The Importance of Entropy

How Randomness Becomes Sovereignty

Jeremy Bufford

Emergency Field Manual — First Edition

Disclaimer: This book is educational. It is not legal, financial, tax, or investment advice. Bitcoin and self-custody involve real risk of permanent loss. Software and firmware change. Always verify procedures against official vendor documentation and current best practice before moving funds. The author is not responsible for losses arising from use or misuse of the information herein. Security events described (including the Coldcard seed-generation advisory of July–August 2026 and service outages) are summarized from public sources as of the freeze date noted in the preface; always check primary sources for updates.

ISBN: To be assigned (KDP)

Companion books: *Practical Bitcoin*; *Seeds Are Not Enough: The Definitive Bitcoin Wallet Recovery Guide*

Dedication

To everyone who trusted a default button labeled “Generate,” and to everyone who still has time to do it right.

Table of Contents

Preface: The Week Defaults Failed

In late July and early August 2026, two different shocks hit people who believed they were doing Bitcoin the "right" way.

The first was a **seed-generation failure** on COLDCARD hardware wallets. Coinkite's public security advisory described a serious problem: on affected firmware, **device-generated** seeds did not receive the entropy users thought they were getting. For some model and firmware ranges, independent analysis and the vendor's own updates pointed to far fewer bits of effective randomness than a 12-word seed is supposed to encode. Funds controlled by those weak seeds were swept. The fix was not "update firmware and stay put." Updating firmware does **not** repair an old seed. The only durable answer is a **new seed** generated correctly, then a careful migration of coins.

The second shock was quieter but familiar: **Boltz**, a widely used non-custodial swap service connecting Lightning and other Bitcoin layers, announced that swap services were unavailable until further notice, with no ETA. Privacy-minded users who depended on that bridge felt the floor shift. Whether the outage is temporary or permanent matters less for this book than the pattern: **sovereignty tools can go dark**. When they do, people who never practiced alternatives rush back toward custodians.

These are not the same bug. One is about **how secrets are born**. The other is about **how private movement depends on living infrastructure**. Together they teach the same lesson:

> ***Self-custody is not a product you buy. It is a practice of generating, protecting, and using secrets that no one else can predict—and of never depending on a single vendor's defaults or a single privacy rail.***

This is an emergency field manual. It is short on purpose. It exists because **low-entropy keys are being hunted**, and the hunt will only get cheaper. GPUs, specialized hardware, and AI-assisted tooling do not "break Bitcoin." They industrialize the search of every key that was **never random enough**—brainwallets, broken RNGs, hand-picked words, buggy firmware paths, patterned passphrases. That is an arms race. If your livelihood sits behind a key, you need to understand which side of the race you are on.

What this book will do

• Explain entropy in plain language: what "enough" means in bits you can feel.

• Show how Bitcoin turns randomness into seed words and keys—without drowning you in algebra.

• Take Greg Maxwell's criticism of BIP39 seriously: word lists and key-stretching are not substitutes for good randomness.

• Walk a history of real failures so you stop thinking "it won't happen to me."

• Give you **concrete ceremonies** for generating entropy yourself (dice and related methods).

• Teach you to use hardware wallets as **signing devices**, not as oracles of trust.

• Cover passphrases, multisig, and backups without turning inheritance into a second disaster.

• End with checklists you can print and follow under stress.

What this book will not do

• Replace *Seeds Are Not Enough*, which covers recovery, derivation paths, descriptors, and PSBTs in depth.

• Teach you to run a full node stack (see the Sovereign Stack materials if you need that).

• Give you an exploit recipe or a "how to drain wallets" guide. Defenders need clarity; attackers already have tooling.

• Promise that any single brand is forever safe. **Trust procedures, verify entropy, diversify failure modes.**

Freeze date. Current-event claims in this edition are frozen around **3 August 2026**. Firmware versions, service statuses, and loss totals change. Before you move funds, read the primary advisory (for COLDCARD: Coinkite's security advisory) and verify software signatures yourself.

If you only remember one sentence from this preface, make it this:

The seed is not magic. The randomness is.

Chapter 1: The Arms Race

1.1 What attackers actually do

Bitcoin private keys are numbers. Large numbers. A properly generated 256-bit private key sits in a space so vast that guessing one at random is not a practical attack against the network. That fact is why Bitcoin can work without a bank vault.

Attackers know this. So they do not try to "guess Bitcoin." They try to **find keys that were never full-sized secrets**:

- Keys generated with broken or undersized random number generators.
- Keys derived from passwords, song lyrics, or "clever" phrases (brainwallets).
- Keys whose effective entropy collapsed because of a software bug, a bad library, or a firmware path that used the wrong generator.
- Keys partially constrained by known patterns (reused nonces, weak passphrases on top of weak seeds, known device quirks).

Every reduction in real entropy multiplies the attacker's chance. Cut a 128-bit secret down to 40 or 72 bits of effective randomness and you have moved from "age of the universe" into "data-center afternoon"—or worse.

1.2 Why this gets worse, not better

Three forces tighten the noose on weak keys:

Cheaper computation. What was expensive to brute-force in 2013 is routine in 2026. GPUs parallelize simple cryptographic operations. FPGAs and ASICs can be aimed at specific constructions when the economics justify it.

Better automation. Scrapers watch the blockchain and known address sets. Tooling checks candidate keys against funded outputs. When a class of wallets is revealed to be weak, the time from disclosure to drainage can be measured in minutes, not months.

AI as a force multiplier—not magic. Large models do not "solve" 256-bit keys. What they do well is **pattern work**: guessing human-chosen passphrases, ranking likely brainwallet phrases, clustering weak generation

styles, assisting reverse-engineering of bad RNG seeds, and coordinating search strategies. The danger is not science fiction. The danger is **industrialized hunting of the low-entropy tail**.

If your key was generated with true high entropy and protected sanely, you are not in that tail. If it was not, you may already be competing with machines that never sleep.

1.3 Self-custody is under social attack too

Not every pressure is cryptographic. Custodial platforms, regulated intermediaries, and "convenient" apps constantly invite you to hand keys back. When a hardware wallet bug hits the news, scammers arrive in replies offering "help." When a privacy tool goes offline, fear pushes people toward hot wallets and exchanges.

The arms race has two fronts:

1. **Technical:** never create a guessable key.
2. **Human:** never abandon self-custody for comfort at the exact moment calm procedure matters most.

1.4 The only durable defense

You cannot out-lobby a GPU farm with hope. You can:

- Generate secrets with **enough independent randomness** that search is futile.
- Prefer **user-supplied entropy** when a device's generator is untrusted or unverified.
- Add **independent barriers** (strong unique passphrases, multisig across vendors).
- Practice **migration under calm**, not panic.
- Assume that **defaults optimize for convenience**, not for adversarial conditions.

The rest of this manual is how to do those things without a PhD.

1.5 How search actually scales

Security people talk in powers of two because each extra bit **doubles** the work.

• **20 bits** ≈ one million possibilities. A laptop can walk that for fun.

• **40 bits** ≈ one trillion. Serious, but a determined attacker with cloud GPUs can chew through structured 40-bit spaces, especially when each candidate can be checked quickly against known funded addresses.

• **64 bits** is large for casual attackers, still not "retirement grade" against a well-funded adversary over time.

• **72 bits** (the ballpark Coinkite cited for some affected COLDCARD generations) is vastly smaller than 128. It is the difference between "not worth trying" and "build a pipeline."

