Smudged Seed Word: Narrowing Candidates

November 27, 2026 · Document #441 · bip39-phrase.com



Photo by Bernd □ Dittrich on Unsplash

Use readable letters, word position, length, and the BIP39 checksum to narrow a damaged word, then verify candidates against a known receive address.

Contents

Record what is known 1

Apply the BIP39 checksum 2

Match a known receive address 3

Handle the recovered wallet safely 3

FAQ 4

Sources 4

Example BIP39 word sequences 4

Reading a BIP39 phrase 5

See also 6

A smudged BIP39 word can often be reduced to a short list when you know its position, some letters, and the word length. The BIP39 checksum can then reject candidates that do not belong to a valid mnemonic. A checksum match still does not prove that the candidate is your word. Confirm the result by deriving addresses and matching a receive address you already know.

Record what is known

Work from a copy of the backup when possible. Keep the original seed phrase offline and do not enter it into a website, chat service, cloud note, or unknown recovery tool. If the phrase has ever been exposed, treat the wallet as potentially compromised and move funds to a newly generated wallet after recovery.

Write down the following details:

- The total number of words: `12`, `15`, `18`, `21`, or `24` for standard BIP39 mnemonics.
- The position of the damaged word, counting from `1`.
- Every readable letter and its position, including whether the word starts or ends with a known letter.
- The word's length.
- The language and wordlist used by the wallet.
- Whether the wallet used an optional BIP39 passphrase. This passphrase is sometimes informally called a "25th word," although it is separate from the mnemonic and is not limited to one word.

BIP39 English words are selected from a fixed 2,048-word list. Filter that list by the known length and letters. A word with three known letters in fixed positions may leave only a few candidates. Preserve the exact word spelling from the official list; guessing from a similar word in a dictionary can produce a phrase that fails later.

The position matters because the damaged word is one 11-bit word index inside the complete mnemonic. The checksum is calculated over the original entropy before the mnemonic is divided into 11-bit groups. Moving the same candidate to a different position changes the bit sequence and can change whether the checksum passes.

Apply the BIP39 checksum

BIP39 mnemonics contain entropy followed by a checksum. If the entropy length is `ENT`, the checksum length is `ENT / 32` bits. The combined bit string is divided into 11-bit numbers, and each number selects one word from the 2,048-word list.

For each candidate, reconstruct the complete phrase and perform this test:

1. Convert every word to its index in the selected BIP39 wordlist.
2. Encode each index as an 11-bit binary value and concatenate the values in word order.
3. Split the result into the entropy bits and the checksum bits. For a 12-word phrase, this is `128` entropy bits followed by `4` checksum bits.
4. Calculate `SHA-256` of the entropy bytes.
5. Compare the first `ENT / 32` bits of that hash with the mnemonic's checksum bits.

Keep candidates whose checksum matches. For a 24-word phrase, use `256` entropy bits and `8` checksum bits. The same process applies to the other valid BIP39 lengths.

The checksum is a filter, not a recovery oracle. A damaged word may leave several candidates that pass, especially when only a few letters are known. A phrase can also pass the checksum while containing a word different from the one originally written. The checksum only says that the phrase has a valid BIP39 bit structure.

A small script or an offline wallet recovery utility can perform the calculation. Inspect its source or use a well-established implementation, keep the computer offline while processing the real candidates,

and avoid saving the candidate phrases to disk. Test the procedure first with the official BIP39 vector `abandon abandon abandon abandon abandon abandon abandon abandon abandon abandon abandon about`, then erase the test data.

Match a known receive address

Use the surviving candidates to derive wallet addresses. The candidate that produces a known receive address is the useful result. This step is necessary because BIP39 checksum validity does not identify the wallet that received your funds.

The derivation settings must match the original wallet:

- The BIP39 passphrase, including capitalization and spaces. An empty passphrase and a nonempty passphrase produce different seeds.
- The network, such as Bitcoin mainnet or testnet.
- The address script type and derivation path. Common Bitcoin paths include `m/44'/0'/0'` for legacy addresses, `m/49'/0'/0'` for nested SegWit, and `m/84'/0'/0'` for native SegWit under BIP44-style conventions.
- The account number, change branch, address index, and any wallet-specific account structure.

Start with a receive address that you recorded independently, such as one shown on an old invoice or transaction record. Derive the first several receive addresses for each candidate and compare them character by character. If the wallet had used many addresses, continue far enough to account for its address gap limit and account selection. A match at a known address is strong evidence; a checksum match alone is not.

