ONI GAMES

Transparency, Provable Fairness & On-Chain Verifiability

Technical Whitepaper | Version 1.0 | 2026

1. Introduction

Oni Games is a Web3 arcade and GambleFi platform built on the OneChain blockchain. Every game outcome, every wager, and every payout is governed by open-source smart contracts — not hidden algorithms. This document explains exactly how fairness is achieved and how anyone can independently verify it.

Traditional online casinos ask players to trust a company's word that their random number generators are fair. Oni Games eliminates this requirement entirely. Because our casino and game portal logic is deployed as immutable on-chain contracts, the rules cannot be changed mid-game, outcomes cannot be manipulated retroactively, and no one — including the Oni Games team — can alter a result after a bet is placed.

CORE CLAIM	*Every game outcome on Oni Games is determined by code deployed to the blockchain. No outcome can be altered by any party after a wager is locked.*

2. Smart Contract Architecture

Oni Games operates through two primary smart contracts, each handling a distinct layer of the platform:

Contract Module	Responsibility
oni_games::casino	Handles all casino game logic — instant wagers (Coin Flip, Dice, Roulette) and session-based games (Crash, Minesweeper, High-Low).
oni_games::game_portal	Handles Score NFT minting, on-chain leaderboards per game, and the NFT marketplace with configurable platform fees.

2.1 Open Source & Immutable

Both contracts are published to the OneChain blockchain. Their source code is publicly accessible. After deployment, the core game logic cannot be modified. Admin functions exist only for operational parameters (fee rates, bet limits, treasury address, pause switch) and cannot change the outcome logic of any game.

2.2 Shared Bankroll (HouseBankroll)

All player wagers and house payouts flow through a single on-chain object called HouseBankroll. Its current balance, lifetime wager volume, lifetime payout volume, and total

games played are all readable by anyone on-chain at any time. There are no hidden reserves or off-chain funds involved in game resolution.

3. Randomness — How Outcomes Are Generated

The single most important fairness question in any casino game is: who controls the random number? Oni Games does not generate random numbers on our servers. All instant games use OneChain's native on-chain randomness module.

3.1 OneChain Native RNG (one::random)

For instant-play games (Category A), randomness is sourced from OneChain's built-in random module. This works as follows:

- A random generator is seeded using on-chain context from the transaction itself (block hash, epoch data, and transaction digest).
- The result is computed inside the smart contract's execution — no external call, no server involvement.
- The random value is generated after the bet is locked. It is not possible to know the result before committing a wager.

```
let mut generator = random::new_generator(rng, ctx);let result =
random::generate_u64_in_range(&mut generator, 0, game_range - 1);let won = (result
== guess);
```

The random module is part of OneChain's core framework and is not under Oni Games' control. We consume it as a read-only dependency.

3.2 PTB Rollback Protection

A known attack vector in blockchain gaming is Programmable Transaction Block (PTB) composition: a malicious actor could compose a transaction that plays a game and then inspects the outcome — rolling back the transaction if they lose, and only submitting it if they win.

SECURITY	*All instant-play game functions are declared as `entry` functions, not `public`. This prevents them from being composed inside a PTB, completely eliminating the rollback exploit.*

Because the functions are entry-only, they must be the terminal step of a transaction. A player cannot conditionally abort after seeing the result.

3.3 Verifiable Outcomes

Every game played emits an on-chain event containing:

- Player address
- Game identifier
- Wager amount
- Player's guess / selection
- Actual random result
- Win/loss status
- Payout amount
- Multiplier in basis points

These events are permanently recorded on the blockchain. Any player can retrospectively verify their outcomes by querying blockchain explorers or indexers using their wallet address.

4. Session-Based Games (Two-Step Escrow)

Some games — Crypto Crash, Minesweeper, High-Low — involve real-time state that cannot be resolved in a single transaction. These use a two-step escrow model:

Step	Description
Step 1: lock_wager	Player submits wager to the blockchain. Funds are immediately deposited into the HouseBankroll. A SessionReceipt NFT is issued to the player as cryptographic proof of their bet.
Step 2: resolve_session	After the game concludes off-chain, the backend signs the final multiplier with its Ed25519 private key. The contract verifies this signature on-chain before releasing any payout.

4.1 What the Signed Message Contains

The backend cannot forge an arbitrary payout. The signature covers a specific, tamper-evident message:

```
Message = BCS( session_id | player_address | multiplier_bps | nonce )
```

- session_id — unique ID of the on-chain session object; cannot be reused.
- player_address — binds the payout to the specific player who created the session.
- multiplier_bps — the outcome expressed in basis points (0 = loss, 10000 = 1x, 25000 = 2.5x, etc.).
- nonce — a unique single-use number preventing replay attacks.

SECURITY

Once a session is created on-chain, the player's wallet address is locked into the receipt. No one can redirect a payout to a different address — not even the Oni Games backend.

