Alien Artifacts: Independent Prize Recovery

UFO Coin Mainnet prize locks - version 1, 31 October 2026. Keep an offline copy of this paper, your lock address, winner wallet, and locking transaction. This guide does not require the UFO Coin website or MongoDB to remain online.

> **Important limitation:** The current contract uses an authority-written market-cap feed, NOT an independent oracle. If the developer disappears while the stored feed is below your target, a winner cannot update the feed or force-unlock alone. A higher real-world market cap does not bypass that restriction. There is no documented timeout or automatic abandonment escape hatch. This paper cannot guarantee recovery in that situation.

1. What survives a shutdown?

The Solana lock account and Token-2022 vault persist independently of the project server. Solana must still operate. Your award is held in the vault, not yet in your wallet. A permissionless `claim` instruction sends the prize ONLY to the winner wallet recorded in the lock. Any transaction fee payer may submit that instruction when the stored feed meets the lock target; the winner is not a required signer of the claim instruction.

If the target is not met, recovery needs the recorded authority signer to update the feed or invoke `force_unlock`, or another legitimate program-maintenance route. Program upgrade authority, availability and any successor arrangements must be independently checked; no successor or lost-key recovery arrangement is promised here.

2. Identify your accounts using an independent Mainnet RPC

```
Program: 6fXjjAvziyTbhjxLvzdhsjUUUPFvmM5VYxv4vqAiH25o
UFOCOIN mint: EsZHf4AeokW6FyyhVvj6XnZ9Viapw18L6ZHosh7opump
Token program: TokenzQdBNbLqP5VEhdkAS6EPFLC1PHnBqCXEpPxuEb
Mainnet genesis: 5eykt4UsFv8P8NJdTREpY1vzqKqZKvdpKuc147dw2N9d
```

Use a reputable Solana explorer or RPC, not an unsolicited recovery link. Verify the lock is owned by the program above, the mint is correct, the winner is your recorded wallet, and the vault is its canonical Token-2022 associated token account.

```
Lock account layout (186 bytes; offsets in bytes):
0..7: first 8 SHA-256 bytes of "account:Lock"
8: authority public key (32 bytes)
40: winner public key (32 bytes)
72: mint public key (32 bytes)
104: vault public key (32 bytes)
136: amount, little-endian u64 (6 token decimals)
144: market-cap target, little-endian u64 (USD)
152: raffle ID (32 bytes)
184: claimed flag (0 or 1); 185: PDA bump

Lock PDA: findProgramAddress(["lock", authority bytes, raffle ID], program)
Feed PDA: findProgramAddress(["feed", authority bytes], program)
Feed layout (49 bytes): "account:Feed" discriminator at 0,
authority at 8, little-endian u64 market-cap value at 40, bump at 48.
The feed has no timestamp: a stored value is not a fresh price proof.
```

3. Decide whether a direct claim is possible

1. If claimed is already 1, inspect the winner token account and prior claim transaction. Do not pay someone to "unlock" an already claimed prize.
2. Read and validate the feed PDA, program ownership, discriminator and matching authority.
3. If feed.mc is below lock.mcTarget, STOP. Independent claiming is blocked. Do not invent a feed, forge a market cap or send SOL to a recovery service.
4. If feed.mc is at or above the target, an independent claim can be built and simulated. A fee payer needs enough SOL for the fee and, if absent, the winner token account rent.

4. Build a claim without the UFO Coin server

A technically competent Solana helper can use the standard packages `@solana/web3.js` and `@solana/spl-token`. Obtain packages from their official distribution. After reading and verifying the accounts above, construct the following instructions. This is a technical construction recipe, not a hosted recovery service.

```
const winnerAta = getAssociatedTokenAddressSync(
  mint, winner, true, TOKEN_2022_PROGRAM_ID);
const setup = createAssociatedTokenAccountIdempotentInstruction(
  feePayer, winnerAta, winner, mint, TOKEN_2022_PROGRAM_ID);
const claim = new TransactionInstruction({
  programId: program,
  data: createHash("sha256").update("global:claim")
    .digest().subarray(0, 8),
  keys: [
    { pubkey: lock, isWritable: true, isSigner: false },
    { pubkey: winner, isWritable: false, isSigner: false },
    { pubkey: mint, isWritable: false, isSigner: false },
    { pubkey: vault, isWritable: true, isSigner: false },
    { pubkey: winnerAta, isWritable: true, isSigner: false },
    { pubkey: TOKEN_2022_PROGRAM_ID,
      isWritable: false, isSigner: false },
    { pubkey: feed, isWritable: false, isSigner: false },
  ],
});
// Transaction instructions: [setup, claim]
// Fee payer signs through their trusted wallet, not the project server.
```

Required imports: `createHash` from Node's `node:crypto`; `TransactionInstruction` from web3.js; both associated-account helpers and `TOKEN_2022_PROGRAM_ID` from spl-token. All named account variables must be verified `PublicKey` values read or derived as described above.

1. Fetch a fresh Mainnet blockhash. Build the transaction with the two instructions and your chosen fee payer.
2. Simulate first, requesting the lock, vault and winner ATA post-state. A below-target error is not a reason to bypass the feed.
3. Verify simulated claimed=1, vault debit equals the award, and the exact same amount reaches the recorded winner ATA. Check fee and rent before approving.
4. Sign using a trusted wallet interface, such as a locally reviewed Phantom-compatible transaction tool. Never disclose a seed phrase/private key to a website or helper.
5. Submit once and wait for finalized confirmation. Save the signature. If uncertain, query that signature and lock state; do not blindly construct another transaction.
6. Verify finalized claimed=1 and the token credit in the recorded winner ATA. Claiming never authorizes changing the winner or withdrawing to the helper.

5. Concrete October 26, 2026 example

```
Lock: 3ZT1eD2dXWsG5uribNVzhPvkGPGXCLTi2px1yUuapUMp
Amount: 398,705.289033 UFOCOIN
```

```
Unlock target: USD 1,000,000
Lock transaction:
3A3RvV8jQ41mbCiUEovWR8Akz2WUwLhKeepAph2cowpG52auQNKNsDyYwJUhVNGCUh2BfxjjwnbDtaigth5jQNbj
```

This example is not an assertion that it is currently claimable. Read the live feed and lock before acting.

6. Security and verification limits

Keep the wallet controlling your recorded winner address accessible. Do not trust screenshots as proof. Check actual account owners and token deltas. The instruction layout here matches the repository's IDL and production client as of the date above, and the current below-target claim was tested by unsigned Mainnet simulation. The Rust program source is not present in this repository; the above-target independent claim path has not been demonstrated live for this award. Inspect the deployed program and simulate before signing, especially after any upgrade.

Public reference: www.ufocoin.site/alien-artifacts/locks. If unavailable, use the saved addresses with another Mainnet explorer/RPC. This guide never asks for a recovery payment or private key.