NESA PROTOCOL

THE LAYER-1 FOR TRUSTED AI ON CHAIN

Nesa Whitepaper

First edition, published in 2024.

Contents

Chapter 1

Introduction

In the rapidly evolving landscape of blockchain and artificial intelligence, Nesa represents a groundbreaking convergence of two technological frontiers.

While smart contracts have been a transformative force in the blockchain domain, their potential remains inherently limited if they lack the capability to integrate AI models into their operational framework. True to their name, smart contracts can only achieve genuine intelligence and versatility when empowered with the advanced decision-making and analytical abilities of AI. This is the crucial step towards realizing the full potential of Web3.

The need for this integration is especially apparent in real-world applications today. Consider, for example, the incorporation of real jurisdictional laws into a DAO's smart contracts. Legal statutes are often fraught with ambiguities and complexities that cannot be directly translated into code. In such a scenario, the nuanced interpretation and judgment akin to a legal expert is essential.

This is where AI, particularly advanced models like Large Language Models, come into play, acting as impartial 'judges' to assess and apply legal principles. Indeed, merging AI and smart contracts is not merely an enhancement but a fundamental step to bridge the gap between the binary world of code and the multifaceted reality of human laws and societal norms.

Nesa is the Layer-1 for AI, bringing critical inference to the blockchain in a scalable, secure, and computationally efficient end-to-end system powered by ZKML. This whitepaper introduces two major innovations distinct to Nesa. We begin with a detailed look at the AIT, the *Artificial Intelligence Terminal*, which is our fully integrated, decentralized system for AI model inference queries, tailored to the unique needs of real-world applications.

The AIT is a revolutionary solution to facilitate AI tasks trustlessly on the blockchain, while query execution occurs off-chain. Unlike traditional mining where participants are rewarded for solving arbitrary cryptographic problems, our system awards fees in our native asset, $NES, to pools of miners utilizing GPU-powered nodes when they have successfully solved AI model inference queries posted by platform users. This approach represents a paradigm shift in the mining process, aligning the computational power of miners not just with securing the blockchain but also directly with the practical application of AI inference.

The second technological innovation of Nesa is its decentralized inference protocol. This protocol implements dual transactions at the blockchain level, and upon release

further breaks down its latter transaction into bifurcated subphases for a commit-reveal mechanism. This implementation enables the first trustless environment where AI computations are performed transparently and are reliably reported on-chain using ZKML, while safeguarding against dishonest behavior and free-riding.

Nesa's cryptographic hybrid-privacy addresses one of the most significant hurdles in crafting a privacy-focused decentralized inference system - the simultaneous need for confidentiality and verifiability. Traditional encryption methods, while adept for preserving confidentiality by securing data at rest or in transit, render data unusable for practical computation by also obscuring the data from the very systems that need to process it. This makes it unintelligible and non-operable for computational algorithms, creating a substantial barrier in a system where primary utility is derived from the ability to perform complex computations and inference on user data.

The intersection of these challenges—ensuring confidentiality while enabling computation and providing verifiability—requires a sophisticated balance of cryptographic innovation through ZKML and system design that Nesa has engineered to ensure not just the confidentiality of their data or models but also the assurance that the computations performed are correct, trustworthy and intended. Our hybrid-privacy system architecture is designed for scale and production, researched, conceived of and built from scratch by a veteran team of researchers in AI and cryptography at Nesa.

This whitepaper delves into the intricacies of our architectural design, these cryptographic advancements, and the platform's tokenomics structure, as well as the broader implications of our project in reshaping the landscape of decentralized AI applications.

Chapter 2

Introducing the AIT - The Uniform Execution Environment

We introduce the AIT, a pioneering new system architecture that is the new standard for blockchain-powered artificial intelligence. AIT stands for the *Artificial Intelligence Terminal,* the first uniform execution environment connecting on-chain assets with off-chain activity to achieve trustless, large AI model inference. The AIT ensures uniform execution across all nodes in the Nesa network, analogous to the role played by the Ethereum Virtual Machine (EVM) in the Ethereum ecosystem. This standardized design across all system requests regardless of origin, function and purpose solves the major industry problem of discrepancy in model execution that can prevent consensus on AI inference results.

At its foundation, the AIT is designed to streamline the AI computation process by providing a consistent set of rules and execution protocols, which every participating node must follow. This not only guarantees that all nodes produce identical results given the same model parameters and input data, but also relieves node operators from the intricacies involved in setting up execution environments, facilitating easier adoption and participation in our network.

The AIT framework offers the flexibility to accommodate a wide range of AI models. By incorporating AIT instances with model parameters on the blockchain, we create an on-chain repository of AI models that can be accessed and utilized by users for various inference tasks.

This chapter will detail the architecture of the AIT, its role in our decentralized inference system, and the mechanisms through which users and developers can interact with it to deploy and execute AI models. Before diving into the details of the AIT, we first present a general architecture of the whole Nesa system in Figure 2.1, where the AIT serves as a critical component.

2.1 AIT Architecture and Design

The Artificial Intelligence Terminal (AIT) serves as the bedrock of our decentralized AI platform, providing a standardized and secure execution environment analogous to the Ethereum Virtual Machine's (EVM) role in Ethereum. However, unlike the EVM, which is geared towards general-purpose smart contract execution, the AIT is specifically optimized for the complexities and nuances of AI model inference, both large and small.

At its core, the AIT is designed to execute AI models in a uniform manner, ensuring

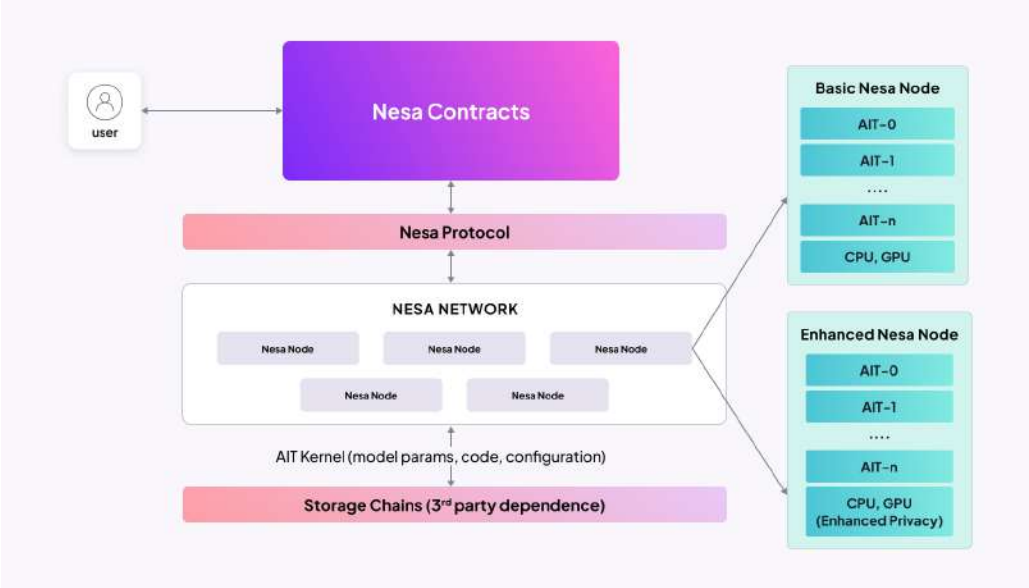

Figure 2.1: Nesa system architecture. End users interact with the Nesa smart contracts that coordinate AI task distribution and aggregation transparently. Interaction occurs through front-end applications and client dApps connected to Nesa by adapter, that are not shown in the figure. The tasks are distributed for computation across the Nesa network, where each node (miner) provides computational power (CPU and GPU), runs the AIT, and earns tokens in return for query inference. Stronger privacy and security features can be enabled through advanced hardware and cryptography integration, based on Nesa query request presets or user designated preference. The AIT kernels are stored on chain for decentralization and end-to-end model updates.

that regardless of the underlying hardware or software of the individual nodes, the output remains consistent. This is critical to achieving consensus within the network, as even minute discrepancies in model execution can lead to divergent results, thereby undermining the veracity and reliability of the entire system, and wasting time, resources, and gas fees in the process.

Each AI model in the AIT ecosystem comprises four integral components:

- **Model Parameters**: These are the weights and biases that define the AI model. They are the product of the training process and dictate the model's behavior and capabilities.
- **AIT Configuration File**: Functionally similar to a Dockerfile but specific to the AIT, this file contains the specifications for the virtual environment in which the AI model will execute. It details the dependencies, libraries, and runtime needed

to run the model, ensuring that every node sets up an execution environment with identical configurations.

- **Inference Code**: The code that runs the AI model. This includes the logic for processing inputs and generating predictions or outputs. The inference code also comes with necessary compilation information to ensure it can be seamlessly executed within the AIT.
- **Aggregation Code**: This piece of scripting code determines how the decentralized VM will aggregate and reach consensus from results returned from different nodes.

These components together form what we call the *AIT kernel*. The kernel encapsulates all of the necessary building blocks for a node to download and correctly execute an AI model. To facilitate transparency and repeatability, the AIT kernel is stored on the blockchain, providing an immutable and verifiable record of the model's execution environment and logic. In addition to the kernel, a fully operational AIT also contains various protocols that support decentralization and security. These protocols will be discussed in later chapters. The full architecture is depicted in Figure 2.2.

The AIT is architected with a highly simplified user interface intended to abstract away the complexities of the AI execution process. Nodes operating within the network can easily download an AIT kernel and run it locally. The interface is designed to accept user inputs, such as prompts for language models or data for predictive analytics, and then carry out the inference task, returning the results to the user or writing them back to the blockchain, as required by the application.

Key design features of the AIT include:

- **Isolation**: The AIT ensures that the execution of AI models is isolated from the host environment of the node, preventing any external factors from affecting the inference process.
- **Reproducibility**: The comprehensive specification of the AIT's environment ensures that models can be executed reliably and with the same results across different nodes.
- **Security**: The AIT is powered by a proprietary hybrid cryptographic security protocol built by Nesa, to be detailed in later sections, that minimizes the risk of malicious code execution and safeguards the integrity of inference tasks.
- **Agility**: While the AIT aims to provide a full-fledged execution environment, it is designed to be light-weight and lean so as not to impose significant overhead on the node's resources.

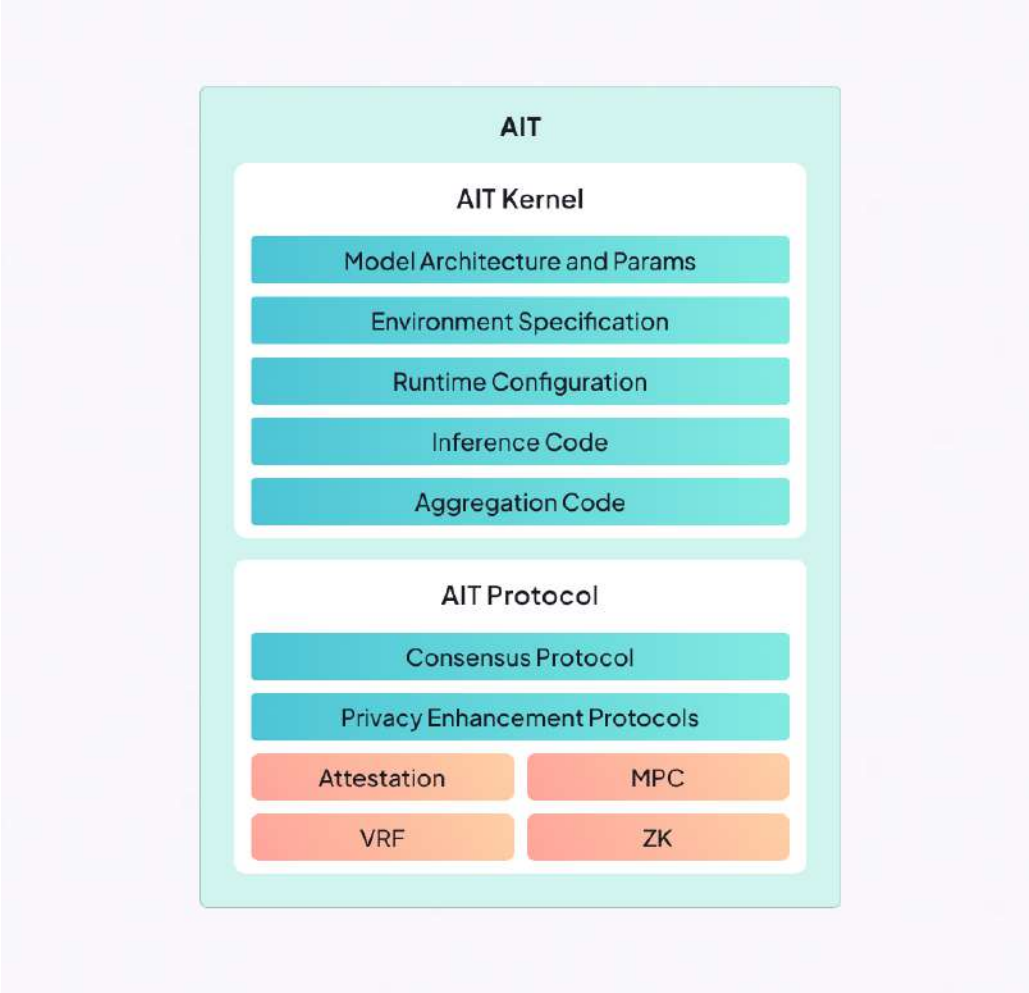

Figure 2.2: AIT Architecture. The AIT contains two core components: the AIT Kernel and the AIT Protocol. The kernel is stored on chain. Its primary purpose is ensuring that the query inference is carried out in a consistent way in the committee of nodes. The AIT Kernel contains specific information related to model inference, including model architecture and parameters, environment specifications, runtime configuration, and inference and aggregation code. Together this makes up the AIT Configuration File, a file functionally similar to a Dockerfile that details the dependencies, libraries, and runtime needed to run the model to ensure identical execution environment. The AIT Protocol's primary purpose is to facilitate communication among the nodes (mostly for security and privacy guarantee) and is packaged in software provided by us. The AIT Protocol includes the consensus protocol and privacy enhancement protocols, including Nesa's security mechanisms.

The AIT architecture lays the foundation for a network where diverse participants can collaborate and contribute to AI tasks with confidence in the consistency and reliability of the results. With the AIT, we ensure that our platform remains open, secure, and accessible, fostering a thriving ecosystem of shared AI capabilities.

2.2 Model Consistency and Inference Reliability

The AIT's primary purpose is to guarantee that AI models execute consistently across various nodes in the network. Model consistency and inference reliability are the linchpins of our platform, ensuring that every node produces identical results given the same inputs and model parameters. This section delves into the measures and specifications that we have put in place within the AIT to uphold these principles.

- **Configuration Specificity**: To achieve uniform model execution, the AIT configuration must encompass every aspect that could influence the computational outcome. This includes specifying the operating system and compiler versions, along with precise compilation options and flags. By rigorously defining the execution environment, we eliminate variability that could otherwise arise from different software stacks.
- **Hardware Specifications**: If a model demands particular hardware characteristics, such as GPU acceleration or specialized processing units like TPUs, these requirements are explicitly stated in the AIT configurations. Moreover, features provided by the hardware that could potentially lead to inconsistent execution, such as non-deterministic hardware instructions, are either strictly enabled or disabled as appropriate. This approach ensures that all participating nodes can adequately prepare and align their computational capabilities with the model's needs.
- **Randomness in Inference**: Many AI models introduce randomness during inference, which can pose a challenge for achieving deterministic and reproducible results. To mitigate this, we have implemented the strategy of fixing the random seed, which ensures that any pseudo-random number generation during inference leads to the same sequence of numbers across all executions. In scenarios where public randomness is necessary, we integrate the cryptographic method of Verifiable Random Functions (VRFs) that produce randomness that is both unpredictable and provably unbiased. This use of VRF in our system not only lends credibility to the random number generation process but also makes it possible to verify the randomness after the fact.
- **AIT Execution Protocol**: The execution protocol within the AIT pre- scribes a series of steps that every node must follow. This protocol includes initialization

procedures, data input conventions, model execution, and output handling. By standardizing the execution flow, we can reliably predict and replicate the behavior of AI models across the network.

- **Validation and Testing**: Before an AIT kernel is approved and stored on the blockchain, it undergoes rigorous validation to ensure compliance with the specified configuration and to confirm that it yields consistent results across diverse environments. A suite of tests is run in simulated multi-node scenarios to affirm that the kernel's execution is deterministic and immune to variances in the underlying systems.

The measures above coalesce to create a robust framework for model consistency and inference reliability within the AIT. These provisions are critical for maintaining the integrity of our decentralized inference system, guaranteeing that any node, regardless of its individual hardware or software configurations, can reliably participate in the network and contribute to collective AI tasks. With this standard of uniformity, we enable a diverse ecosystem of nodes to work together seamlessly and trustlessly.

2.3 On-Chain Model and AIT Repository

The on-chain model and AIT repository represent the decentralized storage and management system for AI models and their associated virtual machine configurations on Nesa. This repository acts as a global library, enabling users to access and deploy pre-existing AI models and developers to contribute new models and AITs. In this section, we outline how Nesa leverages decentralized storage technologies and encryption to maintain a secure, transparent, and persistent repository of AI models and execution environments.

