Why Song Lyrics and Famous Quotes Make Weak Passphrases

November 14, 2026 · Document #428 · bip39-phrase.com

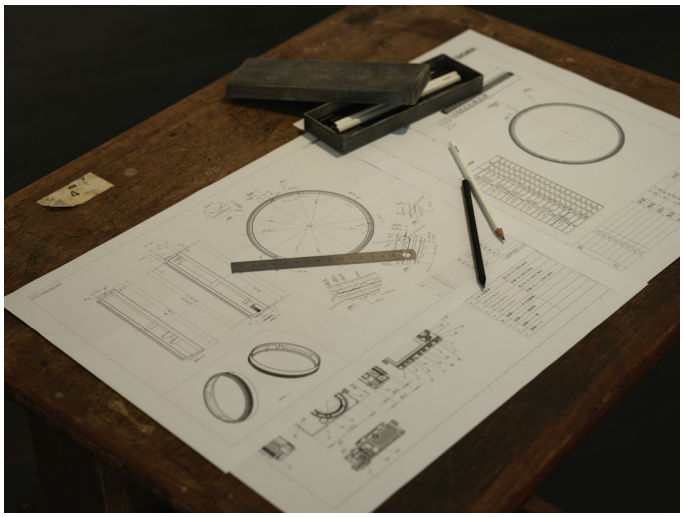


Photo by Declan Sun on Unsplash

Why lyrics and famous quotes are easy to guess, what brain-wallet attacks revealed, and how to create memorable random passphrases.

Contents

Why attackers try lyrics and quotations	2
Length does not measure real randomness	2
Build a memorable passphrase from random choices	2
Apply this carefully to wallet recovery	3
FAQ	3
Sources	4
Example sequences from the BIP39 wordlist	4
Reading a BIP39 phrase	5
See also	6

A song lyric can be 60 characters long and still be an easy passphrase to guess. Attackers can test the complete line as one candidate, without discovering each character separately. The same problem affects dialogue from films, passages from books and famous quotations: their familiarity makes them memorable, but also puts them in collections an attacker can search.

Why attackers try lyrics and quotations

Password cracking relies heavily on trying likely choices first. Attackers build candidate lists from leaked passwords, dictionaries, names and public text. Lyrics, quotations and film dialogue are useful because many people independently choose the same memorable lines.

An attacker does not need to know your favourite song. They can collect popular lyrics, split them into lines and phrases, and generate variations. Capitalizing words, removing spaces, substituting digits for letters or adding a year creates predictable candidates. These changes increase the attacker's workload somewhat, but do not turn a familiar line into a randomly generated secret.

Brain wallets demonstrated the financial consequences. In a classic construction, software hashes a user-selected phrase and uses the result to derive a cryptocurrency private key. Anyone who guesses the phrase can reproduce the key and spend its funds.

Researchers studying Bitcoin brain wallets tested candidates drawn from passwords and public text, including literary material. Their work showed that attackers could discover weak phrases and steal funds. Because blockchain activity is public, an attacker can also check derived addresses for activity without asking a wallet service for permission.

The lesson applies beyond that particular construction: a secret selected from public culture is vulnerable to a search through public culture. A fast hash makes each guess cheap; it does not add unpredictability to the phrase.

Length does not measure real randomness

A quote's character count says little about its resistance to guessing. Once you choose a familiar sentence, its words are strongly related. Knowing the opening often makes the ending predictable.

For comparison, choosing uniformly at random from a collection of one million quotations would provide about `log2(1,000,000)`, or `19.9` bits of entropy. That is an illustrative calculation, not a measurement of everyone's quotation choices. Actual human choices are uneven: famous lines attract far more users than obscure passages, so attackers can prioritize them.

You cannot estimate a self-selected quotation's strength by multiplying its length by the entropy of a random character. That calculation assumes each character was independently sampled from the stated alphabet. A sentence remembered from a film does not meet that assumption.

Combining two favourite quotes also gives no dependable entropy figure. Both choices may come from a small, predictable set. Personal details can narrow that set further if your public profile identifies the books, musicians or films you enjoy.

A passphrase's useful randomness comes from **how it was generated**. Memorability can be added after random selection.

Build a memorable passphrase from random choices

Random words offer a practical balance between unpredictability and recall. Use a documented word list and select each word independently, with replacement.

