

Biscuit Wallet

Architecture, keys, network and swaps

The Biscuit developers

Version 1.0, October 2026. Describes Biscuit 1.2.2.

<https://biscuitwallet.com/whitepaper/>

Abstract

Biscuit is a free desktop wallet for Windows, macOS, Linux and Tails, built on Feather Wallet. Monero is the only coin every wallet holds; Bitcoin, Litecoin, Dogecoin, Ethereum and Arbitrum (with USDT and USDC) and Solana are optional. This paper describes how Biscuit is built: its components, how seeds and wallet files are protected, every network connection it makes, its Tor mode, exchange swaps through a public relay, atomic swaps from Bitcoin to Monero, and how releases are built and updates checked. It ends with what Biscuit does not protect against.

Contents

1 What Biscuit is	1
2 Architecture	2
3 Seeds and keys	3
4 Wallet files	3
5 Network	3
6 Tor	4
7 Our server and exchange swaps	4
7.1 Public data	4
7.2 Swaps through Trocador	4
7.3 Our own exchanges	5
7.4 In the app	5
7.5 Abuse limits	5
8 Atomic swaps	5
9 Releases and updates	5
10 What Biscuit does not protect against	6

1 What Biscuit is

Biscuit is a desktop wallet. Monero comes first: it is the only coin every wallet has. The other coins can be added one by one, or not at all. Biscuit is free software under the BSD 3-Clause License, built on Feather Wallet, whose Monero code it keeps.

A few rules decide most of what follows:

- **No account.** Nothing to sign up for, no identifier sent with any request, no analytics, no crash reports.

- **No remote configuration.** What Biscuit does is decided by the version installed. Our server sends data, never instructions.
- **Every connection is listed.** Each one has a reason, and can go through Tor or be replaced by a server of the user's own.
- **Small first.** A new coin is judged by the code and the network surface it adds. A coin that is not added makes no connection at all.
- **Checkable.** Releases are reproducible, and the server code is public.

2 Architecture

Biscuit is one C++ program with a Qt 6 interface, plus a helper for atomic swaps. A small server, run by us, serves the website, price data and the swap relay. Figure 1 shows the parts and what each one talks to.

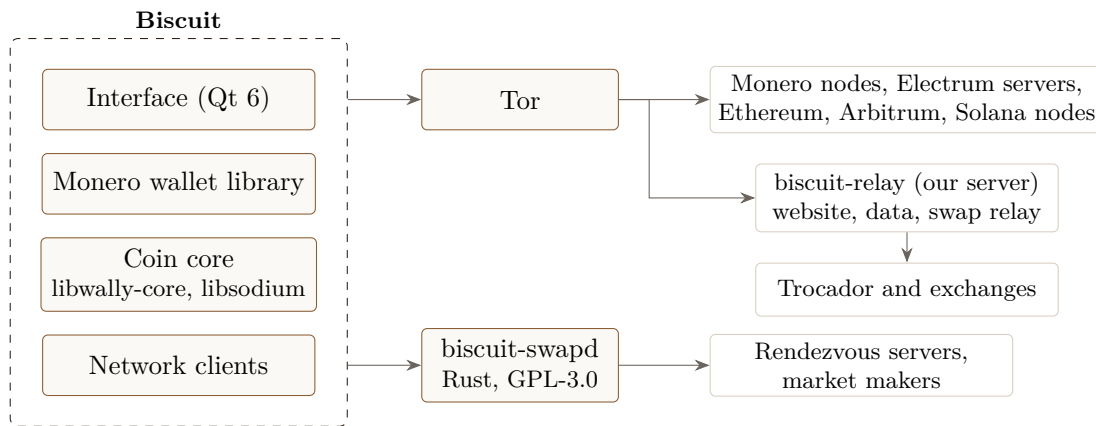


Figure 1: The parts of Biscuit. In Tor mode, every connection to the outside goes through Tor; without it, the same connections are made directly.

Part	Language	What it does
Biscuit (the app)	C++, Qt 6	Interface, settings, Tor control, update checks, exchange swaps
Monero wallet library	C++	Built from Monero's own source: keys, sync, transactions, hardware wallets
Coin core	C++	Keys, addresses and transactions for the other coins. No network, no interface, covered by unit tests
Tor	C	Shipped with Biscuit, started and stopped by it
biscuit-swapd	Rust	Atomic swaps, built from eigenwallet's code
biscuit-relay	Go	Our server: website and onion service, public data, swap relay

The coin core is kept apart from the network code on purpose: keys are derived and transactions are built and signed by code that cannot open a connection. The network clients around it speak the Electrum protocol over TLS (Bitcoin, Litecoin, Dogecoin) and JSON-RPC (Ethereum, Arbitrum, Solana). Ethereum transactions are EIP-1559 transactions, hashed with the Keccak-256 code Monero already ships. Solana signatures are Ed25519, from libsodium, which Monero already uses too. Reusing what is already in the program keeps the number of cryptographic libraries down.

