Powerball $BALL$:

The First Hourly, Automatic, On-Chain Lottery for Token Holders

Anonymous Team

June 3, 2025

Abstract

Powerball $BALLisafullydecentralized,provablyfair,andautomatedlotterysystemonEthereum.Bysimp$
tokens, users are automatically entered into hourly draws for a chance to win the entire ETH prize pool, funded by a transaction tax. The system leverages Chainlink VRF for verifiable randomness and a Merkle root mechanism for efficient, privacy-preserving eligibility verification. This whitepaper details the protocol's design, tokenomics, security, and technical implementation.

Contents

1 Introduction

1.1 Motivation

Traditional lotteries are opaque, centralized, and often require manual participation. Powerball $BALLaimstorevolutionizethelotteryexperiencebymakingitautomatic, transparent, andprovablyfair,$

1.2 Key Features

- **Automatic Entry:** Hold at least 10,000 $BALLtokenstobeeligibleforeveryhourlydraw.$**Hourly Dra**

- **Provably Fair:** Chainlink VRF ensures tamper-proof, auditable randomness.
- **Instant Payouts:** Winners receive ETH directly to their wallet, automatically.
- **No Custody:** No staking or lockup; users always control their tokens.
- **Open Source & Audited:** All contracts are open source and will be audited.

2 Tokenomics

2.1 Token Overview

- **Name:** Powerball $BALL$**Symbol:**BALL
- **Standard:** ERC20
- **Total Supply:** 10,000,000
- **Decimals:** 9
- **Initial Liquidity:** 100% of supply paired with 1 ETH on Uniswap V2

2.2 Transaction Tax

- **Tax Rate:** 5% on all buys and sells
- **Destination:** All tax proceeds are swapped for ETH and sent to the lottery contract, forming the prize pool for each draw.

3 Lottery Mechanism

3.1 Eligibility

- Any address holding $\geq$ 10,000 $BALLtokensatthesnapshottimeiseligible.Eligibilityisdeterminedoff-chainviaasnapshotandpublishedon-chainasaMerkleroot.$

3.2 Draw Cycle

1. **Snapshot:** Off-chain service snapshots all eligible addresses and constructs a Merkle tree.
2. **Merkle Root Submission:** The Merkle root and participant count are submitted on-chain by a trusted submitter.
3. **Randomness Request:** Anyone can trigger a Chainlink VRF request for the current draw.
4. **Winner Selection:** When VRF fulfills, the random number determines the winner's index in the Merkle tree.
5. **Winner Proof:** The off-chain service submits the winner's address and Merkle proof for verification.
6. **Payout:** The contract automatically transfers the entire ETH prize pool to the winner.

3.3 Prize Pool Funding

- The prize pool is funded by ETH sent from the $BALLtokencontract'staxmechanism.AnyonecansendETH$

4 Technical Architecture

4.1 Smart Contract Overview

- Written in Solidity `0.8.20+`
- Uses OpenZeppelin's `Ownable`, `ReentrancyGuard`, and `MerkleProof`
- Integrates Chainlink VRF v2 for randomness
- Prize payouts in ETH only
- Emergency recovery functions for stuck ETH/tokens

4.2 Key State Variables

- `ballToken`: Address of the $BALLERC20token(immutable)$`DRAW_INTERVAL` : $1hour(3600seconds)$
- `currentDrawId`: Incrementing draw counter
- `Draw struct`: Stores Merkle root, participant count, VRF request ID, winner, prize status, etc.
- `merkleRootSubmitter`: Trusted off-chain service address

4.3 Core Functions

- `requestDraw()`: Starts a new draw if interval has passed
- `submitMerkleRoot(bytes32 root, uint256 count)`: Sets Merkle root and participant count for the draw
- `requestRandomWinner()`: Requests randomness from Chainlink VRF
- `fulfillRandomWords()`: VRF callback, stores winner index
- `submitWinner(drawId, winner, proof, index)`: Verifies winner and pays out ETH

- `recoverStuckETH() / recoverStuckTokens()`: Owner can recover stuck ETH/tokens (except *BALL*)

4.4 Security Features

- **ReentrancyGuard:** All payout functions are non-reentrant.
- **Access Control:** Only owner or trusted submitter can call sensitive functions.
- **Checks-Effects-Interactions:** State is updated before external calls.
- **Custom Errors:** Gas-efficient error handling.
- **Input Validation:** All user input is validated.
- **No Upgradability:** For simplicity and auditability.

