decentralize pNetwork?

The recent appearance of Layer 2 (L2) blockchains to facilitate scalability issues has introduced new approaches to achieve decentralization. A particular approach is Optimistic Rollups[4]. In the Ethereum case, L2 blockchains move computation and storage off-chain, i.e. transactions are executed outside of Ethereum, then these transactions are bundled together in large batches which are then submitted back to Ethereum. As a direct consequence, fixed costs are spread across multiple transactions, allowing a reduction of fees for L2 users.

Optimistic rollups are considered "optimistic" because they assume off-chain transactions are valid and actors in the system do not publish proofs of validity for transaction batches posted on-chain. Instead, the security of optimistic rollups relies on a fraud-proving scheme to detect cases where transactions are not calculated correctly: a batch submission can be challenged within a time frame called the challenge period by computing and submitting a fraud proof. This forms evidence of the discrepancy, and is generally delivered to the layer one blockchain with an attestation as to the proof-sender's identity. If the fraud proof passes validity checks, the submitted batch needs to be reverted. Otherwise, if the submitted batch remains unchallenged (no fraud proof proving its incorrectness), it is considered as valid, subsequently allowing execution of batch of transaction to complete unhindered.

Optimistic rollups are implemented in already-existing L2 blockchains such as Arbitrum One and Optimism.

From a software perspective, the entire rollup implementation requires a lot of code, both off-chain and on-chain, thus introducing the possibility to have bugs in the system. In [1], Vitalik Buterin highlighted that risk and proposed a solution based on two provers plus a governance tie-break. Recalling the fraud proof schema mentioned above, a batch submission to the layer one network could be considered invalid whenever two different provers demonstrate incorrectness. Ideally, the two provers should have a very different construction to minimize the chance of simultaneous bugs. The governance should only be requested in extreme cases.

The further decentralization of the pNetwork protocol will be achieved by the abovementioned approaches of Optimistic rollups and multi-provers.

Based on the optimistic approach, pNetwork would process transactions with an instant issuance request of the pTokenised assets on the host blockchain or with a tentative pre-unlocking of native assets on the native blockchain in the case of a peg-out. This will begin a challenge period during which elements may be triggered within the pNetwork protocol to propose the dismissal of the transaction.

Anyone (the so-called Relayer 4.1.1) can send on the destination chain a transaction to propose a peg-in/peg-out execution to occur after the challenge period has elapsed. By contrast, in pNetwork v2 transactions were always and only processed by bridges run by a limited set of permissioned nodes.

The pNetwork v3 system is made secure by the interoperation of different

actors whose role is to dismiss the cross-chain transactions in case they are fraudulent:

- Sentinels, a set of registered entities emitting fraud proof backed by a TEE attestation (4.1.2);
- Guardians (4.1.3), a set of DAO elected nodes that may submit dismissal requests independently;
- DAO (4.1.4), whose involvement is required just in case conflicts arise from particular situations.

The optimistic approach is challenged by the multi-provers approach. Should one of these three elements (Sentinels, Guardians and DAO intervention) be triggered, a Relayer's request may be dismissed. In this case, the challenge period is extended. This extra time allows for additional checks to be made and for a potential second dismissal proposal to be triggered by either one of the other parties involved. Ideally, if a Sentinel fraud proof is issued, Guardians would execute the same check automatically, dismissing the manipulation attempt quickly. A DAO vote should only be necessary in emergency or cumbersome situations, as it incurs higher costs and longer resolution times. When at least 2-out-of-3 of these elements are triggered, the transaction is fully dismissed.

Sentinels will be required to have some assets at stake to prevent them to misbehave, i.e. they submit invalid fraud proofs to dismiss legitimate transaction, or they are inactive despite their revenue participation.

There are various scenarios that may arise:

A Under normal circumstances, transactions are processed assuming everything is correct (the optimistic approach). None of the security actors detect requests as malicious, and, after the challenge period has expired, the tokens are issued to the end user. For example, if a cross-chain transaction is detected requesting to move 100 PNT-on-Ethereum from Ethereum to Polygon, it would be instantly processed with an optimistic approach, meaning that the issuance of 100 PNT-on-Polygon is instantly requested (after the required confirmation times to avoid double-spend or other issues arising from blockchain reorganizations) on Polygon.

B A Sentinel detecting a malicious or incorrect transaction issues a fraud proof for the Relayer's request, and this would represent a first trigger for dismissal. As a consequence, the challenge period is extended to allow for a potential second trigger. Any Guardian detecting that activity as malicious could submit a dismissal request, thus activating the second trigger. In this case, 2-out-of-3 types of elements are triggered, hence the network agrees on dismissing the transaction at this point. Note that the DAO vote is not needed in this case. Due to its proposal of an incorrect transaction, the Relayer may get slashed, thus losing some portion of what they staked. Should this drop their stake to below the threshold

required to take part in the pNetwork, they would thusly lose their ability to participate entirely pending further staking, and possible intervention from the DAO.