4.2 Replay Attack Prevention

The contract maintains a used_nonces table on-chain. Every nonce submitted with a session resolution is permanently recorded. Any attempt to reuse a signed message will be rejected at the contract level, regardless of whether the signature is valid.

4.3 Fund Flow Guarantee

Funds are deposited into the HouseBankroll at the moment lock_wager is called — before any game play occurs. There is no possibility of the house withholding a player's wager mid-session. If a session is never resolved, the wager remains visible on-chain and can be audited.

5. House Edge & Payout Limits

5.1 House Edge

The house edge is embedded in the multiplier_bps passed to each game function. For example, a Coin Flip has two equally probable outcomes. A perfectly fair game would pay 2.0x (20,000 bps). Oni Games' default configuration pays 1.96x (19,600 bps), representing a 2% house edge on this game type.

The house_edge_bps value stored in HouseBankroll is an informational parameter. The actual effective house edge for any game is determined by the multiplier used when calling the play functions. Both are publicly readable on-chain.

5.2 Maximum Payout Cap

To protect the bankroll from catastrophic single-event risk, a max_payout_bps limit is enforced per payout. By default, no single payout can exceed 5% of the total bankroll at the time of resolution. This check is enforced at the contract level:

```
let max_allowed_payout = (balance::value(&bankroll.balance) *
bankroll.max_payout_bps) / MAX_BPS;assert!(potential_payout <= max_allowed_payout +
wager_amount, E_PAYOUT_EXCEEDS_MAX);
```

This means that as the bankroll grows, maximum possible payouts grow proportionally. The current bankroll size and thus the current payout cap are always publicly queryable.

5.3 Bet Limits

Minimum and maximum bet limits (min_bet and max_bet) are stored transparently in HouseBankroll and publicly readable. Any change to these limits is a blockchain transaction — not a silent backend configuration change.

Parameter	Default Value
Minimum Bet	0.01 OCT (10,000,000 MIST) — configurable by admin
Maximum Bet	10 OCT (10,000,000,000 MIST) — configurable by admin
Max Payout	5% of bankroll per single payout — configurable by admin
House Edge	2.5% default — varies by game type

6. Score NFTs & Leaderboards

6.1 Score Verification

Score NFTs are minted via one of two paths:

- Admin-Minted (Gasless): The Oni Games backend verifies the score and submits the mint transaction, covering the player's gas cost. Nonces prevent any score from being minted twice.
- User-Submitted (Signature-Verified): The backend signs a message binding the player's address, game ID, score, and a unique nonce. The player submits this on-chain. The contract verifies the signature before minting.

In both cases, scores cannot be fabricated. A player cannot mint a score they did not legitimately achieve, because the backend is the sole holder of the signing key.

6.2 On-Chain Leaderboards

Leaderboards are stored entirely on-chain inside the GameStore object as a Table mapping game IDs to sorted score vectors. The top 10 entries per game are maintained automatically by the smart contract on every mint. There is no off-chain database involved — the leaderboard state is the blockchain state.

TRANSPARENCY	*The leaderboard you see in the Oni Games UI is read directly from the blockchain. It cannot be manually edited by the team.*

6.4 Can Someone Call the Mint Function Directly and Fake a Score?

This is the most common question raised about on-chain NFT minting — and a legitimate one. The short answer is: no. Here is why, in full technical detail.

The mint_verified_score function is public, meaning anyone can call it directly from any wallet or script without going through the Oni Games frontend. However, calling it successfully requires

passing a valid Ed25519 signature produced by Oni Games' backend server private key. Without that signature, the contract will unconditionally reject the transaction.

```
// The contract verifies this BEFORE doing anything else:assert!(
ed25519::ed25519_verify(&signature, &store.server_public_key, &msg),
E_INVALID_SIGNATURE,);
```

To forge a valid signature, an attacker would need to either: (a) obtain the server's private key — which never leaves the backend and is not embedded in the frontend, or (b) break Ed25519 — a cryptographic primitive considered computationally infeasible to crack with current or near-future hardware.

The signed message itself is also structured to prevent misuse even if a real signature were somehow intercepted:

- player — the signature encodes the caller's wallet address. If an attacker intercepts a valid signature issued for player A and replays it from their own wallet (player B), the reconstructed message will not match the signature. The transaction fails.
- game_id + score — the exact game and score are baked into the payload. A signature for a Tetris score of 5,000 cannot mint a score of 5,000,000 or a different game's NFT.
- nonce — every signature includes a unique single-use nonce. Even if a genuine signature were somehow captured in transit and resubmitted, the nonce table on-chain will reject it as already used.

CANNOT FAKE	*A direct call to mint_verified_score with a fabricated or replayed signature will always fail at the Ed25519 verification step. The contract has no fallback or bypass path.*

For the admin-minted path (admin_mint_score_nft), the protection is even more fundamental. That function requires passing a reference to the AdminCap object — a unique on-chain capability object held only by the Oni Games deployer wallet. A non-admin calling this function will be rejected at the Move object-capability layer; the object does not exist in their account and cannot be spoofed.