- **Decentralized Storage Solutions**: To facilitate the storage of AIT kernels —comprising model parameters, configuration files, and inference code—we utilize the InterPlanetary File System (IPFS) and Arweave. IPFS offers a peer-to-peer network for storing and sharing data in a distributed file system, ensuring that AIT kernels are accessible across the network without relying on centralized servers. Arweave provides a platform for permanent storage with its blockweave technology, ensuring that once an AIT kernel is uploaded, it remains available indefinitely. By combining these decentralized storage solutions, we achieve resilience against data loss and censorship, as well as enhanced accessibility for model deployment.
- **Integration with the Blockchain**: The blockchain maintains a registry of the AIT kernels, storing metadata such as model descriptions, versioning information, and pointers to the actual data on decentralized storage. This registry

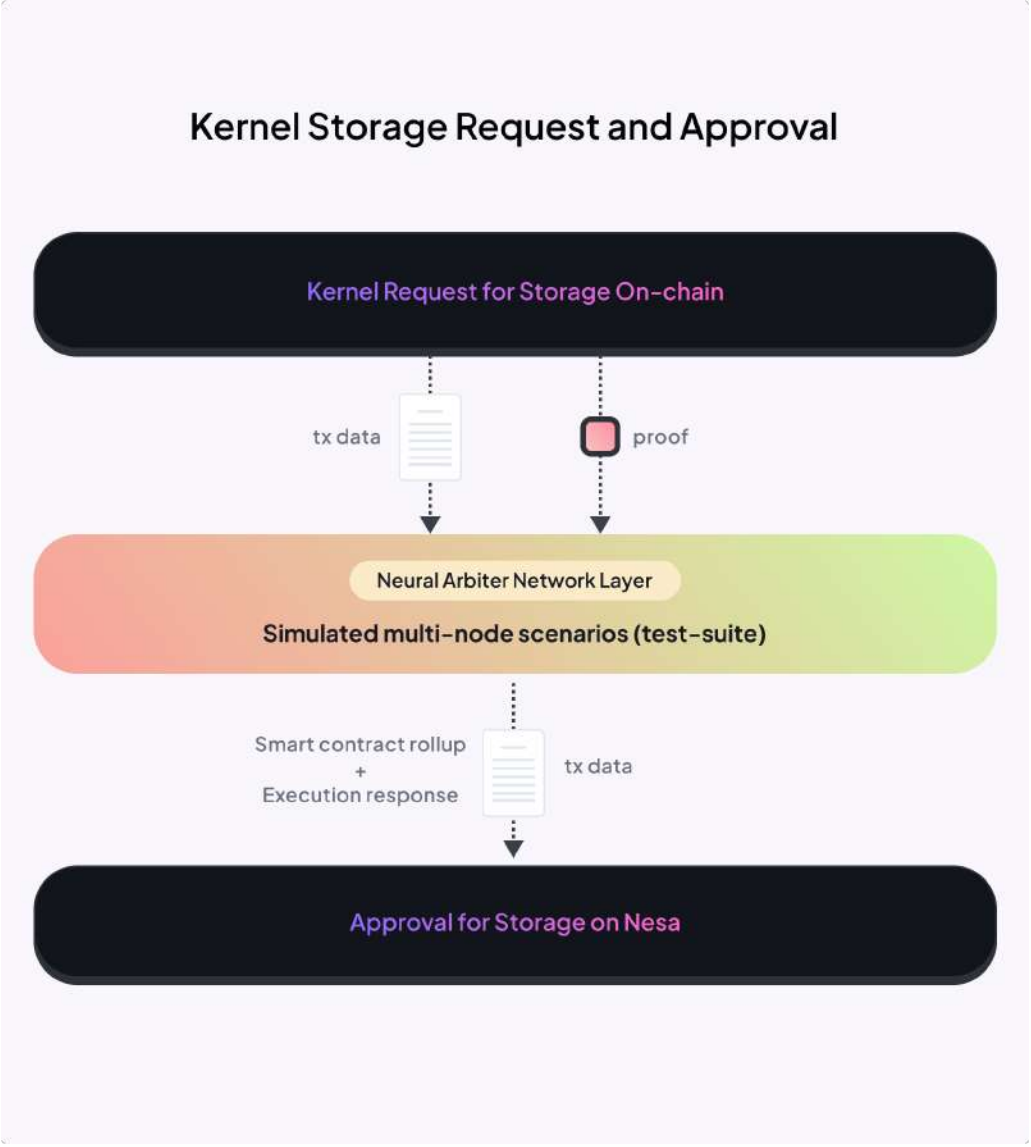

Figure 2.3: AIT kernel storage request and approval. Before an AIT kernel is approved and stored on the blockchain, the kernel undergoes rigorous validation to ensure compliance with the specified configuration and to confirm that it yields consistent results across diverse environments. A suite of tests is run in simulated multi-node scenarios to affirm that the kernel's execution is deterministic and immune to variances in the underlying systems. Nesa utilizes an AI jury system called NANs to assess the reliability of execution, which by user preference can focus on performance, ethical alignment, data consistency, hallucination levels, and/or congruence with predefined model behaviors by rapidly generating simulated environments where new updates are tested against varied and unpredictable conditions through their unique adversarial evaluation framework.

allows for the discovery and verification of models, as Nesa's immutability and transparency provide a reliable source of truth for the available kernels.

- **Privacy for Proprietary Models**: For models that are private or proprietary, encryption is employed before uploading the kernel to the decentralized storage. The model owner is responsible for managing the encryption keys and can distribute them to authorized parties who have the necessary permissions to execute the model. This ensures that private models remain confidential and secure, while still benefiting from the decentralized infrastructure of the platform.
- **Encryption and Key Distribution**: When a private model is uploaded, the model owner uses robust encryption algorithms to secure the data. The corresponding decryption keys are not stored on the blockchain or in the decentralized storage. Instead, the model owner maintains control over the key distribution, which can be facilitated through secure channels, ensuring that only authorized nodes can execute the private model.
- **Updating and Versioning**: The repository supports updates and version control for AIT kernels. When a model is updated, the new version is uploaded to decentralized storage, and the blockchain registry is updated to point to the latest version. This versioning system allows users to access both historical and current versions of models, fostering model evolution while preserving the lineage of development.
- **Verification and Quality Assurance**: Before a model is added to the on-chain repository, it undergoes a verification process to ensure compatibility with the AIT execution standards. This process includes checks for correctness, consistency, and compliance with the platform's security protocols. By enforcing a rigorous verification process, we maintain high-quality standards for the models available in the repository.

Through the on-chain model and AIT repository, we provide a stable and scalable infrastructure for the storage and deployment of AI models. This infrastructure results in persistent model security, consistency and accessibility to all users and contributors, whether they are engaging with public models or managing proprietary ones.

2.4 AIT Interface for Model Interaction

The AIT frontend interface serves as the gateway for users to interact with the on-chain model and AIT repository, facilitating the uploading, managing, and monitoring of AI models and their corresponding virtual machines. The frontend interface caters to both novice users with Nesa's Minimal Code Suite which offers presets for smaller or non-proprietary models, as well as complete customization ability for experienced

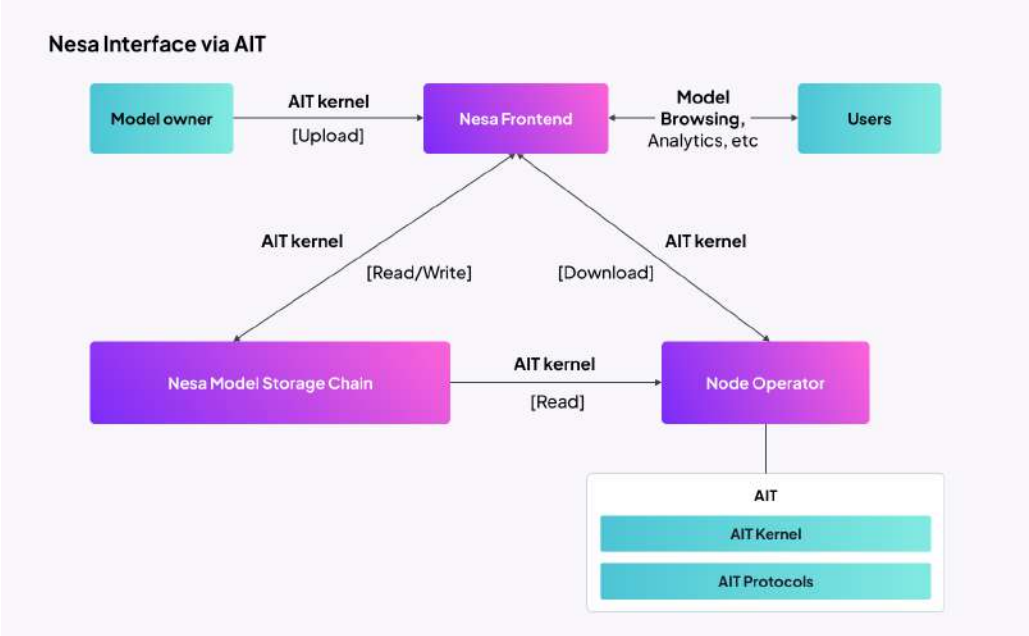

Figure 2.4: Nesa inference and management via AIT. AI Model owners can upload and manage AIT kernels by using the Nesa frontend. The kernels are stored on a dedicated decentralized storage chain connected to the Nesa network. Node operators can either download kernels from the Nesa frontend or directly from the chain. AI dApp users can browse the frontend to obtain rich information about various models supported by our community.

developers. In this section, we outline the features and functionalities of the interface that facilitate user communication with Nesa. The overall flow of our system for managing the AIT is shown in Figure 2.4.

Model Browsing and Discovery. Users explore the repository through the frontend interface, which provides a catalog of available AIT kernels with rich associated model information. The interface supports granular search and filtering to find models that suit a user's specific needs based on criteria such as model type, complexity, and performance metrics.

Detailed Model Information. The interface presents detailed metadata for each AI model uploaded to Nesa's AIT repository, including authorship, version history, and performance benchmarks, that can be reviewed when selecting a model to deploy for inference.

Model Upload and Registration. For users and developers looking to contribute

new models to the platform, the interface provides a guided upload process. This includes the submission of model parameters, the AIT configuration file, and the inference code, along with any necessary documentation. The upload and registration process ensures that all models on Nesa adhere to the required standards for AIT compatibility.

Model Deployment and Execution. The interface facilitates turnkey deployment of AI models from repository selection to inference task initiation. Users can input their data, configure execution parameters, and submit their request directly through the interface. Nesa handles the execution of the model within the AIT and returns the results to the user.

Security and Privacy Controls. When dealing with private or proprietary models, the frontend interface provides tools for managing encryption keys and access controls. Users can specify which nodes are authorized to execute their models and distribute decryption keys through secure means.

Real-time Monitoring and Analytics. The interface provides real-time monitoring of model performance and usage statistics, letting users track the activity of their deployed models, analyze usage patterns, and gain insights into the performance of their inference tasks.

The AIT interface democratizes access to AI by allowing users from diverse backgrounds to participate in the sharing, creation, deployment, management, and collective evaluation of AI models. In the future this will become a full-fledged AI model marketplace on-chain, complete with version control, reviews, and lookbacks for model performance and example output, at the discretion of the model creator and model's users.

The AIT has been engineered to cater to the full spectrum of developer skillsets, from beginner developers who are seeking default presets that allow them to upload their model and then plug and play, through to advanced developers who seek complete customization ability of the end-to-end system, from control over their kernel configuration, to the type of inference, security, consensus, and output that they seek.

Nesa's future expansion plans for AIT include further interoperability of the system with other blockchain networks to drive universal adoption to the platform, and a roadmap of new modular features that will give creators and users more control in how they build, present, and implement AIT kernels. For a detailed look at these plans, please see this whitepaper's Addendum.

Chapter 3

Decentralized Inference

Nesa's decentralized inference process is the cornerstone of our autonomous AI oracle network, enabling the first trustless environment where AI computations are performed transparently and reliably reported on-chain. This section outlines the intricacies of Nesa's decentralized inference framework, which is composed of several core components: users who submit inference requests, chain contracts responsible for verification and aggregation of results, and nodes that process these requests. This framework leverages a two-phase transaction structure, utilizing a commit-reveal paradigm, to safeguard against dishonest behavior and free-riding. This ensures that nodes are incentivized to perform their computations honestly and that users can trust the integrity of the inference results. The system maintains a decentralized approach by employing smart contracts for the key processes of verification and aggregation, allowing for a scalable network that harnesses the collective computational power of its participants. We give a short summary of the key components in Figure 3.1.

3.1 The Two-Phase Transaction

Nesa's inference technique employs a unique two-phase transaction mechanism that decouples the inference request from inference response to streamline the network's inference process and enhance system scalability.

Nesa has been engineered with this design because a major challenge of executing AI inference on the blockchain comes from the computational intensity of large model inference which can severely impact transaction throughput and block generation times. Traditional approaches by blockchain projects, such as the method adopted by the Cortex project, involve integrating an INFERENCE command directly into the blockchain protocol. While this approach has the merit of simplicity, it is not without its drawbacks. Specifically, it is characterized by slow performance when dealing with complex models, which subsequently leads to a bottleneck effect on the chain's overall throughput.

Instead, Nesa takes a different, bifurcated approach.

The First Phase: **Inference Request Queueing**. In the first phase, a user submits an inference request transaction, which includes the necessary details for the inference task but does not trigger the execution immediately. Instead, this transaction is registered into a queue within the blockchain ecosystem. Our system utilizes a priority queue to order these requests. The priority for each request is directly correlated with the fee paid by the user; higher fees result in higher priority, ensuring that users with urgent needs can opt for faster processing by electing to pay a premium. This dynamic

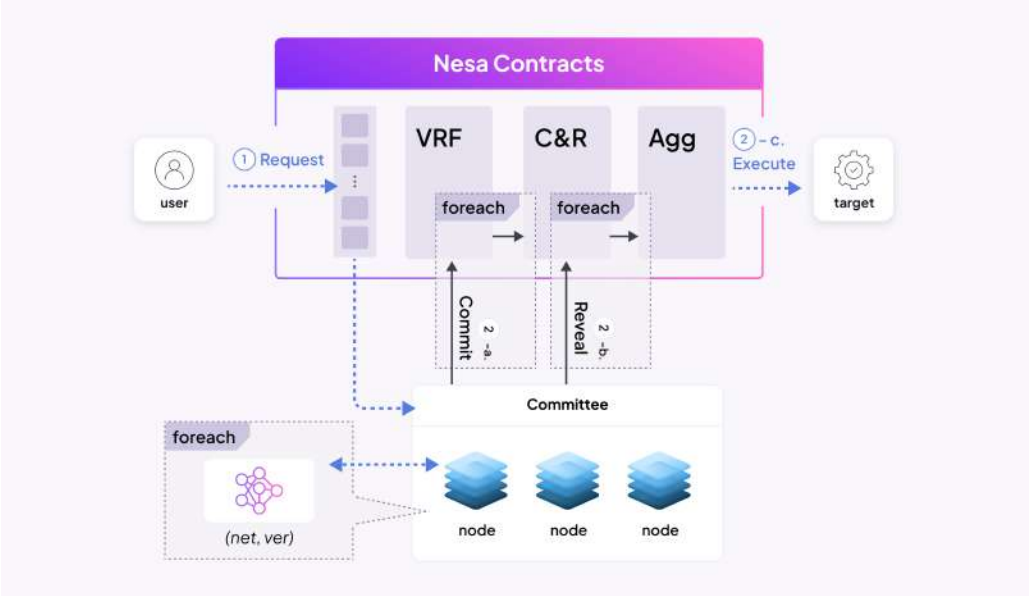

Figure 3.1: High-level view of our decentralized inference. Nesa's inference procedure is separated into two phases: a request phase, and an inference and response phase. The separation achieves higher transaction throughput than a traditional processing method where a single lengthy transaction is generated for the full procedure. The second phase is further deconstructed with a commit and reveal mechanism to prevent free riding.

pricing model aligns resource allocation with market demand, thereby optimizing system efficiency. Requests are registered and enqueued within smart contracts, which act as decentralized and transparent priority queue managers. These smart contracts are programmed to organize the requests according to their assigned priority, ensuring that the system's resources are allocated in a fair and economically rational manner. This non-blocking transaction allows the blockchain to continue processing other transactions, maintaining high throughput and low latency.

The Second Phase: **Inference Execution and Response**. The second phase is initiated once the inference request reaches the front of the queue. Separate transactions are created by the designated inference committee, which is tasked with actually performing the inference task. Upon completion, the results are recorded and disclosed. This phase is conducted mostly off-chain to prevent the computational load from affecting the blockchain's performance.

The separation of request and execution transactions in Nesa's design offers several advantages. Firstly, it avoids the congestion that can occur when the blockchain waits for computationally intensive inferences to complete. Secondly, it provides flexibility

in resource allocation, as the inference task can be processed in parallel with other blockchain operations. Finally, it ensures that the blockchain maintains a consistent and fast block generation time, regardless of the complexity or size of the AI models being inferred.

Through two-phases of transaction, Bifurcated Inference Ledgering ensures that Nesa can scale as more requests enter this system. This design forms the backbone of our decentralized inference system.

3.2 Robust Inference Committee Selection

A fair and secure method for selecting the inference committee is critical to maintaining the integrity and trustworthiness of our decentralized inference system. To facilitate this, we incorporate VRF as the means for random yet deterministic committee selection, drawing inspiration from the organizational structure of duties within Ethereum 2.0 (ETH2).

The Role of VRF in Committee Selection. On Nesa, each computational node is assigned a public-private key pair (pk, sk). These cryptographic keys serve as the node's identity and secure its participation in the network's inference processes. The nodes are responsible for executing AI tasks, and their selection is organized using a framework of slots and epochs, similar to how validators' attestation duties are arranged in ETH2. At the onset of each slot, the highest-priority inference request in the queue is identified. Each node then utilizes its private key (sk) to invoke the VRF's evaluate function, generating a random number unique to that node but consistent across evaluations. This random number is used to determine the node's eligibility to be part of the inference committee for that particular slot.

Transparent and Verifiable Committee Formation. Thanks to the cryptographic guarantees of the VRF, the selection process is both unpredictable to prevent gaming the system and independently verifiable to ensure transparency. Every node in the network, alongside the governing smart contracts, is able to verify the validity of the VRF proof. This serves as evidence that a node has been legitimately chosen to serve on the inference committee. In the event that a node attempts to submit an inference result without a valid VRF proof or if it was not selected for the committee, the system is designed to enforce penalties akin to slashing in ETH2. This punitive measure serves as a deterrent against malicious behavior and safeguards the system's integrity by ensuring that only legitimately selected committee members can contribute to the inference process and be rewarded for their work.

Nesa has designed this VRF-based selection mechanism to instill confidence in the fairness and security of the committee formation process while upholding the

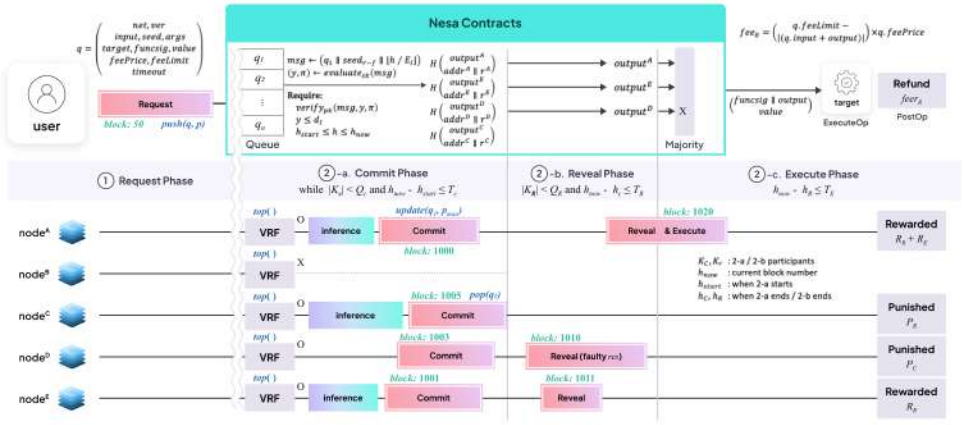

Figure 3.2: Step-by-step details of the decentralized inference process. A user sends an inference request in phase 1. When the request emerges from the priority queue, phase 2 begins. The results are then committed in phase 2-a through an inference committee. In phase 2-b, the original results are revealed. If a quorum is reached, the final result is obtained through a majority in phase 2-c and the requested operand is executed. The reward payment and punishment are made in the POSTOP step. In addition, the contract refunds the remaining fee to the user.

decentralized nature of our network. By integrating such a robust selection protocol, we ensure that our system is resilient against manipulation and that the responsibilities of AI computation are distributed across a diverse set of participants, enhancing overall reliability and credibility of the inference outcomes.

3.3 Free-Riding Prevention

A potential vulnerability in any decentralized system is the risk of free-riding, where one party seeks to gain the outcomes of a shared effort without contributing its fair share of work. In the context of our decentralized inference system, this would manifest as a node waiting for others to submit inference results, then copying and submitting those results as their own, thereby conserving their computational resources illegitimately.

To mitigate this risk, we have designed a unique security mechanism that divides the second transaction phase into two distinct sub-phases, each with its own transaction: the Commit phase and the Reveal phase.

