Nugget Wager Protocol

by adrian

April 14 2025

Abstract

We introduce a poultry-based betting game centered around how many McDonalds' Chicken Nuggets Adrian can AT LEAST eat in one sitting. An idea sparked days before the event, the goal is to define a payout structure that is fair to participants, potentially with a small house edge.

Adrian will attend the cinemas viewing **A Minecraft Movie (2025)** starring Jason Momoa and Jack Black this Sunday, 20 April 2025 AEDT (at an undisclosed time due to anonymity reasons).

During the film's estimated 1h 40min runtime, he will attempt to consume as many nuggets as his body allows from a predefined supply.

Participants may anonymously submit their guesses (bets) on how many nuggets Adrian will AT LEAST eat. These guesses are recorded via a smart contract on the Solana blockchain, and the final nugget count will be submitted post-movie by Adrian himself. Adrian has no visibility into who submitted which guesses, preventing foul play (or fowl play). The betters must claim their rewards via the same smart contract, at which point bets are revealed.

The nugget supply consists of 4x 24-pack boxes (96 McNuggets at a competitive $35.85 AUD) and a Minecraft Movie Meal (10 McNuggets), giving us 106 total nuggets. However, due to reasons, 6 nuggets will be discarded and will not enter Adrian's digestive ledger, leaving a maximum possible count of **100 nuggets**. Beverages, condiments, and fries are available ad libitum to Adrian's pleasure. (*These may influence overall strategy and nugget velocity.*)

Updates may be posted throughout the day depending on Adrian's existence, with final submissions closing once the 100 target nuggets are officially in hand. May the odds—and sauces—be ever in your favor.

Will Adrian devour ONE HUNDRED McNUGGETS? Or will he face his demise at the hands of the mighty chicken nugget?

1 Rules Summary

- Participants guess an integer g such that $0 \leq g \leq 100$.
- A fixed bet amount B is placed (e.g., 1 SOL).
- Let n be the actual number of nuggets consumed, $n \in \{0, 1, \ldots, 100\}$.
- **Winning Condition:** A player wins if their guess g is less than or equal to the actual outcome n (i.e., $g \leq n$).
- **Losing Condition:** A player loses if their guess g is strictly greater than the actual outcome n (i.e., $g > n$).

2 Payout Structure (Profit Range Adjusted - Intermediate)

The payout depends on meeting the winning or losing condition. Meeting the winning condition ($g \leq n$) results in a net profit only if the guess g is sufficiently close to the actual result n.

2.1 Losing Payout ($g > n$)

If a participant meets the losing condition ($g > n$), they forfeit their entire stake.

$$\text{Net Profit (Loss)} = -B$$

$$\text{Total Return (Loss)} = 0$$

2.2 Winning Payout ($g \leq n$)

If a participant meets the winning condition ($g \leq n$), their total return depends on the distance $d = n - g$, where $d \geq 0$.

The total amount returned is calculated using the decay constant $k = 0.14$:

$$\boxed{\text{Total Return (Win)} = B \times \left(3.9e^{-0.14(n-g)} + 0.1\right)}$$

The net profit (or loss) is:

$$\text{Net Profit (Win)} = \text{Total Return (Win)} - B$$

$$\boxed{\text{Net Profit (Win)} = B \times \left(3.9e^{-0.14(n-g)} - 0.9\right)}$$

Key characteristics of the winning payout (with $k = 0.14$):

- The payout depends exponentially on the distance $d = n - g \geq 0$.
- **Profit Zone:** Net profit is positive only if $3.9e^{-0.14d} - 0.9 > 0$. This occurs when $e^{-0.14d} > 0.9/3.9 = 3/13$, which requires $d < -\ln(3/13)/0.14 \approx 10.47$. Therefore, if the distance $n - g$ is between 0 and 10 (inclusive), the player makes a net profit.
- **Loss Zone:** (despite $g \leq n$) If the distance $n - g$ is 11 or greater ($n - g \geq 11$), the net profit is negative, meaning the player gets back less than their initial stake B.