C A Sentinel issues a fraud proof for the Relayer's request. The challenge period is extended. The Guardians detect that the activity is legitimate, and the fraud proof was a false-positive, and, as a consequence, they do nothing. Without activity from the Guardians, the DAO might decide to open a vote for dismissing the proposed transaction. If the vote is not initiated, or DAO users vote against the dismissal if a vote was opened, only one out of three of the elements is triggered. Once the challenge period is over, the transaction is tacitly approved and everything continues as usual. The Sentinel proposing the request dismissal may get slashed whether it is established it operated in a malicious way censoring the Relayer's request.

D A Sentinel issues a fraud proof for the Relayer's request. The challenge period is extended, Guardians detect that the activity is legitimate and do nothing. The DAO opens a vote for dismissing the Relayer's request and a consensus is reached to dismiss it. With two out of three triggered elements, the original request will be canceled. The Relayer may get slashed, thus losing some portion of what they staked and the ability to operate in the network.

E There is no Sentinel issuing a fraud proof for the Relayer's request. Instead, a Guardian detects that same requests as malicious, thus it submits a dismissal request. The challenge period gets extended. With just one trigger, the DAO should intervene. This situation needs to be promptly addressed: there might be a bug in the Sentinel codebase from which a Relayer has taken advantage. In this case, the DAO should open a vote to put the system in a locked state. Otherwise, the aforementioned Guardian may have submitted a wrong or invalid dismissal request, and the DAO may open a vote to decide if its role has to be revoked.

The last two cases would occur in rare situations where the Sentinel codebase might have a bug, or the Sentinel has been compromised in a way it can submit incorrect fraud proofs.

Notice that the entire system can be kept secure with just one honest Sentinel, at least for preventing illicit requests by Relayers: one fraud proof is sufficient to dismiss the malicious attempt as, in the worst scenario, the DAO would intervene. Otherwise, with an additional honest Guardian, the DAO would not be needed to solve the conflict in any case.

Recall that a malicious Sentinel or Guardian may just create illegitimate fraud proofs, thus censoring valid requests from Sentinels. Also, if reiterated repeatedly, this may represent a Denial of Service attack because of the higher costs and time required by subsequent DAO vote. The DAO should be able to promptly detect these situations so that the proposing actor could get penalized.

4 Modelling the new pNetwork v3

4.1 Actors

The pNetwork v3 would introduce new actors permitting to achieve full decentralization. These are Relayers, Sentinels, Guardians, and the DAO. Their functionality and structure will be outlined in the following sections.

4.1.1 Relayers

The Relayers are those players in the pNetwork protocol who propose the execution of peg-in/peg-out transactions.

Ideally, the Relayer will be a resource-efficient process listening to a blockchain for swap requests, and creating the issuance/unlocking transactions in the destination chain.

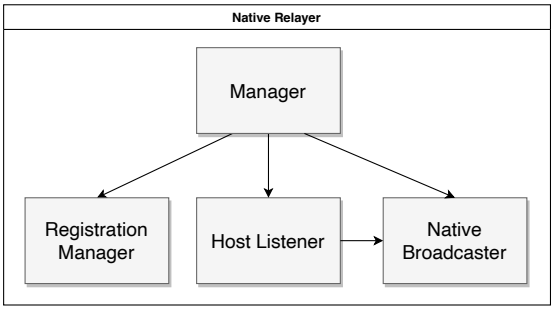

Figure 2: Native Relayer block diagram.

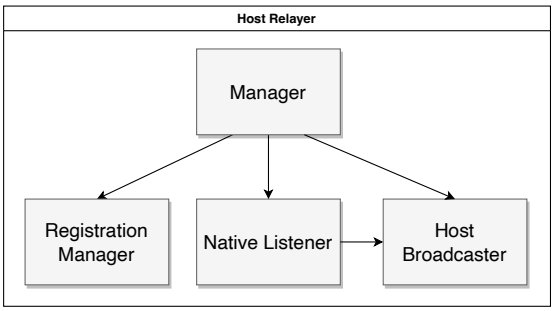

Figure 3: Host Relayer block diagram.

Figures 2 and 3 show the block diagram representing the working structure. In particular, the native Relayer proposes fund-unlocking transactions in the native blockchain (hence the name), conversely, the host Relayer requests token issuance in the host blockchain.

The common manager component's role is to orchestrate the other services running within the Relayer: it will manage updates, spin-up, and, in general, ensure all the components are running as expected.

