LOG Protocol and Ecosystem Whitepaper

Version 1.0. Initial technical edition.

Revision date: 25 November 2026.

Project: LOG.

Network: Sui Mainnet.

Website: https://uselog.xyz

X: https://x.com/glog_g

Telegram: https://t.me/gglog_tg

Coin type: 0xcc88e22d5042c96a1e6c234c00656d9849c6ef43ff8d59d96f7ae8d90d8a4aae::log::LOG

Abstract

LOG is a fixed-supply fungible token and an early-stage digital ecosystem project deployed on Sui. The current Mainnet Move package implements the LOG currency, a reserve-accounting object, and an inventory-backed sale that accepts exact SUI payments for whole-LOG quantities. It does not implement the project's proposed referral programme, social-task verification, LOG Box, phased claims, governance system, or integrations with independent products.

The LOG supply is fixed at 100,000,000 tokens with nine decimal places. The initial allocation assigned 30,000,000 LOG to public sale, 30,000,000 LOG to rewards, 15,000,000 LOG to liquidity, and 25,000,000 LOG to the project treasury. The public-sale inventory is held in a separate shared Sale object. The sale was verified active on Mainnet on 25 November 2026 with 30,000,000 LOG remaining and a configured price of 0.005 SUI per whole LOG.

This document describes the deployed system separately from concepts under discussion. A feature is not represented as available merely because it appears in a roadmap or has been discussed by the project. Mainnet object values and administrative state can change; the on-chain objects and transaction history are the authoritative record of current state.

1. Document purpose, status, and terminology

This whitepaper is the project's technical and economic description of LOG. It is intended to explain what has been deployed, how the deployed components operate, what assumptions they make, and which proposed components remain undecided or unimplemented. It is a versioned working document and may be revised as the system changes.

The project intends to expand this document over time, potentially into a long-form reference of up to 100,000 words where the technical and economic scope justifies that depth. New material will be added in stages and grounded in specific protocol decisions, implementation evidence, and independently checkable information. The length is not a reason to repeat claims or present unresolved proposals as facts.

The following terms indicate implementation status:

- **Deployed:** the relevant package or object exists on Sui Mainnet.

- **Verified state:** an object or transaction was queried directly on Mainnet on the stated date.
- **Tested:** the feature has passed project tests; this is not equivalent to an independent audit.
- **Planned:** the project intends to consider or build the feature, but it is not deployed.
- **Under discussion:** the idea or parameters have not been approved as a final specification.
- **Not specified:** no final parameter or commitment is asserted in this document.

Mainnet values in this edition were checked on 25 November 2026 unless a different date is stated. The document is not a live dashboard. Readers should query the identified objects before relying on a current balance, sale status, capability owner, or package version.

2. Project description and scope

LOG is an independent digital ecosystem project organized around its native fungible token and a future network of digital experiences, utilities, and participation opportunities. The present on-chain implementation is narrower than that ecosystem ambition: it creates a fixed-supply token, accounts for designated reserves, and operates a SUI-denominated sale.

SimSoccer and Lucky Six are separate existing applications. Their gameplay, accounts, databases, and operational rules are not contained in the LOG Move package. Any future integration would require distinct application and backend work, including a defined method for verifying wallet ownership and on-chain payment before crediting an external account.

The project does not claim that every future product will use LOG, that every activity will generate rewards, or that token ownership grants an interest in project revenue, governance, or assets. Any such rights would require explicit terms and an implemented mechanism; they are not created by the current token or sale contracts.

3. Design objectives and boundaries

The deployed protocol is designed around the following objectives:

1. Establish a fixed and publicly identifiable LOG supply.
2. Record reserve balances in explicit on-chain state.
3. Separate sale inventory from long-term reserve balances and SUI proceeds.
4. Require exact SUI payment and whole-token purchase quantities in the sale function.
5. Emit events for purchases and selected administrative actions.
6. Distinguish deployed functionality from future product and reward proposals.

