# Arcium Purplepaper: The Encoded Supercomputer

Yannik Schrade Leopold Joy Daniel Filipe Nunes Silva

Nicolas Le Bel Lukas Steiner Arihant Bansal

Nicolas Schapeler

May 11, 2026

**Abstract**

*This abstract should be read as an introduction to the crypto-asset white paper. The prospective holder should base any decision to purchase this crypto-asset on the content of the crypto-asset white paper as a whole and not on the summary alone. The offer to the public of this crypto-asset does not constitute an offer or solicitation to purchase financial instruments and any such offer or solicitation can be made only by means of a prospectus or other offer documents pursuant to the applicable national law. This crypto-asset white paper does not constitute a prospectus as referred to in Regulation (EU) 2017/1129 of the European Parliament and of the Council (36) or any other offer document pursuant to Union or national law.*

The Arcium Network introduces a decentralized, confidential compute protocol powered by secure Multi-Party Computation (MPC) and a bespoke token-economic model. By combining state-of-the-art MPC protocols with the latest blockchain infrastructure, Arcium enables scalable, trustless execution of confidentiality-preserving computations—from confidential voting to federated machine learning.

At the core of the Network lies the ARX token, a fixed-supply utility token used to align node operator incentives with the Network's computational throughput. Node operators stake ARX to unlock hardware resources proportional to their attested computational capabilities, aligning infrastructure growth with real-world Network demand. Computation Customers pay a base price plus a customer-defined Priority Fee for each computation, with all such fees accruing entirely to the Arx Nodes that execute the work.

Arcium's disintermediated architecture and crypto-economic incentive design enable a self-reinforcing ecosystem where participants—node operators, delegators, and Computation Customers—collectively drive Network efficiency. By anchoring token utility to physical hardware provisioning and on-chain governance, Arcium pioneers a paradigm where on-demand confidential compute becomes both economically sustainable and universally accessible.

**v0.1.0**

Key innovations include Multi-Party eXecution Environments (MXEs) enabling self-contained Byzantine fault tolerant MPC execution, on-chain computational priority fee markets, and a fixed-supply utility token whose demand is tied to Network usage via staking. The result is a Network poised to scale from niche cryptographic operations to enterprise-grade confidential computing, all governed by transparent, incentive-aligned mechanics.

# Contents

# Disclaimers

This crypto-asset white paper has not been approved by any competent authority in any Member State of the European Union. The offeror of the crypto-asset is solely responsible for the content of this crypto-asset white paper.

This crypto-asset white paper complies with Title II of Regulation (EU) 2023/1114 and, to the best of the knowledge of the management body, the information presented in the crypto-asset white paper is fair, clear and not misleading and the crypto- asset white paper makes no omission likely to affect its import.

The crypto-asset referred to in this white paper may lose its value in part or in full, may not always be transferable and may not be liquid.

The utility token referred to in this white paper may not be exchangeable against the good or service promised in the crypto-asset white paper, especially in the case of a failure or discontinuation of the crypto-asset project.

The crypto-asset referred to in this white paper is not covered by the investor compensation schemes under Directive 97/9/EC of the European Parliament and of the Council. The crypto-asset referred to in this white paper is not covered by the deposit guarantee schemes under Directive 2014/49/EU of the European Parliament and of the Council.

# 1 Token Offeror and Issuer Information

All information regarding the issuing entity, its legal form, management, corporate governance, and all other required disclosures under Annex I of MiCAR "DISCLOSURE ITEMS FOR THE CRYPTO-ASSET WHITE PAPER FOR A UTILITY TOKEN" Part A, are contained within the following sections.

| | |
|---|---|
| Name | Arcium Association |
| Legal Form | Association under Swiss law |
| Registered Address | Grabenstrasse 25, 6340 Baar, Switzerland |
| Date of Registration | 8 July 2022 |
| Identifier | CHE-355.161.855 |
| Telephone | [number] |
| Email | contact@arcium.com |
| Response Time | 1-2 business days |
| Issuer business | IT, Software development |

## 1.1 Financial Condition

The Arcium Association was established in mid-2022, which allows for representation of its financial condition over the span of a first overlong business year starting in mid-2022 until the end of 2023, instead of the required three years as per Annex 1, Article 10.

The Arcium Association's financial condition for the period of 07.2022-12.2023 is surmised to have been satisfactory with the company starting operations, establishing its headquarters, management board, and governance. The starting capital was from investment from the founding members, who also form the management board. Additional funds in the total amount of USD 9 million were raised via a seed and then strategic round with investors.

The detailed report on the financial condition of the Arcium Association for its first overlong business year can be found here:

[insert financials; for now, you can just add a placeholder]

### 1.2 Core Business

The Arcium Association promotes and fosters the development and adoption of the Arcium Network. The Arcium Network is a confidential computing network enabling secure data collaboration. By leveraging a distributed Multi-Party Computation (MPC) architecture, it ensures data remains encoded throughout processing, preventing single points of failure and safeguarding against cyber threats.

The network's on-chain orchestration manages computation scheduling, compensation, and security enforcement. Nodes form clusters to execute secure computations, integrating seamlessly with blockchain systems to enhance efficiency while preserving confidentiality.

Arcium's flexible configuration allows users to customize their MPC setups for specific security needs. By prioritizing confidentiality and decentralized trust, Arcium is redefining secure data utilization across industries.

### 1.3 Management Body

Yannik Schrade, CEO, c/o Arcium Association, Grabenstrasse 25, 6340 Baar, Switzerland

Lukas Steiner, COO, c/o Arcium Association, Grabenstrasse 25, 6340 Baar, Switzerland

Nicolas Schapeler, CTO, c/o Arcium Association, Grabenstrasse 25, 6340 Baar, Switzerland

Julian Descher, CSO, c/o Arcium Association, Grabenstrasse 25, 6340 Baar, Switzerland

## 2 Introduction

### 2.1 The Need for Decentralized Confidential Compute

Data exists in three states: at rest, in transit, and in use. While encoding for data at rest (storage) and in transit (transmission) is well-established, data

in use—actively being processed—remains dangerously exposed. Traditional approaches to securing runtime data rely on hardware-based Trusted Execution Environments (TEEs), which isolate computations in secure enclaves. However, TEEs suffer from critical flaws:

- **Hardware Centralization**: TEEs (e.g., Intel SGX, AMD SEV, etc.) depend on proprietary hardware, creating centralized chokepoints. Vulnerabilities like Spectre, Meltdown, and Plundervolt[1] have repeatedly compromised TEE integrity.
- **Trust Assumptions**: TEEs require users to trust both hardware manufacturers and third-party attestation services, introducing significant trust assumptions and centralization.
- **Performance Limitations**: Scaling TEEs across distributed systems introduces latency and coordination overhead, making them impractical for real-time, high-throughput applications.

Fully Homomorphic Encoding (FHE), another cryptographic approach, allows computations on encoded data without decoding. Yet FHE's computational overhead—often 1,000x slower than plaintext operations—renders it unusable for most applications.

Arcium addresses these shortcomings by pioneering decentralized confidential computing (DeCC) through Multi-Party Computation (MPC). Unlike TEEs or FHE, Arcium's MPC-based framework:

- **Eliminates Hardware Reliance**: Computations are secured cryptographically, not through vulnerable hardware enclaves.
- **Distributes Trust**: No single entity controls the computation. MPC protocols split data across nodes, ensuring no single party accesses plaintext.
- **Matches Plaintext Speeds**: Parallel execution across Arcium's Clusters achieves near-native performance, unlike FHE's prohibitive latency.

Arcium's architecture advances MPC beyond niche use cases:

- **Parallel Execution**: MXEs (Multi-Party eXecution Environments) enable atomic, parallelized computations with dedicated states, bypassing bottlenecks of global consensus.
- **Configurable Trust Models**: Developers capitalize on tailored MPC protocols, enabling them to strike a balance between considerations of speed, security, and decentralization—e.g., healthcare applications might prioritize Byzantine fault tolerance, while gaming uses lighter protocols.

[1] S. van Schaik, A. Batori, A. Seto, B. AlBassam, C. Garman, T. Yurek, A. Miller, D. Genkin, E. Ronen, and Y. Yarom, *SGX.Fail: Epic Fail of SGX Security*, 2023. Available at: https://sgx.fail.

- **Chain-Agnostic Encoding**: arxOS has the capacity to integrate with any blockchain or off-chain system, encoding data flows across fragmented ecosystems.

This shift from hardware-dependent TEEs to cryptographic MPC democratizes confidential computing. Financial institutions can run risk analyses on encoded datasets, AI models can be trained on consented, encoded enterprise datasets without exposing raw inputs, and decentralized autonomous organizations (DAOs) vote securely—all without relying on centralized hardware vendors. By encoding the final frontier of runtime data, Arcium transforms the internet into a confidentiality-first infrastructure.