The listener will be a simple service that will listen to events in a blockchain:

- the native listener will listen for peg-in events in the native blockchain, i.e. deposits into a vault;
- the host listener will listen to peg-out transactions in the host blockchain.

Whenever an event is fired, the listener would witness it, and, based on the event data, a new transaction would be created:

- upon a deposit of N tokens, the host Relayer will create a transaction requesting the issuance of N pTokens in the host blockchain;
- in case N tokens are redeemed, the native Relayer will create a transaction requesting the unlocking of N pTokens from the native vault.

Eventually, the created transaction will be broadcasted to the relevant chain by the broadcaster service.

The created transaction should match data from the originating transaction. Otherwise, Sentinel and/or Guardians may detect it as a malicious transaction, thus initiating a dismissal attempt.

As an example, Polygon transaction 0x7aafde5..., representing a peg-out of `PNT` tokens from Polygon to Ethereum, generated a `Redeem` event: a native Relayer would detect it and create an unlocking transaction of the equivalent quantity of `21 PNT` in the redeem manager contract deployed in the native blockchain, and destined to the address `0x36b9bB0Fb89f8E74251E45b6d9BbA2926560028A`.

Initially, there would be a whitelist of trusted Relayers run by trusted actors to avoid malicious attempts while the system security checks get strengthened. In the medium term, the whitelist would be gradually expanded to add other trusted parties. Finally, in the long term, the whitelist would be lifted entirely so that anyone would be able to become a Relayer.

In particular, there will be two kinds of Relayers:

- Gasless Relayers. A set of PNT-staking entities operating the Relayer role which is used when the user chooses the gasless option for cross-chain transactions, i.e. transaction fees on the destination chain are subsidized by the pNetwork protocol, and the user is willing to pay more in fees to sustain gasless Relayers. Those Relayers would be required to stake PNT tokens to perform their roles and would get a small fee back when users use their gasless Relayer's services. In the long run, the gasless Relayer's role could be performed by pNetwork light nodes (Yolo Parrots -powered pNetwork nodes), so that they could run nodes at a fraction of the cost of a Sentinel, whilst being exposed to less severe slashing conditions in the event of misbehaving, e.g. proposing invalid transactions.

- Users acting as Relayers. The users themselves can act as a Relayer for their own transactions when they choose the gas-payment option. Transaction fees on the destination chain are paid directly by the user, resulting in lower fees compared to using gasless Relayers. These users will be required to stake PNT tokens as well to ensure they do not engage in misbehaviour.

To act in the Relayer role, one will have to stake a pre-defined quantity of PNT tokens. Extra fees paid by a user will be earned by those Relayers operating in the right way. The ones misbehaving will get slashed (4.1.5).

4.1.2 Sentinels

These are pNetwork full nodes aimed to invalidate transactions emitted by Relayers. Whenever the node has correctly registered as a full node, it can submit fraud proofs for a Relayer transaction within the challenge period and thus potentially dismiss it.

Sentinels would share much of the code the existing pNetwork v2 uses, with only a few adaptations required.

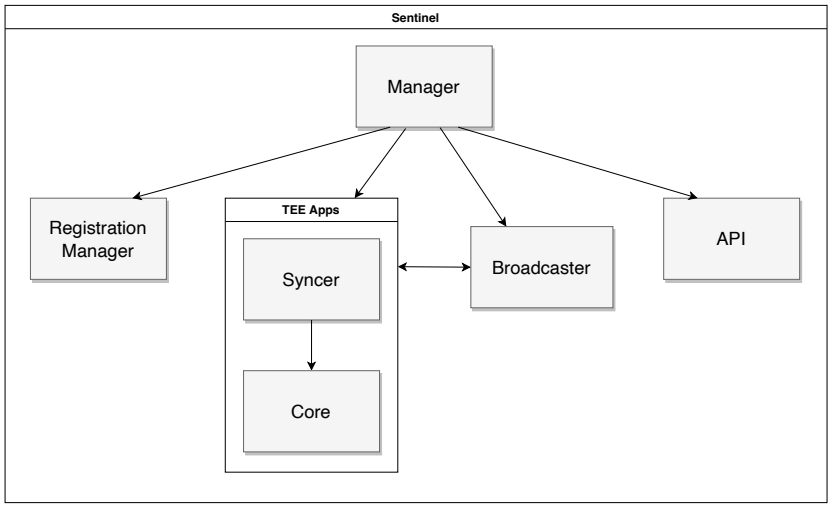

Figure 4: Sentinel block diagram.

Figure 4 shows the block diagram representing the working structure of a Sentinel node.

The manager component's role is to orchestrate the other services running within the Sentinel: it will manage updates, spin-up, and, in general, ensure all the components are running as expected.