The Commit Phase

In the Commit phase, nodes in the inference committee must submit a cryptographic commitment of their inference results. This commitment is a one-way hash of the result combined with a secret nonce, providing a way to lock in the result without exposing its content, i.e., $H(m||r)$, where H is the hash function, m is the result, and r is a random nonce. The commitment ensures that the node has performed the computation and is ready to reveal the valid result. This phase has a specified timeout limit within a fraction of the slot duration. Failure to submit a commitment within this window results in the node being ineligible to reveal its results and facing a penalty.

We list the detailed off-chain and on-chain algorithms for the commit phase in Figure 3.3.

The Reveal Phase

Following the Commit phase, the Reveal phase allows nodes to disclose their previously committed results. Each node must reveal both the result m and the nonce r used in the commitment, allowing others to verify the hash against the commitment made in the previous phase. This phase also has a timeout limit, and failure to reveal on time, or revealing a different result from what was committed, will result in punitive measures against the offending node.

The introduction of the Commit/Reveal scheme effectively prevents free-riding by ensuring that each node provides evidence of its contribution before any results are made public. This two-step process requires nodes to stake their claim on an outcome without knowledge of other nodes' work, thus ensuring that each node contributes independently. By employing a cryptographic commitment scheme, we create a trustless environment where work cannot be copied, and honesty is enforced through the threat of penalties.

These penalties for non-compliance act as a strong deterrent against malicious activity, helping to maintain a fair and secure environment for all participants. The Commit/Reveal protocol guarantees that only nodes that genuinely perform the computations are rewarded, thereby upholding the integrity of the decentralized inference process and protecting the system from exploitation. This careful orchestration of transactions and timings ensures that our network remains resilient, efficient, and free from the inefficiencies and unfair practices that free-riding would introduce.

We list the detailed off-chain and on-chain algorithms for the reveal phase in Figure 3.4.

3.4 Aggregation of Inference Results

The final step in our decentralized inference system is the aggregation of results submitted by individual nodes in the inference committee. This crucial step determines the official outcome of the inference task. To ensure the integrity and accuracy of the aggregated result, our system implements a default majority vote strategy within a smart contract. We also support customized aggregation outside smart contract, which will be described later in this section.

Here is how the default aggregation strategy works:

- Majority Vote within Smart Contract: A smart contract is tasked with aggregating the results. It tallies the submissions and identifies the majority result, meaning the outcome that has been reported by more than half of the nodes in the committee. This majority result is taken to be the correct and official outcome of the inference task, and it is this result that is ultimately communicated to the user or utilized for further on-chain actions.
- Fault Detection and Punishment: Any submitted result that does not align with the majority is flagged as faulty. The node responsible for a faulty result is subject to a penalty. This could manifest as a loss of a security deposit (slashing), a reduction in reputation score, or both. The specific nature and severity of the punishment are pre-defined in the smart contract's rules and are automatically enforced. The penalization protocol serves as a deterrent against submitting incorrect or dishonest results, thus motivating nodes to perform their computations diligently and accurately.
- Finalization and Reward Distribution: Once the official result is determined and faulty nodes are penalized, the smart contract finalizes the result and triggers the appropriate reward distribution to the nodes that contributed to the majority result. Rewards are allocated as per the predefined incentive structure, balancing the costs incurred by nodes and incentivizing continued honest participation in the system.

This majority vote strategy ensures that the aggregated result reflects the consensus of the committee, thereby reducing the likelihood of erroneous outcomes due to individual node failures or malicious behavior. It also reinforces the reliability and trustworthiness of the decentralized inference system, as all nodes are accountable for their contributions and the overall process is transparent and verifiable. The smart contract serves as an impartial and incorruptible arbiter that enforces the rules of the aggregation protocol, ensuring that the system remains fair and resilient.

Customized Aggregation

Nesa is designed to offer flexibility in how inference results are aggregated to produce a final output. While the majority vote strategy serves as a robust default aggregation method within the smart contract, certain use cases may demand more tailored approaches. To cater to these needs, we introduce a new concept called "Customized Aggregation" where model owners can specify their own aggregation logic that goes beyond the capabilities of a smart contract. This subsection outlines the components and considerations of the customized aggregation process.

Inference Results Set Retrieval. After all participating nodes have revealed their inference outputs and the smart contract has recorded these revelations, the contract will now provide the raw set of inference outputs without directly applying any aggregation strategy. This raw result set will be available for further processing as defined by the model owner's custom aggregation logic.

Custom Aggregation as Part of AIT. The custom aggregation code is authored by the model owner and is an integral part of the associated AIT. This code is stored on the blockchain alongside the model parameters and the AIT configuration file, ensuring that the entire inference and aggregation process is verifiable and transparent. The model owner can then tailor the aggregation process to the unique requirements of the model and the inference objectives.

Diverse Aggregation Methods. Customized aggregation strategies can range from simple methods like majority voting or averaging to more complex techniques such as averaging with outlier removal. The choice of aggregation method is determined by factors such as the nature of the model, the desired robustness against aberrant results, and the level of consensus needed among nodes.

Result Verification. Ensuring the integrity of the inference results and identifying nodes that may have submitted outputs without performing the actual computation is critical to ensuring the integrity of the inference results. The AIT addresses this by implementing various verification methods: a) Outlier Identification: This involves statistical analysis within the aggregation code to detect and handle outputs that deviate significantly from the consensus or expected range of results. b) Publicly Checkable Proof: Similar to approaches used in academic works like TownCrier, the model owner can require nodes to produce zero-knowledge proofs or leverage trusted hardware attestations to publicly verify the correctness of their computations. c) Outlier Quota and Proof Submission: Each node could be assigned an outlier quota, limiting the number of times it can deviate from the consensus results before it is required to submit a proof of computation. If a node exceeds its quota, it must provide such proof to maintain its standing in the network and avoid penalties.

With these customized aggregation strategies, the AIT provides model owners with a rare level of flexibility to define how results are synthesized to best suit their use cases.

3.5 Step-by-Step Process of Decentralized Inference

The following outlines the decentralized inference process (shown in Figure 3.2), taking into account the techniques and protocols established within the previous sections:

Step 1: User Request Submission

Users submit inference requests to the blockchain. Each request includes the user's input data for the AI model and the fee paid by the user, which determines the priority of the request in the system.

Step 2: Priority Queue Management

Requests are registered in a smart contract that acts as a priority queue manager. The smart contract organizes the requests based on the respective fees, ensuring those who paid higher fees receive higher priority in the execution queue.

Step 3: Inference Committee Selection

At the beginning of each slot, a committee of nodes is selected to perform the inference task. The selection is based on the output of a VRF that each node executes using its private key. Nodes generating qualifying random numbers are chosen for the committee.

Step 4: Inference Execution

The selected committee of nodes independently performs the requested AI inference task. Each node computes the result using the input data provided in the user's request.

Step 5: Commitment to Inference Results

To prevent free-riding, each node in the committee commits to its computed inference result using a cryptographic commitment scheme. This involves sending a hash of the result and a nonce to the blockchain within a specific timeframe.

Step 6: Reveal of Inference Results

Within a subsequent timeframe, each node reveals its committed inference result by submitting both the result and the nonce to the blockchain. If a node fails to reveal its result, submits after the deadline, or reveals a result that does not match its commitment,

it faces a penalty.

Step 7: Result Aggregation and Majority Vote

Submitted inference results are collected by a smart contract that performs a majority vote to determine the consensus result. The result most frequently reported by the committee is considered the correct outcome.

Step 8: Penalty Enforcement and Reward Distribution

The smart contract identifies and penalizes nodes that submitted incorrect results (not aligning with the majority). Penalties may include slashing of stakes or reduction in reputation. Nodes that contributed to the majority result receive rewards, as specified by the system's incentive structure, for their accurate and honest work.

Step 9: Finalization and User Notification

The smart contract finalizes the consensus result of the inference task and records it on the blockchain. The user is notified of the result, which concludes the inference process.

This step-by-step summary encapsulates the core sequence of operations within Nesa's decentralized inference system. It details the journey from request submission to the final delivery of the consensus inference result, highlighting the roles of smart contracts, priority queues, committee selection, result commitment and reveal, and aggregation via majority vote, all without any specific privacy-preserving or cryptographic enhancements from Chapter 4. The system ensures a transparent, fair, and secure process, motivating nodes to produce accurate results and maintaining reliability across the network.

Nesa's default aggregation system by smart contract epitomizes an equilibrium between democratic principles and technological enforcement, allowing us to mitigate risks associated with node aberrance or malicious intents. By provision of penalties and reputation diminution for deviant or deceitful submissions, we are engendering a self-regulating ecosystem that encourages computational precision and integrity. The process maintains detailed control over results set retrieval, an array of sophisticated statistical methods for conclusive consensus, and reward allocation commensurate with the predefined incentives ensures adversarial resilience and motivation for continued honest engagement by Nesa miners.

[Off-Chain]
Require: $top() \neq null$
procedure NodeCommit(Quorum_C, $\mathsf{Timeout}_C$, d_I)

$q_1 \leftarrow top()$
for block height $h \leftarrow h_{start}, \cdots$ **do**
  for $k \in \{nodes\} - K_C$ in parallel **do**
    $msg \leftarrow (q_1 || seed_{r-f} || \lfloor h/\mathsf{Epoch}_I \rfloor)$
    $(y, \pi) \leftarrow \mathsf{evaluate}_{sk}(msg)$
    if $y \leq d_I$ **then**
      $output^k \leftarrow$ Inference(q_1)
      $H \leftarrow H(output^k || addr^k || r^k)$
      Commit$^k_{q_1}(y, \pi, h, H)$
    end if
  end for
  if $|K_C| \geq \mathsf{Quorum}_C$ or $h - h_{start} > \mathsf{Timeout}_C$ **then**
    break
  end if
end for

[On-Chain (Contract)]
Require: $\mathsf{verify}_{pk}(msg, y, \pi), y \leq d_I, h_{start} \leq h \leq h_{now}$
transaction Commit$^k_q(y, \pi, h, H)$

Store H^k_q to Nesa contract; $K_C \leftarrow K_C \cup \{k\}$
if $|K_C| == 1$ **then**
  $update(q, p_{max})$
end if
if $|K_C| \geq \mathsf{Quorum}_C$ **then**
  $pop(q)$
  $H_C \leftarrow h_{now}$
  $(seed_r, \pi) \leftarrow \mathsf{evaluate}_{sk}(seed_{r-1} || r)$
else if $h_{now} - h_{start} > \mathsf{Timeout}_C$ **then**
  $pop(q)$
  $K_C \leftarrow \emptyset$
  $seed_r \leftarrow H(seed_{r-1} || r)$
  Refund $(q.feeLimit - |q.input|) \times q.feePrice$
  Refund $q.value$ ETH
end if

Figure 3.3: Off-chain and on-chain algorithms for the Commit phase. The Commit phase is stage one of the two-part subphase for Nesa's commit-reveal paradigm within the system's decentralized inference protocol.

[Off-Chain]
Require: $K_C \neq \emptyset$
procedure NodeReveal$_q$(Quorum$_R$, Timeout$_R$, τ)

for block height $h \leftarrow h_C, \cdots$ **do**
  for $k \in K_C - K_R$ in parallel **do**
    Reveal$_q^k(output^k, addr^k, r^k)$
  end for
  if $|K_R| \geq$ Quorum$_R$ or $h - h_C >$ Timeout$_R$ **then**
    break
  end if
end for

[On-Chain (Contract)]
Require: $H(output||addr||r) == H_q^k, addr == addr^k$
transaction Reveal$_q^k(output, addr, r)$

Store $output$ to Nesa contract; $K_R \leftarrow K_R \cup \{k\}$
if $|K_R| \geq$ Quorum$_R$ **then**
  call Execute
else if $h_{now} - h_C >$ Timeout$_R$ **then**
  if $\tau == \tau_I$ **then**
    $h_R \leftarrow h_{now}$
  else
    $push(q, p_{max} - 1)$; $K_C \leftarrow \emptyset$; $K_R \leftarrow \emptyset$
  end if
end if

Figure 3.4: Off-chain and on-chain algorithms for the Reveal phase. The Reveal phase is stage two of the two-part subphase for Nesa's commit-reveal paradigm within the system's decentralized inference protocol. The Reveal phase allows nodes to disclose their previously committed results. Each node must reveal both the result m and the nonce r used in the commitment, allowing others to verify the hash against the commitment made in the previous phase. This phase also has a timeout limit, and failure to reveal on time, or revealing a different result from what was committed, will result in punitive measures against the offending node. After the reveal, submitted inference results are collected by a smart contract that performs a majority vote to determine the consensus result. The result most frequently reported by the committee is considered the correct outcome.

Chapter 4

Hybrid Design for Enhanced Privacy

This chapter introduces Nesa's cutting-edge hybrid approach to privacy enhancement. The essence of this hybrid design lies in the thoughtful integration of hardware-based and cryptographic-based solutions, each selected and optimized for varying scenarios within our ecosystem.

The hybrid-privacy methodology on Nesa is grounded in the recognition that privacy concerns manifest in different forms—users may wish to conceal their input data or the results of their inferences, while node owners might seek to protect the confidentiality of their model parameters. Our hybrid design acknowledges the unique requirements of these use cases by deploying the most appropriate privacy-preserving technologies. Through the synergy of the robust, hardware-centric protections of Trusted Execution Environments (TEEs) and the advanced cryptographic techniques of Zero-Knowledge Proofs (ZKPs) and Secure Multi-Party Computation (SMPC), we ensure that privacy is a foundational pillar of the system. This chapter elucidates the rationale behind Nesa's hybrid strategy, offering a comprehensive blueprint for achieving the highest standards of privacy using Split-Flow while maintaining the usability and efficiency of the decentralized inference process.

4.1 Challenges in Achieving Confidentiality and Verifiability

In engineering a privacy-centric decentralized inference architecture, the primary challenge lies in reconciling the twin imperatives of data confidentiality and computational verifiability. These concurrent objectives, each indispensable, represent an intrinsic paradox that proves difficult to harmonize. While conventional encryption techniques are adept at bolstering confidentiality, they incapacitate the data that they encrypt for substantive analytical processing. So in environments like Nesa's where the system's operational utility is contingent upon the facilitation of intricate data computations on private data, encryption doesn't work.

Confidentiality typically involves encrypting data to prevent unauthorized access. While encryption secures the data when static or during exchange, it also obscures the data from the very systems that need to process it. Standard computation on encrypted data is not feasible, as the process of encryption changes the data format and structure, making it indecipherable and inoperative for standard computation. This necessitates the development of the specialized techniques of Homomorphic Encryption (HE) and Secure Multi-Party Computation (SMPC), which are designed to enable computations on encrypted data without ever needing to decrypt it.

However, the introduction of such techniques to enable computation on encrypted

data engenders a second challenge: verifiability. Users and node owners inherently desire not just the confidentiality of their data or models but also the assurance that the computations performed are correct and trustworthy. Verifiability means providing a way to prove that the computation was executed as intended and that the results are accurate reflections of the computation on the expected data. Zero-Knowledge Proofs (ZKPs) can be employed to demonstrate the correctness of computations without revealing the underlying data. But integrating this into a system already grappling with encrypted data adds layers of complexity.

The intersection of these challenges—ensuring confidentiality while enabling computation and providing verifiability—requires a sophisticated balance of cryptographic innovation and system design. Any proposed solution must be sufficiently secure to guard against breaches and robust enough to withstand the scrutiny of verification without compromising on performance or scalability. As we proceed through this section, we will dissect these challenges in detail and explore Nesa's Split Flow Protocol that harnesses composite architectures to address the combined imperative of confidentiality and verifiability within the inference framework.

4.2 Split-Flow

Split-Flow is an orchestrator protocol that implements a dual-strategy to preserve confidentiality and ensure computation verifiability in Nesa's decentralized inference network. It constitutes an intelligent task-directed allocation system that leverages both hardware and cryptographic composites, whose employment is determined by an automated assessment mechanism based on input sensitivity, computational burden, model-oriented criteria, and consensus conformity parameters.

At its core, the Split-Flow Protocol operates by dissecting the workflow of any given computation into two principal streams: the confidentiality stream and the verifiability stream. These streams are aligned with security controls that respond dynamically to variable requisites of the data model and inference objectives, yielding an efficient, secure, and verifiable computation cycle.

The Confidentiality Stream

The confidentiality stream utilizes a dual-modality operation to tackle the problem that traditional static encryption poses to computation. Upon initial assessment, the model size and computational complexity dictate whether a TEE-hosted enhanced node or cryptographic constructs take precedence, and the extent of composite interplay.

The Verifiability Stream

To buttress the protocol's trust in computation outcomes, Split-Flow utilizes ZKPs to establish the correctness of computations without any requirement for data disclosure. It does so in a combinative nature, in concert with the confidentiality measures of both composites to ensure a system that maintains user privacy while concurrently producing attestable, accurate results.

Protocol Operation Workflow

Upon receiving an query request, the Split-Flow Protocol stratifies and directs the inference task through its security controls:

1. Automated Evaluation: The protocol firstly appraises task characteristics – sensitivity, model size, computational complexity, and desired consensus.

2. Confidentiality Routing: Based on the assessment, an optimal privacy approach is selected, engaging either hardware or cryptographic composite or a hybrid blend as warranted by the task.

3. Verifiability Assurance: Concurrently, ZKPs are orchestrated to align with the chosen confidentiality mode, ensuring that computations are demonstrably accurate without revealing sensitive data.

4. Result Synthesis: The multi-party computations are conducted within the confines of the confidentiality construct, producing aggregate inference data.

5. Validation: Results are aggregated and validated to ensure correctness and avert any adversarial influence, at which time they are rolled up for settlement

4.3 Composite 1: Hardware

To address the challenges of privacy and verifiability, our system leverages specialized hardware provisioned with enhanced security features for both CPU and GPU to fortify computation nodes. Some nodes in the Nesa network are designated as enhanced nodes equipped with TEEs, which offer a secure enclave for processing sensitive data. This setup is particularly adept at protecting the user's input from exposure while still enabling computation.

Complimentary to the hardware requirement, Nesa relies on a threshold cryptosystem, because the user submits a request that is publicly accessible, hence the request should contain encrypted information that can be decrypted (in secure memory) by a randomly selected committee. Basically, in a (t, n)-threshold cryptosystem, we have:

- PK: public key (publicly known by the world);

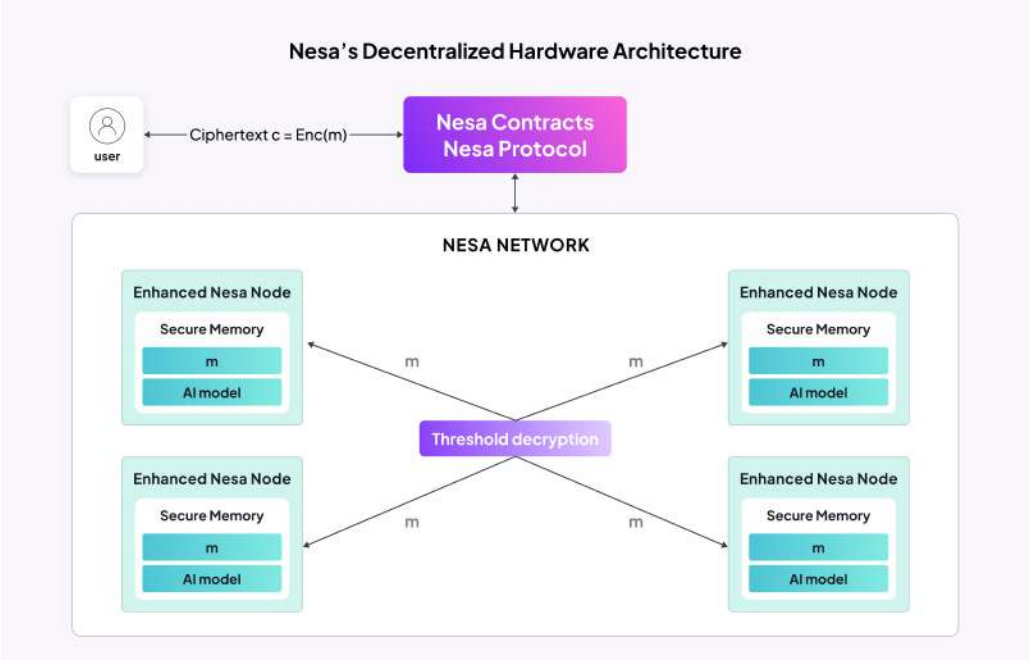

Figure 4.1: Nesa's Hardware Composite. Each participating node in the inference is equipped with advanced hardware with confidential computing capability. A user can submit ciphertext in the request transaction. The ciphertext can be decrypted only inside secure memory in the committee of nodes, through threshold decryption. The decryption reveals the plaintext in the secure memory, enabling each node to continue performing subsequent inference computation.