## 2.2 Arcium's Vision

Arcium envisions a future where data confidentiality is seamlessly integrated into the fabric of digital infrastructure, enabling trustless, performant computation across industries without compromising confidentiality. By replacing centralized intermediaries and vulnerable hardware enclaves with decentralized MPC and transparent, stake-aligned crypto-economic incentives, Arcium transforms encoded compute from a niche tool into a universal utility. The Network empowers developers, enterprises, and individuals to process sensitive data—financial records, healthcare analytics, AI models, and beyond—with cryptographic guarantees, while aligning incentives for node operators, delegators, and users through the fixed-supply ARX token model.

Beyond securing data-in-use, Arcium bridges Web2-scale computation with Web3's permissionless innovation. By abstracting cryptographic complexity into configurable MXEs and integrating with the latest blockchain infrastructure (e.g. Solana), the Arcium Network lowers barriers for developers to build confidentiality-first applications. This vision culminates in an encoded supercomputer: a globally distributed, self-sustaining ecosystem where computational power scales dynamically with demand, and sensitive operations execute in parallel across chains, jurisdictions, and industries—all without exposing a single byte of plaintext.

# 3 Architectural Overview

## 3.1 Core Components

### 3.1.1 Arx Nodes

An Arx Node is an individual compute provider that contributes to collective MPC processes. Nodes participate in multiple computations concurrently via Clusters (see section 3.1.2). For each process that an Arx Node participates in, it holds a fragment of the overall data being computed—a keyshare for the given encoded data. Nodes then collaboratively perform computations on these fragments without revealing their individual pieces of data to one another. This

ensures the confidentiality and security of the data while enabling the Network to compute a function over an entire dataset. Nodes collaborate via secure MPC protocols (see section 3.2) to share computation results, enabling the Network to effectively derive a publicly cryptographically-verifiable final output without any single node gaining access to the original complete dataset.

#### 3.1.2 Clusters

In the Arcium Network, Clusters are groupings of Arx Nodes that collectively execute computations. Clusters are created by Computation Customers (see section 4.2), with the set of nodes being selected based on their unique properties, including their historical reputations of trustworthiness on the Network, their available computational resource capacities, as well as other factors. Clusters also can undergo migration in the event that one or more nodes become unavailable or are deemed untrustworthy, with new nodes being selected to replace them (see section 8.5).

#### 3.1.3 Multi-Party eXecution Environments (MXEs)

MXEs are isolated, dedicated virtual environments for secure computation execution. They are highly configurable, enabling fine-tuning of data provisioning and handling, encoding schemes, and more. Created by Computation Customers, MXEs are MPC protocol-agnostic, and run on top of Clusters—either a particular chosen Cluster or rotating between multiple pre-selected Clusters based on their availability. MXEs are defined as either "persistent", meaning that they can be reused indefinitely into the future, or "single-use", indicating that the MXE may only be used for one computation and then automatically discarded.

#### 3.1.4 Computations

In the Arcium Network, a computation is a discrete, standalone operation to be computed by an MXE. Computations are first created as "definitions" by Computation Customers, each consisting of an associated logic circuit definition, the parties granted execution authority over the definition (e.g. individuals, on-chain contracts, etc.), input and output data handling configurations, callback settings (e.g. to schedule future computations, execute functions in other on-chain programs, etc.), as well as other attributes (see section 10.2 for more on defining computations). Each Computation Definition must be associated with exactly one MXE. Individual computations (calls) are always based on a pre-configured Computation Definition.

#### 3.1.5 On-Chain Contracts

All orchestration of the Arcium Network is handled via the blockchain, with smart contract programs (e.g. on Solana) handling all of the on-chain mempool organization and computation ordering, rewards calculations and distributions,

token staking and slashing, as well as many other organizational responsibilities. All network participant interactions are coordinated via the Arcium Network's on-chain programs. The Arcium Network leverages existing blockchain infrastructure (e.g. Solana) as its consensus layer, removing the need for an independent layer-1 solution for state and consensus management. This architecture enables full parallelization within the Arcium Network by offloading consensus to the blockchain, allowing network orchestration tasks to scale efficiently.

Arx Nodes can be thought of as off-chain workers, with the on-chain Arcium Network programs functioning as the organizational hub, collecting computations in its on-chain mempools, designating them out to Clusters of Arx Nodes for execution, and then handling the results and payments.

### 3.1.6 Computation Units (CUs)

The Computation Unit (or CU) is a Network-wide constant that represents a fixed amount of computational work. The CU is the smallest unit of computation in the Arcium Network, and all types of arithmetic operations on the Arcium Network (that computations are made up of) are measured in CUs.

### 3.1.7 Epochs

Timing on the Arcium Network is tracked and handled via Epochs. Epochs are of a consistent duration, serving as a timing mechanism for the execution and scheduling of work, the distribution of rewards to Arx Nodes, the lock-up periods of staked tokens, and more.

## 3.2 Multi-Party Computation (MPC) Protocols

Initially the Arcium Network supports two different MPC protocol backends, nicknamed Cerberus and Manticore. These two backends make different tradeoffs and enable a wider range of applications. The Arcium Network is designed to be extensible and protocol-agnostic, and additional MPC protocol will be added in the future.

As an example, it would be possible to train a machine learning model using Manticore since it is a computationally intensive task, and then use Cerberus to make predictions using this newly trained model.

The Cerberus and Manticore protocols natively execute circuits authored in Arcis (see section 10.2.1), leveraging its type system for efficient secret-sharing and verifiable output generation. Developers, using circuits written in Arcis, can seamlessly toggle between protocols based on application requirements (e.g., Byzantine resilience vs. honest-but-curious assumptions).

### 3.2.1 Cerberus

Cerberus is Arcium's main backend for MPC computations. Each share is authenticated with a Message Authentication Code (MAC), allowing honest nodes to verify integrity and detect tampering, ensuring that computations are aborted

if malicious behavior is detected. Under the assumption that at least one node in the given Cluster is honest, the computation's result is guaranteed to be correct. Cerberus provides the strongest security guarantees of the two initially supported MPC protocols.

#### 3.2.2 Manticore

Manticore operates in the "honest but curious" setting. Furthermore, it also relies on a "trusted dealer", a node that generates the preprocessing data needed for the online phase and then goes offline. This leads to faster execution time, enabling more complex operations typical of machine learning and artificial intelligence (AI) applications. However, this speed comes at the expense of weaker security guarantees. Manticore is designed to be deployed in settings where the nodes in the given Cluster are trusted operators and have an incentive to behave honestly (see section 8.7).

### 3.3 ArxOS: Distributed Execution Engine

ArxOS is a semantic distinction referring collectively to the Arx Nodes, Clusters, and MXEs that form the execution layer of the Arcium Network. This term encapsulates the decentralized coordination of computational resources, where:

- **Arx Nodes** act as distributed processing cores (see section 9).
- **Clusters** operate as dynamically assembled compute groupings, each with its own unique configuration and capabilities (see section 8).
- **MXEs** define encoded execution environments (see section 7).

ArxOS is not a standalone system but a conceptual framework for describing how these components interoperate to power confidentiality-preserving computations. It abstracts the orchestration of MPC protocols, resource allocation, and parallelized task execution, mirroring the role of an operating system in a traditional computer—but decentralized and trustless.

## 4 Network Participant Types

The Arcium Network consists of three participant types who all engage with the Network economically to either obtain or provide services.

### 4.1 Arx Node Operators

Arx operators are the primary service providers of the Arcium Network, operating the Arx MPC node software on their physical hardware infrastructure in order to participate in the execution of computations, earning rewards proportional to the computational jobs they complete. When launching an Arx Node, the operator specifies the amount of computational resources their node

is capable of providing to the Network, however, in order for these hardware resources to become eligible to do work, the total amount of collateral staked to the Arx Node must be proportional to its computational resources (see section 5.4). Operators also specify details of the available security features of their Arx Node configurations (e.g. TEE support). In addition to available computational resources and node features, Computation Customers also select Arx Nodes (in Clusters) based on their historical reputations (see section 5.5). Thus, an Arx operator focuses on designing and operating a maximally reliable service to maintain their reputation in order to receive a continual flow of computational work. Reputation is assessed off-chain by Computation Customers based on, for example, historical slashing record, completed computation statistics, etc. To further grow their business, an operator can opt to increase the amount of infrastructure resources that they provide to the Network.

Arx Node operators receive two forms of compensation for the computational work they perform: protocol-level rewards corresponding to their self-delegated capacity, and operational fees paid for hardware-resource activation services provided to third-party delegators (see section 4.3). The obligatory minimum self-delegation of stake is used to cover the costs of planned migrations if needed (see section 8.5.2).