The TEE apps block runs a syncer fetching and submitting blocks from the originating and the host blockchains to the pTokens core application: blocks will be processed to determine if a user requested a cross-chain operation and a

Relayer took charge of it; possibly, valid operations may be indexed in a Merkle tree that will permit to efficiently determine if a transaction has already been seen. The Merkle tree may be maintained and updated for an epoch duration, e.g. 30 days. At the epoch termination, Sentinels should discard the existing Merkle tree, and initiate a new one from a pre-determined root.

Whenever a request is detected, the listener would interact with the TEE app to determine if the originating transaction is legitimate, and, in that case, check if the originating request matches the Relayer's one. If a mismatch is found, the Sentinel would build a transaction with a fraud proof backed by a TEE attestation. That would be pushed on-chain by the Broadcaster component.

The API part will expose information about the Sentinel for monitoring purposes.

The Sentinel role will be possible for pNetwork full nodes only, i.e. entities staking 200,000 PNT or more in the pNetwork DAO. It is expected pNetwork DAO v2 to allow users to lend and borrow PNT tokens.

Cross-chain operations fees accrued by the pNetwork protocol will be distributed amongst those full nodes. Misbehaving Sentinels will get slashed (4.1.5).

The Sentinel validation steps are illustrated in Figure 5. After the listener component has detected a new request from a Relayer, the TEE app would check if it is in sync, i.e. the last processed block is comparable with the blockchain height: in that case, the Merkle tree would be inspected to determine if the originating transaction has already been seen by the core; if it is, the originating transaction is checked against the Relayer request for possible mismatches; if a mismatch is found, a fraud proof is generated and finally submitted by the broadcaster. Whenever the core is in sync and the originating transaction is not found in the Merkle tree, it means the originating transaction has not been considered valid by the core, so the Relayer proposal would need to be dismissed. Otherwise, the Sentinel would do nothing.

Recall that the system can be secured from illicit Relayers' proposals if there's at least one honest Sentinel in sync.

4.1.3 Guardians

Guardians are another class of provers that should propose a transaction dismissal in case it is detected as potentially malicious or incorrect. Ideally, Guardians would operate via dedicated software (different and less computationally demanding from the Sentinel one) that automatically checks transactions, and, eventually, should any of them be detected as malicious, requests for dismissal.

Figure 6 shows the block diagram for a Guardian.

The manager component is aimed to orchestrate the other services running within the Guardian: it will manage updates, spin-up, and, in general, ensure all the components are running as expected.

The listener will be a simple service that will check for new issuance/unlock requests in the on-chain manager queue: a getter function from the contract may be preferable as it can efficiently return multiple results, and it would be more

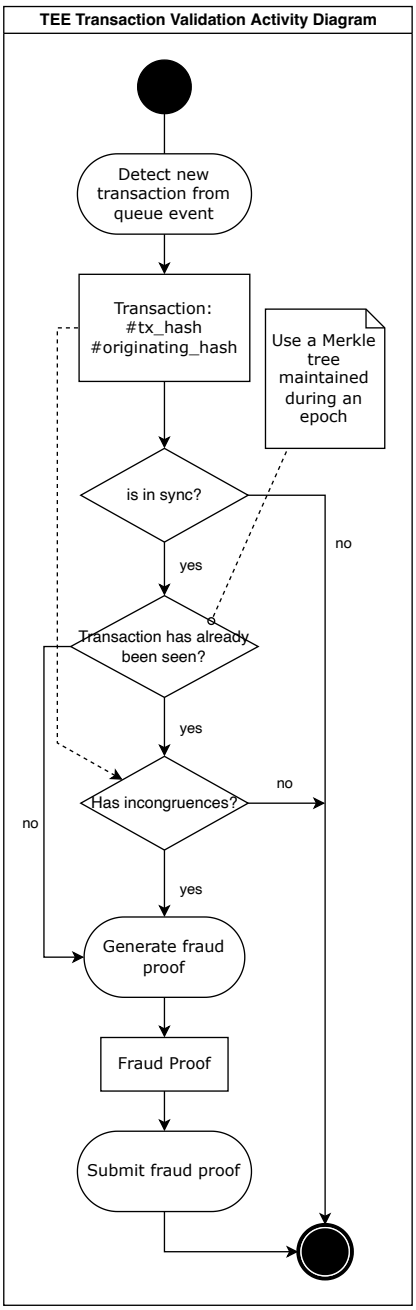

Figure 5: Sentinel transaction validation activity diagram.