In summary: minting a fake score requires either breaking Ed25519 cryptography or stealing the Oni Games server private key — neither of which is possible through any smart contract interaction.

6.5 NFT Marketplace

The NFT marketplace is entirely on-chain. When a player lists an NFT for sale, the NFT is transferred into a shared Listing object (escrow) on the blockchain. The seller retains no custody once listed. The marketplace fee (2.5% default) is applied transparently:

```
let platform_fee_amount = (price * store.market_fee_bps) / MAX_BPS;// Fee ->
treasury// Remaining -> seller// Change -> buyer (refunded automatically)
```

There are no hidden fees. Every price, fee, and fund transfer is executed by the contract and recorded as an on-chain event.

7. Admin Capabilities & Security Boundaries

7.1 What Admins Can Do

Admin functions are restricted to holders of the CasinoAdminCap and AdminCap objects. These capabilities allow:

- Adjusting bet limits (min/max), house edge bps, and max payout bps
- Updating the treasury address for fee collection
- Funding or withdrawing liquidity from the bankroll
- Setting the Ed25519 server public key used for session verification
- Pausing the casino in an emergency
- Minting Score NFTs and setting mint/marketplace fees

7.2 What Admins Cannot Do

CA NN OT DO	*Admin capability holders cannot alter the outcome of any game that has already been played. They cannot modify on-chain event history. They cannot redirect a player's payout to a different address. They cannot fake a server signature.*

The admin capability is an object on the blockchain. Its holder is publicly visible. Any exercise of admin power is a signed blockchain transaction — permanently logged and attributable.

7.3 Emergency Pause

A paused flag exists in HouseBankroll and is checkable by anyone. When paused, all play functions (play_instant_wager, play_range_wager, lock_wager) revert. This mechanism exists for the protection of players in the event of a discovered vulnerability — it cannot be used to selectively block specific players or outcomes.

8. How to Independently Verify

Oni Games encourages players and auditors to verify all claims in this document directly on-chain. Here is how:

8.1 Read the Smart Contracts

- Navigate to the OneChain block explorer.
- Look up the package address for oni_games.

- The source code of both modules is viewable directly in the explorer.

8.2 Check Bankroll Status

- Look up the HouseBankroll shared object.
- Read balance, total_wagers, total_payouts, total_games, min_bet, max_bet, max_payout_bps, paused.
- All fields are publicly readable — no API key or authentication needed.

8.3 Verify Your Game History

- Query InstantWagerPlayed and RangeWagerPlayed events filtered by your wallet address.
- Every event contains the random result, your guess, and payout — all on-chain and immutable.
- Session games emit SessionCreated and SessionResolved events with identical transparency.

8.4 Check Leaderboard Integrity

- Look up the GameStore shared object.
- Inspect the leaderboards table for any game_id.
- Entries are sorted descending by score and capped at the top 10 — enforced by contract logic.

8.5 Audit Nonce Usage

- The used_nonces tables in both HouseBankroll and GameStore are publicly readable.
- Any nonce that appears in the table has been used and cannot be replayed, regardless of whether the original signature is valid.

9. Glossary

Term	Definition
Basis Points (BPS)	A unit equal to 0.01%. 10,000 BPS = 100%. Used throughout to represent multipliers and fee rates precisely without floating-point arithmetic.
MIST	The smallest denomination of OCT (OneChain's native token). 1 OCT = 1,000,000,000 MIST.
PTB	Programmable Transaction Block — a feature of OneChain that allows composing multiple operations in one transaction. Oni Games prevents PTB composition of game functions to block rollback exploits.
Ed25519	A digital signature algorithm used for the server's signing key. Signatures are verified on-chain by the smart contract.

Term	Definition
BCS	Binary Canonical Serialization — a deterministic encoding format used to construct the messages that the server signs.
Nonce	A unique one-time number included in server signatures. Once used, the nonce is permanently recorded on-chain, preventing replay attacks.
HouseBankroll	The on-chain shared object holding all casino liquidity. All wagers flow in and all payouts flow out of this single pool.
SessionReceipt	An on-chain NFT issued to a player when they lock a wager. It is the cryptographic proof that the bet exists and cannot be backdated or forged.

10. Closing Statement

Provable fairness is not a marketing claim — it is a technical property. Oni Games achieves it by placing all game logic on an open, immutable blockchain, using a native randomness module that neither Oni Games nor any player can manipulate, and producing a permanent, queryable audit trail for every action taken on the platform.

We welcome scrutiny. The contracts are public. The events are public. The bankroll is public. If you believe you have found a discrepancy between what this document describes and what the contracts actually do, please contact our security team.

Oni Games — Building trust through transparency.