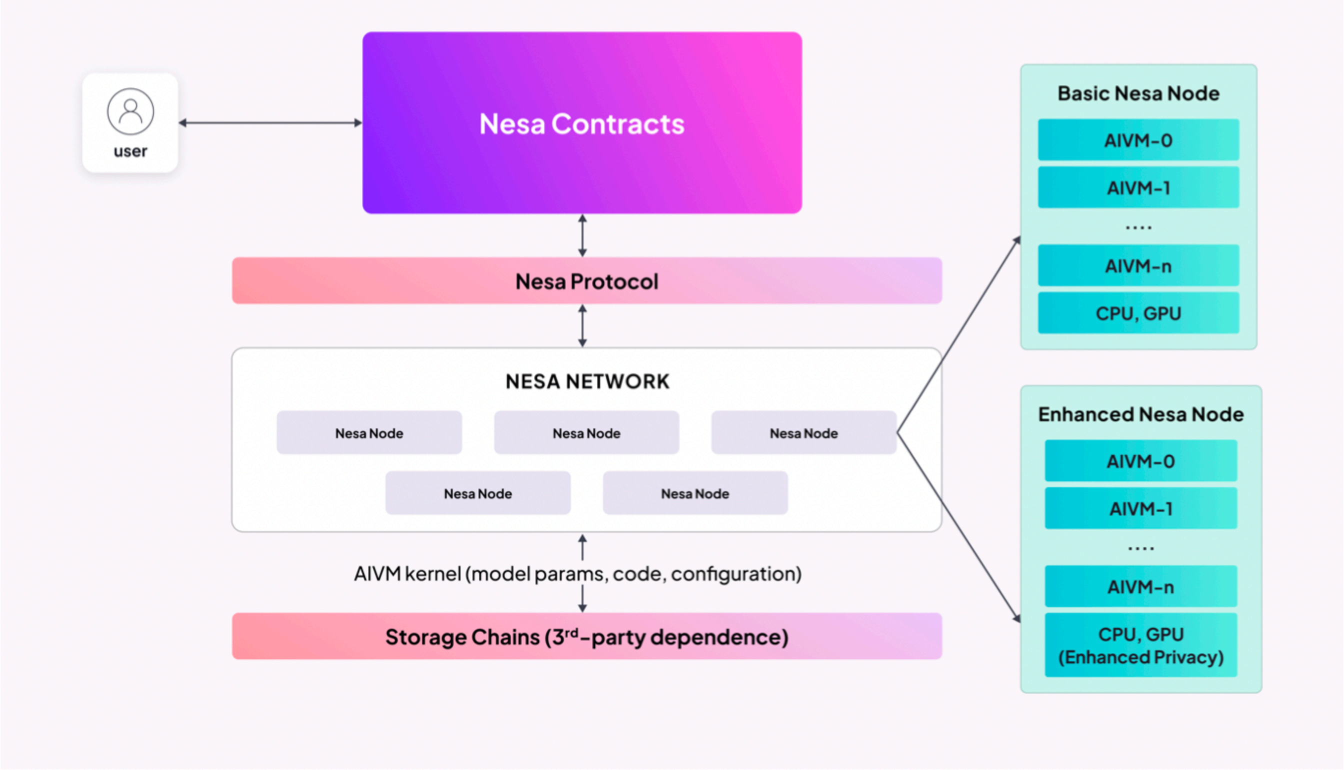
AIVM Architecture and Design

The Artificial Intelligence Virtual Machine (AIVM) 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 AIVM is specifically optimized for the complexities and nuances of AI model inference, both large and small.

At its core, the AIVM is designed to execute AI models uniformly, ensuring 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.



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, which 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 AIVM, 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 preferences. The AIVM kernels are stored on the chain for decentralization and end-to-end model updates.

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

- **Model Parameters** 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.
- **AIVM Configuration File** functions similar to a Docker file but specific to the AIVM, 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** is 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 AIVM.
- **Aggregation Code** is a piece of scripting code that determines how the decentralized VM will aggregate and reach consensus from results returned from different nodes.

These components together form what we call the AIVM 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 AIVM kernel is stored on the blockchain, providing an immutable and verifiable record of the model's execution environment and logic.