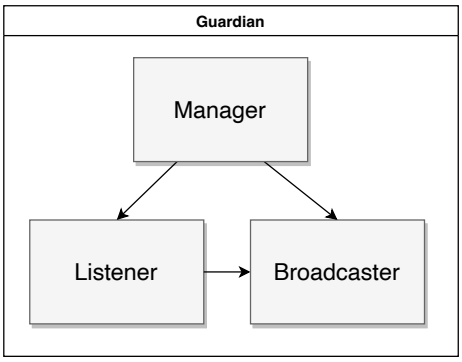

Figure 6: Guardian block diagram.

reliable than listening to events. Whenever a request is detected, the listener would check if it matches the originating one. If a mismatch is found, the Guardian would build a dismissal transaction. That would be pushed on-chain by the Broadcaster component. In particular cases, where a Sentinel and/or other Guardians have already submitted a fraud proof, the Broadcaster will not broadcast to save on gas fees.

Guardians do not have a registration manager as they are elected by the DAO. They will not put anything at stake, but they will need to identify when proposing to the DAO. Misbehaving Guardians may have their role revoked and incur in reputational damage.

In the first iteration, the Guardians could be a whitelist including trusted actors only. In the long term, Guardians could be directly elected by the DAO.

Similar to Sentinels, Guardians will also earn fees from the operation of pNetwork.

Table 1 reports the main differences between a Guardian and a Sentinel.

	Sentinel	Guardian
Enrollment	Staking at least 200,000 PNT	DAO Election
Detection	Block and transaction validation	Event listening
Intervention	Invalid requests by Relayers	Invalid requests by Relayers
Slashing	Staked PNT transferred to DAO	Reputational damage
Rewards	% of protocol fees	% of protocol fees
Service Cost	High	Low

Table 1: Brief comparison between Sentinels and Guardians.

4.1.4 pNetwork DAO

pNetwork already has its own Decentralized Autonomous Organization (DAO) that governs the network, manages pNetwork Treasury, and takes crucial decisions for the protocol development.

The DAO is an essential pillar for the decentralization of pNetwork, as it allows everyone that has PNT at stake to contribute to the development and growth of the project. DAO members have the ability to open and vote for Improvement Proposals (IPs). The IP passes if quorum is reached and if the PNT voting weight in favor is higher than the PNT weight against. For example, members will be responsible for electing which pTokens bridges to develop and support next, deciding on the fee mechanism of the network, and resolving any upgrade proposals.

A new and improved version of the DAO is under development, and it will be integrated with pNetwork v3 mechanism. In particular, DAO v2 will permit a user to lend and borrow PNT tokens: this will represent a new way for PNT holders to monetize from their assets, once they are also actively participating to governance proposals; it will also permit those users not owning PNT tokens to borrow tokens for running a Sentinel without being exposed to PNT price fluctuation, and have a share in the protocol revenues. Further information about the built-in lending and borrowing mechanism, as well as other pNetwork DAO v2's features will be shared separately.

With pNetwork v3, DAO will play an even more pivotal role. In fact, its intervention will be required in case conflicts arise from particular situations where Sentinels and Guardians do not reach a consensus on a Relayer proposal. In this case, a new vote will be automatically open, and the DAO will express itself by voting for or against the disputed transaction. Ideally, these should be rare conditions as the DAO intervention should be limited as much as possible because of the additional costs and increased processing times for pNetwork operations. DAO will also be responsible for electing Guardians. Finally, the DAO should intervene to decide whether a v3 actor shall be slashed, like, a misbehaving Relayer.

4.1.5 Misbehaviour and slashing

Anyone in the pNetwork will have the possibility to act as any of the Relayers, Sentinels, and Guardians roles. Because of this, a third party may decide to participate in the network and operate maliciously with the clear intent to damage the network. This section will analyze how the three types of actors could operate incorrectly or maliciously, and the possible ways to mitigate such behaviour.

For those actors required to stake something to act in their role, i.e. Relayers and Sentinels, slashing consists in temporarily locking a portion of a misbehaving actor's staked tokens to allow further checks, and, should the actor indeed be found to have been misbehaving, the locked amount will be transferred to the DAO.

Relayers are possibly the most dangerous element in the system as they request issuance and/or redemption transactions, and the protocol 'optimistically' expects these requests to be coherent with the originating blockchain state. Recall that to act in the Relayer role, one will have to stake a pre-defined quantity of PNT tokens.

In particular, a Relayer may:

- propose illegitimate transactions requiring the issuance of tokens that have never been locked on a native blockchain;
- propose illegitimate transactions requiring the unlocking of tokens that have never been redeemed on a host blockchain;
- propose transactions where the specified quantity and/or destination address differ from the user's request.

The challenge period will permit Sentinel and Guardians to detect such requests. A particular case is where a Relayer might propose multiple incorrect/malicious transactions in quick succession with the intent of creating a Denial of Service (DoS): mechanisms will be in place to prevent this by limiting the number of proposals any Relayer can do, and the challenge period may be extended to permit Sentinel and Guardians to detect this kind of activity.