- **Maximum Profit:** Occurs at the minimum distance $d = 0$ (when $g = n$). Net Profit $= B(3.9e^0 - 0.9) = B(3.9 - 0.9) = 3.0B$. Guessing the exact number yields 300% net profit (total return is 4B).
- **Maximum Loss:** (while $g \leq n$) Occurs at the maximum distance $d = 100$ (e.g., $g = 0, n = 100$). Net Profit $= B(3.9e^{-0.14\times100} - 0.9) = B(3.9e^{-14} - 0.9) \approx B(3.9 \times 8.3 \times 10^{-7} - 0.9) \approx -0.89999B$. The minimum total return approaches $0.1B$ (a 90% loss) for large distances.

3 Illustrative Examples (Profit Range d≤10, k=0.14)

Table 1 shows payouts for a bet $B = 1$. Net Profit is calculated as $3.9e^{-0.14(n-g)} - 0.9$ for wins ($g \leq n$) and -1 for losses ($g > n$).

Table 1: Examples of Payouts (Profit Range d≤10, k=0.14, Bet $B = 1$)

Guess g	Actual n	Win? ($g \leq n$)	Distance $d = n - g$	Net Profit ($B = 1$)	Total Return ($B = 1$)
50	30	No	-	−1.0000	0.0000
60	60	Yes	0.0000	3.0000	4.0000
59	60	Yes	1.0000	2.4982	3.4982
60	70	Yes	10.0000	0.0617	1.0617
59	70	Yes	11.0000	−0.0638	0.9362
55	70	Yes	15.0000	−0.4228	0.5772
10	90	Yes	80.0000	−0.9000	0.1000
0	100	Yes	100.0000	−0.9000	0.1000
0	0	Yes	0.0000	3.0000	4.0000
5	0	No	-	−1.0000	0.0000

4 Protocol Execution: Solana Smart Contract Mechanics

To bring this medium-rare-stakes poultry contest to life, the Nugget Wager Protocol is implemented as a smart contract deployed on the Solana blockchain. This choice ensures a transparent, auditable, and fundamentally trustless environment for managing bets and payouts, minimizing the need for blind faith in the host's administrative prowess.

4.1 Commit-Reveal Scheme

Participant anonymity and fair play are paramount. To prevent strategic betting based on others' revealed choices (or Adrian getting hungry ideas), a **commit-reveal scheme** is employed:

1. **Commit Phase (The Bet):** Participants don't submit their guess g directly. Instead, they generate a cryptographic commitment on their end. This involves a specific encoding process before hashing:

   - The integer guess g (where $0 \leq g \leq 100$, fitting within a single byte) is converted into a single-byte representation.
   - A secret, randomly generated 64-bit number (the "salt") is encoded into exactly 8 bytes using Little-Endian byte order.
   - These are concatenated: the single guess byte comes first, followed immediately by the 8 bytes representing the salt. This results in a compact 9-byte data structure.
   - This specific 9-byte sequence is then fed into the `Keccak-256` hashing algorithm.

   The resulting hash digest is the commitment submitted to the smart contract along with the fixed bet amount B (where $0 \leq B \leq 1SOL$).

   $$\text{Commitment} = \texttt{Keccak256}(\texttt{Concat}(\texttt{Byte}(g), \texttt{BytesLE}_8(\text{salt})))$$

   The contract records this commitment hash and the associated stake. Crucially, the hash reveals nothing about the original guess g or the salt.

2. **Reveal Phase (Reward claiming and reveal):** After the cinematic experience concludes and Adrian, through sheer force of indigestion, submits the final nugget count n to the smart contract, the reveal phase begins. Participants now submit their original guess g and the unique 64-bit random salt they used. The smart contract performs the *exact same* encoding and hashing process: it converts g to a byte, encodes the salt to 8 Little-Endian bytes, concatenates them into the 9-byte sequence, and computes the Keccak-256 hash. If this re-computed hash perfectly matches the commitment stored earlier, the contract verifies the bet's authenticity. It then proceeds to:

   - Compare the revealed g with the official n.
   - Calculate the payout according to the formula defined in Section 2.
   - Transfer the calculated winnings (if any) back to the participant's wallet.

Participants **must** safeguard their chosen guess g and random salt used. This is the *only* key to unlocking potential winnings. The interface will help you store this locally in browser local storage, but do treat it like the nuclear codes. Loss of this information renders the bet unclaimable. Contact Adrian if you end up winning and treasury still has money after all the deadlines. This string is referenced as 'bet_commitment'.