These objectives do not imply that the current contracts provide decentralized governance, trustless administration, price stability, guaranteed liquidity, or an audited security guarantee. The project retains administrative capabilities and a package UpgradeCap. Those authorities and their custody are material trust assumptions.

4. Sui execution and object model

Sui represents on-chain state as typed objects with unique identifiers, ownership information, and versions. LOG uses shared objects for the reserve Treasury, Sale, and SUI proceeds Treasury. Administrative capabilities are owned objects. The LOG currency and its metadata are managed through Sui's Currency Standard and Coin Registry.

Sharing an object makes it available for transactions to reference; it does not itself authorize a caller to perform privileged actions. The Move module must check authorization. LOG's purchase function is intentionally available to buyers while the sale is active. Sale configuration and reserve or proceeds withdrawals are administrative operations controlled by capability objects.

Sui transactions are programmable transaction blocks (PTBs). A PTB can compose Move calls, coin operations, and transfers atomically. LOG's purchase function returns a Coin<LOG>; the caller must consume that value, usually by transferring it to the buyer in the same PTB. When a command fails, the transaction does not partially commit its state changes.

Shared-object operations are subject to Sui's consensus and transaction processing. The LOG sale therefore depends on the availability of the Sui network and the current shared Sale and SUI treasury objects.

5. Deployed package architecture

The Mainnet package is named log_core and uses Move edition 2024. Its deployed modules are log, log_treasury, and log_sale. A published package version is immutable; an upgrade publishes a new version subject to the package's upgrade policy and Sui compatibility rules.

Module	Principal state	Responsibility
log_core::log	LOG currency and metadata state	Creates the LOG currency through Sui's Currency Standard, issues the initial supply, fixes the supply, and finalizes metadata.
log_core::log_treasury	Treasury and TreasuryAdminCap	Partitions the initial LOG coin among named reserve balances and provides capability-gated withdrawals and a recycled-token deposit.
log_core::log_sale	Sale, SaleAdminCap, SuiTreasury, and SuiTreasuryAdminCap	Holds sale inventory, validates exact-price purchases, tracks the sale lifecycle, and accumulates SUI proceeds.

Object roles

- **Currency:** shared Currency Standard metadata and supply record for LOG.
- **Treasury:** shared object with distinct balances for public sale, rewards, liquidity, project, and recycled LOG.
- **Sale:** shared object containing sale inventory, a SUI treasury identifier, price in MIST per whole LOG, and an active flag.
- **SuiTreasury:** a separate shared object holding SUI collected by its associated sale.
- **Administrative capabilities:** owned objects authorizing specified operations; the package UpgradeCap is a separate authority.

6. LOG token specification

Property	Mainnet specification
Name	LOG
Symbol	LOG
Asset model	Sui Currency Standard fungible asset
Decimals	9
Fixed supply	100,000,000 LOG
Supply in base units	100,000,000,000,000,000
Mainnet coin type	0xcc88e22d5042c96a1e6c234c00656d9849c6ef43ff8d59d96f7ae8d90d8a4aae::log::LOG

One LOG equals 1,000,000,000 base units. During currency creation, the package mints the initial total and passes the TreasuryCap to the Currency Standard's fixed-supply operation. That operation consumes the cap. The deployed package does not retain a live TreasuryCap or expose a later mint function.

Fixed supply describes the maximum issued amount under the deployed supply configuration; it does not mean that every token is in circulation or that balances cannot move. Transfers among wallets, reserves, sale objects, and applications change ownership or location, not the fixed supply. The current package does not implement a LOG burning mechanism.

The coin type is the authoritative type identifier. A token with the same name or symbol but a different package address is not this LOG asset. Users should verify the full coin type before interacting with a wallet, exchange, pool, or sale.

7. Initial allocation and current reserve accounting

The initial distribution allocated the fixed supply to four categories. The amounts below describe the launch allocation, not necessarily the current balance of every associated object.