Should a Relayer be found to be misbehaving, their staked amount of PNT will be slashed and transferred to the DAO.

The Sentinel role will be possible for registered pNetwork full nodes only, and will be required to have staked at least 200,000 PNT.

A Sentinel can misbehave if:

- it submits invalid fraud proofs, thus creating a censorship problem;
- it does not propose dismissal transactions for malicious requests in a timely manner.

In the former case, no Guardian should propose a dismissal transaction, and the DAO intervention will be needed to decide the outcome. Should the DAO find the Sentinel operated maliciously, slashing of their stake will occur. The latter case could happen if the Sentinel is inactive, or the codebase has been compromised. This Sentinel inactivity may be recognized by a mechanism in which tailored requests are periodically broadcast, which requests are expected to be responded to by a specific Sentinel in a timely manner.

A particular case would arise where Sentinels run a full node staking borrowed PNT from DAO v2. In such a situation, slashing the staked amount would not prevent a malicious Sentinel to misbehaving because their tokens are borrowed. A possible solution would be to require them to stake a different token, e.g. USDC or USDT, in a way they would not be exposed to the PNT market price, while still having something at stake to be slashed if acting maliciously.

As for a Relayer, a misbehaving Sentinel will be subject to slashing.

Finally, recall that Guardians do not need anything at stake as they are elected by the DAO. They will be required to identify themselves though: the reputational damage will be a deterrent for Guardians to misbehave. Guardians operate similarly to Sentinels, thus they can misbehave in the same way by submitting invalid fraud proofs, or by not proposing dismissal transactions. In the former case, the DAO should decide on the Guardian's activity, and, possibly, revoke its role. The latter case may be addressed with the aforementioned tailored-requests mechanism.

4.2 Smart contracts for pNetwork v3

A new on-chain component is required to manage Relayers' proposals. Depending on the proposal type, i.e. issuance or redemption, the component will be referred to as the issuance manager and the redeem manager respectively. Their functionality is almost the same: they should expose a method that will be called by Relayers to propose issuance or redemption; requests may be pushed into a queue (an event should be emitted upon insertion); another exposed method should be called to require the effective issuance or redemption in the tokens or vault smart contract, respectively; this method should implement a check for the challenge period expiration and fraud proofs submission. The latter is a critical part: the fraud proof needs to be checked for its validity; moreover, fraud proofs from unauthorized entities, e.g. unregistered Sentinels or Guardians, should be discarded. In particular, an oracle service may be needed to check if entities are entitled to act on their role, i.e. they are correctly registered in those blockchains where the DAO is not present.

The overall structure for the issuance manager is pictured in Figure 7. The

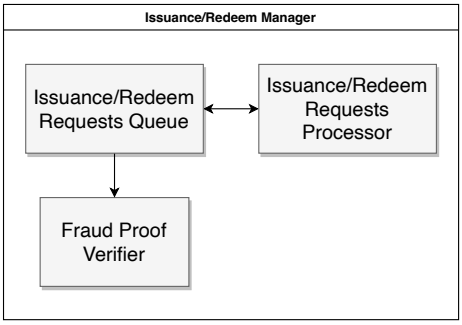

Figure 7: Issuance manager block diagram.

structure for the redeem manager is similar, but for the opposite chain.

The main component is the requests queue which is aimed to keep track of tentative requests by the Relayers. Sentinels and Guardians will be able to invalidate proposals in the queue: in particular, the fraud proof verifier will evaluate attestations from the Sentinels. When Relayers will call for the effective issuance or redemption, the request processor will check if the proposal request has been challenged, or if the challenge period has expired.

4.3 Use case diagram

The UML use case diagrams presented in this section will help to determine the various actors involved in pNetwork v3 and the possible actions they might take within the system.

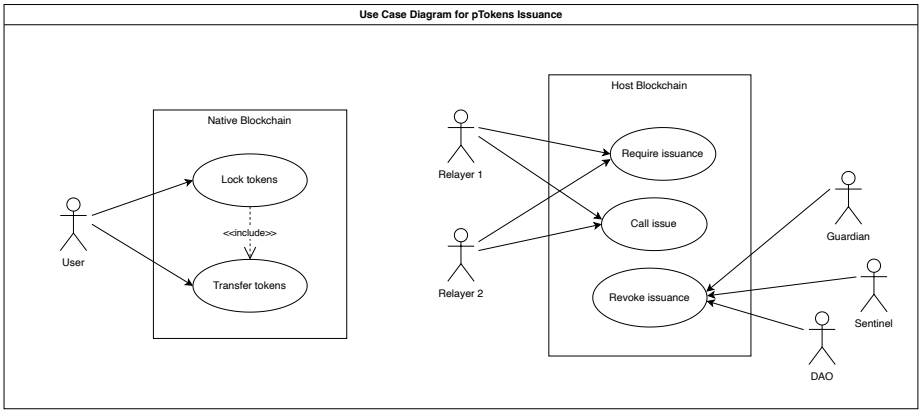

Figure 8: Use case diagram for tokens issuance.

