WINDFALL LOTTO

Technical Whitepaper and Contract-Derived Protocol Specification

This document describes the current Windfall Lotto implementation as expressed in the latest uploaded Solidity package. It focuses on the protocol as implemented: ticket economics, jackpot logic, decentralization properties, shareholder mechanics, randomness fallbacks, draw progression, and NFT-based result recording. Where design intent and current code behavior differ, the code behavior is treated as authoritative.

Reviewed contracts: WindfallLotto.sol, WindfallTicket.sol, WindfallDrawNFT.sol, WindfallFeeShare.sol, WindfallSVG.sol

Version date: 2026-04-02

Primary model: weekly on-chain lottery with permissionless draw progression and multi-source randomness fallback

Executive Summary

Windfall Lotto is a token-denominated lottery protocol built around transferable ERC-721 ticket NFTs, weekly draw cycles, permissionless state progression, and a layered randomness path that escalates from Chainlink VRF to Supra dVRF and finally to a blockhash-derived fallback. The system is designed so that anyone can help move a draw from sale period to reveal, ticket counting, payout finalization, fee distribution, and next-draw opening.

The latest codebase implements a streak-based, order-sensitive winning model. Each ticket contains five numbers in the 0-99 range. A ticket does not win by matching numbers in any order. Instead, it wins only when it matches a consecutive streak of three, four, or five numbers in the exact positions of the revealed winning array.

The economic model combines three distinct flows: ticket revenue, jackpot donations, and host-fee redistribution. Ticket purchases split into a jackpot portion and a host-fee portion. Donations increase only the jackpot and do not incur protocol fee extraction. Host fees are redistributed through a separate shareholder contract that rewards qualifying donors and preserves a continuing economic role for the host treasury.

1. Protocol Overview

Core contract responsibilities

Contract	Primary role	Key notes
WindfallLotto	Core protocol engine	Handles ticket sales, donations, draw lifecycle, randomness requests, counting, payout finalization, claims, fee distribution, and opening the next draw.
WindfallTicket	Ticket NFT	ERC721Enumerable ticket collection. Stores draw id, draw end, original minter, packed numbers, and visual tier state for metadata/SVG rendering.
WindfallDrawNFT	Result NFT	Mints two result NFTs per completed draw: one to the host treasury and one to the lotto contract itself.

Contract	Primary role	Key notes
WindfallFeeShare	Shareholder ledger	Tracks donor qualification, expiry, fee-share accruals, and claims for redistributed host fees.
WindfallSVG	Rendering library	Provides ball definitions and SVG helpers used by the NFT metadata contracts.

- Ticket price is fixed at 1e18 token units per ticket, meaning the deployed token is assumed to use 18 decimals.
- Maximum batch buy size is 50 tickets per transaction.
- Winner determination is computed from exact-position consecutive streaks, not unordered matching.
- Draw progression after close is intentionally permissionless: any user can trigger the next valid transition.
- The protocol keeps jackpot accounting and host-fee accounting separate inside each draw.

2. Decentralization Concept

Windfall Lotto is not purely adminless, but it is deliberately structured so that the operational flow of each draw can continue without requiring the host treasury to be online. This is the protocol's main decentralization concept: administration exists for initial configuration and a few bounded parameter controls, while the weekly economic and settlement path is publicly executable.

How decentralization appears in the implementation

Area	Current implementation
Ticket purchase	Permissionless while the draw is OPEN and before endTime.
Jackpot donations	Permissionless while the draw is OPEN and before endTime.
Closing a draw	Permissionless; any caller can request randomness or escalate to the next fallback path when timeouts are reached.

Area	Current implementation
Reveal finalization	Permissionless once randomness is delivered or blockhash fallback becomes ready.
Ticket counting	Permissionless and batched, reducing dependence on a single operator.
Tier finalization	Permissionless once all tickets for the draw have been processed.
Winner claims	Self-service by the owner of each winning ticket.
Opening next draw	Permissionless after the prior draw reaches COUNTING_DONE.
Admin powers retained	Host treasury can set Chainlink callback gas limit, Chainlink confirmation count, and Supra confirmation count. Child contracts also require one-time lotto address wiring by host.