- SK: secret key (a virtual piece of information not known by any one);
- $SK_1, SK_2, \cdots, SK_n$: n secret shares of SK, each of which held by a party.
- $c \leftarrow Enc_{PK}(m)$: encryption of message m with public key;
- $m_i \leftarrow Dec_{SK_i}(c)$: partial decryption of ciphertext c by party i who holds the secret share SK_i.
- $m \leftarrow Comb(m_{i_1}, m_{i_2}, \cdots, m_{i_k})$: combining a list of k partially decrypted messages into the original message m, with $k >= t + 1$.

Here is a step-by-step breakdown of the protocol employed by our system that utilizes hardware-based solutions to ensure privacy-preserving computation (Figure 4.1):

1. Inference Request Preparation: The user begins by encrypting his input data m,

by

$$c = Enc_{PK}(m),$$

using a public key that is part of a public-secret key pair associated with the enhanced nodes. This key pair is specially designed so that the corresponding secret key is not held by any single node but is rather distributed amongst the computation nodes using secret sharing techniques. The user then creates an inference request transaction, embedding the encrypted input (ciphertext) rather than plaintext data. The user also specifies that this is an inference request with enhanced privacy so that only nodes with proper hardware will participate in the committee selection.

2. Transaction Submission: The inference request transaction, containing the encrypted input, is submitted to the blockchain and queued according to the priority system outlined in Chapter 3.

3. Committee Selection: Upon reaching the head of the queue, an inference committee S is selected using the VRF technique, ensuring a fair and random choice of enhanced nodes from among those equipped with TEEs.

4. Threshold Decryption: The selected committee nodes (with size $|S| > t$) retrieve the encrypted input from the transaction. Inside the secure enclaves provided by their TEEs, committee nodes collaborate to perform a threshold decryption operation. Basically, they perform

$$m_i = Dec_{SK_i}(c),$$

followed by a threshold combining

$$m = Comb(\{m_i\}_{i \in S}).$$

This process ensures that only a collective effort by the committee can reconstruct the secret key and decrypt the input, thus no single node has access to the plaintext data.

5. Inference Execution: Once decrypted, the plaintext input remains within the protected memory space of the TEEs, where the inference computation is securely executed. The TEE ensures that the computation process is isolated from the rest of the system, providing a safeguard against potential leaks or attacks.

6. Result Encryption and Submission: The output of the inference computation, still within the TEE, is encrypted with a public key provided by the user.

An encrypted inference result is then generated and submitted to the blockchain as a transaction. This result can only be decrypted by the user, maintaining the confidentiality of the data. This hardware-based solution provides strong security guarantees for user input by combining encryption, threshold decryption within TEEs, and secure computation. The protocol addresses the dual challenges of confidentiality and verifiability by ensuring that the sensitive data is never exposed in an unencrypted form outside of the secure enclaves and that the computation is performed within an environment that is resilient to tampering. By maintaining the secrecy of the data throughout the process and leveraging the trusted hardware, the system provides a robust framework for performing privacy-preserving computations on a decentralized network.

TEE Setup and Attestation Protocol

To ensure that enhanced nodes with TEEs are trustworthy and properly configured, we implement an attestation protocol similar to the decentralized trust mechanisms present in Chainlink's oracle network. This attestation process is critical as it not only provides the assurance that the TEE is genuine and secure but also serves as the initiation protocol for new nodes entering the system. The node admission procedure is shown in Figure 4.2 and discussed below:

1. Node Initialization: When a new computation node equipped with a TEE wishes to join the network, it must first perform a local attestation of its TEE. This step involves generating an attestation report that certifies the authenticity of the TEE and provides details about the environment's configuration. This is called a quote $\mathsf{Qte}(\mathsf{hw}, \mathsf{env})$.

2. Network Attestation: The new node then broadcasts its attestation report $\mathsf{Qte}(\mathsf{hw}, \mathsf{env})$ to the existing nodes in the network. A threshold number (larger than t) of established nodes must validate the newcomer's attestation report. This is a crucial step as it ensures that no single node can unilaterally admit a new member into the network, thereby decentralizing trust.

3. Secret Share Distribution: Upon successful verification of the attestation report, the existing nodes collaborate to spin up a new share of the secret key for the new node. This process utilizes secure multi-party computation protocols to ensure that the new node receives its share without any single node ever having possession of the complete secret key. The threshold secret sharing is basically n evaluations of a randomized polynomial

$$p(x) = s + r_1 x + r_2 x^2 + \cdots + r_t x^t,$$

with s being the secret, and $x = 1, 2, \cdots, n$. Hence, to accept a new node with index $n+1$, a threshold of the existing nodes collectively evaluate the polynomial

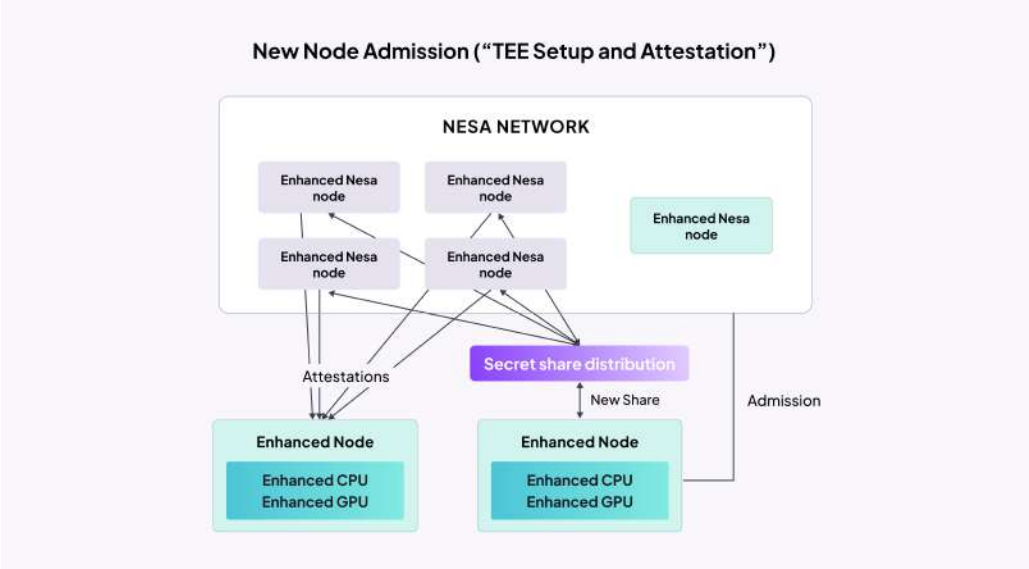

Figure 4.2: New node admission. To admit a new node into the network, the new node has to attest to a number of existing nodes. Upon successful attestations, the existing nodes will collectively generate a new share of the secret key for the new node, enabling it to become a member that is capable of participating in the hardware-based secure inference procedure shown in Figure 4.1.

on a new point $p(n+1)$. Basically, the way to perform such an evaluation in a decentralized way is by using a distributed version of the Lagrange interpolation. Suppose we have $t+1$ evaluations:

$$(i_1, a_{i_1}), \cdots, (i_{t+1}, a_{i_{t+1}}),$$

where $a_i = p(i)$ is the secret share held by node i. By applying the Lagrange interpolation formula, we have:

$$p(n+1) = \sum_{j=1}^{t+1} a_{i_j} \lambda_j,$$

where λ_j is the Lagrange interpolation coefficient:

$$\lambda_j = \prod_{k=1, k\neq j}^{t+1} \frac{i_k - n - 1}{i_k - i_j}.$$

Note that $a_{i_j}\lambda_j$ can be computed locally by node i_j. Therefore each node can compute a Lagrange interpolation term and send to the new node that will aggregate and obtain $p(n+1)$ as a new share of the secret.

4. Key Share Integration: The new node integrates the received share of the secret key into its TEE. With the combination of the key share and the TEE's secure enclave, the node is now capable of participating in threshold decryption operations as part of an inference committee.

5. Node Admission: The node is now considered a trusted member of the network and can participate in inference tasks. For ongoing assurance, the attestation process can be periodically repeated to confirm the integrity of the TEE.

The attestation and key share distribution processes are essential for maintaining a secure and decentralized environment, preventing any single point of failure in the security model. Each node's TEE serves as a strong anchor of trust, with the assurance that it has not been tampered with and is running the correct software. By distributing the responsibility of attesting and generating secret shares across multiple nodes, we ensure that the system remains resilient and can dynamically adapt as new nodes join or leave the network.

This subsection outlines how TEE attestation is integral to ensuring that all computation nodes meet the stringent security standards required to protect the confidentiality and integrity of user data, resisting any form of participant collusion.

4.4 Composite 2: Cryptography

Nesa's ZKML cryptographic method for secure and verifiable computation pair with its hardware composite to provide versatility in execution environment based on the size of the model, the preference of the user, and the complexity of the task. The amalgamation of the two composites is such that each can in some capacity offset the other's limitations. The resource-intensive nature of cryptographic solutions leads Nesa to default to their usage particularly for small and medium-sized AI tasks requested through the AIT. This minimizes the computational overhead that cryptographic privacy-preserving techniques could face on a large model. As a result, our cryptographic solution employs *broadcast secret sharing (BSS)*, a protocol that enables a user to distribute a public message that allows a group of recipients to obtain secret shares of a confidential message without revealing the message itself.

Nesa operates on BSS to ensure that at no point is the unencrypted input data exposed to any of the participating nodes. Instead, secret-shared inputs are used to perform SMPC, allowing each node to process the data with its model and generate an individual inference result. This approach maintains the privacy of both the input and the model. This section describes the steps of result submission, majority vote, rewards, and punishment that comprise the cryptography composite.

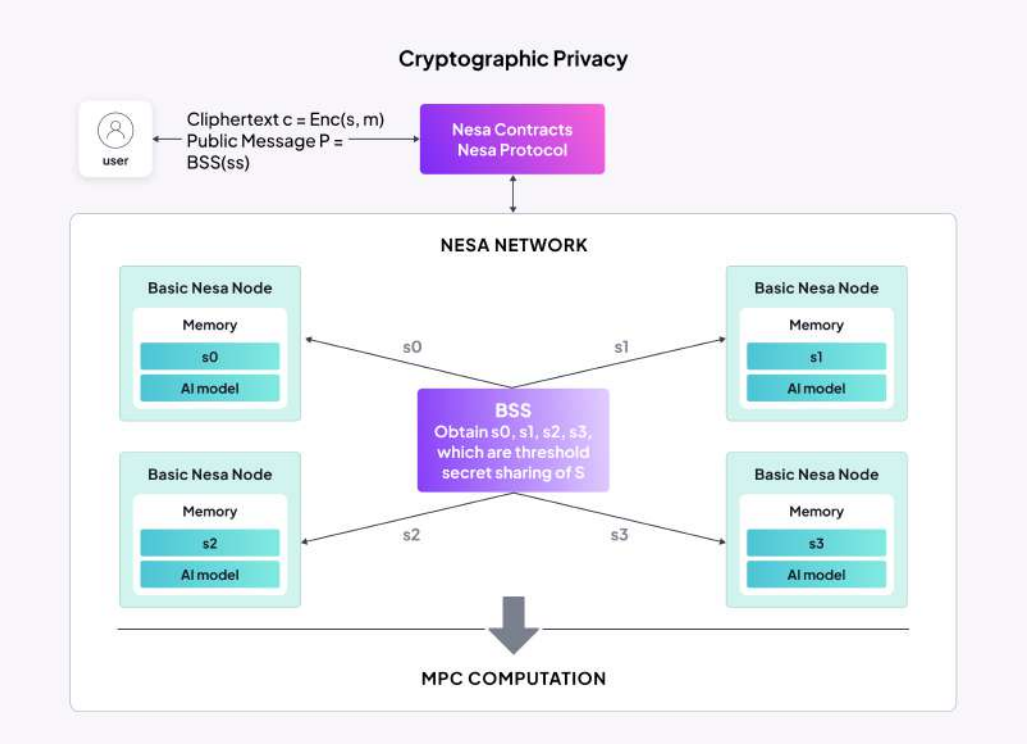

Figure 4.3: Nesa's Cryptography Composite. A user submits ciphertext in the request transaction, along with a public message that enables threshold sharing of a secret. The ciphertext can be decrypted in the committee of nodes, through broadcast secret sharing and threshold decryption. The decryption reveals the plaintext, enabling each node to continue performing subsequent inference computation. The decryption can alternatively reveal only plaintext shares such that subsequent computation is carried in MPC.

Computation on Fragmented Data

By way of this hybrid composite, Nesa allows the user to encode their sensitive input data into a format suitable for broadcast secret sharing. The user's data is transformed into a public message consisting of multiple secret shares, each intended for a different computation node. When nodes receive their respective shares, they perform computations on this fragmented data without the ability to reconstruct the original input individually. Through this process, nodes collaboratively engage in the computation of the AI inference task using an SMPC protocol, where each node contributes its computational power while the data remains distributed and confidential.

Protocol Steps

We break down Nesa's cryptography composite in the following steps (shown in Figure 4.3).

1. Input Data Preparation: The user prepares their input data encrypted by a secret s. The secret s is processed into a public message using a broadcast secret sharing scheme. The user submits the ciphertext c and the public message to the blockchain along with the inference request.

2. Node Committee Formation: Similar to the process outlined in Chapter 3, a committee of nodes is selected based on a VRF technique to ensure a random and fair choice of participants.

3. Secure Multi-Party Computation: The nodes runs the BSS algorithm collectively to obtain a (t, n)-threshold secret sharing of the secret s. Upon receiving their secret shares of the secret, the nodes initiate an SMPC protocol. Each node processes the input with its own AI model, yielding an inference result revealed to itself. If we let $[\cdot]$ denote something that is secret shared, then this protocol contains several sub-steps: 1) $[m] \leftarrow \mathsf{Decrypt}(c, [s])$; 2) $[R] \leftarrow \mathsf{LLM_i}([m])$; 3) $R \leftarrow \mathsf{Reveal}(i, [R])$.

4. Result Commitment: Nodes commit to their results by sending a cryptographic commitment to the blockchain, ensuring commitment to the individual computations.

5. Result Reveal: After a specified timeframe, nodes reveal their results by submitting them to the blockchain, alongside any necessary cryptographic proofs of correctness.

6. Result Aggregation and Majority Formation: A smart contract aggregates the partial results, applying the majority vote strategy to determine the final outcome of the inference task. The smart contract identifies the consensus result based on the majority of matching partial results.

7. Reward and Punishment Enforcement: The smart contract dispenses rewards to nodes that contributed to the consensus result. Nodes that submitted deviating results are penalized as outlined in the system's rules.

8. Inference Result Finalization: The consensus result, representing the secure and private computation of the AI inference task, is recorded on the blockchain. Users can then retrieve and make use of the final result for their purposes.

Beyond ensuring the confidentiality of input data through broadcast secret sharing, Nesa utilizes of ZKPs to verify that nodes have correctly performed their part of

the secure multi-party computation without revealing the underlying data or model parameters.

The zero knowledge scheme on Nesa for proving AI model computation (zkAI) consists of the following algorithms:

- $\mathsf{pp} \leftarrow \mathsf{zkAI.KeyGen}(1^\lambda)$: Given the security parameter, the algorithm generates the public parameters pp.
- $\mathsf{com}_W \leftarrow \mathsf{zkAI.Commit}(W, \mathsf{pp}, r)$: The algorithm commits the parameters W of the model using the randomness r.
- $(y, \pi) \leftarrow \mathsf{zkAI.Prove}(W, X, \mathsf{pp}, r)$: Given a data sample X, the algorithm runs inference algorithm to get $y = \mathsf{pred}(W, X)$ and generates the proof π.
- $\{0, 1\} \leftarrow \mathsf{zkAI.Verify}(\mathsf{com}_W, X, y, \pi, \mathsf{pp})$: The algorithm verifies the prediction y with the commitment com_W, the proof π and the input X.

Recent efforts in the field have yielded a specialized protocol, called the zkCNN, which focuses on the verification of neural network evaluations within the zero-knowledge proof framework. This protocol enables the construction of proofs that attest to the correct execution of computations related to AI models, such as convolutional neural networks, without compromising data privacy.

The adoption of ZKP for verifiable computation within AI model evaluation comes not without significant computational overhead. The generation and verification of zero-knowledge proofs for complex neural network operations are a resource-intensive process, which can lead to prolonged execution times and increased costs. As a result, Nesa orchestrates the application of ZKP through Split-Flow when it ascertains the requested query is on a relatively small or medium-sized AI model, where the trade-off between the added security and the computational burden is justifiable to the user.

Our system strategically incorporates ZKP-based verifiable computation where the model complexity and inference task size allow for the practical application of these advanced cryptographic methods. This selective integration ensures that enhanced privacy and verifiable computation when appropriate without disproportionately affecting the system's overall throughput and performance, safeguarding scalability. Split-Flow is the arbiter protocol that is constantly evaluating and calibrating the security-utility trade-off for each inference request and AIT kernel activated on Nesa.

Chapter 5

Large Language Models in the Nesa Ecosystem

LLMs, such as OpenAI's GPT (Generative Pre-trained Transformer), are advanced AI systems designed to understand, generate, and interact with human language in a way that is both contextually aware and highly nuanced. These models are trained on vast amounts of text data, allowing them to perform a wide range of language-related tasks such as translation, summarization, question answering, and creative content generation.

Mathematical Formulation and Notations

We now introduce the formulation and notations we will be using to describe our LLM inference algorithm. In essence, LLMs model a language sequence $(w_1, w_2, \cdots w_N)$ by computing the conditional distributions

$$\log p(w_1, w_2, \cdots w_N) = \sum_{i=1}^{N} \log p(w_i | w_1, w_2 \cdots w_{i-1}) \tag{5.1}$$

For convenience, we write $p^i(w_i) = p(w_i|w_1, w_2 \cdots w_{i-1})$, we also write context $C^i = (w_1, w_2 \cdots, w_i)$ for $i < N$. In an LLM task, the inference code will be given C^K, and the model is asked to predict continuations $(w_{K+1} \cdots, C_N)$. The prediction is done by ancestral sampling from each $p^(w_i)$, where $i = K + 1, K + 2 \cdots N$.