3 Seeds and keys

Monero has its own seed: a 16-word Polyseed, which carries the wallet’s creation date, so a restore never needs a block height. Older 25-word and 14-word seeds can be restored. Before the seed is made, the user can mix dice rolls or coin flips into the system’s randomness.

The other coins share one 12-word BIP39 seed by default, so a single sheet of paper covers them. When a coin is added, it can be given its own seed instead, new or restored. Biscuit uses the standard derivation paths (table 1), so each seed opens the same coins in other wallets.

Coin	Path	Addresses
Bitcoin	m/84'/0'/0'	Native SegWit, <code>bc1q...</code> , a new one after each payment
Litecoin	m/84'/2'/0'	Native SegWit, <code>ltc1q...</code>
Dogecoin	m/44'/3'/0'	Legacy, D...
Ethereum and Arbitrum	m/44'/60'/0'/0/0	One <code>0x...</code> address for ETH, USDT and USDC
Solana	m/44'/501'/0'/0'	One base58 address

Table 1: Derivation paths of the coins besides Monero.

Ledger and Trezor devices can hold the Monero keys, which then never reach the computer. The other coins keep their seed in the encrypted wallet file.

4 Wallet files

A wallet is three files, all encrypted with the user’s password:

- `name.keys`, the Monero keys, and `name`, the Monero cache (transactions, labels, notes, the Polyseed, the swap history). These use Monero’s own format: ChaCha20, with a key derived from the password by CryptoNight. Feather and Monero’s own wallets open them.
- `name.btcltc`, the other coins: seeds, addresses, labels and contacts. It is sealed with XChaCha20-Poly1305, an authenticated cipher, so a modified file is refused rather than misread. The key comes from Argon2id at libsodium’s moderate setting: three passes over 256 MiB of memory, which makes each password guess slow and expensive.

The password never leaves the computer and is not part of any seed. There is nothing to reset: a lost password means restoring from the seeds.

5 Network

Table 2 lists every connection Biscuit makes, and what the other side learns without Tor.

Light wallets have a cost: an Electrum server, an Ethereum node or a Solana node knows which addresses it is asked about. Tor hides who asks, not what is asked. Each of them can be replaced by the user’s own server in the settings, and Biscuit then talks to that one only.

Electrum connections must use TLS. A certificate that no public authority signed is pinned on first use, and refused if it changes later. Unencrypted connections are allowed only to `.onion` addresses and the local network. Only real local addresses (`localhost`, `.local` names and private IP ranges) are reached without Tor, so that a node at home keeps working; a public name that merely starts like one, such as `10.example.org`, does not count.

To	For	Learns
A Monero node	Blocks, sending	The IP address. Not the addresses or the balance: the wallet scans every block locally
Electrum servers	Bitcoin, Litecoin, Dogecoin	The IP address and the addresses asked about
Ethereum and Arbitrum nodes, Blockscout	Balances, fees, sending; history	The IP address and the address
Solana nodes	Balance, history, sending	The IP address and the address
biscuitwallet.com	Prices, currency rates, the Journal, updates	Nothing kept: no access logs
Rendezvous servers, market makers	Atomic swaps	The IP address, unless the swap uses Tor
Our relay, Trocador, exchanges	Exchange swaps	See section 7

Table 2: Every connection Biscuit makes.

6 Tor

Biscuit ships Tor and manages it: it starts its own instance and stops it when it closes, even after a crash. A Tor already running on the computer can be used instead, and on Tails and Whonix the system's Tor always is.

In Tor mode, every connection goes through Tor: Monero nodes (to their `.onion` addresses by default), Electrum servers, Ethereum, Arbitrum and Solana nodes, Blockscout, our server, both kinds of swap. Our server is also an onion service, so prices, the Journal, updates and exchange swaps never leave the Tor network. A stricter mode allows onion services only.

A wrong proxy setting, or another program listening where Tor should be, would send traffic in the clear while the interface says Tor. So in Tor mode Biscuit asks our server, now and then, whether its request arrived through Tor. The answer is yes or no, nothing is logged, and a no is shown in the status bar.

7 Our server and exchange swaps

Our server is one Go program, `biscuit-relay`, hosted by 1984 Hosting in Iceland. Its source is public. It keeps no access logs, and drops Go's own connection errors, which contain IP addresses.

7.1 Public data

The server fetches coin prices from CoinGecko, currency rates from the European Central Bank (through Frankfurter) and Monero's crowdfunding proposals from `getmonero.org`, and serves the same copy to every user. Those sources never see who uses Biscuit, and every copy of Biscuit makes the same requests. Nothing about a wallet is sent.

7.2 Swaps through Trocador

Trocador gathers offers from many exchanges. Biscuit never talks to it directly: the request goes to our relay, which adds our partner key and passes it on, so the key is not in the app. The relay forwards five read and create calls and refuses the rest. Trocador receives the key and a fixed user agent, nothing from the user's connection.

