POCKET NETWORK

(VERSION 0.2.0)

Andrew Nguyen
andrew@pokt.network

Michael O'Rourke
michael@pokt.network

Luis C. de Leon
luis@pokt.network

ABSTRACT

Reliable node infrastructure is fundamental to the success of any decentralized architecture. It is not practical for application developers to host full nodes to provide back-end support to their application. Currently, the developer community is heavily reliant on solutions consisting of trusted central entities leading to centralization risk for protocols. In part this is due to lack of native relay node incentivization, but also a high level of complexity and inconvenience to the developer. To solve this problem, Pocket Network incentivizes individuals and companies globally to deploy nodes for any blockchain that has developer demand, by financially rewarding them for providing access points for decentralized applications.

This paper describes a decentralized relay network comprised of a federation of nodes, ensuring the honesty and integrity of every read and write request performed within the network. Pocket Network significantly reduces centralization risk, decentralized application development costs, and lowers the barrier for developers to create peer-to-peer applications for any blockchain.

1. INTRODUCTION

The world is headed towards a multi-blockchain future. Infrastructure is a vital part of what allows for the maturation of the blockchain application ecosystem in this vision of the future. In April 2017, Ethereum infrastructure platform, Infura, was handling 175 million API requests per day. Two years later, that number increased by more than 7,000% to over 12 billion requests per day. By lowering the barrier of entry for developers to build decentralized applications and providing easy to use, scalable infrastructure, Infura laid the groundwork for Ethereum's exponential growth in 2017 [1].

However, centralized blockchain infrastructure solutions lead to central points of failure for decentralized applications. This is a significant worry to the Ethereum, Bitcoin and broader blockchain community [2], [3], [4]. This concern can largely be attributed to the node incentivization problem, where individuals are not incentivized to run full nodes for any blockchain. Bitcoin and Ethereum have infrastructure and developer advantages that other blockchains do not. They have well supported and documented infrastructure and APIs, allowing developers to build applications easily on their networks [5], [6]. New and future blockchains need similar support. Pocket's single interface can provide this much-needed infrastructure support for *any* blockchain.

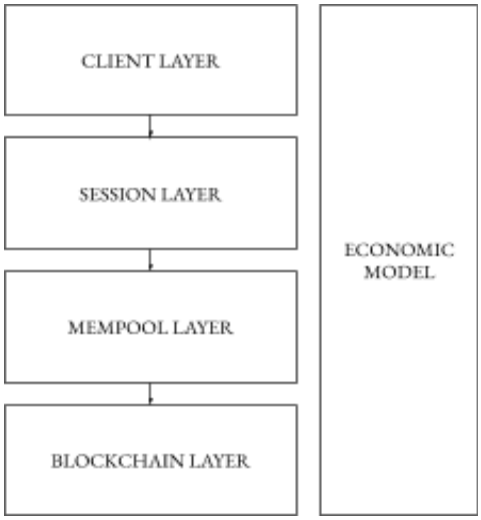
This paper introduces Pocket Network, a decentralized system that provides an incentive for individuals and businesses to run full nodes. This, in turn, solves the node incentivization problem, relieving infrastructure centralization risks and providing a common interface for all blockchain infrastructure.

2. POCKET DESIGN

Within the context of this document, the Pocket Network architecture is represented through the characterization of four abstraction layers:

1. Client Layer
2. Session Layer
3. Mempool Layer
4. Blockchain Layer

Similar to that of other conceptual computing models, the four layers of the Pocket Network interact and communicate with adjacent layers [22].

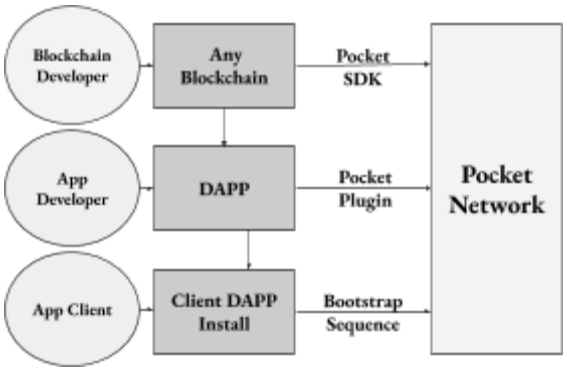


Though there are many mechanisms and complexities within the four layers, it is essential to start with the high-level structural design to understand the network as a whole. For the sake of clarity and breadth of the intended audience, this paper takes a top-down approach in explaining the abstraction layers.