5.1 Training and Uploading LLMs to Nesa Offline

The initial phase of deploying an LLM on Nesa involves offline training. This process is conducted independently from Nesa to ensure optimal training conditions and efficiency. The model constructor focuses on creating a robust and accurate LLM, utilizing extensive datasets and computational resources. This phase culminates in a fully trained model ready for integration with Nesa's infrastructure.

Uploading Model Parameters

Once the LLM is trained, Nesa provides an upload pipeline to upload its model parameters to decentralized storage solutions IPFS or Arweave. This makes the model accessible to Nesa nodes. The uploaded package includes not only the model parameters but also a comprehensive configuration file detailing the model's architecture and running environment requirements.

Inference Code Integration

The core of the on-chain LLM functionality lies in the inference code. This code is responsible for generating natural language responses based on given prompts. To achieve this, the code employs a sequence of Gumbel noises as a deterministic factor in the Softmax sampling process. By using these noises, the model can produce consistent outputs across different nodes, given the same prompt. This consistency is crucial for maintaining the integrity and reliability of the decentralized network.

Given a prompt $P = w_1, w_2, \cdots w_K$ where $K < N$, the model generates a continuation by sampling from distribution $p(w_{K+1}, \cdots w_N|P) = p^{K+1}(w_{K+1}) \cdots p^N(w_N)$. In LLMs, the sampling of each $p^i(w_i)$ is through Softmax sampling. However, it is well known that the softmax distribution can be approximately reparameterized using Gumbel distribution.

We briefly introduce Gumbel softmax and Gumbel distribution. The standard Gumbel distribution $\text{Gum}(0, 1)$ is defined with probability density function:

$$p_G(x) = \exp -(x + e^{-x}) \tag{5.2}$$

The multivariant Gumbel distribution $p_G(\mathbf{x}) = p(x_1)p(x_2) \cdots p(x_N)$.

Our goal is to sample from a Softmax distribution $p(w)$. $p(w)$ can be representated as:

$$p(w) = (p_1, p_2, \cdots p_M) \tag{5.3}$$

where M is the size of vocabulary. A hard sample w_i can be representated with a one-hot vector:

$$w_i = (0 \cdots, 0, 1, 0 \cdots, 0) \tag{5.4}$$

In Gumbel Softmax, we use a differentiable approximation of hard sample $y = (y_1, y_2 \cdots, y_M)$. We define:

$$y_i = \frac{\exp 1/\tau(\log(p_i) + g_i)}{\sum_j \exp 1/\tau(\log(p_j) + g_j)} \tag{5.5}$$

In this equation $g = (g_1, \cdots g_M)$ is a vector of independent noise. The noise obeys the Gumbel probability distribution: $g \sim \text{Gum}(0, 1)$. Then we have the following result:

$$y \rightarrow^d p(w), \tau \rightarrow 0 \tag{5.6}$$

Namely, y converge in distribution to $p(w)$ when $\tau \rightarrow 0$.

Now looking back to our problem. We want to sample from $w_i \sim p^i(w)$ Roughly speaking, there is a deterministic function f as above, such that $w_i' = f(w_1, w_2 \cdots, w_{i-1}, g_i)$ when $w_i \sim p^i(w)$ when $\tau \to 0$, where g_i is an independent Gumbel noise at timestep i.

In this way, if the sequence of Gumbel noises $(\epsilon_{K+1}, \cdots, \epsilon_N)$ is pre-determined, the sequence $(w_{K+1}, \cdots, w_N)$ will be determined. So each node can generate the continuation given the pre-given Gumbel noises, making them easy to reach consensus.

Consensus Among Nodes

To achieve consensus among nodes, consensus code is included by the model constructor alongside the inference code. A series of presets for default majority response are provided by Nesa, with customizable consensus mechanisms available to the model constructor based on preference and nature of the model and query request.

Enhancing Deterministic Output Generation

The use of Gumbel noises in the Softmax sampling process is a key innovation of Nesa's inference system. It allows the LLM to generate deterministic outputs, which is a critical requirement for consensus in a decentralized environment. This approach ensures that all nodes, given the same input prompt and noise sequence, will produce identical outputs and the deterministic nature of Nesa's inference here is vital for the consistency and reliability of LLM responses across the network.

Nesa's unique combination of offline training, decentralized storage, on-chain deployment, Gumbel noise-based inference for the reconstruction of the exact sequence of inferential steps by any participating node, and unique method of consensus allows LLM inference requests on the network to enjoy a unique level of robustness, reliability, security, and execution speed.

The adaptability of consensus protocols on Nesa is further strengthened with threshold cryptography that enables a subset of nodes to generate a valid group signature, optimizing network latency, and throughput and protect against threats such as Sybil attacks and node manipulations.

Our system is bolstered by the concept of 'Convergence through Divergent Paths', entailing that even though the nodes in the system may execute the inference process independently, the application of Gumbel noise ensures that the divergent paths converge to a single, deterministic result. This sets a new paradigm for how decentralized artificial intelligence systems can operate with high availability and determinism.

Chapter 6

Business Implications of Nesa

6.1 Business Use Case: AI-Based Illegal Content Detection on Storage Blockchains

In the era of digital content explosion, blockchain and decentralized storage technologies have emerged as game-changers, offering data permanence and resistance to censorship. These features however present a significant challenge in content governance. Unlike centralized platforms that can moderate and remove inappropriate content, decentralized networks lack these control mechanisms. This gap becomes critically problematic when it comes to illegal and harmful content as the permanence of such content on the blockchain poses legal and ethical issues.

The Objective and Solution

The primary objective of this business case is to create a robust, blockchain-integrated solution to effectively monitor and prevent the upload of illegal content to decentralized storage networks. This requires decentralization, immutability, and responsible content governance for ethical and legal compliance.

The solution on Nesa would be an AIT Kernel that specializes in vision AI and image-based detection. Through Nesa's modular infrastructure, this model would seamlessly interface with the smart contract protocols of connected decentralized storage services like Arweave for access to all visual content it is meant to evaluate.

The models housed in this AIT Kernel in particular would be trained on datasets focused on illegal content identification, and would be continually updated through Nesa's model upload pipeline to adapt to ever-evolving digital content forms. Leveraging Nesa's smart contract infrastructure, these models would also be embedded directly into the smart contract layer of Arweave, ensuring that all content undergoes real-time analysis before being uploaded to the blockchain. This level of vertical integration would allow for an immediate response to any detected illegal content on Nesa.

Nesa's unique mining process, where miners equipped with GPU-powered nodes are incentivized to process AI inference queries, would be activated to secure the network and reach consensus on content detection for requests initiated out of this Kernel.

Nesa would apply Dynamic Resource allocation for this business case, allowing for flexible scaling of resources, where the expenditure of NES tokens determines the number of miners allocated for each content analysis request. Furthermore, stakeholders would inure the architectural benefits of Nesa's modular network design, including enhanced trust through Nesa's token staking paradigm, whereby miners are required

to stake a certain amount of NES tokens when participating in inference, creating an economic incentive for accurate and honest processing of content. Nesa's dynamic allocation of resources serves as a deterrent against spam or abuse, as higher token costs for larger volumes of content detection data would ensure that the system is used judiciously.

The Outome

This solution on Nesa offers a proactive approach to content moderation, ensuring that illegal materials are identified and blocked before they become part of the permanent blockchain record. This preemptive strategy is crucial in preventing the spread and permanence of harmful content. By leveraging Nesa's decentralized network of miners and having AI models stored on distributed networks avoids central points of control or failure, maintaining security while safeguarding transparency despite introducing an additional layer of content moderation.

6.2 Business Use Case: Enhancing DAO Governance with AI on Nesa

In the evolving landscape of decentralized governance, Decentralized Autonomous Organizations (DAOs) challenge traditional hierarchical structures by enabling a flat, democratic decision-making process, where governance is executed through collective member consensus and smart contracts. This model allows for a high degree of transparency and community involvement. However, the novelty of DAOs brings with it inherent complexities. The sheer volume of governance decisions, ranging from financial transactions to proposal vetting, can be overwhelming, and ensuring that these decisions are made in compliance with both the internal rules of the DAO and external legal requirements is a formidable task, often laden with nuances that are difficult for individuals to consistently interpret and enforce.

One of the most challenging structural issues of DAOs is that the decentralized and often anonymous nature of the vehicle can lead to problems in maintaining accountability and ethical standards. Without a centralized authority, it becomes crucial to have a robust, impartial mechanism for governance oversight. The lack of such a mechanism can result in inefficiencies, disputes, and a potential loss of trust among members, which are detrimental to the longevity and success of a DAO. We believe at Nesa that this is the reason so many DAOs fall apart.

The Objective and Solution

The primary objective of this business case is to integrate an AI model (particularly an LLM) into a customer's DAO to revolutionize governance through the automation

of its decision-making process. The goal would be to create a system where governance proposals, financial transactions, and operational decisions are automatically reviewed and assessed for compliance with both the internal rules of the DAO and applicable legal standards. The requirements for solution would be to address the challenges of scalability, accountability, and regulatory compliance in DAO operations.

The solution on Nesa would be to upload an LLM that has been trained offline to understand and interpret local laws, regulations and the legal framework and internal rules governing the DAO, and integrate it into its own decentralized inference framework to analyze, interpret, and provide insights on governance-related content within the DAO codex. Through its modular network design, Nesa would also provide the computational resources and secure environment necessary for their operation.

On Nesa, an LLM inference request would be triggered periodically to automatically review proposals submitted within a DAO, assessing for coherence, relevance, and adherence to the DAO's predefined rules and ethical guidelines. The model would flag proposals that are inconsistent, irrelevant, or violate the DAO's standards to prevent unnecessary or harmful proposals from proceeding to a vote.

Concurrently, the LLM would monitor and analyze transactions from the DAO's treasury on-chain address to ensure that each transaction aligns with the DAO's financial and operational rules, and does not compromise the interests of the DAO members.

Each query transaction, after validation is performed, would roll up to Nesa's settlement layer and at which point it would be added to the ledger to ensure that the results of all LLM operations for governance are transparent, verifiable and even revertible by DAO members.

The Outcome

By leveraging the computational heuristics of an LLM within the constructs of a DAO, governance propositions could be autonomously analyzed with a lens prioritizing compliance and underlying jurisprudential guidance. An LLM operationalized within Nesa's AIT would mitigate the risks associated with decentralized adjudication while allowing the DAO to conform to the canonical statutes that the DAO codex is built upon. This would foster an equilibrium between autonomy and structured regulatory fidelity, resulting in a self-regulated machine of compliance, ethics, accountability, and decentralized governance.

Chapter 7

Nesa's Impact for AI

No researcher today can explain exactly how neural networks are able to work so well in practice. No one knows exactly why, for example, significantly less neural layers are needed in an artificial network than in a biological one for it to still far outperform the brain in cognitive tasks. This underscores the immense lack of understanding that we have around this technology that we are building so quickly.

LLMs hallucinate regularly today despite deep investment into methodologies to prevent this. The top researchers in the world at OpenAI and Google have said that their models "breathe on their own" and cannot be fully understood. And everywhere you look, there is bias and negativity in AI output. Indeed, this is why safety is such a huge concern among the top companies in the space. These are the earliest days of AI evangelization around the world and already, there are unknowns even from AI's foremost creators.

The only way to truly harness AI, to reign it in and successfully focus its implementation while guarding against misdirection, is by putting the process for its querying on-chain. This was the challenge that we embarked on solving over a year ago. And this is how Nesa's unique technology was born.

7.1 Nesa's Beneficiaries

Nesa is putting AI on-chain for the future of humanity. If AI is on-chain, then for the first time in its existence, AI's performance can be legitimately tracked, and its output can be easily monitored and evaluated.

People, companies, and the world need this.

People

Every human in the future will need their personalized AI, their companions, and their copilots on Nesa because it provides proof of their ownership, provides ease of settling royalties and payments to them for their usage, and provides traceability for underlying model updates that can fundamentally change how their digital beings represent them, so that they have clear version control. Furthermore, when querying their personal AI's, they will want full transparency and verifiability into the responses from the underlying model used.

Proof of ownership

In a world where AI-generated content is becoming increasingly prevalent, it is imperative to establish and maintain ownership of your IP to protect your intellectual

rights. Nesa provides a legally enforceable way to assert and prove ownership over your AI-generated content and digital embodiments such as your image, your likeness, and your voice.

Data privacy and control

Your personalized AI's, as companions and copilots, will handle incredibly sensitive personal data. Proof of ownership ensures that you have control over your data, which is essential for safeguarding privacy, and for deciding who has access to the insights that are generated by their AI.

Settlement of payments

Every time your AI produces value, be that in the form of content, data processing, or some other service, Nesa can automatically execute the transfer of funds to you as the IP holder. This removes the need for intermediaries, reducing the potential for disputes and delays in payments. Maintaining proof of ownership on-chain is critical for individuals interested in monetizing their IP or trading their digital AI services and solutions within online marketplaces.

Version control

Your personalized AI's will be eminently dynamic, evolving through ongoing training and interaction with data, which is the primary reason why we created Nesa. The ability to trace the evolution of your AI system over time is fundamental because it ensures accountability and enables you to understand how, when, and why your AI changed, progressed, or degraded.

Digital history

Nesa has the profound ability to conserve digital heritage through AI, incorporating one's knowledge, experiences, and even part of their personality. By creating a digital artifact on Nesa, online life can be augmented through a digital reach and level of automation that one cannot come close to achieving through manual activity.

Personal empowerment

Your AI's will invariably become increasingly personalized over time. By utilizing an on-chain version-controlled approach, you will have the agency to select which version of your AI you wish to interact with, and which AI you wish others to interact with. This is akin to choosing a specific software update based on your preference for features or performance of query response.

Businesses

Every application and service integrating AI into its business today needs to have their their inference querying, the training processes of their models, and their model updates touch the blockchain somewhere in their stack, because they will receive unparalleled monitoring and evaluation ability, quality control, and shared gains in doing so.

Performance monitoring

Nesa ensures that every action taken in the lifecycle of an AI model is recorded permanently. This helps in performance monitoring by providing a transparent and tamper-proof audit trail for all model output results. With Nesa, stakeholders can accurately trace the evolution of their model, facilitating a clear understanding of changes and their impact on the performance of query response.

Quality control

Nesa automates quality control checks for businesses looking to ensure that there is predefined commercial criteria that their model update needs to meet before it is deployed out to their users or customers. Nesa provides the assurance that a business's AI system is being maintained responsibly and that its model is open for scrutiny.

Resource sharing

Every application will want to become a dedicated services provider or active dApp on Nesa's Utility Suite so that it can benefit form the ocean of AI services that the ecosystem provides. Nesa provides a turn key suite for everything from training and accessing models to sharing computational resources.

Safety

Each AIT Kernel can be programmed to adapt to new regulations automatically via smart contract, ensuring continuous compliance and reducing legal risks for companies navigating a quickly changing regulatory environment. The integration of AI governance frameworks in Nesa's consensus based design can also enforce ethical guidelines, such as avoiding bias in decision-making and respecting user privacy

The World

The people of our planet need AI on-chain because the power currently rests in the hands of a few companies that are building AGI privately and at an alarmingly fast pace. There must be greater visibility and openness around the performance of new models and the outcomes of training updates. Until now, there has not been a cohesive solution that is easy to integrate into. Nesa makes it brutally simple.

Decentralizing AI research

Nesa's modular network will democratize AI development, allowing federated groups of researchers to collaboratively query and evolve models, with on-chain mechanisms ensuring the integrity and direction of a model's growth according to shared goals. AI querying and evolution on-chain under Nesa could see major decisions about models' development pathways being made through community consensus rather than being left to a few major corporations and their executives.

Sharing performance data

Nesa is a conduit for on-chain AI performance data sharing, which will help standardize key AI protocols for all humans, and deliver global awareness of the capabilities of AI models in production. Visibility of all major models and comprehension about their performance and implications on society is the first critical step to keeping everyone safe at a global scale.

Bridging educational gaps in AI

Nesa will play a pivotal role in bridging the educational divide in AI by providing an open, transparent repository of AI development and performance data. Such a resource will be invaluable for educational institutions and students around the world, particularly in regions where access to cutting-edge AI knowledge and training models is limited. By democratizing access to AI learning materials and allowing for the replication of experiments and model evolutions, Nesa fosters global literacy in AI, accelerates innovation, and empowers a new generation of AI practitioners, bringing a standard of equality to the industry that today is remiss.

Diverse Perspectives

Nesa will contribute to the global benefit by enabling diverse groups, ranging from multinational teams to local communities, to collaborate within the AIT repository ecosystem, so that a myriad of perspectives are incorporated into AI. This diversity can guard against the creation of echo chambers and biases that occur when AI development is restricted to homogenous groups or regions. Nesa has the unique potential to be reflective of the real, multicultural world we live in, ensuring that artificial intelligence serves all and not just a privileged subset.

Controlling unchecked progress

Nesa, in managing AI model querying on-chain, can implement mechanisms that control how and when AI systems are allowed to respond to queries and force the integration of safeguards and ethical considerations at each step for the betterment of society as a whole.

Chapter 8

Platform Tokenomics

8.1 Overview of $NES

Nesa's native asset, $NES, is an essential part of how developers build on the layer-1 for AI. To use Nesa for AI model inference, rollup developers submit PayForQuery transactions on the network for a fee, denominated in $NES.

Role of NES

A core part of the Nesa vision is that deploying an AI update for critical inference should be as easy as calling a smart contract is. In this era of AI progress, developers no longer need to handle every part of the pipeline if they want their AI output to be trusted, accurate, and immutable. Nesa does all the heavy lifting.

Requesting AI model queries

Just as ETH is used on Ethereum-based rollups, developers can opt to bootstrap their AI model update quickly through Nesa by using $NES as a gas token and currency in addition to paying for query availability.

Proof-of-Stake Consensus

As a permissionless network, Nesa uses Proof-of-Stake (PoS) to secure its own consensus. Like in other Proof-of-Stake networks, any user can help secure the network and vote on model updates by delegating their $NES to a Nesa validator for a portion of their validator's staking rewards.

Decentralized Governance

$NES staking also allows the community to play a critical role in decentralized governance over key parts of Nesa such as voting on model query parameters through governance proposals and governing the community pool which receives 2% of block rewards.

Inflation

$NES inflation starts at 8% annually and decreases by 8% every year until it reaches the long term issuance rate of 1.8%. Figure 8.1 displays annual inflation over the next 20 years.