4.5 Chainlink VRF Integration

- **Coordinator:** Set at deployment
- **Key Hash, Subscription ID, Callback Gas Limit, Confirmations:** Configurable by owner
- **Randomness Request:** Anyone can trigger after Merkle root is set
- **Callback:** Only VRF coordinator can call

4.6 Merkle Proof Verification

- Uses OpenZeppelin's `MerkleProof` library
- Winner's address and index are verified against the Merkle root
- Only the trusted submitter can submit the winner/proof

5 Security Considerations

5.1 Attack Vectors and Mitigations

- **Reentrancy:** All external calls are protected by `nonReentrant`
- **Oracle Manipulation:** Chainlink VRF is used for unbiased randomness
- **Access Control:** Only trusted addresses can submit Merkle roots and winners
- **Prize Pool Safety:** Emergency recovery cannot interfere with active draws
- **Gas Griefing:** No unbounded loops; all loops are over small, bounded sets
- **Front-running:** Draw and winner selection are fully on-chain and cannot be manipulated

5.2 Auditing

- All contracts will be audited by a reputable third-party firm before mainnet launch.
- Audit reports will be published for public review.

6 Governance and Upgrades

- The contract is not upgradeable for maximum security and auditability.
- Future governance (e.g., DAO, voting) may be introduced via new contracts.

7 Roadmap

1. **Testnet Launch & Public Testing**
2. **Audit & Security Review**
3. **Mainnet Launch**
4. **Community Growth & Marketing**
5. **Feature Expansion (Governance, New Games, etc.)**

8 Risks and Disclaimers

- Participation involves risk, including the potential loss of principal.
- The $BALLtokenisautilitytokenforparticipationinthelottery, notaninvestmentsecurity.Thisdocum$

- Users are responsible for compliance with their local laws and regulations.

9 Contact and Community

- **Website:** `https://powerballeth.com`
- **Telegram:** [Your Telegram Link]
- **Twitter/X:** [Your Twitter Link]
- **Medium:** [Your Medium Link]
- **Email:** contact@powerballball.com
- **GitHub:** [Your GitHub Link]

10 Glossary of Terms

Powerball *BALL*	The name of the project and the ERC20 token used for eligibility in the lottery.
ETH	Ether, the native currency of the Ethereum blockchain, used for prize payouts.
ERC20	A widely-used standard for fungible tokens on Ethereum.
Chainlink VRF	Chainlink's Verifiable Random Function, a decentralized oracle service providing provably fair and tamper-proof randomness to smart contracts.
Merkle Root	The root hash of a Merkle tree, used to efficiently and securely prove the inclusion of an address in a set (e.g., eligible lottery participants).
Merkle Proof	A cryptographic proof that a specific address is included in a Merkle tree with a given root.
Snapshot	An off-chain process that records all addresses holding at least 10,000 *BALLataspecifictime*
Prize Pool	The total ETH available to be won in a given draw, funded by transaction taxes and direct contributions.
Reentrancy	A class of smart contract vulnerability where an external call can re-enter the contract and manipulate state.
Non-custodial Wallet	A wallet where the user controls the private keys, as opposed to a centralized exchange wallet.
Uniswap V2	A decentralized exchange protocol on Ethereum, used for initial *BALLliquidity.DecentralizedAutonomous* *governedstructureforprotocolupgradesanddecisions*.
OpenZeppelin	A library of secure, community-vetted smart contract components.
VRF Coordinator	The Chainlink contract responsible for managing VRF requests and responses.
Callback Gas Limit	The maximum gas allowed for the VRF callback function.
Subscription ID	The identifier for a Chainlink VRF subscription, used to pay for randomness requests.

11 Live Stats and Analytics

Powerball $BALLprovidesatransparentandengaginguserexperiencebydisplayinglivestatisticsandanalytic$

11.1 Displayed Metrics

- **Next Draw Countdown:** Real-time countdown timer showing when the next hourly draw will occur.
- **Current Estimated Jackpot:** The current ETH prize pool available to be won in the next draw.
- **Total** $BALLHoldersEntered : Thenumberofuniqueaddresseseligibleforthecurrentdraw(holdinga$
- **Total Prizes Paid Out: The cumulative amount of ETH distributed to winners since launch.**
- **Current Draw Number: The sequential identifier for the ongoing draw.**
- **Recent Winners: A list of the last three winners, including their address, prize amount, and timestamp, with links to verify each transaction on Etherscan.**

11.2 Data Sources and Transparency

- All stats are fetched from on-chain data using web3 calls or subgraph queries.
- The frontend code is open source, allowing anyone to verify how stats are calculated and displayed.
- Users can independently verify all draws, winners, and payouts on Etherscan or via the contract's public events.

11.3 Importance for Users

- Live stats foster trust and engagement by making the lottery process fully transparent.
- Users can track their eligibility, jackpot size, and historical results in real time.
- Public analytics help demonstrate the protocol's fairness and success to new users and the broader community.