2.1 Client Layer

In the Pocket ecosystem, there are three types of end users:

1. Blockchain Developer
2. Application Developer
3. Application User



2.1.1 Blockchain Developer

Within the context of the Pocket ecosystem, a blockchain developer is an individual or group who is building a blockchain or publicly verifiable database. Any blockchain developer can allow their particular chain to be accessed through the Pocket Network by utilizing the developer toolkit to build a Pocket Developer Plugin (allows developers to have a single interface to develop decentralized applications on the new network).

The developer toolkit is an SDK built by The Pocket Network core team. The toolkit provides a single interface that allows emerging blockchains to be supported by the Pocket Network.

Blockchain Developer Toolkit

Pocket is "blockchain agnostic," meaning Pocket protocol can provide infrastructure to any blockchain.

This plugin system introduces a single interface that allows any decentralized architecture to access the global Pocket Network, removing the need for blockchain developers to maintain their own nodes.

2.1.2 Application Developer

Within the context of the Pocket ecosystem, an application developer is an individual or group who is building an application that utilizes a blockchain. For an application developer to access Pocket's global infrastructure of full nodes, two things must happen:

1. Use the Pocket Developer Plugin for whatever blockchain the decentralized application requires.
2. Stake POKT for utilization within the network.

A Pocket Developer Plugin is an extension of the developer SDK. The plugin system provides a single interface that allows developers to connect to whatever blockchain they want to use for their decentralized application.

Pocket Developer Plugin

To provide complete support to developers who require less common or multiple blockchains, Pocket introduces a plugin system that supports any blockchain. From the developer's perspective, a few lines of familiar code is the difference between a transaction on Ethereum or a transaction on Bitcoin.

2.1.3 Application User

Within the Pocket Ecosystem, a client is any person using an app built by an application developer (defined in 3.1.2).

Application User Bootstrapping Sequence

The Pocket bootstrapping sequence for an application accessing Pocket Network for the first time consists of a two-tiered approach. This design can be considered similar to bootnodes within the Bitcoin or Ethereum clients.

1. The first tier of the Pocket bootstrapping sequence is through hardcoded DNS seeds hosted by the network.
2. The second tier is a master list of the connected IP's hosted on the Pocket block explorer.

For the sake of efficiency, a Pocket Dispatch Node is selected by geolocation proximity using the Application User's client.

2.1.4 Client Layer Nodes

NOTE: It is important to note that the separation and differentiation of nodes only refer to the roles that nodes play within the abstraction layer for ease of explanation and not a tangible divide in practice. Thereby, for clarity, nodes, in this paper, are delineated based on the function that they perform within the specific abstraction layer.

```
Node interface{
  Hash uniqueID;
  // identifier of the node

  Geolocation geo;
  // location of the node based on IP

  PeerList peerlist;
  // a list of other nodes

  SessionList sessions;
  // a list of ongoing Sessions
}
```

The first point of contact between the Application User and the Pocket Network is a Dispatch Node.

```
DispatchNode interface extends Node{
  boolean SessionExists(Hash devID);
  // Dispatch Node checks for ongoing sessions for a developer

  void NewSession(Hash devID, Geo geoLoc, List blockchains);
  // Dispatch Node creates a new session for a developer

  void AddClient(Hash uniqueID, Hash devID);
  // Dispatch Node adds a client to an ongoing session
}
```

The Dispatch Nodes are responsible for connecting Application Users with a Session. This process is known as the dispatching sequence:

1. The Application User requests a connection through the client bootstrapping sequence and finds the nearest Dispatch Node.
2. The Application User passes its geolocation information, its unique identifier, the developer id, and the public keys from any given wallet that is writing transactions to a hosted blockchain.
3. The Dispatch Node connects the Application User to the given Session.

2.2 Session Layer

A Session is an ongoing relationship between an application developer and the Pocket Network. A Session is linked by a developer's public address, which Application Users access along with their unique identifier during the dispatching sequence.