Initial category	Share	LOG	Launch location
Public sale	30%	30,000,000	Transferred into the separate Sale object.
Rewards	30%	30,000,000	Rewards balance in the shared Treasury.
Liquidity	15%	15,000,000	Liquidity balance in the shared Treasury.
Project treasury	25%	25,000,000	Project balance in the shared Treasury.
Total	100%	100,000,000	Equals the fixed supply.

The Treasury initializer accepts amounts as transaction parameters and checks that the initial balances partition the complete supply. The public-sale balance in the Treasury is zero after the sale inventory is moved into the Sale object. The recycled balance began at zero; it is a separate receiving balance and is not an additional initial allocation.

On 25 November 2026, 5,000,000 LOG was withdrawn from the project treasury to the publisher address. The shared Treasury was then queried on Mainnet and showed 20,000,000 LOG remaining in the project balance, 30,000,000 in rewards, 15,000,000 in liquidity, zero in public sale, and zero in recycled balance. The withdrawal did not change the fixed total supply.

The Mainnet Sale object was queried on 25 November 2026 and showed an active status, 30,000,000 LOG inventory, and a price of 5,000,000 MIST per whole LOG. The associated SUI proceeds object showed zero proceeds at that time. These values are a dated observation, not an assurance that they will remain unchanged.

Withdrawals move LOG between on-chain locations. They do not create new supply. Allocation categories are project accounting designations; they do not, on their own, create contractual rights for token holders.

8. Public sale protocol

The deployed sale is an inventory-backed exchange. Its shared Sale object holds LOG inventory and records the associated SUI treasury, price, and active state. SUI payments are accumulated in a separate SuiTreasury object.

Purchase conditions

For each purchase, the Move function checks that:

1. The sale is active.
2. The supplied SuiTreasury is the object associated with the Sale.
3. The requested LOG amount is positive and is an exact whole-token amount.
4. The sale inventory is sufficient.
5. The required payment calculation does not overflow the supported integer range.

6. The SUI coin supplied by the buyer equals the required price exactly.

At the verified Mainnet configuration, the price is 5,000,000 MIST, equal to 0.005 SUI, for each whole LOG. For a purchase of N whole LOG, the required payment is N multiplied by the configured MIST-per-LOG price. The contract does not accept overpayment or calculate change.

On successful execution, the payment is added to the associated SUI proceeds balance and the requested LOG is split from the sale inventory. The resulting Coin<LOG> is returned to the caller for transfer in the same PTB. A PurchaseCompleted event records the buyer address, sale and treasury identifiers, LOG amount, and SUI amount. When inventory reaches zero, the sale becomes inactive.

Sale administration and limits

The administrator can configure the price while the sale is inactive, start or stop the sale, and withdraw unsold inventory only while the sale is inactive. The proceeds administrator can withdraw SUI from the associated SuiTreasury. The sale has no per-wallet purchase limit and stores no calendar end date. Its global inventory is the purchase limit.

The current sale is open to any address while active. It has no allowlist, private-buyer registry, vesting schedule, refund period, or delayed claim. The contract enforces the on-chain payment and inventory invariants described above; it does not provide a website, checkout interface, fiat conversion, customer support, or off-chain account crediting.

The sale price is a configured SUI amount, not a guaranteed USD value, market price, redemption value, or promise of secondary-market liquidity. SUI's external exchange value can change.

9. Private-sale concept under discussion

A separate private-sale tranche has been discussed but is not deployed and is not available for purchase through a private-sale contract. The current working discussion is for up to 2,000,000 LOG at the same 0.005 SUI per LOG rate as the configured public sale. At that rate, a fully subscribed 2,000,000 LOG tranche would correspond to 10,000 SUI.

The project has not finalized buyers, buyer wallet addresses, dates, allocation limits per buyer, sale documentation, or the handling of proceeds. The project has discussed beginning with outreach and collecting expressions of interest before deciding whether purchases will be restricted to approved wallets. No private-sale payment should be treated as accepted under this discussion.