This produces a hybrid model: the host treasury still matters for deployment, treasury ownership, and a small parameter surface, but draw execution, randomness escalation, counting, and claims can all proceed without centralized intervention. In practical terms, the protocol behaves as a publicly runnable machine once deployed and linked.

The child contracts reinforce this model through one-time wiring. WindfallTicket, WindfallDrawNFT, and WindfallFeeShare each expose setLotto(address), callable only by the host treasury and intended to be used once. After linkage, their privileged actions are restricted to the core lotto contract.

3. Ticket Model and Winning Logic

Each ticket stores five uint8 values packed into a bytes32, plus the draw id, draw end timestamp, and original minter address. Numbers must each be between 0 and 99 inclusive. The implementation explicitly allows duplicates; there is no uniqueness check across the five positions.

Ticket model details

Item	Behavior
NFT standard	ERC721Enumerable

Item	Behavior
Mint authority	Only WindfallLotto can mint tickets
Per-ticket price	1 token (1e18 base units)
Allowed number range	0-99 for each of five slots
Duplicates	Allowed
Tier storage	Updated during processing through setTicketTier(tokenId, tier)
Metadata format	Fully on-chain Base64 JSON + Base64 SVG image
Utility view	tokensOfOwner(address) returns owned ticket ids

The winning algorithm derives a five-number winning array from a random word. It then compares each ticket against that array using _matchTier(). The algorithm walks left to right and measures the longest consecutive streak of exact-position matches. A streak of 3 yields tier 3, a streak of 4 yields tier 4, and a perfect streak of 5 yields tier 5. Any lower result is a non-winning ticket.

Examples of streak-based matching

Ticket vs winning numbers	Result
[11,22,33,44,55] vs [11,22,33,77,88]	Tier 3 - first three positions form the longest consecutive streak
[10,20,99,40,50] vs [10,20,30,40,50]	Tier 2 is not a winning tier; final result is 0
[01,02,03,04,99] vs [01,02,03,04,05]	Tier 4
[77,10,11,12,13] vs [99,10,11,12,13]	Tier 4 - last four positions form a consecutive streak
[12,12,12,12,12] vs [12,12,12,12,12]	Tier 5

4. Draw Lifecycle and State Machine

The Draw struct stores both economic state and operational state. Key fields include endTime, jackpot, hostfee, hostfeeSent, the winning number array, randomness request identifiers, ticket id range, batch-processing cursor, tier counts, active winning tier, payoutPerWinner, rollover remainder, and paidWinners.

Draw states in code

State	Meaning
OPEN	Ticket sales and jackpot donations are allowed.
CHAINLINK_REQUESTED	Draw closed; Chainlink VRF request is live.
SUPRA_REQUESTED	Chainlink timed out; Supra dVRF request is live.
BLOCKHASH_PENDING	Supra timed out; protocol waits for a future block number to become usable.
REVEALED	Winning numbers have been derived and stored. Ticket counting can proceed.
COUNTING_DONE	Tier counts and payout data are finalized; claims and next-draw opening are enabled.

- A draw starts in OPEN when _openNewDraw() sets the next Friday at 23:00 UTC as endTime.
- If a draw closes with no tickets sold, the contract skips randomness entirely and marks the draw as COUNTING_DONE with a full jackpot rollover.
- If tickets exist, closeDrawAndRequestRandom() advances the draw through its next valid randomness stage.
- Ticket counting is capped at 500 tickets per call, allowing public batch participation and reducing gas griefing risk.
- openNextDraw() automatically mints the draw result NFTs and distributes host fees before opening the next weekly draw.
- openNextDrawLight() exists as an emergency path that opens the next draw without NFT minting or fee distribution.

5. Randomness and Fallback Architecture

The protocol uses a three-step randomness path. First it requests Chainlink VRF v2.5. If Chainlink does not resolve within CHAINLINK_TIMEOUT, the system escalates to Supra dVRF. If Supra does

not resolve within SUPRA_TIMEOUT, the system arms a blockhash-based fallback using a block number seven blocks ahead of the arming transaction.

