Functional Requirements for PTX v1

Document Information

- **Document Title**: PTX V1.0 Functional Requirements
- **Author**: Daniel Ames
- **Date Created**: 3rd November 2024
- **Version**: 1.0
- **Intended Audience**: General Distribution
- **Project Name**: Hemis
- **Approval Status**: Final

Introduction

The first version of Probabilistic Transactions (PTX) is designed to enable decentralised 1:1 games of chance, supporting secure and transparent participation. The platform provides an optional feature for creators to define an advantage condition in their game setup, where the initiator has a minor statistical benefit, implemented transparently. Gamemaster nodes manage these processes using a quorum-based consensus mechanism for Random Number Generation (RNG).

This document outlines the functional requirements for PTX v1, including RNG logic, optional advantage condition implementation, and transaction fee distribution.

1. Functional Requirements

1.1 Game Initialisation

Participants submit paired transactions containing:

- Participation amount (fungible tokens or NFTs).
- Shared secret to match the paired transactions.

- Game type (e.g., numerical roll, card draw, etc.).
- Optional advantage condition**:**
    - A creator-defined rule that provides a small statistical edge in the game setup.
    - Example: In a 1–100 numerical range, the creator's condition could be an additional outcome (e.g., `101`).

1.2 Quorum Selection

- A quorum of Gamemaster nodes is selected randomly to oversee the game process.
- Requirements:
    - Selection uses a Verifiable Random Function (VRF) to ensure fairness and security.
    - Configurable quorum size (e.g., 11 nodes) to balance decentralisation, performance and economic disincentive to bad actors..

1.3 Transaction Validation

The quorum validates:

- Both transactions are in the mempool.
- Participation amounts match.
- Shared secrets are identical.
- Advantage conditions (if enabled) are valid.
- Participants have sufficient balances for their transactions.

1.4 RNG Execution

RNG generates the game result, ensuring fairness and transparency. The process incorporates:

- Shared secret.
- Block hash or transaction ID for entropy.
- Quorum-generated random seed.

Advantage Condition:
The RNG logic accounts for the optional advantage condition defined by the creator. For example:

- Numerical games: The creator may define a special outcome (e.g., `101` in a 1–100 range).
- Card games: The creator could define a unique card or combination as a bonus condition.
- Lotteries: A small percentage of outcomes can be reserved as an additional success condition for the creator.

1.5 Consensus

- Quorum members reach consensus on the RNG result.
- A majority threshold (e.g., 3 of 5 signatures) is required for finalisation.

1.6 Settlement

- Outcome-Based Actions:
    - If a participant wins, their address receives the result according to the game rules.
    - If the creator-defined condition is met, the result is credited to the creator.
- Fees Distribution:
    - Gamemaster nodes receive a portion of the block reward as a transaction fee.
    - Any advantage condition-defined outcomes result in additional credits to the creator.

1.7 Transparency and Auditability

- RNG inputs, outputs, and all game metadata are stored on-chain for verification.
- Participants and creators can independently audit results to verify fairness and transparency.

1.8 High-Velocity Game Support

- The system must support multiple concurrent games, enabling high-frequency participation without bottlenecks.
- Quorums must validate and settle processes efficiently to handle rapid usage.

2. RNG with Advantage Condition: Pseudocode

```
# Inputs
shared_secret = "participant_shared_secret"     # Shared secret between participants
block_hash = "current_block_hash"               # Current block hash for added
randomness
tx_id_1 = "transaction_id_participant_1"        # Transaction ID of Participant 1
tx_id_2 = "transaction_id_participant_2"        # Transaction ID of Participant 2
quorum_seed = "quorum_random_seed"              # Random seed provided by the quorum
advantage_condition = [101]                     # Creator-defined special condition

# RNG Function
def generate_random_number(shared_secret, block_hash, tx_id_1, tx_id_2, quorum_seed,
max_value=100):
    combined_input = shared_secret + block_hash + tx_id_1 + tx_id_2 + quorum_seed
    hashed_input = hash_function(combined_input)  # Use SHA-256 or equivalent
    random_number = int(hashed_input, 16) % (max_value + 1)  # Generate random number
    return random_number

# Example RNG with Advantage Condition
```

```
game_result = generate_random_number(shared_secret, block_hash, tx_id_1, tx_id_2,
quorum_seed)

# Decision Logic
if game_result in advantage_condition:
    print("Creator Wins")                     # Creator-defined condition met
elif game_result > advantage_condition[0]:
    print("Participant 1 Wins")               # Participant 1 wins
else:
    print("Participant 2 Wins")               # Participant 2 wins
```