2.2.1 Session Layer Nodes

To understand the capacity and extent of node involvement within the session layer, nodes are abstracted into three distinct forms.

1. Dispatch Node
    a. Nodes that are responsible for assigning developer applications to the correct pseudo-randomly generated Session.

2. Service Node
    a. Within the session layer, nodes that service the API requests of the Application Users and relay the data through the Pocket Network are Service Nodes.

3. Validator Node
    a. Nodes that verify the receipts from Application Users and Service Nodes.

Dispatch Node

Within the session layer, the role of a Pocket Dispatch Node extends from the dispatching sequence (as defined in the client layer in section 3.1.4).

As previously stated, the Dispatch Node is responsible for connecting Application Users with Sessions. However, if a Session does not currently exist for a specific developer, the Dispatch Node is responsible for creating one. The Dispatch Node generates the validator set by a publicly verifiable pseudo-random algorithm described below.

```
Validator Set = Hash(N Block hash + DevID)
```

The selection of the Service Nodes is based on geolocation proximity and hosted blockchain specifications. The Service Node and Validator Nodes in the Session must independently check whether they belong there. Any incorrect participant's stake will result in burning of staked POKT.

```
DispatchNode interface extends Node {
  function FindServiceNode(Hash devID);
  // finds service node for client based off location and blockchains needed

  function GenerateValidatorNodes(Hash devID);
  // responsible for generating random Validator Nodes for a new session
}
```

Service Node

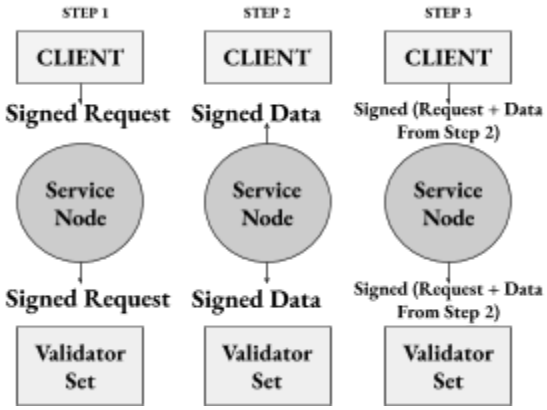
Service Nodes are high availability infrastructure running on existing cloud providers or local data centers. All Service Nodes must participate in Proof of Stake consensus (see section 2.4.1). Service Nodes must be able to scale with the average throughput of the network, or be left behind. They must

pass constant liveness checks from Validator Nodes and other Service Nodes. Service Nodes must run a full blockchain node along with an efficient caching layer to provide up-to-date information and quick responses to the Application User.

Anyone can be a Service Node in the Pocket Network once they have provided the minimum requisite stake. Initially, a Service Node is an unknown, untrusted entity within Pocket Network. By providing good service and generally being a good actor within the system, Service Nodes receive Karma as a reputation score to earn their way into becoming a Validator Node.

The role of a Service Node within the session layer is abstracted into four steps.

1. Receive a data relay request from the Application User and confirm the digital signature.
2. Execute the relay request on the hosted blockchain.
    a. If it is a read request, synchronously verify the read on the hosted blockchain.
    b. If it is a write request, relay and asynchronously wait for data confirmation.
3. Broadcast the receipt from the Application User and Service Node to the Validator Nodes.
4. Repeat steps 1-3.




```
ServiceNode interface extends Node{
  function confirmSignature(Hash signature);
  // confirm signature from the Application User

  function  digitalSignature(Data data);
  // signs the request receipt and response data

  function executeRequest(List blockchains);
  // service node executes request from client on hosted blockchains

  function relayRequest(List validators);
  // service nodes relay request from client to validators
```

```
  function relayData(Client cli, List validators);
  // service nodes relay data to client and validators


  function reimburse(Amount x);
  // service nodes can submit proposals to DAO for compensation of costs
}
```

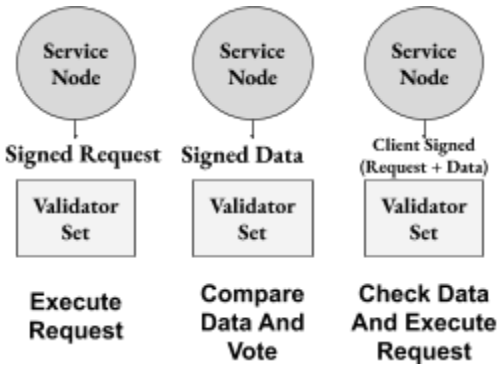
Validator Nodes

A Validator Node is a Service Node who has met the relevant reputation (Karma) and stake requirement to become a Validator Node. Validator Nodes earn the bulk of the block reward (see section 3.2) for verifying the relationship between an Application User and Service Node. Validator Nodes must also act as Service Nodes, meaning the infrastructure costs and stake requirements for being a Validator Node are significantly higher.