Randomness pipeline

Stage	Trigger	Parameters / notes
Chainlink VRF	Default on close	1 random word, configurable callback gas limit and request confirmations, subId/keyHash fixed at deployment.
Supra dVRF	If Chainlink exceeds 1 hour timeout	Uses SUPRA_RNG_COUNT = 1 and a client seed derived from block.prevrandao, block.timestamp, draw id, jackpot, and ticket-range context.
Blockhash fallback	If Supra exceeds 2 hour timeout	Arms a future block at current block + 7. If blockhash is later unavailable, the contract loops back to Chainlink instead of getting permanently stuck.

This design gives the protocol liveness resilience. A single oracle outage does not permanently halt the system. Just as important, the blockhash path includes a recovery branch: if the historical blockhash is unavailable and returns zero, the contract re-requests Chainlink randomness instead of remaining frozen.

The random word is expanded into five numbers by taking modulo 100 on the base word for the first number and on four hashed derivations for the remaining positions. Because the modulo is applied independently to each position, duplicate winning numbers are possible and are consistent with the ticket rules.

6. Jackpot, Windfall Logic, and Tier Payouts

Every ticket purchase contributes to two internal balances inside the active draw. Ten percent of the ticket price is accumulated as hostfee. The remaining ninety percent is added to the draw jackpot. Direct donations bypass the host-fee logic and increase only the jackpot. At finalization, the protocol first determines which tier is active: tier 5 if any tier-5 winners exist, otherwise tier 4 if any tier-4 winners exist, otherwise tier 3 if any tier-3 winners exist, otherwise no active tier and full rollover.

Normal payout shares when jackpot is below the windfall trigger

Active winning tier	Distributable share of jackpot	Rollover behavior
Tier 5	80%	Remaining 20% plus any integer-division dust rolls into the next draw
Tier 4	20%	Remaining 80% plus dust rolls into the next draw
Tier 3	5%	Remaining 95% plus dust rolls into the next draw
No winners	0%	100% rolls into the next draw

The code also introduces a windfall threshold: WINDFALL_TRIGGER = 1e27 base units, which corresponds to 1,000,000,000 whole tokens under an 18-decimal token. Once the draw jackpot reaches or exceeds that threshold, the distributable share becomes 80% of the jackpot regardless of whether the active winning tier is 3, 4, or 5. In other words, the qualifying tier still decides who may claim, but the payout percentage is elevated to the tier-5 share level.

Windfall behavior at or above 1B tokens

Condition	Effect
Jackpot >= 1,000,000,000 tokens and at least one active winning tier exists	Distributable amount is forced to 80% of jackpot
Active tier remains determined by actual streak counts	Tier 3 or tier 4 winners may receive the 80% distributable pool if they are the highest existing winners
Per-winner payout	Computed as floor(distributable / activeWinners)
Remainder	All unused dust and undistributed jackpot balance roll forward

Claim settlement includes a secondary economic split tied to NFT transferability. If the winning ticket is still held by its original minter, the claimer receives the full payoutPerWinner. If the ticket

has been transferred, 10% of the gross winner payout is redirected to the original minter and 90% is sent to the current owner. This creates a continuing royalty right for the original ticket minter.

7. Shareholders Concept and FeeShare Economics

Windfall Lotto separates jackpot funding from host-fee redistribution. The jackpot belongs to the lottery economy. Host fees belong to a separate share-distribution economy implemented in WindfallFeeShare. This system turns qualifying donors into temporary shareholders who accrue claimable balances when host fees are distributed.

FeeShare constants and mechanics

Parameter	Current behavior
THE_MIN	1,000 tokens
Initial minQualifyingDonation	1,000 tokens
Initial minQualifyingDonationToSave	100 tokens
Share duration	365 days
Maximum tracked donor shareholders	200 addresses
Host treasury	Always active for fee-distribution purposes

Qualification works in two stages. First, donations below minQualifyingDonationToSave are ignored by the shareholder registry. Second, a donor becomes active only after its accumulated qualifying amount reaches minQualifyingDonation. Once active, the donor receives an expiry timestamp. Additional qualifying donations extend the expiry proportionally to the current minQualifyingDonation threshold.