If the project elects to take the tranche from the current public-sale inventory, the public sale must first be stopped because the deployed contract permits inventory withdrawal only while the sale is inactive. A 2,000,000 LOG withdrawal would reduce that sale inventory from 30,000,000 to 28,000,000 LOG if no other sale activity occurs. The public sale could then be restarted with the remaining inventory.

The existing public-sale contract cannot enforce private access. A separate sale mechanism would be needed to enforce a wallet allowlist, buyer-specific allocation limits, or a separately accounted proceeds route. A permissionless sale is public in operation even if it is advertised privately. On-chain participation is publicly observable; the word “private” describes access and negotiation, not confidential blockchain execution.

No private-sale module, escrow object, wallet allowlist, vesting rule, or private-sale proceeds destination is represented as deployed in this edition. The figures above are a discussion record, not a binding offer or final tokenomics change.

10. Liquidity and secondary-market considerations

The LOG core package does not create or manage a decentralized-exchange pool. A pool is a separate protocol deployment with its own contract address, assets, price parameters, liquidity positions, fees, and administrative rules. A token's presence in a pool does not guarantee that the pool has sufficient depth, that its displayed price is reliable, or that a user can trade at that price without material slippage.

Liquidity provisioning requires both sides of a trading pair in the amounts required by the selected protocol. For example, at a nominal ratio of 200 LOG per SUI, 10,000 SUI would pair with 2,000,000 LOG. This is a calculation at that ratio, not a recommendation or a promise that a pool will hold that price. A concentrated-liquidity pool may require a selected price range; liquidity outside that range may not be active for trading.

Any project-provided liquidity should be disclosed with the pool address, the assets and amounts deposited, the price configuration, ownership of the liquidity position, and any lock or withdrawal conditions. The project should not represent a pool as locked, audited, deep, or independently managed unless that claim has been verified. Artificial trading volume or self-trading should not be used to create a misleading appearance of market activity.

The 15,000,000 LOG liquidity allocation is a reserve designation. Its existence does not establish that all or any specific amount has been deposited into a DEX pool. Pool balances and positions must be checked independently from the reserve Treasury.

11. Ecosystem components and future integrations

The token is intended to be usable across supported LOG experiences if and when those integrations are built. Each integration must specify whether LOG is used for payment, access, rewards, or another function; how transactions are verified; which system holds relevant assets; and what user actions are reversible or final.

Referral and participation systems

Referral attribution, community campaigns, and social activities have been discussed as possible participation mechanisms. None is implemented in the current Mainnet Move package. Before launch, a campaign would need published eligibility rules, a finite reward budget, duplicate and abuse handling, verification methods, distribution timing, and a process for disputed or failed activity.

LOG Box and randomized rewards

LOG Box has been discussed as a possible ticket-based reward experience. The current package contains no ticket type, random selection logic, prize distribution, claim schedule, or related allocation enforcement. No reward probability, minimum or maximum amount, ticket price, or claim timetable is final in this edition.

Any future randomized feature must have a documented randomness source, clear eligibility and transaction rules, protection against replay, and an independently reviewed economic budget. The project should specify which logic is on-chain and which decisions depend on an off-chain service. A user interface animation would not itself establish that a reward was generated fairly or recorded correctly.

Product payment integration

If an independent product accepts LOG, its backend must verify the relevant Sui transaction before crediting a product account. The integration should bind each credit to a transaction digest and process it idempotently so that retries cannot credit the same payment twice. A wallet address alone does not establish a real-world identity, and linking an address to a product account requires clear privacy and account-recovery rules.

12. Administration, upgrades, and governance

The deployed system is not governed by a DAO. The publisher address controls the deployed administrative capabilities and UpgradeCap. The package's compatible upgrade policy allows eligible future package upgrades; published bytecode is not modified in place. An upgrade publishes a new package version and remains subject to Sui's package compatibility rules.