1. Choose a fixed list, such as the EFF long word list, which contains `7,776` entries indexed by five dice rolls.
2. Roll a fair six-sided die five times for each word, preserving the roll order. Look up the corresponding entry. Repeat for every word.
3. Keep the words in their generated order. Accept repetitions and awkward combinations; rerolling until the result sounds like a sentence changes the selection process.
4. Create a vivid mental scene connecting the selected words. Let the random words determine the story, rather than choosing words to fit an existing story.
5. Rehearse recall and keep a secure physical backup. Record exact spelling, order, capitalization and separators.

With that list, each independently selected word contributes about `12.9` bits. Six words provide about `77.5` bits; seven provide about `90.5` bits. Those figures assume fair, independent selection. The appropriate length depends on the cost of testing guesses and the value being protected.

A trusted password manager can generate random words too. Check its word-list size and generation method. A request to invent a memorable phrase does not establish a measurable random process.

Apply this carefully to wallet recovery

For a **BIP39 passphrase**, an attacker who obtains your mnemonic can test passphrase guesses offline. BIP39 derives a 512-bit seed using `PBKDF2-HMAC-SHA512` with `2,048` iterations. It normalizes the mnemonic and passphrase with Unicode `NFKD`, and uses the prefix `mnemonic` followed by the passphrase as the salt. This adds work per guess, but remains vulnerable to predictable choices.

Every BIP39 passphrase produces a seed, so a typo can open a different wallet without an error. Test recovery before relying on the backup, and store the passphrase separately from the mnemonic where appropriate. Generate the mnemonic itself using the wallet's secure generation process; selecting favourite words defeats its randomness.

FAQ

Does an obscure quotation make a strong passphrase?

Obscurity is difficult to measure. Digitized books and transcripts can enter candidate collections. Independently generated random words provide a clearer basis for estimating strength.

Will punctuation or a year fix a lyric-based passphrase?

Predictable edits provide limited protection because cracking tools generate them automatically. Replace the underlying selection method with random generation.

Should I use a published example passphrase?

No. Published examples are known candidates. Generate your own words privately, and never enter a real mnemonic or wallet passphrase into an online strength checker.

Sources

- BIP39: Mnemonic code for generating deterministic keys (github.com)
- EFF Large Wordlist (www.eff.org)

> Continue reading on bip39-phrase.com: Seed Phrase Examples: 12, 18 and 24 BIP39 Mnemonic Phrases
> More on this topic: Seed Phrases

Example sequences from the BIP39 wordlist

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 mango	2 board	3 calm
4 pottery	5 race	6 orange
7 rigid	8 false	9 person
10 spider	11 beauty	12 charge

Example 2

1 lunar	2 genuine	3 stage
4 during	5 close	6 sketch
7 cigar	8 adjust	9 empower
10 reopen	11 glad	12 alone

Example 3

1 unaware	2 energy	3 teach
4 wrong	5 timber	6 cube
7 ski	8 what	9 garment
10 refuse	11 reflect	12 whale

Example 4

1 close	2 cheese	3 lawn
4 heart	5 dove	6 admit
7 matter	8 kiss	9 oven
10 habit	11 paper	12 reform

Example 5

1 camp	2 smoke	3 explain
4 car	5 helmet	6 very
7 flash	8 picture	9 deliver
10 once	11 discover	12 quick

Example 6

1 safe	2 negative	3 radar
4 style	5 buzz	6 random
7 fantasy	8 load	9 project
10 blue	11 crater	12 mail

Example 7

1 emerge	2 bamboo	3 slab
4 market	5 swallow	6 clever
7 yard	8 ten	9 maid
10 box	11 modify	12 noodle

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	mango	1081	1082	10000111001
2	board	197	198	00011000101
3	calm	260	261	00100000100
4	pottery	1351	1352	10101000111
5	race	1411	1412	10110000011
6	orange	1246	1247	10011011110
7	rigid	1487	1488	10111001111
8	false	658	659	01010010010
9	person	1306	1307	10100011010
10	spider	1677	1678	11010001101
11	beauty	157	158	00010011101
12	charge	308	309	00100110100

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): 87231482547b0737ae7a92a35a344e93
```

This sequence ends in the checksum bits 0100, 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

- Splitting a Passphrase Between Two People: Weak Points · November 11, 2026
- Leaving Off the Last BIP39 Word: A False Trick · January 8, 2025

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/