The annual provisions for inflation are calculated based on the total supply of $NES at the beginning of each year. To calculate how many $NES to issue per block, Nesa uses the block timestamp rather than the block height given the time between blocks

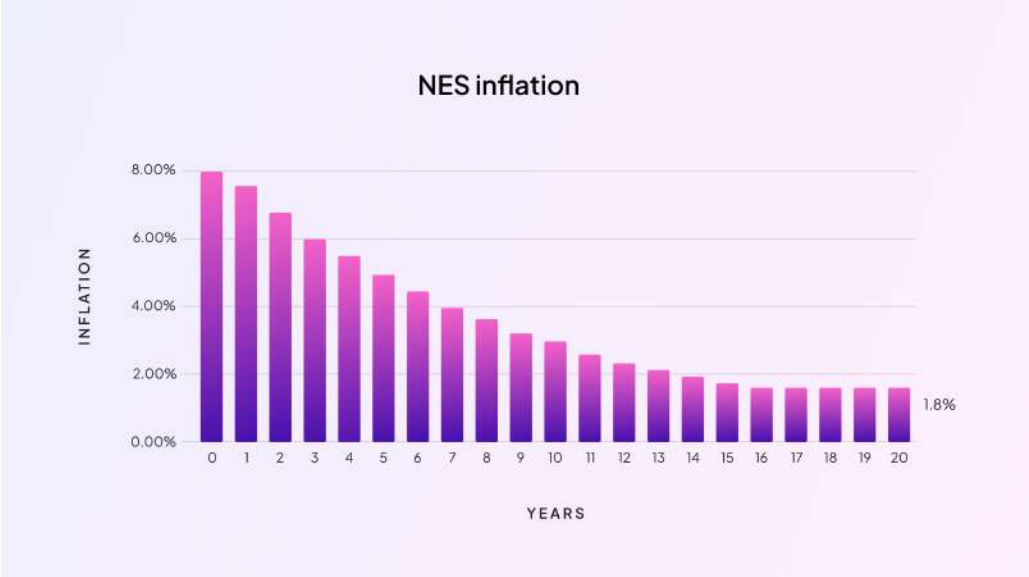

Figure 8.1: NES inflation over the next 20 years. $NES inflation starts at 8% annually and decreases by 8% every year until it reaches the long term issuance rate of 1.8%.

can vary and cause actual issuance to be higher than the target.

$NES Allocation at Genesis

Nesa will have a total supply of 1,000,000,000 $NES at Genesis, split across five core allocation categories. The chart in Figure 8.2 describes each core allocation group.

Nesa plans to use allocation from Incentivized testnet to reward developers who contribute to the Nesa ecosystem, who build on Nesa, who submit their AI Models for use on Nesa, and early miners who lend their compute power for reward.

The earliest supporters of Nesa will be the recipients of the largest portion of the Incentivized testnet token allocation. In addition to being early on the platform, regularly submitting PayForQuery transactions on the system through an AIT Kernel results in a higher score for incentive rewards. For direction on how to earn $NES, the Nesa community channels are the best source of information.

The Genesis drop can include $NES token allocated on both decentralized and centralized platforms, to be announced in Nesa community channels leading up to listings.

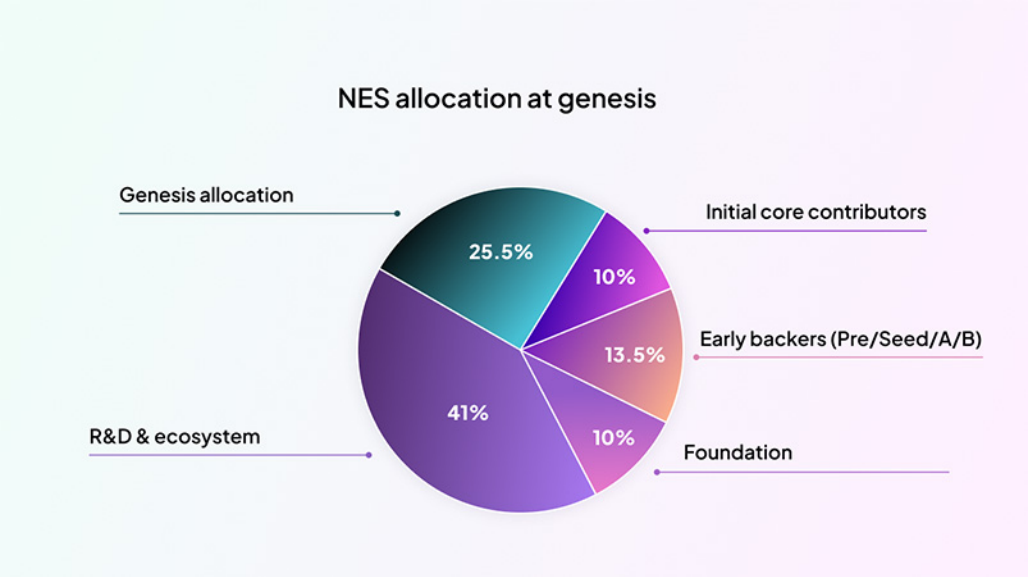

Figure 8.2: $NES allocation at Genesis. $NES will have a total supply of 1,000,000,000 $NES at Genesis, split across five core allocation categories. Nesa plans to use Allocation from Incentivized testnet to reward developers who contribute to the Nesa ecosystem, who build on Nesa, who submit their AI Models for use on Nesa, and early miners who lend their compute power for reward.

Category	Description	%
Genesis Allocation	Genesis Drop, Incentivized Testnet Treasury, Future initiatives	25.5%
R&D & Ecosystem	Tokens allocated to the Nesa Foundation and core devs for research, development, and ecosystem initiatives including: • Protocol maintenance and development • Programs for rollup developers, infrastructure, and node operators	41%
Early Backers (Pre/Seed/A)	Early supporters of Nesa	13.5%
Foundation	Platform and Foundation support	10%
Initial Core Contributors	Members of Nesa Labs, the first core contributor to NES	10%

Note: allocations can move between the fundraise series at the discretion of the team

Figure 8.3: Detailed description for $NES allocation at Genesis, split across the five core allocation categories listed above from Figure 8.2.

Unlocks

Nesa's 1 billion $NES supply at Genesis will be subject to several different unlock schedules (as shown in Figure 8.5). All tokens, locked or unlocked, may be staked, but staking rewards are unlocked upon receipt and will add to the circulating supply.

Circulating Supply

Circulating supply is defined as the amount of $NES tokens in general circulation without on-chain transfer restrictions, including lock-ups.

Available Supply

Available supply (Figure 8.4) is defined as the amount of $NES tokens that are either part of the circulating supply or are unlocked but subject to some form of governance to determine when the tokens are allocated. This includes the unlocked portion of the R&D & Ecosystem tokens and the tokens set aside for future initiatives beyond the Genesis drop and Incentivized testnet.

References to Launch above refer to the day of public access to $NES, meaning $NES listing on one or more public platforms such as decentralized or centralized exchanges.

Category	Unlock Schedule
Genesis Allocation	Fully unlocked at launch
R&D & Ecosystem	6 month lockup. Linear vesting for 2 years.
Early Backers (Pre/Seed/A)	12 month lockup. Linear vesting for 2 years.
Foundation	12 month lockup. Linear vesting for 3 years.
Initial Core Contributors	12 month lockup. Linear vesting for 3 years.

Figure 8.5: $NES supply unlocking schedule. References to Launch above refer to the day of public access to $NES, meaning $NES listing on one or more public platforms such as decentralized or centralized exchanges.

PayForQuery Transactions

To publish data on Nesa developers can submit PayForQuery transactions. A PayForQuery transaction consists of the query request, the query size, the identity of the sender of the data to be made available for, the namespace and a signature. Each PayForQuery transaction is split into two parts: the query which includes the data to be made available along with the namespace, and the executable payment transaction which includes a commitment to the data. Both the query and executable payment transactions are put into the block within the appropriate namespace. The block data is extended using erasure coding and then Merkelized into a data root commitment included in the block header.

Fee Market Overview

Nesa uses a standard gas-price prioritized mempool. This means that transactions with higher fees will be prioritized by validators. Fees are comprised of a flat fee per transaction and then a variable fee based on the size of each evolve in the transaction.

Network Parameters

$NES holders - not just stakers - can propose and vote on governance proposals to change a subset of network parameters. Nesa will publicly list both the changeable and non-changeable parameters and their values of the system to vote on. To learn how to

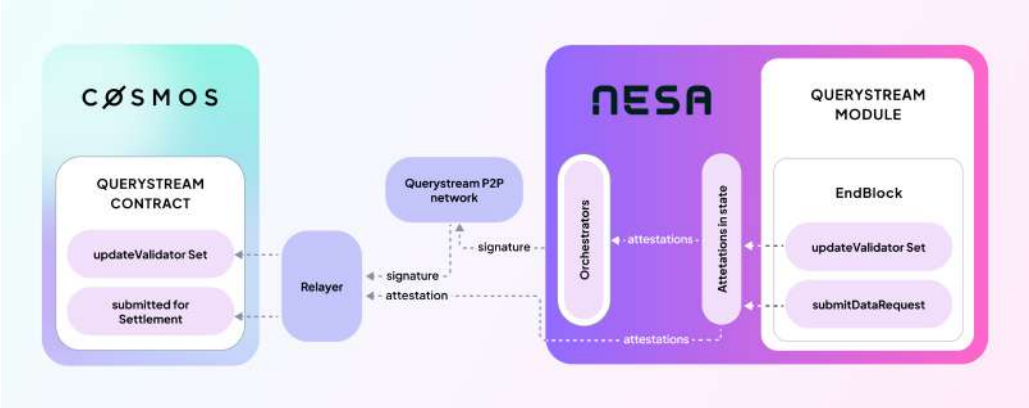

Figure 8.6: queryStream flow of communication. queryStream is Nesa's method to stream verified transaction data from a query inference request to Nesa's Settlement Layer for registry on the blockchain. With Nesa's interoperable structure, users can elect to stream to another blockchain network such as Cosmos or Ethereum at their preference. The module on Nesa flows through attestation and orchestrators to the queryStream P2P Network, and then into a dedicated Relayer that streams the query result to Nesa's Settlement Layer or to Cosmos or Ethereum for settlement. The queryStream contract is then formalized on Nesa, executing the model inference end-to-end.

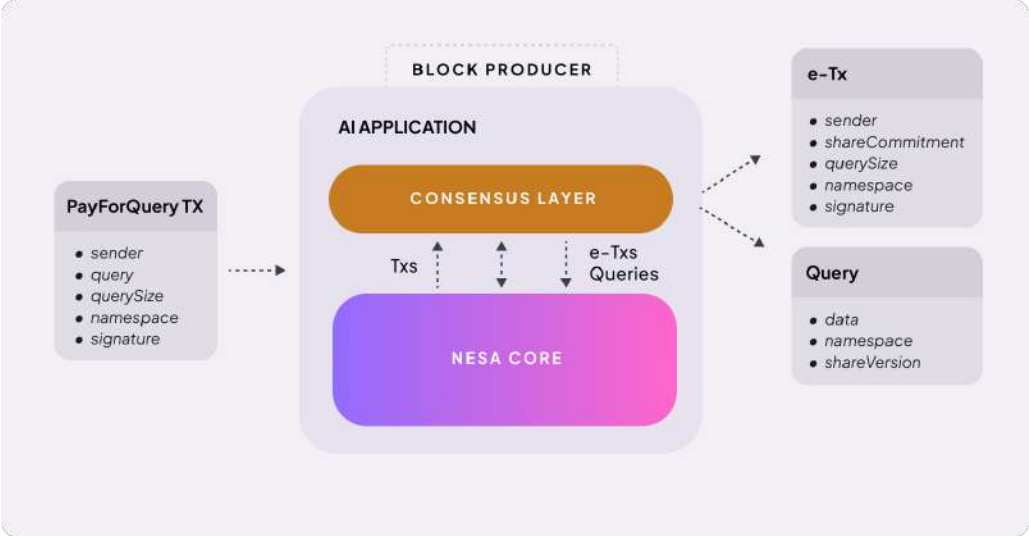

Figure 8.7: Block production for a PayForQuery transaction. When initiated, a namespace and query details are established, and the transaction is signed. It is then passed through the system as it is shared with the Consensus Layer for verification and Nesa Core where it is prepped for settlement.

submit and vote on governance proposals, review the resources in Nesa's online docs.

Community Pool

Starting at Genesis, Nesa's community pool receives 2% of all $Nesa block rewards. $NES stakers may vote to fund ecosystem initiatives.

queryStream

queryStream interfaces with Nesa's crypto-economic framework backed by its Proof-of-Stake consensus protocol. By utilizing Nesa's Consensus Layer, computations for AI model inference are transparent and subject to scrutiny by the network of validators and their delegators through an efficient attestation mechanism.

In scenarios where dishonest activities are suspected, such as data withholding by validators, the system is designed to recognize any malfeasance if 2/3 of Nesa validators are implicated, prompting actions such as slashing to maintain network integrity. This modular verification layer provides strong crypto-economic security assurances, raising the standard for trust in decentralized AI model evolution.

queryStream enables developers to tap into Nesa's scalable namespace-specific storage, which may start at a few hundred kb per inference query on the Mainnet Beta, with potential increases manageable through on-chain governance. The actual storage provision is based on the complexity and size of the query and underlying model, and supports the potential for tens of thousands of model parameters to be processed per second due to the efficient use of erasure coding and Merkle tree data structures.

The communication between Nesa's verification layer and it's settlement layer (or Cosmos/Ethereum's settlement layer given that Nesa is interoperable) is facilitated by a peer-to-peer (P2P) network which includes an queryStream Relayer. The Relayer streams the query data from the verification layer to the settlement layer (or to the Cosmos/Ethereum network), where dedicated smart contracts formally encode and execute the query directives received from Nesa.

Every PayForQuery transaction submitted to Nesa is validated with a Byzantine Fault Tolerant (BFT) signature mechanism. Any AI model data included in Nesa can be independently verified on Cosmos/Ethereum using queryStream, with validation results backed by cryptographic proofs. On settlement, the AI model inference proofs and PayForQuery transaction data are published to Nesa.

For ZK rollups employing queryStream, data inclusion must be verified prior to considering any proof as valid. This can be done by incorporating the inclusion proof within the ZK proof itself for submission to Nesa's settlement layer, or another blockchain network like Cosmos.

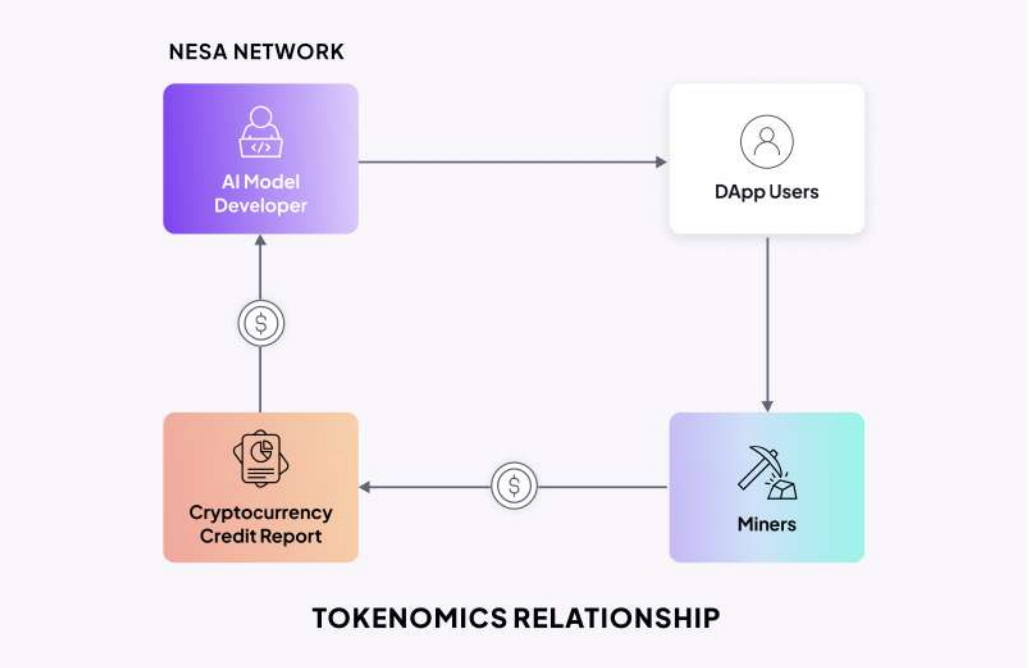

Figure 8.8: Tokenomics relationship on Nesa. Miners are required to stake a portion of $NES for a vested interest in the honest and efficient processing of AI queries on behalf of the dApp user.

Token Fee Mechanism

We summarize these high-level insights into the tokenomics of the platform in Figure 8.8 that describes the token-based relationship between participants in the Nesa ecosystem. On Nesa, miners are required to stake a portion of $NES tokens. This requirement serves as a safeguard, ensuring that miners have a vested interest in the honest and efficient processing of AI queries. This aspect of our system draws parallels to the concept of oracles in blockchain, yet with a unique focus on AI model execution.

Nesa's token fee mechanism introduces a flexible economic model: the more tokens committed, the larger the pool of miners, hence enhancing security with more proofs attached in the queryStream. Conversely, spending fewer tokens results in a smaller miner pool, indicating a trade-off between cost and security.

Chapter 9

Conclusion

In conclusion, the convergence of blockchain and AI embodied by the Nesa ecosystem heralds a profound transformation in technology, economics, and society. By embedding AI capabilities within smart contracts, we establish a foundation for intelligent decision-making that addresses complex, real-world applications with unprecedented ease and security. Nesa's AIT and decentralized inference protocol with ZKML privacy design represent a trifecta of innovations that, together, forge a new paradigm for conducting AI tasks on a blockchain infrastructure.

The AIT is a testament to harmonized execution consistency, spearheading a trail where AI computations are performed without compromising the overarching system's performance, security, or accessibility. Nesa's bifurcated inference methodology ensures not only a trustless computational process but also the integrity of the results produced by its consensus-driven AI models. Our novel ZKML cryptographic protocol and hybrid privacy design dismantle traditional trade-offs between data confidentiality, computational capability, and verifiability, balancing these imperatives to meet the demands of secure, decentralized systems.

Nesa's pragmatic approach to privacy, security, and computational efficiency bridges the cryptographically rigid realm of blockchain with the fluid intelligence of AI. This seamless integration opens up a myriad of possibilities across industries, enhancing financial transactions, automated legal reasoning, personalized digital interactions, objective decision-making on-chain, and ethically-aligned DAO governance, to name a few. Moreover, Nesa's robust tokenomics not only fuel the ecosystem's vitality but also intricately align incentives between miners, developers, and stakeholders, fostering a self-enriching economic loop within an ecosystem that is equitable and transparent by design.

As the foundation for the Nesa ecosystem, AIT Kernels act as autonomous entities with a unique genetic architecture, reflecting an advanced stage of blockchain and AI confluence. This vision goes beyond technological advancement; it embodies a commitment to imbue AI progression with the transparency, ethical considerations, and safety required for industry-wide acceptance and integration.

With an intricate understanding of consensus protocols, federated systems, model version control, and cutting edge artificial intelligence models, Nesa sets out to not only solve the pressing challenges of AI inference queries on-chain but to provide a platform for pervasive, successful AI inference across industries. As AI continues to move towards ubiquity, Nesa positions itself as the pioneering force, preparing the terrain for the next chapter in human-digital symbiosis.

Chapter 10

Addendum

10.1 Dynamic Model Versioning and Fork Management

AI models evolve constantly, and managing these versions on-chain can become complex. Nesa incorporates a sophisticated version control system for models that handle forks and merges in model evolution, akin to Git but for AI training processes. This system tracks and merges various AI training paths, facilitating more complex evolution, adaptations, and experimentations with model updates.

When developing AI, it is imperative to track different versions of your models and corresponding input data. This requires robust version control mechanisms for both models and datasets alike. By logging the lineage of data and model updates on Nesa, companies can provide version histories necessary for ensuring that the correct versions are always used or referenced, and that rejected model updates can be learned from and calibrated on.