## 4.2 Computation Customers

As the Network's buy-side participant type, Computation Customers purchase bespoke confidential computing services from Arx Nodes operating in Clusters. To access computing services, a Computation Customer must first create an MXE (see section 3.1.3) and then create Computation Definitions containing the circuits and details of the operations that they plan to have computed (see section 10.2 for more on defining computations). Customers may then order the execution of computations in accordance with their predefined computation types by paying the corresponding computation fee which consists of a Network-defined base cost, plus a customer-defined priority fee, enabling customers to have their computations prioritized at additional costs (see section 6.1).

## 4.3 Third-Party Delegators

Each third-party delegator delegates their stake to an Arx Node operating on the Arcium Network in order to receive a pro-rata share of the Arx Node's rewards from executing computations. A rate-based fee, charged by Arx operators, is deducted from third-party delegators' rewards (since Arx operators assume the actual physical infrastructure operation costs). Third-party stake helps an Arx operator unlock their hardware, making it eligible to perform work. Furthermore, third-party stake is liable to be slashed in the event of misbehavior, meaning that third-party delegators assume the same collateral risks as Arx operators (see section 5.5).

Arx operators can optionally disable third-party delegations, effectively only permitting their own self-delegation to their Node. While the usage of this

feature may be rare, some node operators may prefer to make a single large self-delegation to their node, rather than collecting additional rewards via fees from third-party delegations (e.g. large institutions that prefer to only have slashing-liability for self-custodied collateral). In such cases, they may prefer to prevent third-party delegations, since their self-delegation may be large enough alone to exactly unlock the entirety of their available hardware.

# 5 ARX Tokenomics

## 5.1 The ARX Token

The ARX token is the utility token of the Arcium Network. The token has a fixed total supply set at genesis, with no protocol-level minting or burning mechanisms after launch. Demand for ARX is tied directly to use of the Network: node operators and third-party delegators must stake ARX to activate Arx Node hardware (see section 5.3), aligning token utility with the Network's overall computational throughput.

Compensation to Arx Nodes flows entirely from Computation Customer payments—a base price plus customer-defined Priority Fees (see section 6.1)—paid in the chain-native token. The protocol does not mint additional ARX to subsidize node operations, nor does it burn ARX from priority-fee flows. All Priority Fees accrue to the Arx Nodes that execute the corresponding computations.

## 5.2 Node Hardware Claim

Each Arx Node in the Arcium Network must publicly declare a hardware claim when it is created. This claim is defined as the number of computation units (or "CUs", see section 3.1.6) per unit of time that the node can handle at peak load. Arx Nodes are incentivized to make honest claims, since, if they underestimate, then the associated node will receive less computational work (and therefore fewer rewards), and if they overestimate, then the node can be slashed for non-participation if it is, for example, too slow to perform a large computation (or, for example, crashes when attempting to perform a computation that is too large for its available hardware configuration).

Arx Nodes may freely improve their hardware claims after launching. However, decreasing a node's declared hardware claim after launch may only be done if the node is either (a) not in any Clusters currently, or (b) willing to pay for the migration (from their self-delegated stake) of any of the Clusters that they are in where the lowering of their hardware claim decreases the overall performance of the Cluster such that the new overall performance falls below the Cluster's defined required maximum computational load (see section 8.1 for more details on the Computation Customer's definition of this limit).

## 5.3 Staking & Hardware Activation

Although each node operator must submit a node-specific claim (see section 5.2), this alone does not render its computational resources as available to the Network. Instead, delegations of ARX tokens must be made to an Arx Node in order to activate its computational resources. The more stake that a node receives, the more of its hardware unlocks, making it eligible to perform work for the Network. Insufficient stake makes an Arx Node ineligible to perform work using the entirety of its available computational resources.

To quantify and calculate stake activation of an Arx Node's hardware, a Network-wide constant ratio exists representing the amount of stake (both third-party and self-stake, see section 5.4 below for more on this distinction) required per unit of computational load capacity—this constant ratio is named `REQUIRED_STAKE_PER_UNIT_OF_CU_LOAD_CAPACITY`. This constant ratio is multiplied by the node's hardware claim in order to calculate the size of the total delegation required to attest to the node's claim. Given that the hardware claim defines the maximum possible computational load that an Arx Node can handle at peak usage, if a node receives less than the amount of total delegation that corresponds to its claimed hardware capacity, it is not proportionally assigned computational jobs up to its maximum possible capability (it can only be assigned an amount of work proportional to the stake it has delegated to it).

When the combined Arx Node delegations to a given node exceed the required amount to satisfy the `REQUIRED_STAKE_PER_UNIT_OF_CU_LOAD_CAPACITY` ratio, a node operator must either upgrade its hardware (in order to increase the amount of total delegation it can receive for its proportionally improved hardware claim) or rewards for additional stake delegations to the Arx Node in question are put through a near-linear curve—thereby enforcing a strong disincentive for over-delegation above the capacity of the hardware claim.

The Arcium Network staking approach disincentivizes centralization, by creating a near-linear relationship between the hardware that an Arx Node provides to the Network and the amount of stake required to activate it, meaning that even the most popular or reputable nodes can only receive stake proportional to the utility (computational resources) they provide to the Network.

## 5.4 Stake Delegation Mechanics

The total combined stake delegated to an Arx Node consists of both the operator's self-delegation, as well as third-party delegations from ARX token holders (see section 4.3). Each node is obligated to make a self-delegation of at least a specific minimum stake amount, this serves to cover the cost of any migrations should they occur (see section 8.5). While both self-delegated stake and third-party stake serve the function of unlocking a node's hardware claim (see section 5.2), the node operator's self-delegation also serves as an endorsement of confidence in their own ability to provide a reliable service, signalling, from a game theoretic perspective, their skin in the game. On the other hand, while third-party delegators do not directly operate node infrastructure, they do still

provide a service to the Network by delegating their ARX token capital to Arx Nodes that they deem to be trustworthy and reliable—since, like node operators, third-party delegators also assume the risk of potentially having their capital slashed in the event of node misbehavior. As such, third-party delegators are incentivized to actively engage in on-going due diligence of nodes, likely scrutinizing both on-chain and off-chain reputation (infrastructure setup, security practices, team reputation, etc., see section 9.4), and redelegating their stake to different nodes when deemed necessary.

## 5.5 Slashing

In the event of Arx Node misbehavior, all of a node's delegations (including the operator's self-delegation) are slashed by the same rate as punishment. The slash acts as a strong disincentive for misbehavior, and also provides a mechanism to pay for the costs associated with a failed computation (to compensate the other nodes that did not misbehave for their work).

### 5.5.1 Types

There are two types of punishments for Arx Nodes in the Arcium Network, non-participation (e.g. downtime or intentional inaction) and cheating (providing incorrect computation results). Both types are punished via slashing, however, non-participation receives a heavier punishment because the cost to detect non-participation is greater than that of cheating. Cheating is detected cryptographically, however, non-participation detection often comes with the increased cost associated with requiring a broader set of Arx Nodes (not just those in the computation's Cluster) to run a consensus protocol that verifies the withholding claim (see section 5.5.2).

Slashing penalties are rate-based, meaning that the amount slashed is always proportional to the size of the computation that was interrupted. As such, larger slashing penalties may be applied to misbehaving nodes with larger hardware specifications and therefore larger total stake delegated to them (since more stake is required to enable the hardware); thus, the increased stake functions as proportional collateral for increased slashing penalties (when needed).

### 5.5.2 Detection and Disputes

Since non-participation cannot be detected cryptographically (unlike cheating, see section 5.5.3), the Arcium Network's non-participation dispute resolution mechanism detects non-participation via a broader consensus of Arx Nodes. The protocol allows any participant to accuse another of non-participation, prompting the accused to prove their participation by sending a specific message to all peers within the Cluster, which then act as relays. If initial attempts fail, the accusation can be escalated to a broader set of Arx Nodes outside the original Cluster, ensuring broader verification and maintaining transparency. This strategy not only eliminates the possibility of censorship or disruptions but also

aligns node incentives with network health, promoting a stable and reliable computational environment.

#### 5.5.3 Cheater Detection

Another form of misbehavior for participants is sending incorrect data to each other. In such cases the receiving peer will detect the invalid data and trigger an abort of the protocol. In the classic BDOZ[2] MPC protocol for example, it is impossible to tell apart a legitimate abort from a malicious one, opening up the possibility of nodes using this mechanism to selectively censor computations with virtually no repercussions. Thankfully, protocols like these can be augmented such that attempted cheating can be cryptographically demonstrated to other peers as well as to third parties. Such an improvement (as exists in the Arcium Network's Cerberus protocol, see section 3.2.1), coupled with a slashing mechanism, as well as the knowledge that another Cluster can pick up the computation subsequently, any incentive to attempt censorship of a computation is all but removed.