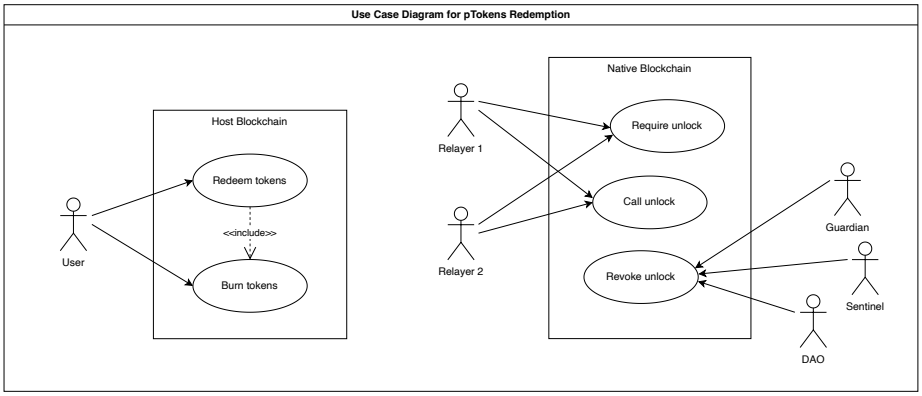

Figure 9: Use case diagram for tokens redemption.

Figure 8 shows how pNetwork v3 actors may interact when a pToken issuance is requested. A user initiates the cross-chain operation by locking his funds into a vault contract in the native blockchain. While locking his funds, the user needs to specify the host blockchain, a destination address in the host blockchain. Additionally, arbitrary data may be included by the user. A Relayer detecting that locking of funds would then propose pTokens issuance in the host blockchain, for the same locked quantity. This request will be submitted to a dedicated smart contract that will be responsible for queuing requests. Notice

the Relayer may be the same user that initiated the operation in the native blockchain. At this point, the Relayer's request will be seen by Sentinels and Guardians that will check for mismatches between the originating request and the one proposed by the Relayer: should mismatches be found, Sentinels and Guardians will propose a transaction to revoke the pTokens issuance. If the same Relayer's request gets challenged by both Sentinels and Guardians within the timelock period, it will be dismissed. Otherwise, the DAO may intervene to solve the dispute, deciding whether to dismiss the issuance or not. Finally, a Relayer may finalize the pToken issuance. This will be successful whenever the proposal has not been dismissed and the challenge period has expired.

The outlined actions will be further explained in Section 4.4.

By contrast, Figure 9 represents the UML use case diagram for the actors and the possible actions they may take when a pToken is redeemed. The process begins with a user redeeming his pTokens in the host blockchain; this results in the funds being burnt. In the native blockchain, a Relayer that has detected this operation will propose tokens to be unlocked from the vault contract. Similarly to the issuance proposal, this one may be challenged within the challenge period by Sentinels and Guardians, and, eventually the DAO. If it does not get dismissed, a Relayer may effectively call for the unlocking.

Notice that for both issuance and redemption, after the challenge period expiration, the same Relayer, or another one, needs to broadcast another transaction requiring the real transfer of tokens to the final user. This approach requiring two transactions, one for the proposal and another one for the real transfer, permits increased flexibility to easily adapt to different blockchains. For example, recall that the ERC20 standard[3] requires a `Transfer` event to be emitted when tokens are moved from one account to another; we cannot expect a transaction proposal to emit that kind of event because the same proposal may get dismissed; also deferring tokens availability is not a viable solution. Instead, the second call by the Relayer will result in a transaction where the real transfer takes place and the event can be emitted accordingly.

4.4 pToken issuance

This section aims to describe the issuance process with the aid of UML sequence diagram: actors' actions will be shown chronologically, and their effect analyzed depending on their order.

Figure 10 represents a successful pToken issuance. As explained in Section 4.3, a user locks his funds in a vault smart contract. The movement is seen by a Relayer in the host blockchain, and, as a consequence, it proposes an issuance transaction. The request is queued on-chain in the issuance manager smart contract queue along with the originating transaction hash, the destination address, quantity, and optional user data. At some point, the challenge period expires, and another Relayer finalizes the issuance: within the period, no other actor attempts to dismiss the request, thus the issuance manager smart contract internally calls the token smart contract to issue new tokens to the end user. The fraud proof from the Sentinel comes after the challenge period expiration, thus the issuance request can no longer be dismissed. Should the Relayer attempt to finalize the issuance during the challenge period, the request would fail. In case the Sentinel submits the fraud proof within the challenge period, its duration would be extended: at this point, Guardians may submit another fraud proof that would form a second-actor dismissal and thus invalidate the initial request.

