PHYSICAL VALUE, PORTABLE BY DESIGN

Redeem a PR3DICT paper wallet

A complete, standalone specification for reading the printed QR codes, decrypting the private key locally, verifying the result, and importing it into MetaMask.

SECURITY RULE: Never upload the encrypted QR image, its password, or the recovered private key to a website, chatbot, cloud notebook, shared computer, or another person. An AI may write the decoder, but it must run locally and offline on the owner's device.

FORMAT	CRYPTOGRAPHY	NETWORK
76-byte binary QR	scrypt + AES-256-GCM	BNB Smart Chain

Scope

This document is sufficient to build an independent decoder. It does not require access to the PR3DICT source repository or server. It describes the current physical paper wallet format exactly; do not invent a different encoding.

For the simplest supported experience, use the public **redeem.html** page supplied by PR3DICT. The same checks below still apply.

1. Identify the printed data

01

Read the correct QR codes

A PR3DICT paper wallet contains two large QR codes. They have different jobs:

- **Large QR on the left:** the public EVM address as text, normally beginning with **0x** and containing 40 hexadecimal characters after it.
- **Large QR on the right:** exactly **76 bytes** of binary encrypted payload. Preserve the decoder's raw **binaryData**; do not treat these bytes as UTF-8 text, Base64, or a hexadecimal string.
- **Small repeated border QR:** human-facing paper wallet information or password hint. It is not the encrypted key payload.

The password is case-sensitive UTF-8 text. No trimming, lowercasing, normalization, separator, or extra newline may be added.

Expected public-address rule

Accept a left-QR value only if an EVM address parser validates it. Keep it for the final equality check; the left QR is not secret.

Expected encrypted-payload rule

Reject the right QR unless the raw binary payload length is exactly 76 bytes. A text-only QR reader is insufficient for this QR.

2. Binary layout

Offset	Length	Field	Meaning
0..15	16 bytes	salt	Random salt passed to scrypt
16..27	12 bytes	iv	AES-GCM initialization vector
28..59	32 bytes	ciphertext	Encrypted 32-byte secp256k1 private key
60..75	16 bytes	authTag	AES-GCM authentication tag

3. Exact cryptographic procedure

02

Derive, authenticate, decrypt

- Encode the password exactly as UTF-8 bytes.
- Derive a 32-byte key with **scrypt**: **N=16384, r=8, p=1**, salt = bytes 0..15.
- Decrypt with **AES-256-GCM**: IV = bytes 16..27, ciphertext = bytes 28..59, authentication tag = bytes 60..75.
- There is no additional authenticated data (AAD). The expected plaintext is exactly 32 bytes.
- Represent the plaintext private key as **0x** followed by exactly 64 lowercase hexadecimal characters.

Authentication failure means wrong password or damaged/incorrect QR. Do not return partial plaintext and do not guess silently.

Language-neutral pseudocode

```
payload = qr.binaryData
assert length(payload) == 76
salt = payload[0:16]
iv = payload[16:28]
ciphertext = payload[28:60]
tag = payload[60:76]
key = scrypt(utf8(password), salt, N=16384, r=8, p=1, dkLen=32)
plain = aes_256_gcm_decrypt(key, iv, ciphertext, tag, aad=empty)
```

```
assert length(plain) == 32
privateKey = '0x' + lowercase_hex(plain)
```

4. Mandatory verification

03

Prove the key belongs to this paper wallet

Derive the standard Ethereum/EVM address from the recovered secp256k1 private key:

- Compute the uncompressed public key from the private key.
- Apply Keccak-256 to the 64-byte X || Y public-key body, excluding the 0x04 prefix.
- Take the last 20 bytes and format as an EVM address; EIP-55 checksum casing is recommended.
- Compare it case-insensitively with the public address read from the large left QR.

Reveal or import the private key only when the two addresses match. On mismatch, clear the plaintext and stop.

5. MetaMask import

04

Move the recovered account into the wallet

In MetaMask, open the account selector, choose **Add account or hardware wallet**, then **Import account**. Choose **Private key**, paste the recovered 0x-prefixed key, and confirm. Wording may vary by MetaMask version.

After import, verify that MetaMask displays the same address printed by the large left QR before using funds.

PR3D token metadata

Network	BNB Smart Chain
Chain ID	0x38 (decimal 56)
Token contract	0xC13fA16331bCbE294726Cbf0bC6C58E001eeBb19
Symbol	PR3D
Decimals	18

Operational safety

- Work on a trusted personal device. Disconnect from the internet before entering the password if using a newly generated decoder.
- Do not take screenshots, sync the key to cloud storage, paste it into chat, or leave it in clipboard history.
- Clear the decoder, clipboard, downloaded images, and temporary files after the MetaMask import.

- Anyone who sees the private key can control the account. PR3DICT support must never ask for it.

6. Prompt for a coding AI

The following prompt can be supplied together with this PDF. Do not include the real password or QR image in the chat. Ask the AI to create the tool first, inspect it, then run it locally and offline.

```
Build a single-purpose local decoder from the attached PR3DICT specification. Use a local image picker and a QR library that exposes raw binaryData. Perform scrypt and AES-256-GCM locally, verify the derived EVM address against the left QR, and show the private key only after a match. Make no network requests, analytics, wallet signatures, or transactions. Do not persist secrets. Include a clear button that overwrites in-memory byte arrays where practical. Give me commands to run it on my own computer. Do not ask me to send you the QR, password, or private key.
```

7. Implementation acceptance checklist

- Reads left QR as a validated EVM address.
- Reads right QR as exactly 76 raw bytes, not text.
- Uses scrypt N=16384, r=8, p=1 and a 32-byte derived key.
- Uses AES-256-GCM with 12-byte IV and 16-byte authentication tag; no AAD.
- Rejects wrong passwords and malformed payloads without exposing output.
- Derives and compares the EVM address before revealing the key.
- Makes zero network, signing, or transaction requests.
- Clears secrets on request and when the page/application closes.

If every item passes, the implementation is compatible with the documented PR3DICT physical paper wallet format.