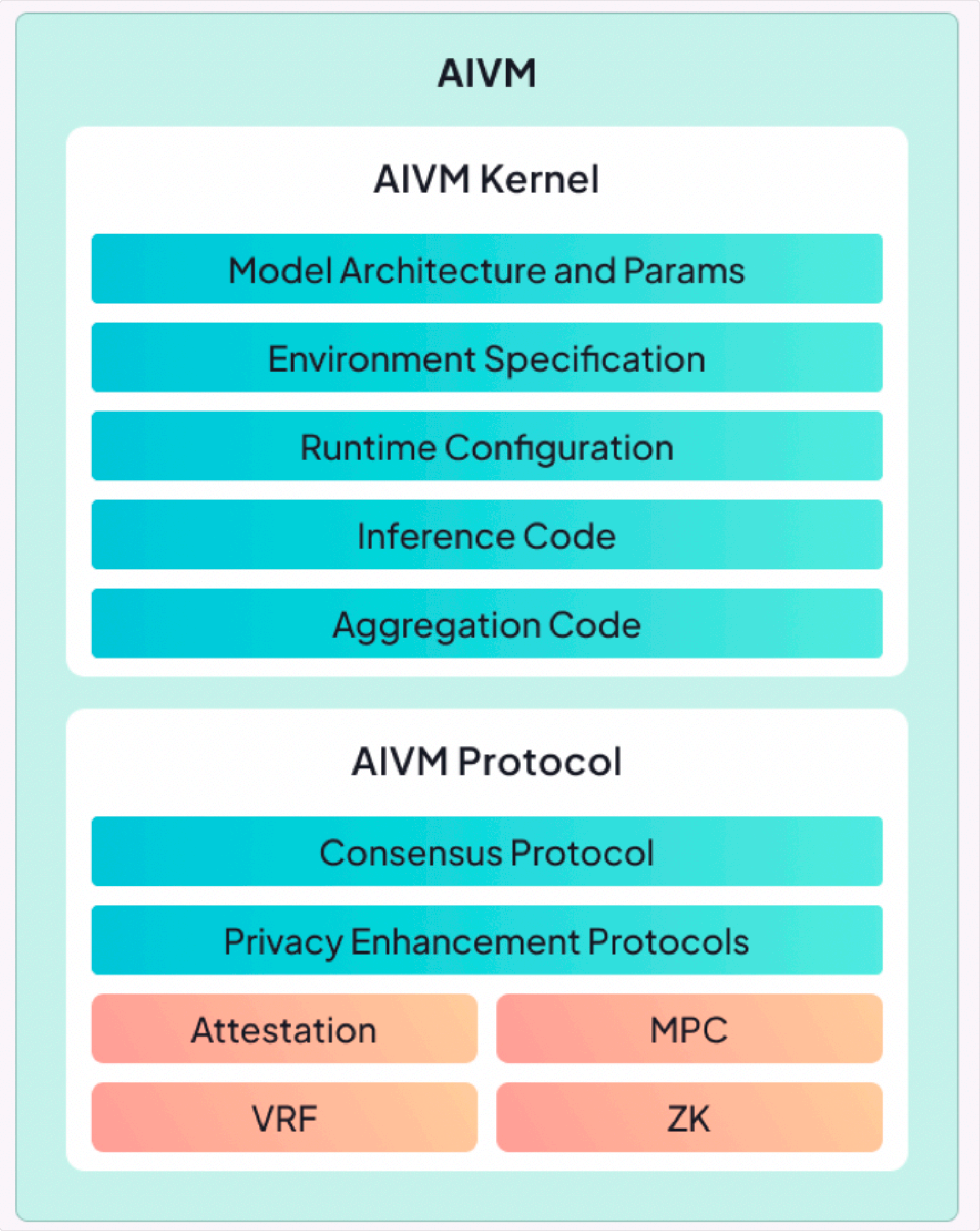
In addition to the kernel, a fully operational AIVM also contains various protocols that support decentralization and security. These protocols will be discussed in later chapters. The full architecture is depicted in the image below.

The AIVM 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 and run an AIVM kernel 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 AIVM include:

- **Isolation.** The AIVM 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 AIVM's environment ensures that models can be executed reliably and with the same results across different nodes.
- **Security.** The AIVM 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 AIVM aims to provide a full-fledged execution environment, it is designed to be lightweight and lean so as not to impose significant overhead on the node's resources.



AIVM Architecture. The AIVM contains two core components: the AIVM Kernel and the AIVM 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 AIVM 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 AIVM Configuration File, a file functionally similar to a Dockerfile that details the depen- dencies, libraries, and runtime needed to run the model to ensure identical execution environment. The AIVM Protocol's primary purpose is to facilitate communication among the nodes (mostly for security and privacy guarantee) and is packaged in soft- ware provided by us. The AIVM Protocol includes the consensus protocol and privacy enhancement protocols, including Nesa's security mechanisms

The AIVM 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 AIVM, we ensure that our platform remains open, secure, and accessible, fostering a thriving ecosystem of shared AI capabilities.

Previous
AIVM - The Artificial Intelligence Virtual Machine

Next
Model Consistency and Inference Reliability

Last updated 1 month ago