Authority	Current role	Important limitation
SaleAdminCap	Configures, starts, and stops its associated Sale; withdraws inactive sale inventory.	Controls the public sale lifecycle and unsold inventory.
SuiTreasuryAdminCap	Withdraws SUI from its associated SuiTreasury.	Controls the destination of withdrawn sale proceeds through the transaction.
TreasuryAdminCap	Authorizes reserve withdrawals.	The capability struct is not bound to a specific Treasury ID; the deployed state has one Treasury and one such capability.
MetadataCap	Controls metadata operations permitted by the Currency Standard.	Owned by the publisher address.
UpgradeCap	Authorizes compatible package upgrades.	Can change future package behavior subject to upgrade rules; owned by the publisher address.

The TreasuryAdminCap has no implemented revocation or rotation flow. SaleAdminCap and SuiTreasuryAdminCap include object identifiers for their respective objects, but the existence of an authorization capability still makes its custody important. No multisig, timelock, DAO vote, or additional administrator set is configured in the deployment described here.

Concentrated administrative authority is an operational risk. A lost or compromised key may prevent intended administration or enable unauthorized actions within the powers of the affected capability. The project should improve key custody and governance before handling materially larger reserves or adding additional value-moving modules.

13. Security, testing, and verification

Move's resource and type system provides useful safety properties, but those properties do not prove that the protocol's logic, configuration, economics, frontend, key custody, or integrations are secure.

Reported validation

The project reports 21 passing Move tests and 92.73 percent line coverage across the package test suite. A local Sui source-verification run was reported successful for the Mainnet package using the publication metadata and matching toolchain. Source verification checks bytecode correspondence; it is not an independent security audit.

As of this revision, no independent third-party security audit is claimed. A SuiScan source-submission or public verification workflow should be treated as pending unless its successful completion can be independently confirmed on the explorer. Explorer reputation warnings are not proof of contract vulnerability, and an absence of warnings is not proof of safety.

Principal risks

- **Administrator-key risk:** reserve, sale, proceeds, metadata, and upgrade authorities are controlled by project-held capabilities.
- **Upgrade risk:** the UpgradeCap holder can publish compatible changes that alter future behavior.
- **Capability design risk:** TreasuryAdminCap does not record the Treasury ID it is intended to authorize.

- **Market risk:** the sale price is not a redemption guarantee; secondary-market price and liquidity may differ materially.
- **Concentration and slippage risk:** a shallow or concentrated pool can experience large price changes from relatively small trades.
- **Integration risk:** future product credit, referral, or reward systems may rely on off-chain services not covered by this package.
- **Operational risk:** incorrect addresses, transaction construction, custody procedures, or user-interface information can cause loss.
- **Disclosure risk:** users may misunderstand a proposed feature as deployed unless status and terms are clearly labeled.

Users should independently verify the full coin type, package ID, current object state, capability ownership, source correspondence, transaction effects, and pool details before interacting with LOG. This document is not a substitute for inspecting on-chain data or obtaining independent technical review.

14. Development plan and decision gates

The following workstreams describe the project's direction; they are not delivery commitments or claims that the work is underway unless separately stated.

1. **Transparency:** maintain accurate deployment records, token metadata, reserve disclosures, and source-verification evidence.
2. **Operational security:** improve capability custody and define a reviewed authority and incident-response model.
3. **Liquidity operations:** define pool selection, initial price, asset amounts, position control, and public disclosure before representing liquidity as established.
4. **Sale development:** finalize whether any separate private sale will exist, its access rules, price, buyer limits, settlement, and proceeds destination before accepting funds.
5. **Product integration:** design transaction verification and account-credit handling for each product independently.
6. **Rewards:** define eligibility, budget, anti-abuse, randomness where applicable, distribution, and claim behavior before implementation.
7. **Independent review:** obtain external review of new value-moving code before production use.

Changes to supply, allocations, sale terms, or governance should be documented with dates, affected object identifiers, transaction references, and a clear statement of whether the change is proposed, approved, or executed.