A pseudo-randomly selected set of 4 Validator Nodes and 1 Service Node will be in a Session.

The role of the Validator Nodes is abstracted into six steps:

1. Receive signed relay receipt from the Service Node and confirm the signature.
2. Compare Service Node data with Application User data.
3. Vote on correctness and integrity.
4. The majority consensus receives positive karma (a scoring system described in section 3.3.1), and the minority consensus receives negative karma scoring.
5. Repeat steps 1 - 6.
6. Submit relay batch when finished.



```
ValidatorNode interface
extends ValidatorNode {
  function validateData();
  // Validator Nodes validate the data received from the service nodes by comparing
it with their results

  function vote();
  // validators vote on integrity and accuracy of data; this allows the karma
scoring of the session by the protocol


  function signRelayBatch();
  // validators sign the batch of relays from any given session

  function submitRelayBatch();
  // validators submit the signed relay batch once consensus is reached
}
```

2.2.2 Relay Batches

The product of a session in the Pocket Network is known as a Relay Batch. The relay batch is a single transaction within the Pocket Network that summarizes the work and karma scoring of a session completed within the confines of a maximum amount of time or relays.

```
RelayBatch interface extends Transaction{
  Hash developerID;
  // the developerID;

  BigInteger relaysCompleted;
  // total number of relays completed

  List serviceNodes;
  // the ids of the service nodes

  List validatorNodes;
  // the ids of the Validator Nodes
```

```
  Hash validatorNodeSignatures;
  // signatures of the Validator Nodes participating



  Score score;
  // structure holding the scores of the session participants. This structure also
allocates mint
}
```

2.3 Mempool Layer

Between the session and the blockchain layer exists an abstraction layer known as the mempool. Node intercommunication, off-chain validation, liveness checks, and relay consensus are features of the mempool layer.

The mempool is a P2P internal table structure that is held by all nodes. This structure holds important network information and potential blocks that could be submitted to the chain after a consensus among Validator Nodes upon validation.

Mempool contains:

1. Peers
2. Open sessions
3. Active Validator Nodes
4. Active Service Nodes
5. Pending block
6. Designated miner

```
Mempool interface {
  List Peers;
  //  list of peers

  List OpenSessions;
  //  list of open sessions

  NodeList validators;
  //  list of active Validator Nodes

  NodeList service;
  //  list of active service nodes
```

```
  Block pending;
  //  pending block

  Node minter;
  //  designated minter
  function CheckTX();
  //  crosscheck transactions based on rules and state conflicts

  function BroadcastChanges();
 // mempool synchronization must be achieved. Project any changes using peer list
}
```

2.4 Blockchain Layer

The purpose of the Pocket blockchain is to maintain a decentralized ledger of the state of consensus of all the transactions relayed by the Pocket nodes. Much like any other blockchain, the Pocket blockchain can be thought of as the ultimate source of truth, meaning any transactions confirmed on the Pocket blockchain are provable and authoritative.

2.4.1 Proof of Stake

To protect the integrity of the blockchain, Pocket Network uses the Tendermint consensus protocol due to it being among the most well-tested and mature Proof of Stake (PoS) algorithms. Pocket Network does not need to optimize for block times or transactions per second as most of the protocol's work happens asynchronously off chain. This enables more flexibility and decentralization in number of total nodes able to participate in consensus. To participate in PoS consensus you must be a Service Node for at least one supported blockchain.

<u>Block Rewards</u>

All block rewards are directly tied to the usage of the protocol, meaning each block will have a dynamic, inflationary reward.