• **128 bits**, drawn uniformly, remains the standard minimum for modern seeds.

• **256 bits** is overkill against brute force for the foreseeable technological horizon of classical computers; it is still good practice for long-horizon funds.

AI does not delete these numbers. What AI and better software do is:

• find **which** keys are in the weak band (bug classes, wallet versions, human patterns);

• generate better **candidate lists** for passphrases and brainwallets;

• automate monitoring so that when a new class is discovered, drainage is immediate.

If your generation process truly samples a 128-bit space uniformly, you are not in the race as prey. If it samples a 40–72-bit space—or a human-pattern subspace—you are.

Address reuse and "I never posted my seed"

Attackers do not need your seed pasted on a forum. They need a **funded address** derived from a weak key. The blockchain is a public scoreboard. Weak-key hunters scan for balances, not for confessions.

Social acceleration after disclosure

When an advisory drops:

1. Researchers and attackers reproduce the weakness.
2. Scripts generate candidate keys in the reduced space.
3. Derived addresses are checked for balances.
4. Sweeps hit the mempool.

5. Social media fills with fear, fake support accounts, and “send me your seed to check.”

Your job is to be **faster than the sweep if you are affected**, and **slower than the scammers** in how you act. That tension is why Appendix B exists.

What industrialized search looks like

Defenders should understand the shape of the threat:

1. **Candidate generation** — enumerate keys in a reduced space (bug parameters, password lists, partial seed knowledge).

2. **Derivation** — compute addresses for common paths.

3. **Lookup** — check against UTXO sets / indexed address balances.

4. **Sweep** — sign transactions moving funds to attacker-controlled outputs.

5. **Automation** — repeat continuously as new blocks appear and as new weak classes are published.

Your counter is not a faster GPU. Your counter is **never appearing in step 1’s candidate set**.

Chapter 2: What “Enough Randomness” Means

2.1 Entropy is unpredictability, not “looks random”

A string of words can look chaotic and still be guessable. “Correct horse battery staple” is famous because it is *relatively* strong as a password and still tiny compared to a cryptographic key. A 12-word BIP39 phrase generated by a broken RNG can look identical to a good one. **You cannot judge entropy by eye.**

Entropy, for our purposes, means: **how many equally likely possibilities did the generation process actually choose among?** Measured in bits. Each independent fair coin flip is about 1 bit. A fair six-sided die roll is a bit more than 2.5 bits ($\log_2 6 \approx 2.585$).

2.2 Targets that matter for Bitcoin

Goal	Rough size	Typical encoding
Minimum modern seed (common)	**128 bits**	12 BIP39 words (+ checksum)
High-security seed	**256 bits**	24 BIP39 words (+ checksum)
Strong BIP39 passphrase (additional)	**≥80+ bits** of *independent* entropy preferred	Diceware / long random string

These are **targets for the generation process**, not for how pretty the words look.

A process that claims 128 bits but only produces ~40 or ~72 bits of real unpredictability is a **lie with a checksum**. The checksum only proves the words are well-formed BIP39. It does **not** prove the entropy was good.

2.3 Why “I picked words I like” fails

Humans are biased. We prefer familiar words, patterns, themes, keyboard adjacency, and stories. Attackers model those biases. Choosing twelve BIP39 words “at random” from memory is not random. It is a **tiny, structured subset** of the full 2048^{12}-ish space (and in practice far smaller than theory because of human pattern).

Rule: If a human brain is the primary entropy source for the seed itself, assume the key is weak.

Passphrases are different only when you deliberately use a **measured** method (e.g., enough Diceware words from fair dice), not a favorite quote.

2.4 Stretching is not a substitute for entropy

Key derivation functions (KDFs) slow down password guessing by requiring work per guess. BIP39 uses PBKDF2-HMAC-SHA512 with **2048 iterations** and an optional passphrase salt. That is mild by modern password-hashing standards. It is **not memory-hard**. It parallelizes.

If your input is high-entropy (a full 128-bit random seed), the KDF's job is mostly format conversion into a seed binary. If your input is a weak password or a collapsed RNG output, the KDF is a speed bump in front of a train.

As the community line goes—echoing long-standing developer commentary around BIP39:

If the entropy is bad, you have already lost.

2.5 A practical feel for "enough"

- **50 fair independent rolls** of a six-sided die contribute on the order of **128+ bits** when used correctly as the entropy source (or mixed as specified by a well-audited procedure).
- **99+ rolls** are often recommended when you want ~256-bit territory from dice alone (see COLDCARD's published dice guidance and similar sober tutorials).
- **One weak device RNG** can erase all of that advantage if you never mixed independent entropy and the device lied about its generator.

You do not need to become a mathematician. You need to **follow a ceremony that forces enough independent outcomes** and to **never trust a single opaque "Generate" click** with your livelihood.

2.6 Measuring what you are doing

Coin flips and dice (worked numbers)

- Fair coin: 1 bit per flip. 128 flips → 128 bits (if fair and independent).

• Fair d6: $\log_2(6) \approx 2.585$ bits per roll.

• 50 rolls ≈ 129 bits.

• 99 rolls ≈ 256 bits.

These are **upper bounds for perfect execution**. Bias, dependent rolls, transcription errors, and "I re-rolled sixes" destroy the math.

The checksum trap

People see a wallet reject an invalid 12th word and conclude "the wallet checks security." It checks **format**. A weak seed that is correctly checksummed is still weak. Validity ≠ safety.

Passphrase entropy (Diceware sketch)

The Electronic Frontier Foundation and others publish long word lists for passphrase construction. If each word is chosen by fair dice from a list of 7776 words (6^5), each word is ~12.9 bits.

• 6 Diceware words ≈ 77 bits.

• 8 words ≈ 103 bits.

That is a serious **additional** barrier when the mnemonic is high quality —or a critical barrier when seed entropy is degraded. It is not a license to use "Password123" as a BIP39 passphrase.

Threat model question that matters

Ask: **Who benefits if I am wrong about my entropy?**

• If the answer is "someone with a script and a week of GPU time," your bar is higher than "my cousin won't guess it."

• If the answer is "a nation-state for years," you need multisig, operational security, and professional threat modeling beyond this manual.

• Most readers are in the first camp with livelihood-sized balances. **Do not use nation-state despair as an excuse to skip dice.**

Chapter 3: How Bitcoin Turns Entropy into Keys

3.1 The chain of custody of a secret

Simplified path used by most modern wallets:

1. **Entropy** — raw random bits (the actual secret).
2. **Mnemonic encoding (often BIP39)** — those bits (plus a checksum) mapped to words humans can write on paper or metal.
3. **Seed** — PBKDF2 turns mnemonic (+ optional passphrase) into a 512-bit seed.
4. **Master key (BIP32)** — hierarchical deterministic wallet: one seed → many child keys.
5. **Addresses** — public keys and scripts derived along paths (BIP44/49/84/86, etc.).

Seeds Are Not Enough covers recovery, paths, descriptors, and PSBTs in depth. Here you only need one idea:

Everything downstream is worthless if step 1 was weak.

3.2 What the words actually are

BIP39 maps entropy to a fixed English list of 2048 words. Each word encodes 11 bits. A 12-word mnemonic encodes 128 bits of entropy plus a 4-bit checksum. A 24-word mnemonic encodes 256 bits plus an 8-bit checksum.