If no candidate produces the expected address, review the wordlist language, word position, letter transcription, wallet type, derivation path, and BIP39 passphrase. Also check whether the damaged item is a word at all: a missing word, an extra word, or a transcription error in another position changes the calculation.

Handle the recovered wallet safely

Once a candidate matches a known address, restore it in a trusted wallet. Keep the seed offline except when entering it directly into that trusted wallet during recovery. Verify several historical addresses and transaction records before treating the recovery as complete. If the seed was entered into an internet-connected computer or third-party tool, create a new wallet on a trusted device and transfer funds to it.

Do not publish the readable letters, candidate list, wallet type, or address together. Combined details can reduce the protection provided by the remaining words. Store the corrected backup in a durable form, and record the wallet's derivation settings separately from the seed when they are needed for future recovery.

FAQ

Can every checksum-valid candidate be the original word?

No. The checksum confirms internal BIP39 structure. Several candidates can pass, and only address or transaction matching can identify the wallet.

Does the word's position really affect the result?

Yes. Each word contributes an 11-bit index at its position, so moving a candidate changes the entropy and checksum bits.

What if no candidate matches a known address?

Recheck the language, word length, letter positions, phrase length, passphrase, derivation path, account, and address type. One additional transcription error can prevent every candidate from matching.

Sources

- BIP-0039: Mnemonic code for generating deterministic keys (github.com)
- BIP-0032: Hierarchical Deterministic Wallets (github.com)
- BIP-0044: Multi-Account Hierarchy for Deterministic Wallets (github.com)

> Continue reading on bip39-phrase.com: What Is a Seed Phrase? How a Recovery Phrase Works
> More on this topic: Recovery

Example BIP39 word sequences

Randomly generated sequences of words from the BIP39 English wordlist. None of them passes the BIP39 checksum, so no wallet will accept them: they show the format only. Never store funds with a phrase you found in a document.

Example 1

1 claw	2 prosper	3 cousin	4 laundry
5 object	6 bubble	7 fitness	8 day
9 miss	10 track	11 aunt	12 vocal

Example 2

1 flame	2 tree	3 mimic	4 bicycle
5 exhibit	6 essay	7 chase	8 engine
9 ivory	10 topple	11 embody	12 settle

Example 3			
price	half	loyal	brother
patch	raccoon	bag	edge
erase	update	grape	mixed

Example 4			
dolphin	mesh	length	dove
discover	fade	flash	bird
develop	wink	mimic	desk

Example 5			
increase	arena	earn	security
ripple	pioneer	gorilla	lecture
pride	ordinary	call	bone

Example 6			
crane	fog	drip	predict
possible	camp	brass	effort
brisk	beef	uniform	start

Reading a BIP39 phrase

Every word stands for an 11-bit number: its position in the 2048-word list, counted from 0. Here is Example 1 decoded word by word. Each word links to its page on bip39-phrase.com, where the list is numbered from 1.

#	Word	Index (0-2047)	Word no.	11 bits
1	claw	336	337	00101010000
2	prosper	1380	1381	10101100100
3	cousin	395	396	00110001011
4	laundry	1005	1006	01111101101
5	object	1216	1217	10011000000
6	bubble	234	235	00011101010
7	fitness	703	704	01010111111
8	day	448	449	00111000000
9	miss	1134	1135	10001101110
10	track	1844	1845	11100110100
11	aunt	122	123	00001111010
12	vocal	1962	1963	11110101010

The 12 words carry 132 bits: 128 bits of entropy followed by a 4-bit checksum. The checksum must equal the first 4 bits of the SHA-256 hash of the entropy.

```
Entropy (hex): 2a1590c5bed9803a95f9c08ddcd03d7a
```

This sequence ends in the checksum bits 1010, but SHA-256 of its entropy starts with 0011, so wallets reject it as an invalid phrase. That is the point of the checksum: about 15 of every 16 random or mistyped sequences fail it.

See also

- Leaving Off the Last BIP39 Word: A False Trick · January 8, 2025
- Seed Word Not on the BIP39 List: Find the Right Word · November 25, 2026
- How BIP44 Account Discovery Decides When to Stop · June 26, 2025
- Recovering Funds Sent to Contract Wallets on Other Chains · April 23, 2026

Educational document. The example sequences are random and fail the BIP39 checksum: they are not wallets and must never hold funds. Security notes: bip39-phrase.com/security/