It is important to note that Service Nodes do not earn directly from work done - this is to avoid any self-dealing conflicts. Service Nodes earn POKT through two primary ways:

1. Being chosen as a block producer
2. Chosen as a single untrusted node in a trusted Validator pool

This enables Service Nodes to have consistent income to cover infrastructure costs and gain Karma as well. Costs are significantly lower due to less validation responsibilities. Block rewards are split amongst all Validators who participated in a specific relay batch for a given amount of time.

Delegation

While Pocket Network is not a delegated PoS network, POKT holders are able to stake on behalf of Validator Nodes and Application Developers for additional rewards. Pocket Network will facilitate an off-chain market through its public communication forums to match POKT holders with protocol participants through traditional legal agreements in revenue and equity.

2.4.2 Karma

Karma is Pocket Network's internal reputation system. Within a session, Service and Validator Nodes are scored based on a majority consensus among participants. Scoring is cumulative and is tethered to the unique identifier of the node. Achieving a certain high karma threshold is the first requirement to become a Validator Node. Scoring is based on various functions such as response times to applications, liveness checks from other nodes, challenges to incorrect votes, and correct votes on validation.

2.5 Known Attack Vectors

Spam-for-profit attack

Definition:
The attacker "self-deals" POKT token by servicing themselves.

Solution:
Prohibiting Service Node POKT payments and introducing the Karma concept.

One-kill-at-a-time attack:

Definition:
A corrupt group of Validator Nodes collude to falsely report a selected individual node as a bad actor anytime they are in a quorum with that victim but play honestly in all other interactions. Eventually, the targeted node gets burned and banned. The corrupt nodes then move on to the next victim.

Solution: A Truebit style challenge-response game that enables the one honest node to prove the colluding set lying on chain. This will lead to another randomly selected set of Validators to verify a set of transactions

DDOS to prevent new block:

Definition:
An attacker will know the governance order for the block minting process.

Solution:
Use a consensus algorithm that does a round-robin leader selection with a timeout such as Tendermint.

Bot Net App (Malicious Developer):

Definition:
The app takes down the service nodes before they can report anything to the validators. In turn, they DDOS the network and their stake is never burned.

Solution:
Middleware security mechanisms like Cloudflare to prevent overflow.

Stealing Developer Stake:

Definition:
A developer using another address' allotted stake for throughput.

Solution:
Developer Policies and revocable tokens embedded in the application code.

3. ECONOMIC MODEL

Pocket's economic model involves a continuous, inflationary token issuance that utilizes staking and burning for long-term economic sustainability. The core concepts in the economic model are:

1. Unit of Work
2. Staking
3. Inflation
4. Burning

3.1 Unit of Work

Each unit of work is a valid relay agreed upon by Validator Nodes. The protocol rewards the participants with POKT and/or Karma. It is important to note that unlike Proof of Work or Proof of Stake the unit of work in Pocket Network is fundamentally separated from how the nodes reach a consensus on the blockchain. Validator nodes reach a consensus on each unit of work and then batch the total as one transaction into the ledger.

By tying the inflation reward as closely as possible to work done, it increases efficiency and significantly reduces the cost of coordination for the protocol as a whole. The cost of validation for a relay is estimated to be $0.0000007 to the network if running on traditional cloud providers. It is expected that competition in pricing will move a large amount of throughput of the network to physical data centers, decreasing costs while increasing decentralization and censorship resistance.

3.2 Inflation Reward

The POKT cryptocurrency is permanently inflationary, based off of a unit of completed work. Relay batches are submitted per Session and inflation reward is awarded to the corresponding parties involved.

For each relay completed, the amount of POKT created will be constant for the life of the protocol unless otherwise changed through governance mechanisms. The total reward is split among the following participants.

1. 89% to Session participants
2. 10% to Pocket DAO
3. 1% to Service Nodes participating in PoS consensus

Four Validator Nodes and one Service Node get chosen per Session, and split the reward evenly per relay validated.

Service Nodes are exposed to financial upside by participating in Proof of Stake to protect the integrity of the network. To provide consistency to their random block rewards, 1 Service Node also gets randomly assigned per Session to earn Karma and an additional inflationary reward for providing validation. This provides a low barrier of entry for anyone who wishes to work their way up to becoming a Validator Node.