15. Mainnet deployment record

The following identifiers refer to Sui Mainnet. Testnet has separate package and object identifiers and must not be used for Mainnet transactions.

Resource	Mainnet identifier
Package, original package ID	0xcc88e22d5042c96a1e6c234c00656d9849c6ef43ff8d59d96f7ae8d90d8a4aae
Coin type	0xcc88e22d5042c96a1e6c234c00656d9849c6ef43ff8d59d96f7ae8d90d8a4aae::log::LOG
Currency object	0xdc1dda07b2865805840b702bf1eeb90988b33eef93e9d1201d052246933033da
Shared reserve Treasury	0xe3dcb801ecd520669eb73f22b6dedddb3d111c05c5b7612e3849e2f4d0865260
Shared Sale	0x194f2249b3d0e032060f2e7f4f8e422380613a51ad711fa91a6265817d612683

Resource	Mainnet identifier
Shared SUI proceeds Treasury	0xe7ced97b997e235a93c20d24ef145a9e27a64c6577a23aa0c40abbfb73a58a5a
Publisher and current capability owner	0x4aaa92440a045d76fa191cc89a447ebd7b0fe932611f92105f4f5daaf2bcf971
UpgradeCap	0x99416e529cd4494506077cd0af9708c659e33b45b3c89476afc2afffaa6ef250
TreasuryAdminCap	0xce221f082bec9dbf0008545754ee18062cb5f1fc535ab77cd08db5f231698430
SaleAdminCap	0xddb18d0541465099df2f4a58bec164946e7a1e4dbdb883e9b42d6b82e9a6e6bc
SuiTreasuryAdminCap	0x501731f06ec8cd083266c7475373f82ecbeaba81f18272b636478cd2c01b8f83
MetadataCap	0x7d1ef746fafb12f24a9d0250c6ff6be96eff28faac6ba2e8fd2f4cd6d1a3d4bc

Transactions referenced in this edition

- Mainnet package publication: FzA8WuCbgSZHCgvBwTdTUdurQR8UCgZ34nUs68HiqiF4.
- Mainnet fixed-supply currency, reserves, and sale setup: CrpKtmnanBJNeTE1D5qDfEpY5C76dgfDVPdhUSmXeh9U.
- Mainnet project-reserve withdrawal of 5,000,000 LOG: 57yn26UxmYc7maJYYTLgSusE61ffu45DFQtmTSqSRVe1.

16. Technical references

Project-specific implementation statements in this document are based on the LOG Move source and queried Mainnet objects. Sui protocol descriptions refer to official documentation:

- Sui Documentation, Create Fungible Tokens: Currency Standard: https://docs.sui.io/onchain-finance/fungible-tokens/create-a-fungible-token
- Sui Documentation, Object Model: https://docs.sui.io/develop/sui-architecture/object-model
- Sui Documentation, Shared Objects: https://docs.sui.io/develop/objects/object-ownership/shared
- Sui Documentation, Programmable Transaction Blocks: https://docs.sui.io/concepts/transactions/prog-txn-blocks
- Sui Documentation, Move Package Management: https://docs.sui.io/develop/manage-packages/move-package-management
- Sui Documentation, Source Verification: https://docs.sui.io/develop/manage-packages/source-verification
- Sui Documentation, Security Best Practices: https://docs.sui.io/develop/security/best-practices

17. Important disclosures

This document is informational. It is not legal, tax, or investment advice; an offer to sell or solicitation to buy a security or other financial instrument; or a promise of profit, price appreciation, liquidity, exchange listing, rewards, governance rights, or future product delivery. Any token sale is subject to its separate final terms and applicable law.

Descriptions of proposed features are not guarantees that those features will be implemented. Unless supported by a deployed contract or a clearly identified external system, a feature should be treated as a concept only. No independent security audit is claimed in this document.

Version 1.0. Initial technical edition, revised 25 November 2026. Website: https://uselog.xyz. Community contact: https://t.me/gglog_tg.