Implications:

- Valid-looking words ≠ strong entropy.
- The last word is constrained by the checksum; you cannot invent twelve arbitrary list words and expect every wallet to accept them unless you compute the checksum correctly.
- The word list is a **human-friendly barcode** for bits. Treat it that way.

3.3 Passphrases create wallets, not decorations

In BIP39, an optional passphrase is mixed into seed derivation. Empty passphrase and `"hunter2"` are different wallets. There is no “password to the same wallet” in the casual sense—there are **different seeds** for different passphrases.

That is power and danger:

- A strong unique passphrase can protect funds even when the mnemonic is exposed—or when seed entropy is weaker than hoped (as Coinkite noted in the 2026 advisory: a strong passphrase is an independent barrier).
- A forgotten passphrase is **permanent loss**.
- A weak passphrase on a weak seed is a speed bump.

3.4 Why “the wallet will remember for me” is not a plan

Cloud backups, screenshots, exchange accounts, and “email me my seed” flows convert self-custody into someone else’s database. The cryptographic story above only matters if **you** control the secret material and the devices that touch it.

Entropy becomes sovereignty only when:

1. Generation is strong.
2. Storage is offline and redundant without digital copies.
3. Signing paths do not leak the seed to hot machines carelessly.
4. You can recover without the original vendor (see descriptors and interoperable tools in the companion recovery book).

3.5 A walk-through without code

Imagine you perform 128 fair coin flips (or the dice equivalent) offline. You now have a bit string. BIP39:

1. Takes those bits.
2. Computes a checksum from a hash of the entropy.
3. Appends checksum bits.
4. Splits the result into 11-bit chunks.
5. Maps each chunk to a word on the 2048-word list.

You write the words on titanium. Later, any compatible wallet can:

1. Map words back to bits.
2. Verify checksum (catches typos).
3. Run PBKDF2 with your passphrase (or empty).
4. Produce the seed.
5. Derive keys along a path.

If step 0—the flips—was replaced by a buggy function that only effectively varied 40 bits, **every later step still "works."** Addresses generate. Backups restore. Balances show. The only thing missing is **unpredictability**, and you will not notice until an attacker's address list matches yours.

That is why weak entropy is so cruel: **the UX is identical to strong entropy until theft.**

Derivation paths in one paragraph

Different wallets historically used different paths (m/44'/0'/0', m/84'/0'/0', m/86'/0'/0', multisig paths, etc.). Your coins live on paths. Recovery requires knowing or discovering them. **Output descriptors** package this knowledge. Generate strong entropy first; then, the day you fund the wallet, export and store the descriptor offline with your recovery documents—not in the same photo as the seed if you can help it.

Chapter 4: Maxwell's Warning — BIP39 Is Packaging, Not Magic

4.1 Why Bitcoin Core developers were cold on BIP39

BIP39 became the de facto standard because users and hardware vendors adopted it—not because it was universally loved by protocol developers.

Greg Maxwell's widely cited critique of the proposal includes a structural point that still matters years later: **BIP39 mnemonics do not include versioning.** There is no in-band way for the words themselves to say which derivation scheme, script type, or wallet policy they belong to. Maxwell called the lack of versioning a **serious design flaw** and said that on that basis alone he would recommend against the proposal.

Electrum's documentation famously explains why Electrum does **not** generate BIP39 seeds: without version information, recovery becomes a guessing game across paths and wallet types—a threat to user funds when software changes.

That critique is about **long-term recoverability and correctness**, not about word aesthetics.

4.2 The weak KDF problem

BIP39's seed derivation uses PBKDF2 with only 2048 rounds. Compared to modern password hashing (scrypt, Argon2, etc.), that is inexpensive. It is CPU-oriented and parallelizable. High-end GPUs can test large numbers of weak inputs per second.

If users treat the mnemonic like a "password they chose," or if the entropy behind the mnemonic is low, attackers get a friendly playground. If users feed BIP39 **true high entropy**, the KDF is not the main story—the entropy is.

Recent public discussion after the Coldcard incident re-lit these old arguments for good reason: **a standard that encodes secrets does not create secrets.**

4.3 The correct use of BIP39 in 2026

Use — Verdict

Encode 128–256 bits of verified entropy as words for backup — Practical, interoperable, fine

Generate those bits with a broken or opaque RNG and trust the words — Dangerous

Hand-pick words from the list — Dangerous

Assume any 12 valid words are equally safe — False

Rely on PBKDF2 to save a weak passphrase/seed — False

Expect the mnemonic to record derivation metadata — It won't—document descriptors separately

Book position (plain):

> *BIP39 is a **container**. Entropy is the **contents**. Maxwell's dim view is a warning against mistaking the container for safety.*

4.4 What to do instead of "BIP39 religion"

1. **Generate entropy** with a method you can reason about (Chapter 6).

2. **Encode** with BIP39 if you need hardware and software interoperability.

3. **Record output descriptors** (or equivalent wallet configuration) so recovery does not depend on tribal knowledge—*Seeds Are Not Enough*.

4. **Consider Electrum-style versioned seeds** only if your whole threat model and tool chain support them; do not mix standards carelessly.

5. **Never** let marketing language ("military grade seed phrase") replace a ceremony.

4.5 Living with BIP39 anyway

Maxwell's critique does not mean "throw away every hardware wallet." It means:

1. **Do not treat BIP39 as a complete wallet backup standard.** Supplement with descriptors and notes.

2. **Do not treat PBKDF2-2048 as armor for bad passwords.**

3. **Do not invent "better" word lists ad hoc** and expect interoperability.

4. **Do respect Electrum's versioned seeds** if you live entirely in Electrum—but understand migration costs if you later want hardware that only speaks BIP39.

Practical interoperability 2026

Most hardware wallets, Sparrow, many mobile wallets, and recovery tools speak BIP39. For a field manual aimed at regular people, the recommendation is:

- **Use BIP39 encoding.**
- **Supply real entropy.**
- **Document descriptors.**
- **Test recovery on a second wallet before you need it.**

That is how you honor Maxwell's warning without isolating yourself from the ecosystem.

The "versioning" failure mode in real life

Someone finds a 12-word phrase from 2017. Which software created it? Which path? Was there a passphrase? Was it P2PKH or SegWit? Without notes, they open five wallets, try three paths, forget a passphrase variant, and conclude the coins are gone—or they paste the words into a phishing "path finder" website.

Entropy starts sovereignty. Documentation finishes it.

Maxwell, standards politics, and you

Standards debates feel academic until your recovery fails. Maxwell's versioning objection is about **decades-long** coin survival. Bitcoin is designed to still matter decades from now. Your backup must still be interpretable.

Practical habits that implement his concern without waiting for a new BIP:

- Store **output descriptors** with backups.
- Note wallet software name and version used at creation.
- Note derivation path and script type in plain language ("Native SegWit single-sig, Sparrow, 2026").
- Prefer tools that show descriptors by default.
- Re-test restore every year as software UIs change.

Chapter 5: A Short History of Weak Wallets

History is not trivia here. It is a list of ways the same mistake recurs: **trusting a generator you did not understand.**

5.1 Android SecureRandom and early mobile wallets (≈2013)

A flaw in Android's SecureRandom usage led to predictable values in Bitcoin transaction signing and key material on some early wallets. Attackers reconstructed private keys from blockchain data. Funds vanished from wallets whose owners had done nothing "wrong" except trust the platform RNG.

Lesson: Mobile OS cryptography can fail. Your threat model must include **platform bugs**.