3.3 Staking

Staking is the locking up of the POKT cryptocurrency within the protocol for a minimum time period. Pocket's economic model is designed to have as much POKT staked within the system as possible. There are two reasons a participant would stake:

1. To pay for throughput as an Application Developer
2. To ensure all actors within the network are economically incentivized to act within the confines of the rules of the protocol

Application Developers pay in advance for access to non-native blockchains by staking POKT. The amount of POKT a developer stakes is directly dependent on the number of relays they intend to use. The protocol throttles the number of API requests a developer can send to a node in a given time period, allowing the developer to pay through inflation with POKT. After an initial bonding period, the developer may exit the network by starting the unstaking process of their POKT.

Nodes stake POKT to ensure the authenticity of the network and earn rewards by validating relays and transactions. All nodes must have a minimum amount of POKT staked before participating, this stake is their incurred risk; their staked POKT will be burned if the protocol is not followed. Service nodes receive rewards from each Session and have the opportunity to participate in PoS. Once, certain threshold requirements are met, a service node can become a Validator node. Validator nodes earn the bulk of the token reward in POKT, which is calculated based on each relay validated in the Session.

Pocket Network economic model utilizes the Tendermint PoS leader election algorithm to protect the integrity of the results on the blockchain. All nodes in the network have an opportunity to produce a block, their chances are proportionate to their stake. By design, Pocket incentivizes a high stake participation rate, through PoS, Application Developer and Node staking.

3.4 Burning

From a security standpoint, burning is the mechanism that economically disincentivizes bad acting. From an economic standpoint, burning is the mechanism that reduces the total supply of the POKT token. At network maturity, the goal is for the circulating supply to be slightly deflationary or with as little inflation/deflation as possible through the following burning mechanisms:

1. Nodes not following rules of the protocol
2. Developers overusing their stake and throttle thresholds
3. Transaction fees
4. POKT being burned through DAO proposals

If a Node miscasts a vote on a relay, the protocol will burn the equivalent amount of POKT that would have been an inflationary reward from it stake. Should a Node continue miscasting votes, burning will continue in a quadratic formula, becoming more severe over each block, until the Node has reached below the minimum stake. If Nodes report Application Developers going over their allotted stake thresholds, their stake will be burned by the equivalent of the related inflationary reward.

4. GOVERNANCE

When considering blockchain longevity and scalability of the Pocket Network, a mechanism for human governance is critical for creating a fault tolerant system.

4.1 Pocket DAO

Pocket's Decentralized Autonomous Organization (DAO) ensures that Pocket Network is eventually governed by its contributing stakeholders, who will have control over protocol upgrades, development, and management of ecosystem resources. The Pocket DAO receives 10% of every validated relay within the network. This funding mechanism helps combat the tragedy of the commons, and ensures that core protocol research will continue in perpetuity. To safeguard against early vulnerabilities, it will launch with a small group of trusted members, then progressively open governance and distribute responsibilities as milestones are achieved.

The Pocket DAO's mission is to use governance mechanisms for the enhancement of Pocket's developer experience, where

i) governance mechanisms include

- proposals
- votes
- bounties
- disputes
- and any other effective mechanisms that emerge.

ii) developers include

- blockchain developers
- node operators
- app developers.

4.1.1 Membership

There are two types of membership in the Pocket DAO.

DAO Voter

A DAO Voter is a Pocket address that has been whitelisted with voting permissions. The Pocket DAO must first verify participation in the Network to be admitted as a DAO Voter. While there is no hard line as to what is enough contribution, this means potential Voters must have proven to be capable and willing to be one or more of the following types of participant within the Pocket Network:

- **Node Operators:** test and run Pocket nodes
- **Developers:** test and build applications using Pocket tools
- **Enthusiasts:** understand and support the Pocket Network.

DAO Contributor

A DAO Contributor is someone who is working on an active proposal or has been invited to contribute to the discussion by another DAO Member. They can participate in most DAO communication channels, and voice their opinion during the decision making process, but they do not have a vote.
Being an active DAO Contributor is a critical path for non-technical enthusiasts who wish to be approved as DAO Voters.

4.1.2 Proposals

You do not need to be a DAO Member to submit a proposal to the DAO. All proposals will carry a POKT fee that is immediately burned.