# 6 Economic Mechanisms

## 6.1 Computation Pricing

The payments that MPC job Computation Customers (see section 4.2) make for computations consist of two different parts: the base price and the priority fee, both of which are split equally between all of the nodes contained in the Cluster that executes the computation. The base price is determined by a stake-weighted node operator vote and represents the minimum cost-covering rate to ensure that, at the very least, node operators can cover their operational costs resulting from participating in computations. On the other hand, the priority fee is set by the free market and determines the execution ordering of computations—higher Priority Fees can be set to expedite computation execution.

While the ARX token is used for Arx Node staking (see section 5.4 above), chain-native token (e.g. SOL on Solana) is used as the medium of payment for computations (both the base price costs and Priority Fees). Once Arcium becomes multi-chain (see section 13.1), node operators will earn job revenue across multiple chain-native tokens, each at a unique, dynamic token rate.

#### 6.1.1 Base Pricing

Execution costs must always be set to a minimally economically viable level since non-participation of Arx Nodes is impermissible. Set by the protocol, Base Pricing serves as a baseline to cover Arx Node operation costs in all market

[2]M. Bendlin, I. Damgård, C. Orlandi, and S. Zakarias, "Semi-homomorphic Encryption and Multiparty Computation," IACR Cryptology ePrint Archive, Report 2010/514, 2010. Available at: https://eprint.iacr.org/2010/514.

conditions. The base price itself is a unit rate, measured in CUs (see section 3.1.6), that is applied to computations—so differently sized computations will have different base costs resulting from the same base price.

Without Base Pricing, in the event that, at some point, insufficient computation-side demand exists on the Network, Arx Nodes could become under-compensated for computations. Unlike in a typical free-market-style decentralized network (e.g. delegated proof of stake chains like Solana, etc.), Arx Nodes cannot opt to not participate in the case that demand decreases (and thereby fees decrease), since this would be classified as non-participation and punished via slashing (see section 5.5). Furthermore, the cost to enter and exit the Arcium Network is substantial—due to lockup times as well as the requirement to pay for the migration of Clusters from the self-delegation deposit. In order to ensure sufficient compensation for nodes to cover their operational expenses in all market conditions (high or low demand), a central-planning-style approach is applied to Base Pricing.

The base price (per CU) is set before a new epoch starts. Node operators cast stake-weighted votes (the stake being their self-delegations) on how much the price of a CU should be in the next epoch. Node operators may submit their votes for the next epoch's price at any point during the current epoch. Abstaining from the vote equates to voting for the current price. The price of a CU for the next epoch is the stake-weighted average price, with abstaining stake counted as voting for the existing price.

Only self-delegated node operator stake is eligible to participate in this once-per-epoch vote. Furthermore, only a node operator's self-delegated stake that is less than the amount of stake required to activate all of the node's hardware is eligible to vote—if a node operator's self-delegated stake is greater than the total amount of stake required to activate all of the node's hardware, ignoring third-party stake, then only the portion of the node operator's self-delegation that is under this threshold would be counted towards the base price vote. See section 5.2 for more on the stake activation of an Arx Node's hardware.

Third-party (non-node) delegators are not permitted to participate in the voting process, since, from a game theoretic perspective, only Arx Node operators themselves have a genuine direct interest in the price being set accurately—third-party delegators do not directly pay for infrastructure (only indirectly via fees changed by Arx Nodes that they delegate to) and could therefore game the system for short-term benefit by setting a higher base price. Node operators have a direct interest in setting a sustainable and competitive price, since, if the price is too high, computation volume (and therefore revenue) will be reduced, while if the price is too low, direct revenue per computation will be reduced, and in either case operation of Arx Nodes may become unsustainable. A balanced base-price is freely found via plutocratic voting.

Lock-up periods force Arx Nodes to think long-term concerning their computation revenue—thus short-term operational cost volatility or changes in job volume will not distract them significantly. Additionally, slashing penalties for non-participation, as well as penalties associated with exiting the Network (see section 8.5.2), dissuade Arx Nodes from non-participation or leaving, even in

the case that market volatility leads to temporary (until the next epoch) under-compensation for computational work.

Arx Nodes will conduct base price voting via a separate subkey in order to mitigate key management risk, since a node's base voting key will often be stored hot (on the node). Arx Nodes may opt to, for example, run bespoke internal automated voting scripts connected to their own external pricing oracles.

#### 6.1.2 Priority Fee Markets

Arx Nodes' computation revenue is the combination of customer-defined Priority Fees and base cost revenue (see section 6.1.1). Varying priority fee values create unique fee markets, as free-market supply and demand control the premiums required for faster computation execution times. Siloed fee markets associated with unions of Clusters emerge, since computations are only competing for execution priority with other computations sharing the same Cluster, as well as with computations in other Clusters sharing any of the same nodes, and so on, recursively—Arx Nodes that are in multiple Clusters join unified fee markets together since each node has a limited amount of computational resources available at one time.

The Arcium Network's on-chain mempool is segregated such that all computations currently in the mempool are organized into priority queues based on these siloed unions of inter-associated Clusters (Clusters that share common nodes with one another). Theoretically, at certain moments, all Clusters on the Network may be part of one single Network-wide union (if computational volume is high and many interconnected Clusters are executing computations concurrently), however this mechanism ensures that when fee markets can be siloed, or broken-down into smaller fee markets, they are.

If a Computation Customer would like to use all of the resources available in a given Cluster for one (likely large) computation immediately (waiting no longer than the completion of any currently in-progress computations within the given Cluster), then the customer must set a priority fee that is higher than those of all other computations in the aforementioned recursive union of Clusters with an overlapping Arx Node-set. Furthermore, Priority Fees alone are used to determine computation execution ordering, and therefore, as a result, if, for example, one Arx Node in a given Cluster is already at capacity (with higher priority computations, or because it is not able to use all of its available hardware due to having a disproportionately small amount of stake, see section 5.3), then other computations simply must wait for execution.

### 6.2 Revenue Distribution

#### 6.2.1 Rewards Distribution to Nodes

Rewards for a computation are split equally among the Arx Nodes in the Cluster where the given computation is executed. The distribution of rewards to Clusters is entirely based on the number of computations performed by each

Cluster, and the distribution of rewards to nodes within a Cluster is entirely egalitarian in the context of that Cluster.

### 6.2.2 Sub-Node Delegation Rewards Distribution

The rewards received by an Arx Node are distributed among that node's delegators on a pro-rata basis based on the sizes of the third-party delegators' stake delegations to the given node. Fees at a fixed rate (defined for each node independently) are deducted from each delegator's rewards and paid out to the associated nodes (see section 9.3.2).

If there is a surplus of stake delegated to an Arx Node—meaning that all of the node's hardware is already active/eligible to execute work, yet there's still more (surplus) stake delegated to the node—then a sub-linear function curve is applied (only) to the delegated stake that exceeds the amount of stake required to meet the node's hardware claim (see section 5.2). As a result, once all of a node's hardware is actively working, returns continually diminish for surplus stake. This enforces a near-linear relationship between the amount of stake delegated to a node and the size of computational resources that the node has available—this way, since slashing is proportional to the size of computations where the misbehavior occurred, there is always a proportionally sufficient amount of stake available to cover any slashing that may occur. Additionally, the application of the curve functions to prevent centralization of stake on single nodes beyond an amount that is proportional to the service they provide to the Network, thus keeping the Network open and competitive to other honest node operators seeking third-party delegations.

Rewards distributions follow the Arcium Network's epoch cycle, meaning that rewards are compounded at the end of each epoch. Reward compounding is automatic, however a delegator may choose to undelegate some of their stake in order to unlock their rewards. Following an undelegation request, one complete epoch must pass before the funds become liquid, however if the undelegation is invoked on new rewards earned from the current epoch then this stake becomes liquid at the end of the current epoch (a full epoch delay is not enforced in this case). Once unlocked, stake may be withdrawn to release the liquid rewards to the delegator.

Delegators may alternatively invoke a redelegation call in order to change the Arx Node that an existing delegation is staked to in a single, combined on-chain transaction. This operation takes effect after one complete epoch, instead of incurring the two-epoch delay that would otherwise be needed to do an undelegation of stake followed by a delegation.

# 7 Multi-Party eXecution Environments (MXEs)

## 7.1 Execution Workflow

To execute a circuit, a certain number of prerequisites have to be met. First, it must be confirmed that the peers of the Cluster currently being used by the

MXE have the capacity to pick up the given computation from the mempool. This entails verifying whether they have enough preprocessing values for it: if they do not, they must generate new values for that purpose. In practice, the Arx Node software automatically schedules preprocessing protocol rounds so as to always be prepared for new computations, doing so in a way that takes advantage of spare computing and storage capacity within the node.

Once these conditions are met, the computation can be executed. The active Cluster's Arx Nodes retrieve any provided inputs and feed them to the circuit accordingly. Gates are processed concurrently by the nodes until the result is reached, at which point it is opened and recovered by all of the Arx Nodes in the Cluster. At this point, they all submit the signed result to the blockchain program, which allows other applications to use it with confidence.