AIT Kernels on Nesa can automate staying in corporate compliance and general compliance. This level of container adaptability allows for the potential rollback of heterogeneous AI models in the event of an unsuccessful update, application downtime, DMCA takedown notice, user/customer complaint, or system exploit.

AIT Kernels benefit from the unique Configuration Charter (Figure 10.1) that they are instantiated with on Nesa, which sets a universal template for the easy replication of their specifications and procedures for such a Marketplace to exist.

10.2 Nesa's Utility Suite

Nesa accessorizes its inference system with a hub for external AI tools that help facilitate model querying, training, upload, and evolution (Figure 10.2). As the point of final validation for any model inference query, Nesa sits at the very bottom of the stack and can plug an assortment of third party AI services on top of it.

These services are serially executed and consigned to a dedicated adapter within the system. The Utility Suite includes provisioned computational power (TPUs, GPUs), almost all major publicly available open source models, the major API-based LLMs, an assortment of Information Oracles, DAO tooling, and Relayers.

Nesa will continue to grow its collection of AI-Web3 partner services that can be invoked during training, upload, orchestration, and inference before Nesa executes final query validation.

Some examples of Nesa's future Utility Suite include:

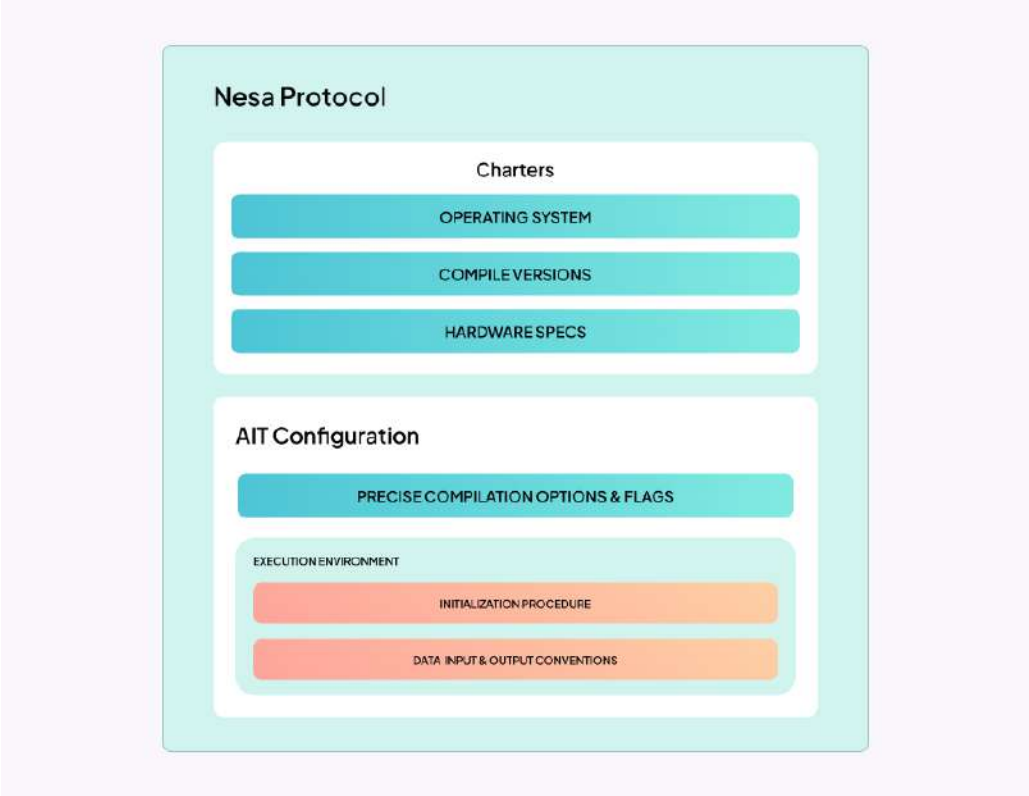

Figure 10.1: AIT specification and configuration. Details are provided in AIT kernels in order to guarantee consistency and inference reliability in the AIT, including operating system, compiler, hardware specifications, initialization procedures, the inference code and aggregation code, data input and output conventions, and other precise compilation options and flags for execution. The execution protocol within the AIT prescribes the exact series of steps that every node must follow. By standardizing the execution flow, we can reliably predict and replicate the behavior of AI models across the network. If a model demands particular hardware characteristics, such as GPU acceleration or specialized processing units like TPUs, these requirements are explicitly stated in the AIT configurations. Moreover, features provided by the hardware that could potentially lead to inconsistent execution, such as non-deterministic hardware instructions, are either strictly enabled or disabled as appropriate. Developers are permitted to customize any aspect of the AIT configuration file including requested hardware for inference and specifics of the query response. Before an AIT kernel is approved and stored on the blockchain, it undergoes rigorous validation to ensure compliance with the specified configuration, whether preset by default or customized by the model's developer.

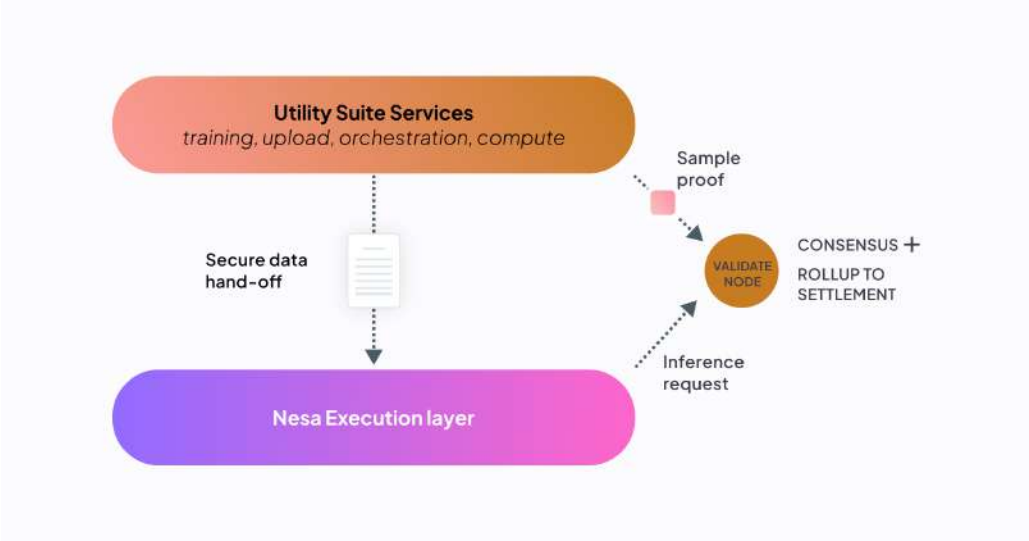

Figure 10.2: Nesa Utility Suite Architecture. Nesa connects a hub of utility services that help facilitate model querying, training, upload, and evolution for AI model developers in the Nesa ecosystem. A utility service is initialized, run, and then securely hands off data to Nesa's Execution Layer to assist in a component of the preparation, orchestration, or computation of the request. Inference is then conducted by Nesa Core and the results are sent to validator nodes for consensus before rollup to settlement.

1) Efficient sharing of computational resources on-chain, such as Render, Akash.

2) Decentralized AI tooling and infrastructure services, such as Olas, Bittensor.

3) Information oracles for off-chain knowledge relayance, such as Chainlink, Band.

4) DAO tooling management systems, such as Gnosis, Aragon.

10.3 Interoperability and AIT's Future Plans

As the AIT matures, our vision for its future encompasses not only expansion in capabilities but also a strong emphasis on interoperability. This is crucial to ensuring the AIT is adaptable platform that can seamlessly integrate with the broader ecosystem of decentralized technologies and cater to a growing range of use cases.

Nesa's LiteBridge facilitates cross-chain interactions on the Nesa network, enabling the AIT to interoperate with different blockchain networks. This protocol allows for the transfer of models, data, and even computational tasks across platforms, contributing to a more integrated and powerful decentralized AI offering for Kernels on Nesa. Nesa's network interoperability has the potential to establish standards for AI model execution

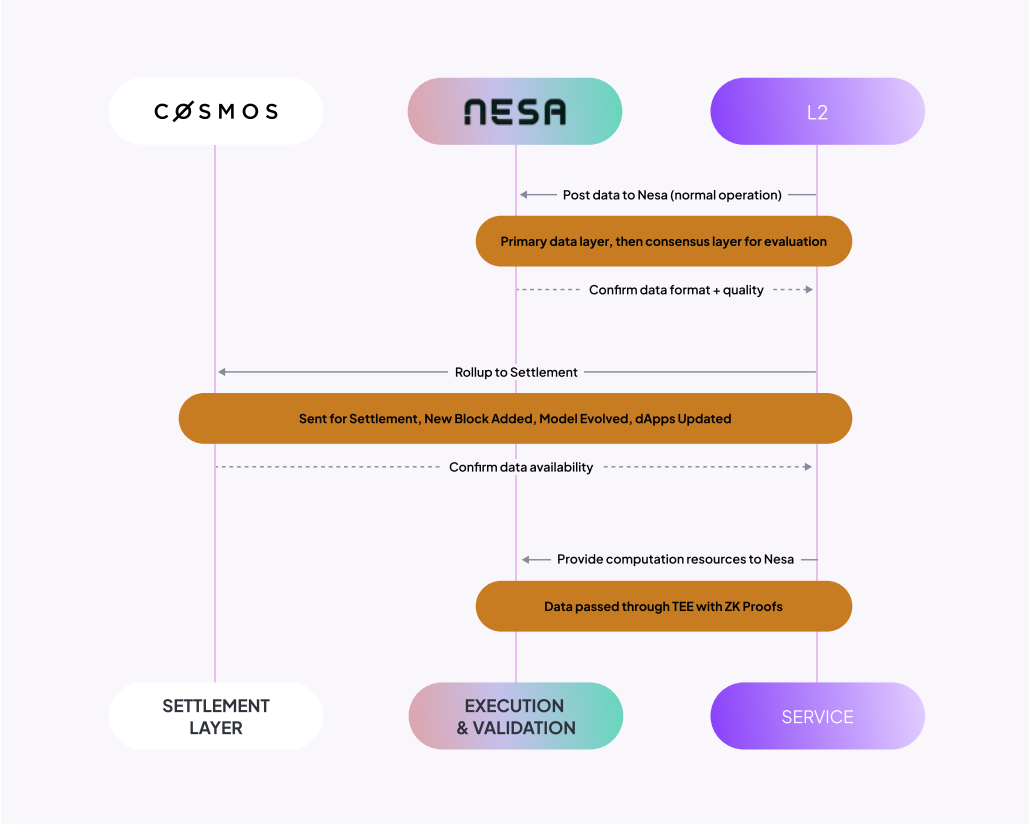

Figure 10.3: L2s connecting to Nesa by adapter. L2's are representative of any blockchain based service in the AI domain, not just a Layer 2. The L2 posts data to Nesa and Nesa communicates back with the L2 to ensure proper data format and quality. From there, results are sent to Nesa's Consensus and Execution Layer, and then rolled up for settlement.

and storage on distributed ledgers for the industry as it brings more partner networks into the ecosystem.

As we move to mainnet, we plan to update the AIT for wider array of AI models and inference scenarios. This includes scaling up to handle larger and more complex models, integrating new machine learning frameworks, and adopting the latest advancements in AI research to provide users with a state-of-the-art execution environment.

The AIT is designed with a modular architecture, allowing for components to be added, removed, or upgraded without disrupting the overall system. This flexibility ensures that the AIT can adapt to new requirements and technologies as they emerge, fostering long-term sustainability and growth

10.4 The AIT Kernel Market

The Nesa team envisions a future where AI models become so critical to daily life that they assume the role and level of importance of today's mobile app ecosystem, like those on operating systems like Google and Apple, where they are sellable and monetizable in an open marketplace.

OpenAI's marketplace of GPTs operate on OpenAI's private model, closed off to the public, meaning a model that is off-chain and not connected to any decentralized environment, and customizable only via OpenAI's blackbox fine-tuning. The work invested into careful fine-tuning and model update provenance can evaporate overnight under such a setup. This is the inherent danger of centralization and AI.

The future that we envision is an AIT Kernel Marketplace where containerized AI's on Nesa could be browsed through, reviewed, piloted, and purchased in an instant to handle any task that a user requires. Nesa wants to give freedom and complete flexibility to creators who want to own, monetize and control their AI evolution. Each container in the Kernel Market would be fully customized, controlled, updated, and managed by its owner.

10.5 The Integration of Evolutionary AI to Evolve the Nesa Ecosystem

This whitepaper previously outlined the fundamental innovations undergirding Nesa: the AIT architecture, its novel hybrid security protocol, and a consensus mechanism tailored for scalable, private, decentralized model inference. This section elaborates on the integration of evolutionary AI within this established framework, underscoring the potentially transformational impact it will have on the ongoing improvement and development of the system.

Evolutionary AI introduces an additional layer of adaptability to the AIT. By embedding system-controlled evolutionary algorithms within its architecture, housed within their very own standalone smart contracts, AI models on Nesa can iteratively self-optimize towards predefined performance metrics and emergent user-defined objectives. This process mimics natural selection within the digital realm, allowing the fittest models to thrive and propagate.

The evolutionary layer interface could be constructed in such a manner that it can efficiently tap into the AIT's capability to execute and evaluate decentralized AI tasks. This would leverage the computational and analytical power distributed across the Nesa when attempting to iteratively adjust model hyperparameters, network architectures, and even learning paradigms in response to feedback from the decentralized network.

Injecting evolutionary processes into an AI system mandates meticulous control to prevent undesirable divergence from optimal operation points. Controlled evolution within the AIT would be regulated through a set of refinement protocols within each evolutionary AI smart contract. These protocols would establish the bounds of permissible evolutionary changes and dictate the procedures for nascence, mutation, and selection of model variants. The key to controlled evolution is the stringent adherence to these protocols by all nodes, ensuring uniform execution and unbiased evaluation across the Nesa ecosystem.

Consensus in an evolving AI context transcends the agreement on state transitions in a blockchain. Within Nesa's consensus mechanism, validators would undertake the additional role of arbiters in the evolutionary process. Through Bifurcated Inference Ledgering and a commit-reveal scheme embedded within the consensus layers, nodes could reach agreement not only on the validity of transactions but also on the acceptability of evolved model states.

Nesa already supports robust decentralized inference mechanisms. With the introduction of evolutionary AI, these mechanisms can be extended to accommodate dynamic and iteratively evolving inference models. Custom aggregation scripts within the AIT kernel could be deployed to facilitate more sophisticated decision-making processes necessary for evolution validation. The result would be an enriched ecosystem where the decentralized inference protocol establishes a resilient, trustless environment capable of supporting the complex lifecycle of evolutionary AI models.

Chapter 11

Background Technology

This section provides a deeper look at some of the security and privacy technologies mentioned throughout the whitepaper

11.1 Trusted Execution Environment (TEE)

Central to our project's commitment to privacy and security in the evaluation of AI models is the integration of TEEs. TEEs provide a secure area within a main processor, ensuring that sensitive data and operations are insulated from the rest of the system. Within our project, this secure environment is utilized to perform computations on encrypted private user data during the AI inference process. By leveraging TEEs, we ensure that the data, although processed by the inference committee, remains confidential and tamper-proof throughout the entire inference lifecycle. This approach means that none of the sensitive information is exposed to any of the inference nodes to preserve the integrity and secrecy of the data. The inference committee, composed of a pre-selected set of nodes, is responsible for collaboratively conducting AI model inference without having direct access to unencrypted data. This creates a robust framework that facilitates user privacy while enabling secure and reliable model evaluations in a decentralized environment, turning our vision of secure and private AI computation into a tangible reality.

Building upon the concept of the TEE, there are several implementations of TEE technologies designed to cater to different types of processing units and their respective architectures. In the CPU domain, prominent players have advanced their offerings to provide robust security solutions:

- Intel's Trusted Domain Extensions (TDX) is designed to enhance the security of virtual machines by providing hardware-level isolation capabilities. TDX creates private regions of memory, known as Trusted Domains, which help to protect code and data from external threats and unauthorized system software.
- AMD CPUs counter security threats with Secure Encrypted Virtualization with Secure Nested Paging (SEV-SNP). This technology adds strong memory integrity protection capabilities to the already existing SEV technology, further fortifying virtual machine isolation and helping to prevent malicious hypervisor-based attacks.
- ARM CPUs introduce the Confidential Compute Architecture (CCA), which aims to fortify application security. CCA provides a secure environment for computation, ensuring that sensitive data can be processed without exposure to the risk of interception or tampering by other software, including the operating

system.

In the GPU landscape, NVIDIA has made significant strides with their Hopper H100 GPU architecture which supports confidential computing. The H100 GPU integrates with the aforementioned CPU TEE technologies, ensuring a secure and seamless interaction between the processing units. This integration allows for the extension of TEE's security benefits into the realm of high-performance computing, making it possible to securely process complex AI and machine learning workloads that require the parallel processing power of GPUs.

These TEE technologies form a multi-layered defense strategy, providing a secure computing backbone for models deployed on Nesa. By leveraging the strengths of each technology, we create a hybrid and interoperable secure environment capable of handling a diverse array of compute demands while maintaining a stringent security posture for confidential computing.

11.2 Secure Multi-Party Computation (MPC)

Nesa employs Secure Multiparty Computation (SMPC or MPC) for provable cryptographic security across the network. SMPC is a cryptographic protocol that enables multiple parties to jointly compute a function over their inputs while keeping those inputs private. Each participant in the computation has a piece of the overall data puzzle, yet none can see the other parties' pieces. This method ensures that intermediate information remains undisclosed to any participating party throughout the process, with only the final output being revealed to the designated recipient.

On Nesa, this is crucial for tasks where both data privacy and collaboration is necessary. However, due to the intensive computational requirements generally associated with SMPC, we have optimized its application to be restricted to lightweight tasks throughout the network. This selective application allows us to benefit from the strong provable cryptographic security guarantees of SMPC where it matters most, without overwhelming the system with undue computational processing demands. As a result, our project not only adheres to rigorous security standards but also maintains a high level of practicality and performance efficiency when handling the strictest data confidentiality measures.

11.3 Verifiable Random Function (VRF)

An additional cryptographic primitive utilized in the AIT Execution layer is the Verifiable Random Function (VRF). VRFs are pivotal for the generation of unbiased and unpredictable random values that are also verifiable by any party with public information. A VRF is a construct that allows its holder to provide a proof that a number

was generated in a truly random and secure manner, akin to a cryptographic lottery that cannot be rigged.

On Nesa, VRFs are utilized to enhance the security and fairness of node selection processes. When executing AI tasks, it is crucial that the committee of nodes responsible for these tasks is chosen without any bias or manipulation. VRFs serve this purpose by allowing for the secure and fair selection of the inference committee. Each node generates a random number with its private VRF key and publishes both the number and its proof. The verifiability aspect of VRFs ensures that all other nodes or participants can independently verify the correctness of the random number, ruling out any foul play in the selection process.

This mechanism preserves the decentralization ethos on Nesa by preventing central authorities from controlling the selection process but also instills confidence in the network participants. This ensures that each AI task is processed by a randomly chosen, yet reliably determined, committee of nodes, which upholds the integrity and unpredictability of the selection protocol for the guarantee of increased transparency and security over assumptions of good intent.

We briefly give some more details about VRF below. It consists of three functions: keygen, evaluate, and verify.