- **DAO Votership Proposals:** Voters will be approved subject to the guidelines in 'Membership - DAO Voter'. These proposals carry a 2,000 POKT fee.

- **Network Improvement Proposals:** These proposals can include infrastructure improvements such as SDKs or any other software that interacts with Pocket Core and SDKs directly. In the bootstrap phase, this can not include changes to the Pocket Core Golang implementation maintained by the Pocket Core team, but could, for example, include creating a second Pocket Core implementation in Rust. These proposals will carry the standard 500 POKT proposal fee.

- **Ecosystem Improvement Proposals:** These proposals can include the creation of custom plugins for different blockchains, using Pocket tools in the development of applications, and building additional tools for the Pocket ecosystem. These proposals will carry the standard 500 POKT proposal fee.

- **DAO Improvement Proposals:** Amendments to this DAO, including operational changes and the adoption/refinement of governance mechanisms. Only DAO Voters can submit DAO Improvement Proposals. These proposals will carry the standard 500 POKT proposal fee.

5. FUTURE CONSIDERATIONS

While Pocket's design doesn't enable direct cross-chain communication between protocols through mechanisms like atomic swaps and Hashed TimeLock Contracts (HTLCs), Pocket is in a unique position to act as connective tissue for any open source protocol. The expectation is that thousands of Pocket Nodes are running dozens or hundreds of full nodes for other blockchains. This enables novel solutions to common areas of research for the blockchain industry.

Multichain Applications and Universal Wallets

Through Pocket's integration with existing developer SDK's, interacting with many chains at once through the same web or mobile user experience is easily accomplished. The advent of blockchain agnostic Web 3 browsers can simply add in Pocket as a dependency and developers can quickly integrate features from decentralized applications on completely different blockchains.

Since Pocket Network consists of a worldwide network of nodes, pluggable to any chain, a universal wallet is now a feasible product. An entire crypto portfolio on a single wallet file is not far off.

Oracle Networks

Pocket plugins can easily be built to support any publicly verifiable database. Instead of Validator Nodes accessing their local copy of a blockchain, they can access public APIs such as sports scores, weather, or even political events. Most attempts at agnostic Oracle protocols force application developers to conform to the specifications of that protocol's rules. Pocket's plugin system acts as a dependency for existing applications, making a truly censorship resistant Oracle much simpler to implement.

Decentralized Exchanges

Millions of dollars of research has gone into atomic swaps and HTLCs to enable cross chain communication and on-chain decentralized exchanges. Due to Pocket Validator Nodes having to stake POKT, they can act as a trustless conduit to provide liquidity across blockchains, without the technical difficulties of on-chain communication.

Meta Reputation

The challenge with blockchains is that they are siloed from each other, and have no way to communicate data across themselves. The record of a relationship between an Application User and the blockchain they are communicating with is effectively being recorded through the Pocket blockchain. If this relationship was backed through individual staking of POKT and able to be transferred across chains, meta reputation would unlock the potential of many areas of current research such as Universal Basic Income, identity and decentralized finance applications.

6. ACKNOWLEDGMENTS

The Pocket Network derives from countless ideas and projects that existed far before its inception. There are many individuals who must be given considerable credit, thanks, and praise. Firstly, to the co-founders of Pocket Incorporated: Pabel Nunez and Valeria Benitez Florez for their expertise in their respective fields and for envisioning a groundbreaking idea of this magnitude. To Tracie Myers for her predictive big data advising and protocol economic analysis. And a personal thanks to Shawn Regan for numerous discussions and contributions to the core protocol. Without the help of many brilliant people, the Pocket Network would not exist.

7. REFERENCES

[1] M. Wuehler, "Scalable Web3 Infrastructure With Infura", Rice University, 2017.

[2] J. Pitts, “ethresear.ch”. 26-Feb.-2018.

[3] J. Ray, “EIP: Reward for clients and full nodes validating transactions #908”. 28-Feb.-2018.u

[4] B. Bernstein, “Twitter”. 10-Jul.-2018.

[5] “Ethereum Javascript API”. 30-Jul.-2015.

[6] W. Bins, “Bitcoin Developer Documentation”. 2009.

[7] M. Castro and B. Liskov, “Practical Byzantine Fault Tolerance,” *Proceedings of the Third Symposium on Operating Systems Design and Implementation*, 1999.