5.2 Blockchain.info and web-wallet RNG failures (≈2013)

Blockchain.info's web wallet suffered random-number issues affecting signing; the company publicly addressed thefts and refunds. Browser environments, JavaScript, and entropy collection are a hostile combination when keys or nonces are born on the page.

Lesson: Generating or signing with weak browser entropy is a recurring disaster class.

5.3 BitcoinJS and "Randstorm" (wallets ≈2011–2015)

Years later, researchers (notably work popularized under the Randstorm investigation) showed that widely used early JavaScript wallet stacks could produce keys with **far less entropy than required**, especially in older browsers with weak `Math.random` / SecureRandom constructions. Wallets created in that era may remain vulnerable indefinitely because **you cannot patch a key**.

Lesson: Weak generation is a permanent chain-level fact. Time does not heal bad entropy.

5.4 Brainwallets

Users hashed passwords or phrases to make keys. Attackers precomputed and continuously search likely phrases. Almost every brainwallet with funds was a piñata.

Lesson: Human language is not 256 bits.

5.5 Trust Wallet browser extension (≈2022–2023)

A browser-extension / Wasm code path used an unsuitable pseudorandom generator (Mersenne Twister class—fine for games, not for keys), collapsing entropy dramatically for newly generated wallets on that path. Mobile apps of the same brand were not the same code. Victims learned that **a logo is not a cryptography audit**.

Lesson: Verify the **code path that generates keys**, not the brand on the icon.

5.6 COLDCARD device-generated seeds (firmware era from 2021; public crisis 2026)

Coinkite's July–August 2026 advisory stated that funds from affected COLDCARD seeds are at risk if the seed was created **without** sufficient independent dice entropy and without a strong unique BIP39 passphrase.

Material points from the vendor advisory (verify live docs before acting):

- **Mk2/Mk3:** affected device-generated seeds on firmware **4.0.1 through 4.1.9** inclusive.
- **Mk4, Mk5, Q:** seeds generated before fixed releases also affected; vendor describes roughly **72 bits** of entropy rather than the expected **128** for those cases.
- **Fixed firmware versions** were published (e.g., Mk2/Mk3 **4.2.0+**, Mk4/Mk5 standard **5.6.0+**, Q standard **1.5.0Q+**, with separate Edge tracks).
- **Updating firmware does not repair an existing seed.**
- **≥50 fair independent private dice rolls** mixed at generation are treated as providing sufficient independent entropy; 99+ rolls for ~256-bit dice-only territory on supported flows.

• **Strong unique BIP39 passphrase** is an independent barrier; still migrate.

• **PIN ≠ passphrase.**

• TAPSIGNER, OPENDIME, SATSCARD: different codebases, not this bug.

• Migration must be **calm**: verify backups, install fixed firmware before generating the replacement seed, test transaction, then move the rest.

Community reports described rapid sweeping of affected wallets once the weakness was understood. Exact totals evolve; the operational fact does not: **reduced entropy turns "cold storage" into a race.**

Lesson: Even respected air-gapped devices can ship a generation path that fails. **User-supplied entropy and independent barriers are not paranoia. They are literacy.**

5.7 The pattern

Failure mode — Example class — User-visible symptom

OS/library RNG bug — Android 2013 — Sudden theft, no phishing

Browser/JS entropy — Web wallets, Randstorm — Old wallets drained years later

Human-chosen secrets — Brainwallets — Near-instant loss after funding

Wrong code path — Extension vs mobile — "I used a big brand"

Device RNG path bug — Coldcard 2026 advisory — Device-generated seeds weak

Weak KDF + weak input — Password-like seeds — Offline cracking at scale

Every line is an entropy story.

5.8 Additional failure notes

Nonce reuse and wallet bugs (related class)

Some historical thefts came not from weak seeds but from **bad signing randomness** (same nonce used twice in ECDSA signatures), which leaks private keys. The theme is identical: **cryptography assumes fresh unpredictability**. Whether at keygen or at signing, "random" that isn't random is fatal.

Library supply chain

Wallet apps depend on libraries. A vulnerability in a common key-generation library can affect many brands at once. Diversifying signers in a multisig is partial insurance against monoculture bugs.

Why refunds are not a strategy

Some companies historically refunded victims of their RNG bugs. Many cannot or will not. Attackers who sweep to mixers or multiple hops do not reverse. **Assume no bailout.**

Timeline mindset

Era — Dominant weak pattern

2011–2015 — Browser JS entropy, early web wallets

2013 — Mobile OS RNG failures

Ongoing — Brainwallets, reused passwords as keys

2022–2023 — Extension/Wasm bad PRNG paths

2026 — Hardware device-generation path failure in respected cold wallets

The industry did not "solve entropy." It **relocated** the failure.

Why open source still failed users (and why it still matters)

The Coldcard issue is a case study in limits:

- Open source allows **eventual** discovery and public analysis.
- It does not guarantee timely discovery before attackers.
- Linker/name collision class bugs are mundanely human—exactly the sort of engineering failure that appears outside crypto too.
- Users who mixed dice were protected by **architecture of entropy**, not by brand mystique.

Takeaway: Prefer open, verifiable tools **and** user-supplied entropy **and** defense in depth. Any one layer can fail.

Chapter 6: How Regular People Generate Real Entropy

This chapter is the heart of the manual. You do not need exotic hardware. You need a **ceremony**.

6.1 Principles before techniques

1. **Independent outcomes.** Fair dice, drawn cards from a well-shuffled deck used correctly, or a hardware RNG you have reason to trust—and preferably more than one source mixed carefully.

2. **Offline.** No cameras pointed at the table. No cloud notes. No "I'll type it into my phone."

3. **Verify.** Prefer tools that show the math (COLDCARD's documented dice verification, offline Ian Coleman BIP39 tool on air-gapped media from a verified hash, SeedSigner-style flows, etc.).

4. **Write once, correctly.** Metal or paper backup verified by reading back and checking a derived address on a trusted screen.

5. **Never photograph seed material.**

6.2 Dice ceremony (recommended baseline)

Materials

• Casino-quality or otherwise fair six-sided dice (multiple dice speed the process).

• Pen and paper for **roll log** if required by your method—or direct entry into an air-gapped device that never leaves your sight.

• An air-gapped hardware wallet or offline computer that will **only** compute the mnemonic from rolls—not a daily-driver laptop.

Target

• At least **50** independent rolls for ≈128-bit class when using methods that map rolls to full entropy (follow your device's documented mapping).

• **99+** rolls when aiming at ≈256-bit dice-only generation on devices that support that path.

Process (generic; align with your device manual)

1. Clear the workspace. No phones with cameras casually facing the desk.

2. Roll. Record each result carefully if recording is required. Do not "fix" rolls you dislike—that injects bias.

3. Enter rolls into the **official dice entry path** of an air-gapped device, or into a verified offline BIP39 tool that accepts dice/binary entropy.

4. Generate the mnemonic. Write it on paper/metal.

5. **Verify:** confirm the device fingerprint / first addresses match what your offline tool shows if you used dual computation.

6. Destroy roll logs if they would allow reconstruction. The mnemonic **is** the secret now; redundant plaintext logs are liabilities.

7. Fund a **tiny** test amount first. Receive it. Spend it. Then deposit real savings.

COLDCARD and other vendors publish exact UI paths and verification math. **Use their current docs**, not memory of a blog post from 2019.