The threshold itself is dynamic. As active donor count rises, minQualifyingDonation escalates in steps. In the current code it stays at 1,000 tokens up to 35 active donors, rises to 10,000 above 35, 100,000 above 70, 1,000,000 above 105, 10,000,000 above 140, and 100,000,000 above 175. The storage threshold minQualifyingDonationToSave always tracks one tenth of the active qualification threshold.

Dynamic qualification thresholds

Active donors	minQualifyingDonation
0-35	1,000 tokens
36-70	10,000 tokens

Active donors	minQualifyingDonation
71-105	100,000 tokens
106-140	1,000,000 tokens
141-175	10,000,000 tokens
176-200	100,000,000 tokens

Expired shareholders are not necessarily removed immediately. Their active flag can be disabled, and after 30 days beyond lastStateChange they can be pruned from the registry entirely. If an inactive donor had an unfinished accumulation balance and then remained inactive for more than 30 days, that accumulation is reset before new accumulation continues. This means continuity matters for reaching qualification.

When host fees are distributed, the contract counts active donors and then gives the host treasury a variable number of shares equal to ceil(activeDonors / 20), with a minimum of one host share. Active donor shares plus host shares form the activeShares denominator. Each share gets the same sharePerMember. The host treasury receives its own shares plus any indivisible remainder from integer division. Each active donor receives one share. All balances are stored as claimable amounts until each participant calls claim().

8. NFT Layer and Metadata Design

Windfall Lotto uses NFTs both as playable tickets and as historical draw certificates. Ticket NFTs are the user-facing game asset. Draw NFTs are protocol records that memorialize the final result of a completed draw.

NFT inventory by completed draw

Asset	Minted to	Transferability	Purpose
Windfall Ticket	Ticket buyer	Transferable	Represents a playable claim-bearing lottery position.
Host draw NFT	hostTreasury	Transferable	Collectible and archival result NFT for the host side.

Asset	Minted to	Transferability	Purpose
Contract draw NFT	WindfallLotto contract	Non-transferable once held by the lotto contract	Permanent protocol-held result record.

Both NFT contracts generate fully on-chain metadata. The tokenURI output is Base64-encoded JSON whose image field is itself a Base64-encoded SVG. Ticket artwork changes according to tier color state: default neon background for non-winners, gold for tier 5, silver for tier 4, and bronze for tier 3. The result NFT stores draw id, reveal timestamp, jackpot snapshot, packed winning numbers, active tier, and active winner count.

Because the ticket contract is ERC721Enumerable and also exposes tokensOfOwner(address), off-chain interfaces can reconstruct player portfolios and per-draw ownership views without relying on a centralized database. The mapping claimed[ticketId] in WindfallLotto also supports direct on-chain claim status checks for UI rendering.

9. Operational Flow of a Full Draw

- 1. A new draw opens automatically through _openNewDraw(). endTime is set to the next Friday at 23:00 UTC and jackpot begins with the carried remainder from the prior draw.
- 2. Users buy one or multiple tickets. Each purchase mints ticket NFTs and updates draw jackpot and hostfee balances.
- 3. Users may donate directly to the jackpot. Eligible donations also register in the FeeShare contract for shareholder qualification.
- 4. After endTime, any caller can invoke closeDrawAndRequestRandom() to start or advance the randomness pipeline.
- 5. Once randomness resolves, the contract stores the winning array and marks the draw as REVEALED.
- 6. Any caller may process tickets in batches of up to 500 ids. The protocol counts tier-3, tier-4, and tier-5 winners and writes ticket tier state into the NFT contract.
- 7. Any caller may finalize the tier once all ticket ids in the draw range have been processed. This locks activeTier, activeWinners, payoutPerWinner, and remainder.
- 8. Winning ticket holders claim individually. If the ticket moved away from the original minter, the minter receives the royalty split automatically.
- 9. Any caller may open the next draw. This carries remainder, mints draw NFTs, and distributes the accumulated host fee through the shareholder system.