- $\mathsf{keygen}(r) \rightarrow (pk, sk)$: generates a public key pk and a secret key sk for a given random input r.
- $\mathsf{evaluate}_{sk}(x) \rightarrow (y, \pi)$: generates pseudorandom output y and a proof π from the secret key sk and x as inputs.
- $\mathsf{verify}_{pk}(x, y, \pi) \in \{\mathsf{true}, \mathsf{false}\}$: takes the public key pk, the pseudorandom output y, the proof π, and the message x as inputs, and returns true if y is actually the output produced by sk and x. If not, it returns false.

11.4 Zero-Knowledge Proof (ZKP)

Finally, Zero-Knowledge Proofs (ZKP) are a foundational core cryptographic technique employed in Nesa's AIT, complimenting the above security implementations. ZKP enables one party, the prover, to demonstrate to another party, the verifier, that a certain statement is true without revealing any information beyond the validity of the statement itself..

There are some notable examples of ZKP adopted in blockchain applications. Zcash, for instance, utilizes ZKP to enable private transactions on a public blockchain. By using zk-SNARKs (zero-knowledge succinct non-interactive arguments of knowledge),

Zcash allows users to conceal transaction details such as the sender, receiver, and amount, while still verifying the transaction's legitimacy.

Ethereum is also exploring the implementation of ZKPs to enhance scalability and privacy on the platform. Other projects like zkSync and Hermez use ZKP to batch transactions off-chain and then settle them on-chain, providing scalability solutions without compromising on security.

On Nesa, ZKP plays a pivotal role in several key areas:

- Interfacing with Smart Contracts: Drawing inspiration from projects like Chainlink but applying it to small and large scale AI inference requests, we leverage ZKP to build secure protocols that enable the feeding of off-chain data into on-chain smart contracts. This mechanism ensures that smart contracts can access the necessary data for AI model execution while maintaining the confidentiality of the off-chain data sources.
- Privacy-Preserving Computation: We implement ZKP to perform computations on private models and data without revealing any underlying sensitive information. This approach is crucial for preserving the privacy of user data and proprietary AI models throughout the inference process. By employing ZKP, we can provide cryptographic assurance that the computation was executed correctly, without exposing the data to external parties.

The integration of ZKP in our system architecture enables private on-chain interactions and confidential computing while providing an easy and accessible channel where users and enterprises can confidently engage with blockchain technology. This alignment of transparency with confidentiality paves the way for broader adoption and trust in Nesa's decentralized system as it evangelizes AI on-chain around the world.

Split-Flow harmonizes the sometimes simultaneous need for these privacy technologies above through its automated evaluation system that analyzes factors such as input sensitivity, workload, model specifics, and consensus criteria to ascertain whether hardware-based secure enclaves,cryptographic techniques, or a combination of the composites should primarily handle data processing. The adaptability of the Split-Flow protocol allocates processes between a confidentiality-preserving stream and a verifiability-focused one, with security measures tailored to match the dynamic demands of varying data models and inference goals, as well as the model's size and computational intensity, creating a balanced and secure operational state across these heavy security technologies.

Chapter 12

Definitions

Autonomous AI oracle network

An autonomous AI oracle network refers to a decentralized and self-operating system that connects off-chain artificial intelligence services with on-chain smart contracts and blockchain networks. This system enables smart contracts to integrate AI capabilities and make complex decisions based on off-chain data inputs, processed by AI models. It revolutionizes the functionality of smart contracts by endowing them with advanced decision-making abilities, akin to an oracle in blockchain but specifically designed to handle AI computations.

AIT (Artificial Intelligence Terminal)

The AIT is a decentralized system architecture that serves as a uniform execution environment for AI model inference on the blockchain. It parallels the Ethereum Virtual Machine (EVM) in providing a standard set of rules and execution protocols that all nodes must follow. This ensures that the execution of AI models is consistent and secure across different nodes, leading to reliable AI computations within a trustless environment. The AIT is adaptable to a wide range of AI models, offering flexibility and facilitating the integration of AI capabilities into a variety of applications.

AI model inference queries

AI model inference queries are requests submitted to the blockchain network by users who seek to leverage AI capabilities for analytics or decision-making tasks. These queries trigger the process of AI model inference, where input data is fed into a trained AI model to generate predictions or insights. The results of these queries are reported back on the chain, providing users with AI-informed outputs that support various applications, such as predictive analysis or automated content generation.

Bifurcated Inference Ledgering (BIL)

BIL is a two-phase transaction structure that Nesa implements to enhance the scalability and efficiency of decentralized AI computations. By decoupling the submission of an inference request from the actual execution of AI model inference, BIL streamlines the network's process, preventing computational loads from affecting blockchain performance. It ensures non-blocking transactions, maintains high throughput, and facilitates a flexible resource allocation, making it a vital component in the system's

decentralized inference framework.

Commit-Reveal Mechanism

A commit-reveal mechanism is a two-step process utilized to ensure honest and independent contributions from nodes executing AI inference tasks. In the Commit phase, nodes submit cryptographic commitments of their results, concealing the content while proving the existence of a computation. During the Reveal phase, the actual results and the nonce used in the commitment are disclosed for verification. This mechanism prevents nodes from free-riding on the work of others, thereby maintaining fairness and integrity in the decentralized inference process.

Aggregation of Inference Results

The aggregation of inference results pertains to the process by which multiple outputs from different nodes are synthesized to produce a final, official outcome for an AI inference task. Nesa employs a default majority vote strategy within a smart contract to tally node submissions and determine the consensus result. Nodes that align with the majority are rewarded, while nodes with faulty submissions are penalized. This method reinforces the reliability and validity of the aggregated inference outcomes.

AIT kernel

The AIT kernel is a complete package that includes the model parameters, configuration file, inference code, and aggregation code required for executing an AI model within the Artificial Intelligence Terminal (AIT). It encapsulates all necessary information and logic for nodes to correctly execute an AI model. The kernel is stored on the blockchain to ensure transparency, immutability, and verifiability of the model's execution environment.

Model Parameters

Model parameters are the set of weights and biases that characterize an AI model, essentially determining its behavior and prediction capabilities. These parameters result from the training process and dictate how the model processes input data to generate outputs. Within the AIT system, model parameters play a vital role in achieving consistent execution and consensus on AI inference results among different nodes.

AIT Configuration File

Similar to a Dockerfile, the AIT configuration file outlines the specifications for the virtual environment needed to run a model on the AIT. It lists dependencies, libraries, and runtime details, ensuring that each node sets up an identical execution environment. This file is essential for maintaining the uniformity and reproducibility of model execution on the decentralized network.

Inference Code

The inference code refers to the actual algorithmic logic executed by the AI model to process inputs and deliver predictions or other outputs. Combined with model parameters, the inference code conducts AI tasks within the AIT and plays a crucial role in interpreting input data and producing consistent and accurate results for users' queries.

Aggregation Code

The aggregation code in the AIT ecosystem is a script that determines how results from different nodes are consolidated to reach a consensus within the decentralized virtual machine. It forms an integral part of the AIT kernel and is vital in synthesizing outputs from various executions to provide a singular, verifiable outcome for AI tasks.

Decentralized Storage Solutions

Decentralized Storage Solutions like InterPlanetary File System (IPFS) and Arweave are employed by Nesa to store AIT kernels in a distributed manner. Such solutions offer a resilient platform for data storage that is resistant to censorship and data loss, ensuring the persistent availability and accessibility of AI models and execution environments.

On-chain Model and AIT Repository

The On-chain Model and AIT Repository constitute a decentralized management system for storing and securing AI models along with their virtual machine configurations. It acts as a discoverable library that allows users to interact with a range of AI models and supports the responsible handling of private or proprietary models through encryption and key management processes.

Privacy for Proprietary Models

Within Nesa's ecosystem, proprietary models are protected through encryption before being stored on decentralized platforms. This privacy protection ensures that

confidential AI models remain inaccessible to unauthorized parties, with decryption keys controlled and distributed at the discretion of the model owner, fostering a secure and trustworthy AI marketplace.

AIT Interface for Model Interaction

The AIT frontend interface is a user-facing platform that facilitates interaction with the on-chain AI model repository. It provides functionalities such as model browsing, uploading, deployment, and real-time monitoring—enabling users across various expertise levels to work with AI models effectively and with ease.

Decentralized inference

Decentralized inference delineates a process wherein AI computations are undertaken across a distributed network of nodes, ensuring a trustless environment where results are transparently reported on-chain. This method allows for AI models to be utilized in a decentralized way, maintaining user privacy and avoiding centralized points of control or bias.

Inference Committee Selection

The Inference Committee Selection process involves deciding on a group of nodes (the committee) that will handle a particular AI inference task on Nesa. It uses VRF to ensure fair and random selection, as well as to verify the committee's composition, bolstering the security and impartiality of the decentralized inference system.

Free-Riding Prevention

Free-riding prevention mechanisms are deployed to deter nodes from profiting from the efforts of others without contributing to the computation work. These mechanisms include a commit-reveal protocol that obligates nodes to demonstrate their own computations before gaining access to the results, promoting fairness and contribution within the decentralized inference network.

Homomorphic Encryption (HE)

Homomorphic Encryption is a type of encryption that allows for computations on encrypted data without requiring access to the decryption key. Nesa's hybrid-privacy system considers the use of HE techniques for carrying out secure computations on private user data within the TEEs of its decentralized AI platform, strengthening the

privacy-preserving capabilities of the project.

Secure Multi-Party Computation (SMPC)

SMPC is leveraged in Nesa's infrastructure for executing secure computations among multiple parties where the data inputs remain private. This cryptographic protocol is valuable for collaborative tasks in a decentralized environment where privacy is of paramount importance, without having to compromise on the integrity of the computations.

Zero-Knowledge Proof (ZKP)

ZKP is a cryptographic technique used by Nesa to enable the verification of the correctness of computations without exposing the underlying data. It helps maintain confidentiality while providing proof that the system's computations are accurate, encouraging trust and security in the decentralized inference system.

Threshold Cryptosystem

A threshold cryptosystem is utilized by Nesa for the secure distributed management of cryptographic keys. In such a system, a secret key is split into multiple shares, with no single party having access to the entire key. Nesa employs this system to ensure secure collaborative decryption during AI model inference tasks, without exposing sensitive data.

Large Language Models (LLMs)

Large Language Models, such as those developed by OpenAI (e.g., GPT), are AI algorithms designed to process and generate human language in a context-aware and nuanced manner. Nesa uses an enhanced version of LLM, optimized for the Web3 environment, to provide smart contract interactions with access to sophisticated language-processing capabilities.

$NES

The native cryptocurrency token used within the Nesa ecosystem. $NES serves various functions such as paying for transaction fees, staking for network security, and participating in governance decisions. It underpins the economic model of Nesa, integrating economic incentives with network operations for AI model inference.

AIT Kernel Marketplace

The AIT Kernel Marketplace envisaged by Nesa represents a future platform where containerized AI models can be publicly traded and monetized. This open marketplace aims to empower creators with the ability to own, update, and sell their AI iterations, promoting a robust ecosystem for AI model evolution and interaction in a decentralized fashion.

Threshold Decryption

Threshold decryption is a cryptographic technique used to decrypt information in a distributed manner where no single participant can reconstruct the complete decrypted data. This method is integral to Nesa's privacy-preserving system and is a key component in securely handling inference on encrypted data without disclosing it to any unauthorized entities.

Utility Suite

A collection of tools and services from external Web3-AI partners provided through Nesa, tailored for various stages of AI development. These include computational resources like TPUs and GPUs, model databases, APIs, and other tools necessary for training AI, such as data oracles and governance platforms, which aid in the smooth and efficient evolution of AI systems.

NANs

NANs are an AI-based jury system within the Nesa infrastructure that evaluate the reliability and performance of kernels by administering a series of tests designed to simulate various operating conditions and uncover any inconsistencies in execution. NANs concentrate on key performance indicators which may include computational efficiency, ethical adherence, data precision, susceptibility to generate erroneous or creative outputs, and compliance with established model protocols. They employ an adversarial evaluation framework to challenge the kernels in unpredictable ways, ensuring they behave consistently and according to the predefined configurations before being cataloged on the blockchain.

Split-Flow

The Split-Flow Protocol dynamically partitions tasks into confidentiality and verifiability streams that operate using either TEEs for heavyweight operations, cryptographic

methods like SMPC for lightweight tasks, or a hybrid of the composites depending on requirements. ZKPs are employed within the verifiability stream to ensure the accuracy of computations without compromising data privacy. The protocol's ability to adjust in real-time to the specifics of computational tasks makes it particularly effective in balancing security, efficiency, and verifiability in Nesa's decentralized inference systems.

Model Version Control

Model Version Control is a process and system within Nesa's decentralized AI framework that enables tracking and management of different versions of AI models and corresponding datasets. It resembles version control systems in software development, like Git for AI training processes, and ensures all model updates are duly logged and managed.

On-chain Fork Management

Nesa's on-chain fork management refers to the protocols established to handle branches in AI model evolution. It ensures that as models fork and merge in their development paths, these changes are accurately reflected on-chain. This system allows for complex evolutionary processes, including adaptation and experimentation on AI model updates.

TEE (Trusted Execution Environment) Implementations

Nesa relies on various TEE implementations, including Intel's Trusted Domain Extensions (TDX), AMD's Secure Encrypted Virtualization with Secure Nested Paging (SEV-SNP), ARM's Confidential Compute Architecture (CCA), and NVIDIA's Hopper H100 GPU architecture. Each provides an additional layer of security and privacy, allowing confidential computations to occur within the Nesa ecosystem.

SMPC (Secure Multi-Party Computation)

SMPC on Nesa is utilized to execute confidential computations collaboratively, ensuring participating parties cannot access others' data input. The implementation is optimized for performance, allowing the network to facilitate secure interactions without compromising efficiency.

VRF (Verifiable Random Function)

Nesa uses VRF to provide unbiased, unpredictable, and verifiable random values

for network processes such as node selection. It ensures that all elements of randomness within the system are fair and transparent, fortifying the security and integrity of the decentralized AI platform.

Proof-of-Stake Consensus

Nesa operates a Proof-of-Stake (PoS) consensus mechanism, where network validators and their delegators secure and maintain the blockchain by staking $NES tokens. This consensus model provides network governance and collaborative decision-making, contributing to the platform's democratization and distributed trust.

Dynamic Pricing Model

Nesa's dynamic pricing model refers to the adaptable fee structure for AI model inference queries. It aligns resource allocation with market demand, enabling users to prioritize their requests by paying higher fees. This model ensures efficient system utilization and optimizes network throughput by managing the queuing of inference tasks based on variable pricing.

Erasure Coding and Merkelization

Nesa employs erasure coding and Merkelization techniques for data redundancy and security, ensuring that AI model data remains intact and tamper-proof as it is stored and processed across its decentralized network.

Model Evolution Attestation

Model evolution attestation is a verification process within Nesa that ensures the integrity of AI model updates. Validators within the network attest to the authenticity and proper execution of model evolutions, providing a trustless environment for model development and deployment.

Namespace-specific Storage Management

Nesa uses a unique storage management system that allocates space for models and data within specific namespaces. This allows for organized and scalable storage solutions tailored to the needs of different AI models and their associated datasets.

queryStream

QueryStream is a protocol function within Nesa designed to facilitate the on-chain querying of AI models by streaming verification layer data to Nesa's immutable ledger or another blockchain like Ethereum. This method ensures the data's integrity through a peer-to-peer network and smart contracts that execute model evolutions end-to-end. QueryStream's P2P relayer component serves as a bridge, streaming query results to the relevant blockchain's settlement layer for final confirmation and execution.

Byzantine Fault Tolerant (BFT) Consensus Mechanism

Nesa's consensus mechanism ensures that even in the presence of faulty or malicious nodes (byzantine failures), the network can reach consensus and operate correctly. This enhances the network's security and resilience.

Validator Nodes

Nodes on a blockchain network with the responsibility to validate transactions, produce new blocks, and maintain the integrity of the network. Validators participate in the network's consensus mechanism and, in some blockchains, are required to stake cryptocurrency as a form of security and commitment to their role.

Intrinsic Rank in AI

Nesa's LLMs utilize the concept of intrinsic rank which postulates that the actual number of parameters needed to perform a task effectively is often much lower than the size of the model itself, allowing for efficient adaptations with minimal parameter changes.

PayForQuery

PayForQuery is a transaction type on Nesa that enables developers to pay for the querying of AI models, with the cost being determined by the complexity and resources needed for the transaction. It capitalizes on Nesa's Byzantine Fault Tolerant signature mechanism to ensure the validity and security of the AI model data queries on the chain. The PayForQuery transactions are essential for the verifiable execution of AI model inferences, allowing for cryptographic validation and submission of data to Nesa or interchain ecosystems like Ethereum.

Chapter 13

Disclaimer

This whitepaper ("Whitepaper") is for informational purposes only and does not constitute an offer to sell, a solicitation to buy, or an endorsement, recommendation, or sponsorship of any product, service, or company. This Whitepaper is not intended to constitute legal, tax, accounting, investment, or other professional advice or services. It is necessary to consult with a qualified professional advisor before making any decision based on information contained in this Whitepaper.

Legal

The concepts and technologies discussed in this Whitepaper are in the development stage and may undergo significant changes, delays, or discontinuation. The creators of the Nesa project make no representations or warranties as to the successful development or implementation of such technologies and concepts, or achievement of any other activities noted in the Whitepaper, and disclaim any warranties, implied or otherwise, regarding the same.

Nothing in this Whitepaper shall be considered a binding agreement or partnership. Participation in the Nesa project, including the acquisition and use of $NES tokens, the interaction with any related software, or engaging with any platforms or services described herein, is at the participant's own risk.

Investment

Cryptocurrencies, tokens, and blockchain-based assets are highly volatile and speculative investments. $NES tokens are no exception. The value of $NES tokens can fluctuate greatly within short periods of time, and there is no guarantee of any return on investment. The purchase of $NES tokens should only be considered by those who are experienced with and willing to accept the high risks associated with such investments. Potential purchasers should conduct their own due diligence and consult with their financial advisors before making any investment decisions.

Crypto

The regulatory status of cryptocurrencies, tokens, blockchain technology, and services related to such technologies is uncertain and may rapidly evolve. Participants are solely responsible for ensuring that their interaction with the Nesa project complies with all applicable laws and regulations of their jurisdiction. The creators of the Nesa project will bear no responsibility for anyone's non-compliance with such laws and regulations.

No part of this Whitepaper is intended to create legal relations between a recipient

of this document or reader of its content and the creators of the Nesa project. It is the responsibility of prospective participants to inform themselves of, and to observe, all applicable laws and regulations of relevant jurisdictions.

This Whitepaper is not intended for distribution to, or use by, any person or entity in any jurisdiction or country where such distribution or use would be contrary to law or regulation, or which would subject the creators of the Nesa project, or its affiliates to any registration requirement within such jurisdiction or country.

By accessing this Whitepaper, the reader acknowledges and agrees that the creators of the Nesa project shall not be held liable for any direct, indirect, incidental, special, exemplary, or consequential damages, including but not limited to, procurement of substitute goods or services, loss of use, data, or profits, or business interruption, however caused and on any theory of liability, whether in contract, strict liability, or tort (including negligence or otherwise), arising in any way out of the use of this Whitepaper, even if advised of the possibility of such damage.

- -

Nesa - The layer-1 for AI.

Website Nesa

nesa.ai

Gitbook Nesa

docs.nesa.ai

Discord

@nesaorg

Twitter

@nesaorg

Email

hi@nesa.ai