6.3 Why "Add Dice" on a weak device still helped

In the Coldcard advisory, dice rolls were hashed together with device entropy on affected firmware. Sufficient independent rolls (≥50) meant the **dice alone** supplied enough entropy even when the device contribution was weak. That is the entire philosophy of this book: **bring your own randomness.**

After fixed firmware, device generation may be fine again—but the habit of user entropy remains healthy, especially for large balances.

6.4 Cards, coins, and multi-source methods

- **Coins:** many flips work but are slow; bias and recording errors are real.
- **Cards:** need a careful mapping to bits and a full shuffle; easy to mis-implement. Prefer well-documented methods.
- **Multi-source:** XOR or cryptographic combination of independent sources can hedge single-source failure—if and only if you use a correct combination method from a trusted procedure. Do not invent crypto.

6.5 What not to do

• Do not use Excel `RAND()`, website “random number” pages, or chatbots to generate seeds.

• Do not reuse entropy across wallets.

• Do not generate on an online computer “just this once.”

• Do not let a stranger or a “helpful” DM walk you through seed entry on a website.

• Do not type seeds into phones to “back up to photos.”

6.6 Dual generation (high assurance)

For serious balances:

1. Generate mnemonic from dice on Device A (or offline tool).

2. Independently compute expected addresses / fingerprint with a second offline method.

3. Only when they match do you trust the backup.

4. Prefer **multisig** across device vendors so one RNG failure cannot unilaterally spend (Chapter 8).

6.7 Operational details that prevent self-owning

Dice quality and handling

• Prefer dice that are not obviously deformed. Casino dice are a nice-to-have, not a religion.

• Roll into a tray so dice do not bounce onto the floor and get “re-rolled” inconsistently.

• If a die cocks against a wall, re-roll **that die** by a pre-committed rule —do not cherry-pick.

• Multiple dice per throw are fine if you read them in a fixed order (left to right) every time.

Air-gapped computer hygiene (if not using hardware dice entry)

If you must use a laptop offline:

1. Prefer a live OS (e.g., Tails or a dedicated offline Debian) booted from verified media.

2. Never reconnect that session to the internet after seed material exists in RAM.

3. Use a BIP39 tool whose hash you verified on a separate machine before going offline.

4. Prefer typing entropy as dice numbers over pasting from a USB stick of unknown origin.

5. Power off and, if paranoid, let memory decay; do not hibernate.

SeedSigner and open air-gapped devices

Open implementations that emphasize transparent seed generation (including dice) are valuable for people who want to **see** the process. Follow their current documentation; verify software like any other critical tooling.

Ian Coleman BIP39 tool — the correct attitude

Widely used, also widely phished. Rules:

• Download from the real repository.

• Verify the hash.

• Open **only** offline.

• Prefer the version you copied to read-only media.

• Never use a random website that "hosts" the tool online for real seeds.

After generation: the first hour

1. Backup

2. Verify backup by restoring to a second device **or** by deriving the same first address via second tool

3. Receive $5–$50 equivalent test

4. Send test out

5. Only then deposit real funds

Skipping the test because "fees are high" is how people discover a wrong passphrase after the life savings move.

Metal stamping checklist

☐ Buy a plate system you have practiced on with blank words.

☐ Stamp in a consistent case and numbering (word 1–24).

☐ After stamping, verify letter-by-letter against paper draft.

☐ Destroy paper draft securely if metal is primary—or store paper as temporary only.

☐ Do not stamp your name, home address, or “Bitcoin” on the same plate if theft of the plate is a concern; security through obscurity is weak, but labeling “LIFE SAVINGS BTC” on a visible plate is unnecessary help to a burglar.

☐ Consider two plates in two buildings.

Chapter 7: Hardware Without Faith

7.1 The right mental model

A hardware wallet is a **signer and a vault for key material**, not a priest.

Healthy assumptions:

- Firmware can have bugs.
- Supply chains can be interfered with (buy carefully, verify packaging and signatures).
- Vendor servers can be down; you still need your coins.
- "Air-gapped" helps only if seeds never touch hot machines and QR/PSBT flows are understood.

Unhealthy assumptions:

- "It's cold, so it's safe forever."
- "The screen would never lie." (Advanced attacks exist; reduce phishing surface anyway.)
- "Defaults are for experts' grandmothers, so they must be fine for me."

7.2 Prefer user entropy on day one

When a device offers **New Wallet** vs **Dice Rolls / Import entropy**:

- For non-trivial funds, prefer **dice or external entropy** paths you understand.
- After any vendor advisory affecting generation, **do not generate on unfixed firmware**.
- After fixed firmware, still consider dice for large stacks—habit and defense in depth.

7.3 Verify firmware

Install firmware only from official sources with signature verification as documented. A "helpful" binary from a forum is an attack.

7.4 Use PSBTs and air-gaps for large moves

Build transactions on an online watch-only wallet (e.g., Sparrow with a watching wallet). Sign on the offline device. Broadcast from the online machine. This is standard practice in 2026 for serious self-custody. Details live in recovery and wallet-tool books; the entropy lesson is: **signing can be air-gapped even when generation was.**

7.5 After a generation advisory (generic protocol)

1. **Stop** generating new seeds on possibly affected firmware.
2. Read the **primary** vendor advisory end to end.
3. Determine whether **your** seed is in the affected set (model, firmware at generation time, dice exception, passphrase).
4. If at risk: prepare migration with Appendix B—**calmly**.
5. Assume scammers will impersonate support. Real support will not ask for your seed.

7.6 Diversity beats brand loyalty

If all keys are born on one device model and one firmware lineage, you share one fate. Multisig across independent implementations (different vendors, one air-gapped software signer, geographic separation) converts a single RNG bug into a non-fatal incident.

7.7 Buying, verifying, and living with hardware

Purchase channel

- Prefer reputable vendors and official stores.
- Be wary of “discount” devices on secondary markets that may be tampered.
- On arrival, inspect packaging seals as documented by the manufacturer.
- Initialize yourself; never accept a device that arrives with a seed.

Firmware verification is not optional for livelihood funds

Vendors publish signatures and verification steps. Do them. An attacker who can make you install malicious firmware owns your generation and signing.

What the 2026 Coldcard event means for all brands

Even if you never bought a Coldcard, the event proves:

1. Air-gapped marketing does not eliminate generation bugs.
2. Community and vendor disclosure can still leave a window where funds are raced.
3. **Dice and passphrases** are not eccentric hobbyism.
4. Multisig would have limited blast radius for many users.

Apply the same skepticism to every device that offers a one-button new wallet.

Signing vs generation

A device can be acceptable as a **signer** for a seed generated elsewhere (imported after dice on an offline tool) even when you distrust its TRNG. Read the manual for import paths. Some advanced users generate seeds on one system and import to another so that no single RNG is trusted alone—combined with multisig for real assurance.

Watch-only online wallet checklist

☐ Import **public** descriptor or xpub only.

☐ Verify that the receive addresses match the hardware screen before announcing an address to payers.

☐ Use PSBT for spends.

☐ Never enter seed into the online machine “to make it easier.”

Chapter 8: Passphrases, Multisig, and Backups That Do Not Leak

8.1 Passphrases done sanely

Do

• Use a **long, high-entropy** passphrase (Diceware with enough words, or a random string generated offline).

• Treat it as **wallet-defining secrets**, backed up separately from the mnemonic if your threat model requires (or together in a sealed scheme you have tested).

• Test recovery of the passphrase wallet with a small amount.