[8] “Introduction to Hyperledger Business Blockchain Design Philosophy and Consensus,” *Hyperledger Architecture*, Volume 1, 2017.

[9] “Tendermint: Byzantine Fault Tolerance in the Age of Blockchains,” *A Thesis Presented To The University of Guelph*, 2016.

[10] C. Dwork, N. Lynch, and S. Larry, “Consensus in the Presence of Partial Synchrony,” *Proceedings of the 3rd ACM Symposium on Principles of Distributed Computing*, 1984.

[11] J. Choi, “Ethereum Casper 101”, 21-Oct-2017

[12] V. Buterin, V. Griffith, “Casper the Friendly Finality Gadget”, 26-Oct-2017

[13] “Ethereum Github CBC Casper”, 2017

[14] E. Duffield and D. Diaz, “Dash: A Privacy-Centric Crypto-Currency,” 2015.

[15] S. Nakamoto, “Bitcoin: A Peer-to-Peer Electronic Cash System,” Oct. 2008.

[16] D. Petkanics, “Inflation and Participation in Stake Based Token Protocols,” Medium, Dec. 2017.

[17] J. Peterson, J. Krug, M. Zoltu, A. K. Williams, and S. Alexander, “Augur: a Decentralized Oracle and Prediction Market Platform,” Jul. 2018.

[18] W. Warren and A. Bandeali, “0X: An open protocol for decentralized exchange on the Ethereum blockchain,” *0xproject.com*, Feb. 2017.

[19] V. Buterin, “Ethereum: A Next-Generation Smart Contract and Decentralized Application Platform,” *ethereum.org*, Feb. 2013.

[20] D. Petkanics and E. Tang, “Livepeer: Protocol and Economic Incentives For a Decentralized Live Video Streaming Network,” *livepeer.org*, 2016.

[21] L. Cuende and J. Izquierdo, “ARAGON NETWORK: A DECENTRALIZED INFRASTRUCTURE FOR VALUE EXCHANGE,” *aragon.one/network*, Apr. 2017.

[22] R. Stewart and B. Gilbert, “CONCEPTUAL MODELING: DEFINITION, PURPOSE AND BENEFITS ”. 2015.

[23] “Amazon AWS Docs: Access Policies”.

[24] C. Lacina, “Coin Telegraph: The Inevitable Failure of Proof-of-Stake Blockchains and Why a New Algorithm is Needed (Op-Ed)”. 24-May-2015.

[25] “Bitcoin Wiki: Protocol rules”. 25-Aug.-2017.

[26] “Bitcoin Improvement Proposals,” *Bitcoin Wiki*. 13-Feb.-2016

8. GLOSSARY

Application User
An individual or group using an app built by an application developer.

Application Developer
An individual or group who is building an application that utilizes a blockchain.

Blockchain Developer
An individual or group who is building a blockchain or decentralized digital database.

Bootstrapping Sequence
The initialization process to first connect to the Pocket Network.

DAO
A decentralized autonomous organization that is governed by the protocol's primary actors.

Dispatch Node
The first point of contact between the network and the Application User.

Eon
A unit of time consisting of a summation of Eras.

Epoch
A unit of time consisting of a summation of blocks.

Era
A unit of time consisting of a summation of Epochs.

Federated Node
A trusted node that maintains a series of threshold requirements.

Karma
An autonomous node scoring system based off of the consensus within a Session.

Mint
Newly created POKT proportional to the number of relays completed.

Overflow
Allowed developer throughput overages per Epoch.

Pocket Improvement Proposal
A formalized publication specifying Pocket propositions.

Pocket Developer Plugin

An extension of the Pocket Developer SDK that provides a single interface for developers to connect to blockchains.

Pocket Developer Toolkit

A single interface for emerging blockchains to be supported by the Pocket Network.

POKT

The cryptographic token needed to participate within the Pocket Network

Session

An ongoing relationship between an application developer and the Pocket Network

Relay

A blockchain API request and response transmitted through the Pocket Network.

Relay Batch

The product of a session that is submitted to the blockchain.

Rollover

Unused developer throughput added to the next Epoch's allocation.

Service Node

The act of a node receiving API requests and responding to the data.