## 7.2 Computation Handling

Computations are defined and scheduled inside of MXEs. Clusters execute computations for their associated MXEs. Since the Arcium Network itself is stateless, the execution of all computations can happen fully parallelized. A Cluster can execute any number of computations concurrently–the only bottleneck being the hardware capacity of the least capable node in the Cluster. Network-wide parallelization is further boosted by the fact that the Network is not required to perform work over shared state (even, often, in the case of individual MXEs), the Arcium Network can instead be viewed as a highly parallelized blackbox, performing computations with arbitrary runtime and complexity in parallel. While MXEs can be configured to use multiple Clusters, each scheduled execution of a computation occurs within a single one of the Clusters associated with the MXE. MXEs that use multiple Clusters, to improve their availability guarantees by not entirely relying on the responsiveness of any one of the individual Clusters they use, can respond to Cluster unavailability faster (e.g. by immediately retrying a failed computation using a different Cluster already associated with the MXE).

## 7.3 MXE Encoding

Encoding of data in the Arcium Network is handled at the MXE level. When creating an MXE, a Computation Customer defines an accompanying encoding scheme. An MXE's encoding scheme is defined as an encoding key used to encode the data within the MXE's computations. Encoding schemes can also be shared between multiple different MXEs belonging to the same Computation Customer, resulting in minor cost savings due to not requiring the creation of multiple encoding schemes. In the case that encoding is unnecessary for a given MXE's use case, the encoding scheme parameter can be optionally set to none.

### 7.4 Side-Channel-Attack Resistance

While some alternative confidential execution platforms such as TEEs (Trusted Execution Environments) rely on hardware to obfuscate sensitive information, this has enabled numerous types of side-channel attacks[3] on them since their inception, leaving them vulnerable to information extraction and manipulation and thereby nullifying their confidentiality guarantees. In contrast, the BDOZ MPC protocol exclusively relies on computational and information-theoretic security, meaning neither a malicious peer nor a majority of them can extract or falsify information as long as the security assumptions are met.

While this is true in theory, some insecure MPC implementations may still be vulnerable to so-called timing attacks, whereby an attacker will deduce some information about another peer's private shares by registering slight variations in latency while receiving data, as some local operations take more or less time depending on the value of the operands. The solution is to always rely on constant-time operations locally, which makes it impossible to extract information using this method. The Arcium Network's MPC implementations are hence implemented in constant time.

## 8 Clusters

### 8.1 Cluster Formation

In addition to the list of Arx Nodes that comprise a Cluster, a number of other properties are defined during the Cluster creation process. The maximum possible computational load of a Cluster (either for a single large solo computation or split across many smaller computations) is implicitly defined by the hardware specification of the weakest Arx Node (computational resource-wise) in the given Cluster (see section 5.2 for more on how this hardware specification is defined). However, when creating a Cluster, the maximum computational load that will be required of the Cluster's computational executions must also be explicitly defined as an expectational lower bound for the hardware available by each of the nodes in the Cluster.

Most properties of a Cluster may not be modified after creation because Clusters are publicly available for use and therefore their specific properties may be relied upon by other MXEs (see section 8.6 for more). However, the exception to this parameter immutability is the maximum computational limit, which may be (only) increased by the Cluster creator on the condition that the increase can be supported by all of the Arx Nodes in the Cluster.

[3] S. van Schaik, A. Batori, A. Seto, B. AlBassam, C. Garman, T. Yurek, A. Miller, D. Genkin, E. Ronen, and Y. Yarom, *SGX.Fail: Epic Fail of SGX Security*, 2023. Available at: https://sgx.fail.

### 8.1.1 Cluster Assignment and Acceptance

During Cluster creation, each Arx Node included in the new Cluster must independently accept its admission to the Cluster. Any number of subjective factors may cause an Arx Node to reject its admission to a Cluster, including, for example, the computational requirements of the Cluster as well as the reputation or jurisdiction of other nodes in the Cluster. In practice, Arx Nodes will likely maintain their own (off-chain) blacklists of Arx Node operators that they refuse to work alongside, as well as the computational requirement parameters of Clusters that they are willing to join, in order to enable on-the-fly analysis of Cluster admission requests.

For non-permissioned Clusters (see section 8.7), prior to Cluster activation, one random Arx Node from the Network is included in the Cluster's node-set. Not explicitly defined by the Computation Customer, this random node is instead independently selected (see section 8.8.1 for more on Sybil resistance with respect to this process). The Cluster immediately becomes active for use once all of the Arx Nodes assigned to it approve the assignment (including the randomly selected node), otherwise, the Computation Customer may cancel the Cluster formation process after a minimum of one complete epoch has passed. If a single Arx Node assigned to a Cluster opts to reject the assignment then the Cluster creation fails.

These same mechanisms of Cluster assignment and rejection also apply to the post-creation addition of Arx Nodes to (existing) Clusters in the event that migrations are needed.

## 8.2 Admission of MXEs

A number of different configurable behaviors are available for handling the admission of MXEs to a Cluster—the process by which an MXE is allowed to use a specific Cluster.

The default behavior for a Cluster to be used by an MXE is that all Arx Nodes in the Cluster must unanimously agree to admit the MXE. For this admission behavior, in practice, Arx Nodes may often choose to maintain their own (off-chain) blacklists of MXEs and Computation Customers (who create MXEs) which they are unwilling to work for, and then automatically accept all other (unidentified) MXE admission requests. In this paradigm, if a single Arx Node rejects the admission of a given MXE to their Cluster, then the MXE is denied the ability to use that Cluster (the Computation Customer must find a different Cluster that will accept their MXE).

Alternatively, Clusters may be configured to automatically approve all MXE admission requests (without unanimous Arx Node approval), effectively making an Arx Node's initial assignment to such a Cluster an open-ended approval of any future MXEs to the Cluster. Finally, Cluster MXE admission behavior may also be configured to delegate MXE admission decisions to a specific party (e.g. one node in the Cluster may be tasked with solely approving the admission of MXEs to the Cluster).

These behaviors apply to both single-use and persistent MXEs in the same manner (see section 3.1.3). Computation Customers who require quick access to a new MXE (single-use or persistent) can opt to select either a Cluster that has the automatic MXE admission approval behavior enabled, or a unanimous-approval Cluster with highly reputable Arx Nodes, which will likely respond to the MXE admission request immediately (since high-quality/professional Arx Node operators will typically run bespoke dynamic MXE admission scripts alongside their Arx Node software processes).

MXE admission requests that remain unapproved may be withdrawn by Computation Customers after one complete epoch has elapsed.

## 8.3 Keyshare Distribution and Security

In the Arcium Network, Clusters handle cryptographic key management, beginning with a Distributed Key Generation (DKG) process. In this step, each Arx Node in an MXE's Cluster receives a fragment of the overall cryptographic key, known as a keyshare. These keyshares are crucial as they collectively form the key of the MXE in that Cluster, enabling secure, joint computation tasks without revealing individual inputs.

The Arcium Network's primary MPC protocol (Cerberus, see section 3.2.1) is designed such that all nodes (N out of N) are required for decoding of a given MXE's data, promoting a highly secure environment. Thus, in this paradigm, as long as at least one node remains uncompromised and trustworthy at all times in a given Cluster, the integrity of computations is safeguarded. Therefore, configuring Cluster sizes can be used, in accordance with the demands of a given application, to control the security of confidential computations. For further enhanced security, nodes within a Cluster may optionally employ TEEs (see section 9.5).

## 8.4 Cluster Forking

Cluster forking occurs when one or more Arx Nodes in a Cluster that previously unanimously agreed to admit a particular MXE to their Cluster (see the Admission of MXEs section) decide to then eject the MXE from their Cluster. Cluster forking allows Arx Nodes to eject an MXE from a Cluster — for example, when the MXE is determined to violate the Cluster's published policies, applicable terms of service, or jurisdictional compliance requirements.

When one Arx Node opts to eject a particular MXE from a Cluster that they are a member of, other nodes in the same Cluster have until the beginning of the next epoch to either join the node in ejecting the particular MXE or ignore the ejection and continue providing computational resources for the MXE. At the beginning of the next epoch, the Cluster will fork, at which time all of the Arx Nodes that opted to eject the MXE must collectively pay (equally split between them all) for the creation of a new Cluster containing only the nodes that opted to continue supporting the particular MXE. The ejecting nodes must also collectively split the cost of migrating the MXE to this newly created Cluster.

Once the fork has occurred, the responsibility to process computations for the particular MXE that was ejected shifts from the old Cluster over to the newly created Cluster. In this way, the old Cluster can continue, unaffected, providing computational support to other MXEs that were unrelated to the fork.