Do not

• Use quotes from movies, single dictionary words, or reused website passwords.

• Store the only copy in your head without a durable backup plan.

• Confuse **device PIN** with **BIP39 passphrase**.

In the Coldcard 2026 advisory, a strong unique passphrase was described as an independent barrier against reduced seed entropy—but **not** a reason to skip migration.

8.2 Multisig as entropy and vendor insurance

A 2-of-3 multisig where keys are generated on different devices (or one dice-generated software key offline) means:

• One compromised or weak key does not move funds.

• One vendor failure does not end self-custody.

• Inheritance and geographic distribution become designable.

Cost: more operational complexity. For livelihood-scale balances, complexity is cheaper than ruin.

Record **output descriptors** and a clear spending policy. Future-you must not reverse-engineer a multisig from vibes.

8.3 Backup materials

- **Metal** backups survive fire and water better than paper. Stamp carefully; verify every character.
- **Paper** is acceptable if sealed, dry, private, and duplicated.
- **Shamir's Secret Sharing** (e.g., SLIP39 where supported) or multisig can avoid a single location holding full spend authority. Use battle-tested tools; test restoration.
- **Never** email, cloud-photo, or password-manager-store seed words unless you have fully accepted that those systems are now co-custodians—and usually that is a bad acceptance.

8.4 Inheritance without accidental publication

Dead-man switches, lawyers' sealed envelopes, and multisig with a trusted attorney or family multisig key are operational choices. Whatever you choose:

- Write instructions that a non-technical executor can follow **without** posting seeds into email.
- Include wallet descriptors and the location scheme for secrets.
- Practice a restoration drill while you are alive.

Entropy that dies with you without a plan is not sovereignty; it is a lost cemetery of coins.

8.5 Worked policy examples

Example policy: household savings (illustrative)

- 2-of-3 multisig.
- Key A: dice-generated on air-gapped device brand 1.
- Key B: dice-generated on air-gapped device brand 2.
- Key C: SeedSigner or offline software key in a bank safe deposit / different city.
- Passphrases: only if every co-signer can operationally handle them; otherwise omit and rely on multisig + metal.
- Descriptor printed and stored with attorney instructions sealed.
- Annual restore drill on testnet or with tiny mainnet amount.

Example policy: individual who travels

- Single-sig cold storage with 24-word dice seed + strong Diceware passphrase.
- Metal backup in location A; passphrase backup in location B.
- Hot Lightning wallet with only weekly spend.
- Travel: no seed materials in luggage; use a separate spending wallet.

Backup anti-patterns

- Photos of seed words in Google Photos / iCloud.
- Seed in password manager that syncs to phone.
- Seed split as "first six words with brother, last six with me" **without** a proper secret-sharing scheme (simple splits can leak or fail awkwardly).
- Only copy stored in a single household drawer during wildfire/flood season.

Human factors during migration

Mistakes that destroy funds

- Transposing two words on the new backup.
- Setting a passphrase on generate and omitting it on restore.
- Sending to an address that was generated on a different passphrase wallet.
- Wiping the old wallet before the new balance confirms.
- Using a phishing site that looks like a "seed path calculator."
- Letting a "technician" screen-share while the seed is visible.

Practices that save funds

- Two-person read-back of metal plates (both family members read aloud).
- Fingerprint (XFP) noted on paper next to the metal.
- Test amount that hurts a little if lost but does not ruin you.
- Overnight pause if exhausted—**if** you are not in an active weak-seed race. If you are in an active race, tag-team with a trusted person so fatigue does not cause typos, but do not hand them the seed.

Chapter 9: When the Rails Break

9.1 Privacy tools are infrastructure

Non-custodial swaps, CoinJoin-class tools, Lightning bridges, and peer-to-peer markets exist so you can move value without handing keys to an exchange. When **Boltz** announced swap services unavailable until further notice (3 August 2026), users who treated it as permanent infrastructure felt stranded.

Whether any one service returns is beside the point. **Single points of failure in the privacy layer are real.**

9.2 What resilience looks like

- Know **more than one** way to move between layers (multiple swap providers, submarine swaps via your own node, on-chain settlement, peer trades).
- Run or use infrastructure you control when amounts justify it (your node, your channels).
- Accept that privacy may become **slower and more expensive** under pressure—plan fees and time.
- Do not respond to an outage by pasting seeds into a random new app.

9.3 Legal and social pressure

Mixing tools and privacy services have faced regulatory and enforcement pressure in multiple jurisdictions. This book does not give legal advice. It states an operational fact: **if your sovereignty design requires a specific company to stay online forever, you have built a single point of failure.**

9.4 The link back to entropy

When rails break, people panic-migrate. Panic-migration is when seeds get photographed, typed into phishing sites, or replaced with exchange custody. The calm generation and backup habits from earlier chapters are what keep a service outage from becoming a total loss.

9.5 Practical alternatives when a swap dies

Without naming a forever-list of “use these brands” (they churn), categories of resilience:

1. **Self-hosted Lightning node** with channels you manage—higher skill, higher independence.
2. **Multiple non-custodial swap providers** bookmarked and tested with dust amounts **before** crisis.
3. **On-chain** settlement when privacy tools are offline—accept the tradeoff consciously.
4. **Face-to-face or reputation P2P** within communities you already trust.
5. **Different layers** (where legally and practically available) without concentrating all privacy hope in one bridge.

Test small. Document what worked. Do not discover your only path during a liquidation event.

Emotional protocol

Service outage → breathe → check official account only via bookmarked URL → use pre-tested alternative → if none, wait or go on-chain → never install a random APK from a reply guy.

Chapter 10: A Durable Doctrine

Write these as personal rules. Revisit yearly.

10.1 Doctrine cards

1. **I generate entropy I can explain.** If I cannot explain where the bits came from, I do not trust the key for livelihood funds.

2. **Defaults are suspects.** “Generate” is convenient; ceremony is safer.

3. **BIP39 encodes; it does not bless.** Maxwell was right to distrust packaging-as-security.

4. **One vendor is not a strategy.** Multisig and multi-path recovery beat brand faith.

5. **Firmware fixes the future, not the past.** Bad seeds stay bad. Migrate.

6. **Test with small amounts.** Always.

7. **Under attack, slow down.** Scammers sprint; you verify.

8. **Privacy rails are temporary.** Practice alternatives before you need them.

9. **Inheritance is part of custody.** Secrets that cannot be recovered by the right people after you are gone are unfinished work.

10. **I am in an arms race.** Attackers automate search of weak keys. I refuse to create weak keys.

10.2 Livelihood tiering

Not every sat needs a 3-of-5 multisig ceremony.

Tier — Example use — Generation bar

Spend — Coffee, LN pocket — Decent wallet, still no brainwallets

Savings — Months of expenses — Dice or fixed HW generation + passphrase

Livelihood / generational — What you cannot afford to lose — Dice/multi-source, multisig, metal, tested recovery, descriptors recorded

Be honest about which tier you are funding when you click generate.

10.3 The moral weight

Self-custody is often sold as freedom. It is also **responsibility**. Weak entropy does not only risk "crypto Twitter embarrassment." It risks rent, payroll, medical buffers, and family savings. Treating generation as a chore to skip is how livelihoods end up in an attacker's clustering script.

Do the ceremony. Teach one other person. When the next advisory hits, be the calm one.