10. Trust Assumptions and Security Surface

The contracts reduce operator dependence, but they do not eliminate trust assumptions entirely. Users still trust the initial deployment, immutable addresses, and the correctness of the linked child contracts. They also trust the host treasury not to misuse its narrow configuration rights, such as tuning randomness confirmation parameters to values that are technically valid but operationally poor.

Main trust and risk observations from the current implementation

Topic	Observation
Admin scope	Host treasury can adjust Chainlink callback gas, Chainlink confirmations, and Supra confirmations. These are bounded but still meaningful.
Liveness	Public callers can move the protocol forward through close, fallback escalation, processing, finalization, and next-draw opening.
Oracle dependency	A single oracle failure does not halt the system because the pipeline escalates from Chainlink to Supra to blockhash.
Gas scaling	Winner counting is explicitly batched with MAX_PROCESS_BATCH = 500.
Claim safety	Claims are protected by nonReentrant guards and an overpayment check tied to activeWinners.
Economic determinism	PayoutPerWinner is locked at finalization and does not change during subsequent claims.
UI integrity	Most user-visible status can be derived directly from chain data: ticket ownership, tier, draw state, winning numbers, and claim status.

A notable implementation detail is that openNextDrawLight() skips both draw-NFT minting and host-fee distribution. It should therefore be treated as an emergency liveness path rather than the preferred routine path, because it omits economically and archivally important side effects that the standard openNextDraw() function performs.

11. Key Constants Appendix

Constant	Value	Meaning
TICKET_PRICE	1e18	One whole token per ticket
HOST_FEE_BPS	1000	10% of each ticket purchase goes to hostfee
MINTER_ROYALTY_BPS	1000	10% of winner payout goes to original minter if transferred
TIER5_SHARE_BPS	8000	80% distributable for tier-5 winner scenario
TIER4_SHARE_BPS	2000	20% distributable for tier-4 winner scenario
TIER3_SHARE_BPS	500	5% distributable for tier-3 winner scenario
WINDFALL_TRIGGER	1e27	1,000,000,000 tokens with 18 decimals
MAX_TICKETS_PER_BUY	50	Upper bound on batch purchase size
MAX_PROCESS_BATCH	500	Upper bound on per-call ticket counting batch
CHAINLINK_TIMEOUT	1 hour	Delay before escalation to Supra
SUPRA_TIMEOUT	2 hours	Delay before escalation to blockhash
BLOCKHASH_DELAY	7 blocks	Future block used for blockhash fallback
DRAW_NFTS_PER_DRAW	2	One result NFT to host treasury, one to the lotto contract

THE_MIN (FeeShare)	1000e18	Base shareholder qualification threshold
SHARE_DURATION	365 days	Base active period granted after qualification
MAX_SHAREHOLDERS	200	Maximum number of tracked donor shareholders

12. Conclusion

The current Windfall Lotto implementation is best understood as a public lottery engine with three intertwined layers: a permissionless draw machine, a collectible NFT layer, and a donor-based fee-sharing economy. Its most distinctive elements are the consecutive-streak winning model, the windfall jackpot trigger that upgrades payout percentage economics at extreme jackpot sizes, and the shareholder system that converts donation behavior into recurring participation in host-fee distribution.

From a protocol-design perspective, the code emphasizes liveness and composability. Anyone can help the system progress, winners claim directly, tickets remain transferable, and result NFTs create a durable on-chain record of each completed draw. The remaining centralized surfaces are comparatively narrow and mostly operational. As a result, the project sits in a strong middle ground between a fully host-operated raffle and a purely static autonomous contract system.

For investors, players, and auditors, the most important takeaway is that the economic truth of Windfall Lotto is explicit in the code: jackpot flows, rollover behavior, fee distribution, qualification thresholds, payout percentages, and claim routing are all deterministic and inspectable on-chain.