## 8.5 Cluster Migration

Cluster Migrations are the processes by which Clusters in the Arcium Network recover in situations where at least one of the nodes in a Cluster can no longer participate in the given Cluster. Migrations can occur for a number of different reasons, all of which are examined below.

### 8.5.1 Forced Migration

The costs of migrations due to slashing (e.g. extended downtime, repeated cheating, etc., see section 5.5.1) as well as migrations due to drops in stake (see section 8.5.3 below) are spread proportionally across all of an Arx Node's participants (the node operator's self-delegation and all third-party delegations) since all of the Arx Node's delegators take collective responsibility for evaluating the node operator as well as monitoring the level of stake delegated to the Arx Node. Since migration costs will typically be quite small, in practice each of the node's participants will likely only pay a minor amount towards a forced migration—likely appearing as only a minor reduction in rewards revenue for the epoch where the forced migration occurs.

### 8.5.2 Planned Migration

In the event that an Arx Node opts to shut down, in order to avoid slashing penalties, it must announce its intention to exit the Network a full epoch in advance. Additionally, the costs associated with migrating to a new Cluster node-set are subtracted from the Arx Node's self-delegation, before the self-delegation is released. The `MINIMUM_SELF_DELEGATION_PER_1000_CLUSTERS` Network constant (see section 9.3) is used to enforce that each Arx Node can always cover migration costs of the Clusters that it is a member of, thus an Arx Node's self-delegation is always sufficiently large enough to cover the costs of the total number of Cluster Migrations that it could potentially need to pay for when shutting down.

The adjusted Cluster (resulting from the migration) is constructed of Arx Nodes based on the Cluster's pre-configured parameters.

Note that when a Cluster is migrated, scheduled computations (currently awaiting execution in the on-chain mempool) could potentially be canceled if the new Cluster node-set has a different maximal computational load capacity (see section 8.1) which is insufficient for any of the scheduled computations—however, this is the Computation Customer's own responsibility when scheduling a computation for execution, since, when creating or selecting a

Cluster, the customer knows the hardware specifications of all of the nodes, including the Cluster's backup nodes.

### 8.5.3 Rapid Drops in Stake

If an Arx Node's eligible hardware goes down due to a decrease in the stake delegated to that particular node (see section 5.3), and as a result the Arx Node then falls below the required computation capacity limit of one of the Clusters that the node is a member of (see section 8.1 for more on how Computation Customers set this computation capacity limit when creating Clusters), then the Arx Node is dropped from the given Cluster and a forced migration occurs (see section 8.5.1 above).

The computation capacity limits for the Clusters that an Arx Node is in are publicly available and thus monitoring tools will likely be developed for node participants to stay updated regarding undelegations and redelegations that could cause forced migrations. Therefore, node participants assume the collective responsibility to actively monitor these changes in total stake delegation to their Arx Nodes and take action in the current epoch (before changes take effect), when needed, to avoid incurring minor costs associated with forced migrations.

## 8.6 Reuse of Existing Clusters

Although created by an initial Computation Customer, Clusters are publicly available and may be freely used and shared by all Computation Customers on the Arcium Network. When creating and launching an MXE, choosing to reuse an existing Cluster (instead of creating a new one) also has the added minor benefit of reducing cost, since an additional on-chain transaction is required for Cluster creation. Sharing of Clusters works in terms of reliability, since Clusters are, for the most part, immutable, with the exception of their maximum computational load capacities which can only be increased (see section 8.1), as well as their Arx Node-set which can undergo migrations when needed (see section 8.5).

## 8.7 Permissioned Clusters

The Arcium Network offers versatile options when it comes to permissioned setups, enabling setups ranging from public (non-permissioned), to partially-permissioned, to entirely permissioned (fully self-operated).

### 8.7.1 Fully-Permissioned Clusters

Some organizations or institutions may opt to operate the entire Network stack themselves, including Clusters containing only self-operated Arx Nodes. Even such siloed organizations are still incentivized to run their use cases inside the Arcium Network (and not fork the protocol itself), since the Network collects

no middleman fees (so no loss of value for such siloed computations) and offers the advantages of the broader consensus mechanism for dispute resolution. Furthermore, such isolated use cases also benefit from the natural price discovery of the Network-wide base price voting mechanism.

### 8.7.2 Partially-Permissioned Clusters

Partially permissioned Clusters are Clusters where an organization or institution runs some of the Arx Node infrastructure themselves, but also includes some external Arx Nodes to diversify their node-set. Such Clusters operate on a hybrid model, benefiting from the assurances associated with operating in a Cluster where, even if all external nodes were compromised, under dishonest majority MPC protocol rules, the Cluster would still remain secure and operable, while simultaneously benefiting from external oversight to an otherwise fully-permissioned Cluster.

### 8.7.3 Data Collaboration

One common use case for permissioned Clusters on the Arcium Network will likely be companies performing, for example, data exploration or AI model training on siloed data held by multiple entities. On the Arcium Network, such entities can readily collaborate with one another without gaining access to each other's private data—while still reaping the mutual benefit from the derived insights.

## 8.8 Sybil Resistance

In the distributed Arcium Network architecture, maintaining integrity and security requires robust mechanisms to mitigate potential Sybil attack vectors. Sybil attacks occur when a single entity creates multiple false identities to gain disproportionate influence over a network. Resistance to this is primarily achieved through typical proof-of-stake mechanics, whereby each node is required to stake a specific amount in order to operate (see section 5.3), but the Arcium Network also employs a number of other strategies outlined in this section.

### 8.8.1 Intra-Cluster Sybil Resistance

Intra-Cluster sybil resistance focuses on mitigating the risks of collusion among nodes within the same Cluster, resulting in secrets being revealed. This is particularly important since, even if the broader network has a highly diverse node-set, if any individual public Clusters suffer from small centralized node-sets, they may pose intra-Cluster Sybil attack risk. The integrity of an Arcium Cluster is maintained, as long as at least one honest node exists within the given Cluster. Thus, the Arcium Network addresses intra-Cluster sybil risks by requiring the inclusion of at least one randomly selected node in all non-permissioned Clusters. This random node, chosen from the broader network, acts as an independent counterbalance, significantly reducing the likelihood of

intra-Cluster Sybil attacks. This measure ensures that Clusters remain secure and secrets confidential, maintaining the integrity and trustworthiness of the Network.

#### 8.8.2 Network-Wide Sybil Resistance

Network-wide Sybil attacks pose the risk of compromising all protocol-wide consensus mechanisms, including base price voting (see section 6.1.1) and the non-participation detection mechanism (see section 5.5.1). Network-wide Sybil resistance in the Arcium Network is achieved by increasing Cluster node-set sizes and the inclusion of a random node in each non-permissioned Cluster (see section 8.8.1 above). Furthermore, Network-wide Sybil resistance is improved through a combination of node operator reputation systems, community engagement, and strategic punishments. Off-chain reputation systems enhance security by allowing node operators to build trust based on their past performance, community interactions, and existing reputation (e.g. from providing network validation services on other networks and protocols, see section 9.4). Active community engagement plays a crucial role, as participants are encouraged to monitor and report suspicious activities, creating a social layer of defense.

# 9 Arx Nodes

## 9.1 Node Configuration and Setup

When launching an Arx Node some metadata details must be provided about the node, and it must also be associated with a node operator (a metadata entity representing the party who operates the node).

#### 9.1.1 Node Operators

Node operators are metadata entities that represent the party who operates an Arx Node. They have a one-to-many relationship with nodes, meaning that node operators may run multiple nodes. The following metadata fields are associated with a node operator:

- Jurisdiction: follows the ISO 3166-1 alpha-2 code standard for jurisdictions (e.g. DE for Germany, FR for France, etc.).
- URL: a URL to a webpage representing the node operator (e.g. social media or a webpage detailing the services and reputation of the operator).

Node operator jurisdictions are declared by the operator at registration and may additionally be attested to by Computation Customers and Cluster co-members through standard verification methods.

## 9.2 Keyshare Management and Security

Arx Nodes store and manage their keyshares for each of the MXEs associated with the Clusters that they are members of. The storage and security of these keyshares is essential for nodes to actively participate in the Arcium Network's MPC protocols.

More sophisticated node setups may choose to additionally store their keyshares inside of TEEs (see section 9.5).

See section 8.3 for more on how keyshare distribution and management occurs in the broader Cluster context.

## 9.3 Operation Incentives

Node operators have two computation-related income sources: the rewards from their self-delegation, and the fees collected from third-party delegations to their Arx Node. In most cases, as the Network grows, fees collected from delegations will likely become the lion's share of nodes' incomes.

### 9.3.1 The Self-Delegation

To operate an Arx Node, a node operator must make a self-delegation of at least the `MINIMUM_SELF_DELEGATION_PER_1000_CLUSTERS` constant amount of stake in order for their node to become active. This constant amount is defined as the required minimum threshold of self-delegated stake, and it enables the Arx Node to be a member of up to 1000 Clusters—since this amount of stake covers the costs of potential migrations for up to 1000 Clusters.