10.4 Annual drill

Once a year, calendar it:

1. Restore metal backup to a wiped air-gapped device or test wallet.
2. Verify descriptors still import into a current Sparrow/Electrum.
3. Rotate passphrases only with full ceremony if you choose to rotate (rotation is optional; botched rotation is catastrophic).
4. Check firmware release notes for generation-related fixes.
5. Re-affirm tiering: has "spend" money crept into a weak hot wallet?
6. Update inheritance packet.
7. Read one post-mortem of a wallet failure (Chapter 5 style) to keep fear healthy, not paralyzing.

10.5 Notes on AI-assisted attacks (precise language)

You will hear extremes: "AI broke crypto" and "AI changes nothing." Both are wrong.

AI changes the economics of attacking human-generated and bug-reduced secrets. It does not replace the need for 128-bit uniform secrets. It makes lazy generation less forgivable because the opponent is stronger every year.

If you generate keys with song lyrics, dates of birth, slight variations of common passphrases, or broken RNGs, you are volunteering for a dataset that improves with every model generation.

If you generate keys with physical dice and correct procedure, you remain in the mathematical shelter Bitcoin promised.

10.6 Notes on "sufficient" for different enemies

Adversary — Weak seed — Strong 128-bit seed — Strong 256-bit + multisig + opsec

Script kid with GitHub tools — Often fatal — Safe — Safe

Crime gang with GPUs — Fatal for weak classes — Safe — Safe

Targeted theft (home invasion) — Seed plates at risk — Seed plates at risk — Multisig + split locations help

Malware on daily PC — Hot keys dead — Cold keys OK if never entered — Same

Vendor RNG bug — Shared fate — User dice / multisig help — Best

Match controls to adversaries you actually face. Most readers need the first two rows solved **perfectly** before obsessing over nation-state theater.

10.7 Teaching script (ten minutes for a meetup)

If you run a Bitcoin meetup, teach this—not price charts:

1. Show two identical-looking 12-word phrases (fake demo words, not real). Explain one is full entropy, one is 40-bit toy space—**indistinguishable by sight**.

2. Explain bits with dice.

3. Explain Maxwell in one sentence: packaging ≠ security.

4. Tell the Coldcard story carefully: device generation failed users who trusted defaults; dice users fared better.

5. Hand out Appendix C and G.

6. Refuse to generate anyone's real seed at a public table.

You may save a stranger's future self.

Chapter 11: Scenario Walk-Throughs

These scenarios turn the earlier chapters into action under stress. Match the one closest to your situation and follow the order carefully.

11.1 “I used New Wallet on a COLDCARD in 2023”

Situation. You bought a COLDCARD, tapped through setup, wrote twelve words, never used dice, used a short passphrase or none. You hold a material balance.

What the 2026 advisory implies. If your model/firmware at generation time falls in the affected ranges published by Coinkite, your seed may have far less entropy than you believed. Community drainage of similar wallets is not a drill.

Actions (order matters).

1. **Do not** post your words anywhere to “check if safe.”
2. Open **only** the official Coinkite advisory URL you typed or bookmarked—not a link from a reply.
3. Match **model + firmware at generation** (if you logged it) against affected ranges. If you never updated and generated years ago on Mk2/Mk3 after 4.0.1, assume concern until proven otherwise.
4. If you used **≥50 private fair dice** at generation, you may be in the exception—confirm against the advisory language.
5. If you used a **strong unique BIP39 passphrase**, you have a barrier; still plan migration.
6. Install **fixed firmware** before generating the replacement seed on that device.
7. Execute Appendix B: new seed (dice preferred), test send, then full migrate.
8. Keep old backup until the new wallet shows the full balance and you have spent a test output from it.

Emotional note. People freeze. Freezing while weak keys are searchable is how livelihoods disappear. Moving without a test is how people fat-finger a wrong passphrase. **Urgency with procedure** is the only adult path.

11.2 “I generated on a phone wallet in 2014”

Situation. Old Android wallet, small forgotten balance now large in fiat terms.

Risk class. Historical OS and library RNG failures; possible Randstorm-era web exports; unknown backup quality.

Actions.

1. Treat the seed as **potentially weak and potentially already copied** if it ever touched email or cloud.
2. Create a **new** livelihood-grade wallet with dice on modern air-gapped tooling.
3. Sweep old wallet to new address ASAP after verifying the new setup with a test.
4. Do not “improve” the old seed with a passphrase and stay—**new entropy**.
5. If you cannot find the seed but have a wallet file, seek professional recovery help carefully; this manual cannot cover every file format.

11.3 “Boltz is down and my LN funds feel stuck”

Situation. You used non-custodial swaps to move between Lightning and on-chain. The service shows unavailable.

Actions.

1. Confirm via official account/site bookmarks.
2. Check whether your wallet supports **other** swap backends or channel management.
3. If you run a node, use your own liquidity paths; if not, consider waiting or moving via another pre-tested provider with a tiny amount first.
4. Do not install unverified apps promising “emergency exit.”
5. Separately: review whether your **on-chain cold storage** entropy is sound. A swap outage is a bad day. A weak cold seed is a bad life.

11.4 “A stranger offers to verify my seed”

Situation. After any hack news, DMs explode.

Script.

> *"I will never share seed words, QR seeds, or partial words. Real help never needs them."*

Block. Report. If you already shared, migrate immediately—those funds are racing.

11.5 "I want to help my parents"

Situation. Family holds BTC on an exchange or a single phone wallet.

Minimum viable upgrade.

1. Sit together offline.
2. Explain: the words are money; photos of words are money.
3. Perform a dice ceremony on a hardware wallet you verified, or a reputable air-gapped process you understand.
4. Metal backup; second location.
5. Practice restore once.
6. Leave a one-page instruction sheet for heirs **without** putting the seed on that sheet—only how to find sealed materials and whom to call.

Do not hand parents a 2-of-3 multisig on day one unless you will be their ongoing operator. Complexity that they abandon is worse than simple strong single-sig they can run.

11.6 A short letter you can send a friend

> *I'm not writing about the price. I'm writing because low-entropy Bitcoin keys are being hunted at scale, and a major hardware wallet just reminded everyone that "Generate" is not a sacrament. If your seed was created by tapping a default button on any device—or by picking words yourself—please read up on dice-generated entropy and consider migrating carefully with a test transaction. Do not share your words with anyone who messages you. The seed is not magic. The randomness is.*

Appendix A: Generation Ceremony Checklist

Print this page.

☐ I know the tier of funds this wallet will hold (spend / savings / livelihood).

☐ Workspace is private; cameras covered or removed.

☐ I am using fair dice or another documented high-entropy method.

☐ Device firmware is current **and** not subject to a generation advisory (or I am using dice-only path as documented).

☐ I will enter ≥**50** rolls for ~128-bit class, or ≥**99** for ~256-bit dice-only where required.

☐ I will not reject rolls I “don’t like.”

☐ Mnemonic written on durable backup; verified by read-back.

☐ Optional passphrase generated with measured entropy; backed up; tested.

☐ Wallet fingerprint / receive address verified on device screen.

☐ Dual-tool verification done (if livelihood tier).

☐ Test transaction received and spent successfully.

☐ Roll logs destroyed if they could reconstruct the secret.

☐ No photos, cloud, or chat of seed material.

☐ Descriptors / recovery notes stored **without** exposing seeds in the same digital file if possible.

Appendix B: Migration Checklist (Calm Protocol)

Use when a seed may be weak or compromised.