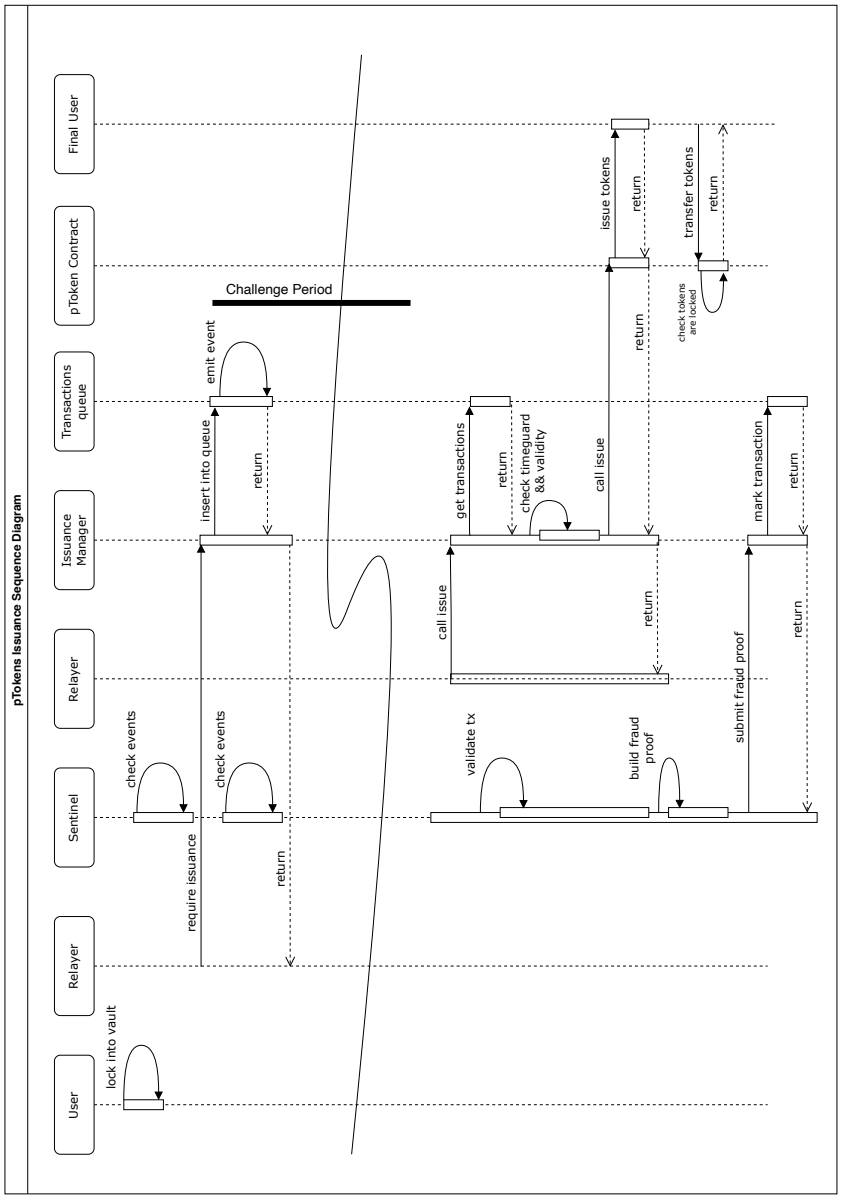

Figure 10: pToken issuance sequence diagram.

5 Conclusions

In this paper, we have presented a solution aimed to decentralize the pNetwork. The optimism inspired approach combined with the multi-provers approach appears as a viable way to secure cross-chain operations without the need for centralized entities. The pNetwork DAO v2 alongside the pNetwork v3 outlined herein allows anyone to play a major role in the network, fostering its growth and cementing its position in the future of web3 and decentralized finance.

References

[1] Buterin, V. Hardening rollups with multi-proofs. https://hackmd.io/@vbuterin/zk_slides_20221010#/6.

[2] pNetwork TVL. https://defillama.com/protocol/pnetwork.

[3] EIP-20: Token standard. https://eips.ethereum.org/EIPS/eip-20#events.

[4] Optimistic rollups. https://ethereum.org/en/developers/docs/scaling/optimistic-rollups/.

[5] Provable Things Ltd. pNetwork litepaper. https://uploads-ssl.webflow.com/60c1acb9d30b474ea009fe17/60f50bfcaeb557e5a9a46253_pNetwork-litepaper.pdf, 2020.