Node operators may self-delegate stake up to any amount, beyond the Network-wide `MINIMUM_SELF_DELEGATION_PER_1000_CLUSTERS` constant amount, and, furthermore, this additional self-delegation enables the node to be a member of more than 1000 Clusters. If `z` represents the total current self-delegation amount of a given node, then we can calculate `x`, the current maximum number of Clusters that a given node can be a member of, as follows:

$$x = \frac{1000z}{\texttt{MINIMUM\_SELF\_DELEGATION\_PER\_1000\_CLUSTERS}}$$

### 9.3.2 Fees

A node operator can opt to set a fee that is collected from the rewards paid to delegators to their node (taken from the curve-adjusted amounts, if the curve is applied, see section 6.2.2). This optional fee is necessary to incentivize node operation over pure delegation.

When an Arx Node is created, the fee to be collected is defined as the `fee_rate` parameter that will be applied to that node's delegator rewards. A `max_fee` rate parameter is also defined during node creation that represents the immutable maximum value that the `fee_rate` can ever be set to in the future (after creation, this maximum value can never be changed). Fee rate

modifications by node operators only take effect after a delay of two full Epochs. This means that if a fee rate change is requested in the current epoch, once it ends and two full Epochs pass, only then will the new fee rate become active. This is done to ensure that even if an Arx Node modifies its fee at the end of an epoch, delegators will have time (in the next epoch) to either redelegate or undelegate in order to adjust their stake if they are dissatisfied with the new fee rate—both the redelegation and undelegation operations enforce a one epoch delay themselves, hence the need for a two-epoch delay when changing the fee rate. The `fee_rate` and the `max_fee` parameters are both defined in basis points (meaning that, for example, a 5% fee rate is represented as 500).

## 9.4 Performance and Reliability Tracking

Reputation is important to Arx Node operators, since this is the fundamental criteria they are evaluated on. Both Computation Customers and third-party stake delegators must critically evaluate the reputation of Arx Nodes, since Computation Customers rely on them for trustworthy and efficient computation and third-party delegators select the most reliable Arx Nodes to delegate to in order to avoid incurring slashing penalties.

Off-chain reputation can be evaluated based on a wide variety of different criteria, such as team reputation, company transparency, infrastructure setup, etc. However, on-chain reputation, on the other hand, is primarily evaluated based on an Arx Node's historical slashing record, which encompasses all past instances of slashing for non-participation (e.g. downtime, system failure, etc.) and cheating (intentionally providing inaccurate results to computations)—see section 5.5.1 for more on slashing. Of course, recently launched Arx Nodes do not have extensive history yet, so evaluations of slashing records will typically also account for the longevity of an Arx Node's participation in the Arcium Network.

## 9.5 Trusted Execution Environments (TEEs)

A Trusted Execution Environment (TEE) is a secure enclave inside an Arx Node's processor that improves a node's security guarantees by keeping keyshares and data solely within the TEE itself—isolated from all other processes on the machine.

## 9.6 Security Considerations

To successfully operate an Arx Node on the Arcium Network, the node software should be run in a highly reliable infrastructure setup, ensuring maximal uptime and failover protections to avoid punishments for non-participation (see section 5.5.1). To meet such reliability needs, Arx Node operators will seek to implement sophisticated systems to safeguard their nodes, including approaches such as network load balancing, solutions combining physical data-centers with

cloud computing services, and direct cloud connections—which bypass the public internet, providing a dedicated network path between an isolated Arx Node and the cloud, significantly reducing latency and packet loss while improving security and reliability.

# 10 Computations

## 10.1 Types of Computation Tasks

In the Arcium Network, two high-level categories of computation exist: system computations and normal computations. System computations are generated by processes related to the operation of the Arcium Network itself, while all other computations are tasks requested by Computation Customers.

### 10.1.1 System Computations

System computations manage essential processes within the Arcium Network itself and come in three types:

- `DKGSystemComputation`: These are Distributed Key Generation (or DKG) operations that are run when a new MXE joins a new Cluster (see section 8.3). Priority Fees for these computations are set by the Computation Customer when they request to add an MXE to a Cluster (see section 6.1.2 for more on Priority Fees).
- `MigrationSystemComputation`: These are Cluster Migration computations (either planned or forced, see section 8.5 for more on Cluster Migrations). Priority Fees in this case can be contributed towards by all of the Arx Nodes in the Cluster that is being migrated—meaning that the portion of the priority fee paid by each node is not equal between them, each may contribute to the priority as much, or as little, as it would like (based on the importance of the execution speed to each individual node).
- `NonParticipationDetectionSystemComputation`: These computations occur when non-participation has been detected inside a Cluster and cannot be resolved by intra-Cluster consensus (a broader consensus of Arx Nodes is needed, see section 5.5.2). Priority Fees in this case are directly inherited from the original computations that led to the non-participation dispute—if the dispute results in a slash, then the cost of this duplicate priority fee is reimbursed to the nodes that executed the system computation, otherwise the duplicate priority fee represents a minor loss in revenue for the nodes involved.

### 10.1.2 Standard Computations

Standard computations in the Arcium Network are submitted by Computation Customers to execute confidential computing. A computation is always defined within the context of an associated MXE.

## 10.2 Defining a Computation

Computations must be defined as Computation Definitions before they may be instantiated. Computation Definitions always exist within the context of an associated MXE (single use or persistent, see section 3.1.3). Each Computation Definition consists of details of the computation's outputs as well as a version field for tracking upgrades to definitions and ensuring that computation execution instantiations are always performed on the intended definition version.

Computation Definitions must also grant associated execution authority—which party or parties are able to execute computations using the definition—from the following options:

- None: no entities currently have authority to execute computations using the definition.
- Private: a single entity has authority to execute computations using the definition.
- Restricted: a list of specific entities that have authority to execute computations using the definition.
- Public: anyone has authority to execute computations using the definition.

The actual execution logic of a Computation Definition is defined as a circuit. Circuits are posted off-chain, with only a hash on-chain along with a reference to a remote bucket from where nodes can fetch it. In the future, this mechanism may be extended to support circuits that are not publicly visible.

The Cerberus protocol (see section 3.2.1) operates on arithmetic circuits, meaning its fundamental units are field elements and elliptic curve points: enabling it to natively handle many cryptographic operations that act on these types without having to emulate them like Fully Homomorphic Encoding (FHE)[4] would. In addition to this however, it can also be made to support logic gates such as conditionals and comparisons, as well as more complex gates such as lookups, thus also enabling true general-purpose computing.

### 10.2.1 Arcis Circuits

Circuits within Computation Definitions are authored using Arcis, a Rust-based framework for developing confidentiality-preserving MPC logic. Arcis abstracts cryptographic complexities by overriding standard Rust types (e.g., `u8`, `bool`, etc.) with masked variants that operate on secret-shared data. Key features include:

- Type-Safe Confidentiality: All operations preserve data secrecy unless explicitly revealed.

[4] C. Gentry, *A Fully Homomorphic Encryption Scheme*, PhD thesis, Stanford University, 2009. Available at: https://crypto.stanford.edu/craig/craig-thesis.pdf.

- Fixed-Point Arithmetic: Supports precision-sensitive computations (e.g., financial models) via a fixed-point representation with 52 fractional and 75 integer bits, enabling rational number handling within finite fields. In the future a more dynamic range will be supported.
- Side-Channel Resistance: Control flow branches execute both paths when conditioned on secret values, eliminating timing leaks.

Arcis-generated circuits are compatible with all Arcium MPC protocols (e.g., Cerberus, Manticore), enabling developers to balance security and performance without rewriting logic.

### 10.2.2 Data Parameters

The data parameters associated with a Computation Definition are either:

- On-Chain Accounts: Computation Definitions can be configured to handle on-chain blockchain-based data.
- Direct Input: Computation Definitions can accept data posted directly via transaction call data.

In the future, as MPC protocol support expands on Arcium, these data provisioning options will likely be extended.

### 10.2.3 Execution Costs

The circuit associated with a Computation Definition also contains metadata about the number of arithmetic operations within the circuit, as well as the number of inputs and outputs. This metadata enables precise calculation of the execution cost of the circuit based on the exact number of each operation type included.

## 10.3 Commissioning a Computation

Once a computation has been defined (see section 10.2), individual computations may be commissioned for execution. First, the exact details for execution, including the arguments, callbacks to be executed afterward, and payment bid details must be provided. The Computation Customer may additionally specify a time window during which the computation's execution is valid (see section 10.4). Then, the computation is sent to the Arcium Network's on-chain mempool.

When commissioning a computation, the Computation Customer must explicitly define the priority fee that they are willing to pay for the computation (see section 6.1.2 for more on how Priority Fees work to define competitive fee markets).