☐ I will not reply to DMs offering recovery help.

☐ I have read the primary vendor or security advisory myself.

☐ I have confirmed whether my seed is in the affected set (model, firmware at generation, dice exception, passphrase strength).

☐ Written backups of the **old** wallet are verified (words + passphrase if any).

☐ Fixed firmware installed and version confirmed on-device **before** generating the new seed (if using that device).

☐ **New** seed generated with strong entropy (dice preferred).

☐ New backup written and verified; fingerprint recorded.

☐ New receive address verified on trusted screen.

☐ Small test amount sent to new wallet; confirmed.

☐ Remaining funds moved in deliberate batches if needed; fees accepted as cost of safety.

☐ New wallet confirmed full balance; old backup retained until completely sure.

☐ Old seed retired; do not reuse.

☐ I slept on any step that felt rushed.

Appendix C: If You Only Do Five Things

1. **Never hand-pick seed words or use brainwallets.**
2. **Generate livelihood keys with dice (or equivalent verified entropy), not blind defaults.**
3. **Use a strong unique BIP39 passphrase on serious funds—and back it up.**
4. **Prefer multisig across vendors for money you cannot lose.**
5. **When bad news hits, migrate calmly with a test transaction—never via a stranger's link.**

Appendix D: Glossary

BIP39 — Common standard mapping entropy to mnemonic words and deriving a seed via PBKDF2.

BIP32 / HD wallet — Hierarchical deterministic derivation: one seed, many keys.

Checksum (BIP39) — Extra bits encoded in the mnemonic that detect typos; **not** a proof of strong entropy.

CSPRNG — Cryptographically secure pseudorandom number generator.

Descriptor — Compact description of a wallet's script and keys for interoperable recovery.

Entropy — Unpredictability of the generation process, measured in bits.

KDF — Key derivation function; slows password-style guessing; does not fix bad entropy.

Mnemonic — Seed words.

Multisig — Multiple keys required to spend.

Passphrase (BIP39) — Optional extra secret mixed into seed derivation; creates a different wallet.

PBKDF2 — Password-based KDF used by BIP39 (2048 iterations in the standard).

PSBT — Partially Signed Bitcoin Transaction; enables air-gapped signing.

TRNG / HRNG — True/hardware random number generator.

Watch-only wallet — Public keys/descriptors online without private keys.

XFP / fingerprint — Short identifier for a master key used to recognize wallets.

Appendix E: Threat Model Worksheet

Copy and fill by hand.

1. What am I protecting? Amount range: _____ Time horizon: _____

2. Who might attack me? ☐ Opportunistic thief ☐ Remote malware ☐ Intimate adversary ☐ Scalpers of weak keys ☐ Other: _____

3. How was my current seed generated? Device/software: _____ Firmware/version if known: _____ ☐ Device default ☐ Dice ≥50 ☐ Dice ≥99 ☐ Other: _____ ☐ Passphrase yes/no Strength: _____

4. Am I in a known affected class? Advisory checked (date/source): _____ Decision: ☐ OK ☐ Migrate ☐ Need help from trained friend (not Twitter)

5. Backup locations 1: _____ 2: _____ Passphrase location: _____

6. Recovery test last date: _____ Result: _____

7. Multisig? ☐ No ☐ Yes (m-of-n: _____) Vendors: _____

8. Next action date: _____

Appendix F: Frequently Asked Questions

Q: Is Bitcoin broken? A: No. Weak generation is broken. The network is doing what it always does: enforcing keys.

Q: Should I panic-sell to an exchange? A: Exchanges fix entropy risk by introducing custodial risk, freeze risk, and honeypot risk. If you must use an exchange temporarily, that is a deliberate trade—not a moral victory. Prefer migrate to a strong new self-custody setup.

Q: Are 12 words enough? A: 12 words encoding **true 128 bits** are standard. 24 words encode 256 bits. Neither helps if the generator was weak. Prefer 24 for livelihood if the ceremony cost is acceptable.

Q: Is a passphrase enough to save a weak Coldcard seed? A: Coinkite stated a strong unique passphrase is an independent barrier but still urged migration. Weak or reused passphrases do not qualify. Migrate.

Q: Can I generate a seed with ChatGPT / Claude / Grok? A: **No.** Language models are not personal entropy sources for keys. Anything that appeared in a chat may be logged. Never.

Q: Dice on a video call with a friend for "witnessing"? A: No. Witnesses learn your entropy. Ceremony is private.

Q: What if I already photographed my seed years ago? A: Assume compromise of that seed's privacy. Migrate to a new ceremony-generated seed. Delete cloud copies if you can; assume attackers already could have them.

Q: Does multisig remove the need for good entropy? A: It reduces single-key failure impact. Each key should still be generated well. Two weak keys are not a clever plan.

Q: Why not only use an exchange ETF? A: That can be a valid financial choice for some people, but it is not self-custody. This book is for people who need or choose sovereignty.

Q: How fast must I move if affected by the Coldcard advisory? A: If you are in the affected set without dice exception or strong passphrase, treat it as **urgent**, not leisurely—but still follow Appendix B. Botched migration loses funds as surely as slow migration.

Q: Where do I learn recovery paths and PSBTs? A: *Seeds Are Not Enough* by the same author.

Appendix G: One-Page Doctrine

THE IMPORTANCE OF ENTROPY — DOCTRINE

1. The seed is not magic. The randomness is.
2. If the entropy is bad, you have already lost.
3. BIP39 is packaging. Maxwell warned you.
4. Defaults optimize convenience. Ceremony protects livelihoods.
5. Firmware fixes the future. Only migration fixes the past.
6. Dice ≥50 (better ≥99 for big stacks). No brainwallets. No chatbot keys.
7. Passphrase ≠ PIN. Strong and unique, or don't bother.
8. Multisig across vendors for money you cannot lose.
9. Test tiny. Then move the rest.
10. When the internet screams, you slow down.

Jeremy Bufford — Emergency Field Manual 2026

About the Author

Jeremy Bufford is an early Bitcoin advocate and entrepreneur who writes about Bitcoin as a monetary phenomenon, a payments technology, and an economic theory of value. His practical self-custody titles include *Practical Bitcoin* and *Seeds Are Not Enough: The Definitive Bitcoin Wallet Recovery Guide*.

Further Reading

- BIP32, BIP39 (Bitcoin BIPs repository) — read critically.
- BIP39 comments wiki — including Greg Maxwell's versioning critique.
- Electrum documentation on seed versioning and motivation.
- Historical reporting on Android SecureRandom (2013), Blockchain.info RNG issues, Randstorm / BitcoinJS, Trust Wallet extension entropy failure.
- *Seeds Are Not Enough* by Jeremy Bufford — recovery, paths, descriptors, PSBTs.
- Device vendor docs for dice-roll math and firmware verification (COLDCARD, SeedSigner, and others).

Closing

Randomness becomes sovereignty when you refuse to outsource the birth of your keys to an opaque default—and when you build practices that survive vendor bugs, service outages, and automated search.

Generate carefully. Verify always. Migrate calmly. Teach someone else.

The arms race is real. **Bring enough entropy.**

This field manual is the **generation** book in a practical set:

1. *Practical Bitcoin* — enter the system.
2. *The Importance of Entropy* — create true keys.
3. *Seeds Are Not Enough* — recover and migrate with descriptors and PSBTs.

Own the secret. Prove you can restore it. Only then treat the balance as yours in the fullest sense.