Trocador requires one thing in return: when a swap is created, the relay writes the time, IP address, user agent, languages and swap ID. Each line is encrypted with age to a public key whose private half is kept offline, so the server cannot read what it wrote. Files are deleted after twelve months. If the line cannot be written, the swap is not created. Because a Tor exit cannot be traced, the relay refuses to create these swaps from Tor exits, using the Tor Project's exit lists; while those lists are missing or stale, it refuses all creations. Biscuit therefore turns Trocador off in Tor mode.

7.3 Our own exchanges

For Tor, the relay also calls a few exchanges itself: PegasusSwap, ETZ-Swap and Stereoswap, chosen because their terms say they never ask for identity and because they accept Tor. For these the relay records nothing, not the IP address, not the swap. The exchange receives the swap itself (coins, amounts, addresses) under the relay's name. In Tor mode, these are the only exchanges offered, reached through our onion service. Their status calls need the payout address, so a swap ID alone reveals nothing.

7.4 In the app

Offers are ranked by one thing: what the user receives. The share of the exchange fee we may get never changes the order. Rates are only asked when the user asks for offers, and a running swap is checked on a slowing schedule, from every three minutes to every fifteen. The swap history is kept in the encrypted wallet file. An exchange holds the coins for the time of a swap, so Biscuit links to a public page on each exchange: its terms, its kycnot.me rating and its open incidents, with sources, checked before each release.

7.5 Abuse limits

To stop abuse, the relay counts requests per IP address (IPv6 per /64) in memory, forgetting them at the end of each window: a minute for requests, up to a day for swap creations. Nothing is written to disk.

8 Atomic swaps

An atomic swap trades Bitcoin for Monero with a market maker, without anyone holding the coins in between. Biscuit runs them in `biscuit-swapd`, built from `eigenwallet`, which implements the protocol described by Gugger [1].

The Bitcoin is locked in an output that needs both parties' signatures; the maker locks the Monero to a key split between them. Biscuit hands the maker an adaptor signature: when the maker uses it to take the Bitcoin, the completed signature published on chain reveals the missing part of the Monero key, and Biscuit sweeps the Monero. If the maker never locks the Monero, the swap can be cancelled after 24 Bitcoin blocks and the Bitcoin refunded. If Biscuit stays closed for a further 144 blocks, about a day, the maker may take the Bitcoin. Biscuit must be opened during that window, and says so.

Makers are found through public rendezvous servers over `libp2p`. Without Tor, those servers and the makers see the user's IP address; in Tor mode, swaps always go through Tor. Atomic swaps don't use our server at all. They are in beta, and go from Bitcoin to Monero only.

9 Releases and updates

Releases are built with Guix, like Bitcoin Core, Monero and Feather: the same source gives the same files, byte for byte, so anyone can check a release against its source. Two parts are not covered, and we say so: `biscuit-swapd` is built outside Guix, and the Guix build stops

unless its SHA-256 matches the one recorded in the source; and the macOS zip gets an ad-hoc signature after the build, so it differs from what Guix produced. The Windows and Linux files are reproducible.

Each release has a list of SHA-256 checksums signed with our release key (RSA 4096, fingerprint D714 332E 6101 C1DA AC60 E644 202F 1FA3 98DE CEFA). The public key is built into Biscuit. Every six hours, Biscuit reads the list of latest versions, from our onion in Tor mode. When there is a new one, it checks the signed checksum list against the built-in key, refuses any hash algorithm but SHA-256, downloads the file and compares its checksum. Nothing is installed until the user clicks.

10 What Biscuit does not protect against

- **A compromised computer.** Malware that reads the screen, the keyboard or memory sees what the user sees. Biscuit cannot fix that; a hardware wallet helps for Monero.
- **What light wallet servers learn.** Without the user's own servers, Electrum servers and Ethereum and Solana nodes learn the addresses, and may link them together. Tor hides the IP address from them, not the addresses.
- **Public addresses.** Ethereum, Arbitrum and Solana addresses never change, and their history is public. Only Monero hides amounts and parties on chain.
- **Exchanges.** During an exchange swap, the exchange holds the coins and may freeze them, whatever its terms say. Atomic swaps avoid this, for Bitcoin to Monero.
- **Our server.** No one can check from outside that the server runs the published code. We publish it, and the app is built so that the server gets as little as possible: no account, no wallet data, and in Tor mode no IP address.
- **Lost seeds.** There is no recovery. The seeds are the wallet.

References

- [1] J. Gugger. *Bitcoin–Monero Cross-chain Atomic Swap*. IACR Cryptology ePrint Archive, Report 2020/1126. <https://eprint.iacr.org/2020/1126>
- [2] Biscuit source code. <https://github.com/BiscuitWallet/Biscuit>
- [3] biscuit-relay, the server. <https://github.com/BiscuitWallet/biscuit-relay>
- [4] biscuit-swapd, the atomic swap helper.
<https://github.com/BiscuitWallet/biscuit-swapd>
- [5] Polyseed. <https://github.com/tevador/polyseed>
- [6] age, a file encryption tool. <https://age-encryption.org>
- [7] Biscuit documentation and privacy policy. <https://biscuitwallet.com/docs/>,
<https://biscuitwallet.com/privacy/>