3. Example Supported Games

3.1 Numerical Roll

- Participants interact with a numerical range (e.g., 1–100) to determine outcomes.
- Optional Advantage Condition: The creator may define an additional result (e.g., `101`) as a special condition.
- Applications: Simple prediction games or decision-making tools.

3.2 Card-Based Interactions

- Players submit tokens representing cards with defined attributes, competing in predefined categories (e.g., strength, agility, or special abilities).
- Optional Advantage Condition: The creator may define specific conditions, such as a unique card or combination as a special outcome.
- Applications: Competitive card games, trading card games, or deck-building games.

3.3 Land Ownership Game

- Inspired by the Hemis team's early concept, participants interact with tokenised land parcels on a hexagonal grid.
- Outcomes determine control of individual parcels, allowing players to expand their domains.
- Applications: Strategic games where players compete for resources or territory over time.
- Optional Advantage Condition: The creator could define unique rules for special tiles or sones, offering additional strategic opportunities.

3.4 Token Lotteries

- Participants contribute tokens into a shared pool, and RNG determines a winner.

- Optional Advantage Condition**:** The creator may reserve a small percentage of outcomes for a predefined condition, such as multiple winners or bonus payouts.
- Applications: Community raffles or fair resource allocation mechanisms.

3.5 Hex-Based Strategy Games

- Players use tokens representing resources, characters, or items on a hex-based map to achieve specific objectives.
- PTX enables fair and transparent resolution of battles or resource allocation between players.
- Applications: Competitive multiplayer games where strategy unfolds on a shared game board.
- Optional Advantage Condition: Special outcomes for tiles or unique scenarios defined by the game creator.

3.6 Custom Games

- The PTX framework supports fully customisable rules, allowing creators to design new games based on their imagination and community preferences.
- Examples include:
  - Cooperative resource-sharing systems.
  - Puzzle-solving scenarios driven by RNG.
  - Dynamic role-playing mechanics where tokens evolve with gameplay.

4. Required Features

- **Wallet Integration**
  - Enable seamless game initiation via the Hemis wallet.
  - Allow participants to select game parameters, including whether or not to enable the optional advantage condition.
- **Quorum Management**
  - Define quorum attributes and manage random selection of Gamemaster nodes for processing games.
  - Provide network monitoring tools to ensure consistent performance of Gamemasters.
  - Implement penalties for malicious or inactive nodes to maintain network integrity.
- **On-Chain Transparency**
  - Record all RNG inputs, outputs, and game metadata on-chain to ensure full auditability.
  - Clearly indicate whether an advantage condition is applied for each game.
  - Ensure participants can independently verify the fairness of the process.
- **Fees and Rewards**
  - Gamemaster nodes receive a portion of the block reward as transaction fees for facilitating games.

    - If an optional advantage condition is enabled:
        - Its parameters must be clearly defined and transparent to all participants.
        - Any resulting additional outcomes (e.g., credits to the creator) are recorded on-chain for verification.
- **Optional Advantage Condition**
    - The advantage condition feature is entirely optional and must be:
        - Clearly communicated to participants before the game begins.
        - Transparent in its implementation, with no hidden mechanics or biases.
    - Ensure fairness by limiting the statistical benefit to a minor, predefined edge (e.g., an extra outcome in RNG).
- **High-Velocity Processing**
    - Ensure the system supports multiple games running concurrently without bottlenecks.
    - Design quorum processes to validate and settle games efficiently, allowing for high-frequency participation.