4.2 Betting Model: Against the House

This operates on "Player vs. Adrian (House)" model, where payouts are determined by the accuracy of participant's guess g relative to outcome n, according to the fixed payout formula. This contrasts with parimutuel systems where odds and payouts shift based on the distribution of bets within a pool.

Justification: Since the outcome n is submitted by the host (Adrian) post-event and is not derived from an external trusted source, If the host's submission of n were momentarily known before payouts are locked or distributed, it could potentially be exploited (e.g., front-running claims if the pool dynamics significantly altered payouts based on n). Betting against a fixed-formula house removes this vector, ensuring payouts depend only on guess-to-outcome relationship.

4.3 Fiscal Prudence: The 1 SOL Limit and Treasury Realities

To maintain operational and manage the non-infinite financial backing of this venture, participation is capped at a **fixed bet B of 1 SOL** per commitment.

Furthermore, potential participants should approach this wager with eyes wide open regarding the payout treasury.

The smart contract treasury fund holds *both* the total value of all player bets (the player pot) *and* the host's (Adrian's) own initial funds. Fear not, dear gamblers, for the host's actual funding level will be implicitly calculated by the difference between the total SOL direct-deposited to the treasury address and the total player pot.

scenarios involving multiple simultaneous high-payout wins (e.g., several participants guessing $g = n$ exactly) could, theoretically, strain the available funds.

Payouts via the smart contract's claim function operate on a **first-come, first-served basis**. Should the unfortunate circumstance arise where the treasury is depleted before all winning claims are processed, subsequent claimants might find their reward... well, likely represented by the spiritual essence of a nugget rather tangible SOL. Consider it contribution to performance art, donation, or contact the host for potential alternative (and likely non-monetary) reparations. You have been advised. Caveat emptor, caveat pullum edentis[1].

4.4 Timelines and Anti-Griefing Measures

To ensure a timely resolution and prevent funds from being locked indefinitely, the following deadlines are enforced by the smart contract:

- **Bet Submission Deadline:** All bets must be committed before **Sunday, 20th April 2025, 11:59 PM GMT** (09:59 AM AEDT Monday). No further bets are accepted after this time.

Following the close of betting, the wager proceeds in phases:

1. **Host Reveal Deadline:** The Host must reveal the true number of nuggets eaten (n) by **Sunday, 20th April 2025, 11:59 PM GMT** (09:59 AM AEDT Monday). *Consequence:* If the Host fails to reveal n by this deadline, Players will be able to reclaim their original stake directly from the contract.

2. **Player Reveal & Claim Deadline:** Players must reveal their guess (g) and salt to participate in the payout by **Sunday, 27th April 2025, 11:59 PM GMT** (09:59 AM AEDT

[1] Let the buyer, and the chicken eater, beware.

Sunday). Winning players must also claim their payout by this deadline. *Consequence:* Any stake associated with a guess not revealed by this deadline is forfeited. Potential winnings not claimed by this deadline are also forfeited. Forfeited funds remain in the contract treasury, potentially benefiting the Host or other claimants.

3. **Final Claim Period (Insufficient Host Liquidity Scenario):** This period addresses the rare case where the Host lacks sufficient on-chain liquidity to cover all winning payouts *after* the Player Reveal phase.

    - **Window:** From the Player Reveal Deadline (**27th April**) until **Sunday, 4th May 2025, 11:59 PM GMT** (09:59 AM AEDT Monday).
    - **Action:** If a winning Player attempted to claim but could not due to insufficient contract liquidity (verifiable on-chain), they have this final window to:
        - Claim their original stake back as a recourse.
        - Attempt to claim their full winnings again if sufficient funds become available (e.g., due to other players forfeiting).
    - **Final Consequence:** After **4th May 2025, 11:59 PM GMT**, any and all remaining funds within the contract treasury will be irreversibly transferred to the Host (Adrian), concluding the wager protocol.

These mechanisms prevent funds from being permanently locked ("squatting") and provide clear steps for concluding the wager under various scenarios, including potential liquidity issues. Players must act within the specified deadlines to secure their stake or winnings.