The arguments passed to a computation during commissioning naturally correspond identically to the parameters specified during the Computation Definition step (see section 10.2).

### 10.3.1 Callbacks

Callbacks enable bespoke actions to be taken following the execution of a computation. Callbacks are defined during commissioning rather than in the definition itself since that way callback actions may be unique to particular instantiations without affecting the underlying logic and circuit of the Computation Definition. Callbacks may be created for both the success and failure cases, meaning that, depending on the outcome of the computation's execution, different instructions are taken.

Callback instructions for the success case may be either static or dynamic. Static instructions enable predefined on-chain actions to be taken, while dynamic instructions enable more complex bespoke on-chain actions (on other programs) using the outputs from the successful computation.

Callback instructions for the failure case are always static on-chain actions since no output is generated.

## 10.4 Computation Lifecycle

Computation Definitions are specifications of logical operations attached to MXEs and therefore they do not expire. However, individual instantiations of computations (commissioned computations) implicitly expire following their execution. Furthermore, a computation may specify a specific window during which the computation's execution is valid. This window consists of two timestamps specifying the times after which, and before which, the computation may be executed.

If a computation's "valid after" timestamp is in the future then the computation waits in a segregated portion of the mempool, while if it is in the past, the computation is valid to be executed. At the same time, as long as the "valid before" timestamp is in the future, the computation is valid to be executed, while if it is in the past, then the computation has expired and therefore is no longer valid to be executed. By default, if this validity window is not specified for a computation, then the "valid after" timestamp is set to zero and the "valid before" timestamp is set to infinity.

# 11 Network Maturity & Sustainability

The Arcium Network's long-term sustainability rests on Computation Customer demand. Arx Nodes are compensated for executed work via a base price plus customer-defined Priority Fees (see section 6.1), with Priority Fees accruing entirely to the executing nodes. Base pricing is set by stake-weighted node-operator voting each epoch (see section 6.1.1), enabling nodes to maintain a cost-covering minimum rate under varying market conditions.

Lock-up periods, slashing penalties (see section 5.5), and the costs associated with planned exits (see section 8.5.2) further dissuade Arx Nodes from rapidly entering or exiting the Network in response to short-term volatility, helping sustain Cluster reliability and Computation Customer experience over time.

### 11.1 Governance

Arcium's governance framework distributes control through two parallel systems: token holders may vote on protocol-level economic parameters via on-chain governance, while node operators vote on network-operations parameters. Neither group has the ability to direct token issuance, conduct market activity, or grant beneficiary payments outside of pre-published, deterministic protocol logic. This separation ensures checks on critical parameters while maintaining agility for protocol adjustments.

Token holders may vote on protocol-level economic parameters via stake-weighted voting, including parameters governing staking activation thresholds, fee distribution, and other economic constants.

Node operators govern protocol parameters also via stake-weighted voting, including supported MPC protocols, slashing conditions, and computation Base Pricing. Critical constants like `MINIMUM_SELF_DELEGATION_PER_1000_CLUSTERS` are adjustable via governance to reflect evolving migration costs (ensuring nodes maintain sufficient collateral for Cluster reliability, see section 9.3.1).

The Network progressively decentralizes through phased governance. Initial upgradeable contracts transition toward immutability as adoption stabilizes, with Arcium team oversight systematically reduced in favor of community-led proposals. However, from launch, the ARX token contracts are only upgradeable via a supermajority stake-weighted vote, whereas the protocol-side contracts are initially upgradeable by the Arcium team, transitioning to community governance over time.

All governance occurs on-chain with full transparency. Core smart contracts are open-source, while proprietary protocol-side components like the Arx Node software remain initially closed-source to protect competitive advantages (but will also ultimately transition to open-source as well). Voting results and parameter changes are immutably recorded on-chain, enabling real-time accountability.

## 12 Comparative Analysis

### 12.1 Differentiation

Arcium diverges fundamentally from both TEE-dependent networks (e.g., Oasis[5], Secret Network[6]) and FHE-focused platforms by prioritizing cryptographic trustlessness over hardware reliance. Unlike TEE solutions vulnerable to side-channel attacks (see section 7.4), Arcium's MPC Clusters enforce Byzantine fault tolerance without proprietary enclaves, while avoiding FHE's prohibitive latency via parallelized MXE execution. Compared to enterprise MPC services

[5]Oasis Protocol Foundation, *Oasis Network: Privacy for Open Finance*, 2020. Available at: https://oasisprotocol.org/whitepaper.

[6]Secret Foundation, *Secret Network: Programmable Privacy*, Graypaper v2.0, 2023. Available at: https://scrt.network/graypaper.

(e.g., Fireblocks[7]) that operate centralized gateways, Arcium's decentralized architecture combines permissionless node participation with stake-aligned tokenomics—transforming confidential compute from walled-garden SaaS to a liquid commodity. This hybrid model uniquely bridges Web3's incentive structures with enterprise-grade confidentiality guarantees.

# 13 Future Developments

While initially deploying the on-chain portions of the Arcium Network to Solana, Arcium will expand to become a multi-chain confidential compute layer, enabling cross-ecosystem confidentiality-preserving operations while maintaining unified tokenomics. The Network will continue to introduce additional modular MPC protocols, allowing developers to dynamically select cryptographic primitives tailored to specific use cases, from low-latency DeFi to regulated enterprise workflows.

## 13.1 Multi-Chain

The Arcium Network is blockchain-agnostic by design, and will therefore transition to being multi-chain. By architecting chain-versatility into the fundamental economics and incentive design of the Arcium Network from the outset, as the Network transitions to being multi-chain, it will be possible for Arx Nodes to activate support for multiple different chains concurrently without needing to self-organize their resource allocation between different blockchain deployments of the Arcium Network protocol.

[7] Fireblocks, *MPC-Based Wallet Infrastructure*, 2023. Available at: https://www.fireblocks.com/.

# Conclusion

The Arcium Network establishes a new paradigm for decentralized confidential computing, where the ARX token serves as both the economic engine and trust anchor of the ecosystem. With a fixed total supply and demand driven by staking to activate node hardware, ARX directly ties token utility to real-world demand for encoded computation. This design ensures node operators, delegators, and customers share aligned incentives, fostering a self-reinforcing cycle of security and scalability.

At its core, Arcium transforms confidentiality from a niche feature into a universal utility. Its cryptographic primitives, coupled with hardware-attested resource claims and Byzantine fault-tolerant Clusters, enable secure computation at Web3 scale. As enterprises and developers increasingly demand trustless data-in-use protection, Arcium's tokenomics and architecture position it not merely as a tool, but as the foundational layer for a new era of confidential applications—where every computation is both private and provable.

# A FAQ

- **Is the ARX supply fixed?** Yes. The total supply of ARX is fixed at genesis. The protocol has no minting or burning mechanisms—Arx Nodes are compensated solely via base price and Priority Fee payments from Computation Customers (paid in the chain-native token, see section 6.1), and all Priority Fees accrue entirely to the executing nodes.
- **Can locked/unvested tokens be delegated to Arx Nodes?** No, locked/unvested tokens cannot be delegated to nodes. Permitting such delegation would dilute the link between staked ARX and actual node hardware provisioning, weakening the signal that ties token utility to real Network usage.

# B Technical References

- **Solana Documentation**: `https://docs.solana.com/`
- **MPC Protocols**:
  - BDOZ Protocol: Bendlin et al., "Semi-homomorphic Encryption and Multiparty Computation"
  - SGX Vulnerabilities: van Schaik et al., "SGX.Fail: Epic Fail of SGX Security"
- **Arcium Network Documentation**: `https://docs.arcium.com/`
- **FHE Foundations**: Gentry, C. *A Fully Homomorphic Encryption Scheme.* `https://crypto.stanford.edu/craig/craig-thesis.pdf`

# C Glossary

- **ARX**: Native utility token of Arcium Network, used for staking, governance, and protocol incentives.
- **ArxOS**: The operating system layer managing Arcium's off-chain compute infrastructure (nodes, Clusters, MXEs).
- **Arcis**: Rust-based framework for developing MPC circuits with type-safe confidentiality guarantees.
- **CU (Computation Unit)**: Standardized measure of computational work on the Network.
- **MXE (Multi-Party eXecution Environment)**: Isolated virtual environment for MPC operations with configurable trust models.

- **Cluster**: Group of Arx Nodes executing computations, with Byzantine fault tolerance guarantees.
- **Priority Fee Market**: On-chain mempool mechanism for computation scheduling priority across Cluster unions.
- **TEE (Trusted Execution Environment)**: Optional secure enclave for enhanced keyshare protection (e.g., Intel SGX).
- **DKG (Distributed Key Generation)**: Protocol for secure cryptographic key sharing across Cluster nodes.
- **Byzantine Fault Tolerance**: System resilience against malicious/erroneous nodes in distributed networks.