DOCUMENT	AUTHOR	REVISION
BitMonero Whitepaper	The BitMonero Community	v1 — 2026
SUBJECT	**BASIS**	**HANDLING**
Protocol & monetary policy	Source code · ANN · Discord	Read the code. Trust nothing.

BitMonero

Ticker: BMR

Bitcoin's issuance schedule. Monero's privacy. No premine.

21,000,000 × [redacted]

A fixed, public ceiling multiplied by a value that stays private. The scarcity is the part everyone can see and count; the amounts moving under it are the part no one can. That is the whole design in one line.

FAIR LAUNCH · NO PREMINE

github.com/BitMoneroNet/bitmonero · bitmonero.net

Launched 16 January 2026

ABSTRACT § 00

BitMonero runs Monero's privacy engine under Bitcoin's monetary rules. Supply is capped at 21,000,000 coins. The block reward starts at 10 BMR and is cut in half every 1,050,000 blocks by a discrete right shift, the same arithmetic Bitcoin uses. There is no tail emission. Transactions hide sender, receiver, and amount by default through ring signatures, stealth addresses, and RingCT, taken from the Monero v0.18 codebase. Proof of work is RandomX, which favors ordinary CPUs. The chain launched on 16 January 2026 from a genesis block seeded by a then-recent Bitcoin block, so its start time can be checked by anyone. There was no premine, no presale, and no developer allocation. This document sets out those mechanics and, wherever a claim can be checked, names the source that settles it.

1 · WHAT BITMONERO IS § 01

Two decisions define BitMonero, and they come from two different projects.

The monetary side is Bitcoin's: a fixed ceiling of 21,000,000 units, a reward that starts high and is halved on a fixed block schedule, and no perpetual issuance once the halvings run out. Scarcity is scheduled and public. Anyone can compute the exact supply at any block height.

The privacy side is Monero's. Sender, receiver, and amount are concealed by default, not as a setting a user has to remember to switch on. This is the property Bitcoin lacks: on Bitcoin, every address and every amount sits in plain view on the public ledger, permanently.

BitMonero is the intersection of the two. It forks the Monero v0.18 codebase, replaces the emission curve, and ships a new genesis block and network identifier. The privacy comes from Monero's cryptography, unchanged. The issuance is Bitcoin's, written into the reward function. It is not Monero with a smaller cap bolted on, and it is not Bitcoin with privacy added afterward. It is one codebase carrying the other's monetary policy.

▪ A note on the name

"Bitmonero" was the original name of Monero when it launched in 2014, before the rename to Monero. This project is unrelated to that history and to the Monero project as it exists today. It reuses the old name and a fork of the code. Read it as its own chain and check it on its own terms.

2 · MONETARY POLICY § 02

BitMonero's monetary policy lives in a handful of constants in the source. The values below are read from the code, not from any announcement.

```
MONEY_SUPPLY             = 21,000,000 × 10^12   (atomic units)
EMISSION_INITIAL_SUBSIDY  = 10 BMR
EMISSION_HALVING_INTERVAL = 1,050,000 blocks
FINAL_SUBSIDY            = 0                    (no tail emission)
COIN                     = 10^12               (12 decimal places)

reward function:
    base_reward = EMISSION_INITIAL_SUBSIDY >> halvings
    halvings    = height / EMISSION_HALVING_INTERVAL
```

The reward function is a right shift, which is Bitcoin's mechanism rather than Monero's. Monero pays a smooth exponential curve: each block releases a fixed fraction of the remaining supply, so the reward falls a little every block and never quite reaches zero, which is the tail emission. BitMonero instead holds the reward constant for 1,050,000 blocks and then halves it in a single step. The reward is 10 BMR through the first era, 5 through the second, 2.5 through the third, and so on down, until the right shift reaches zero and issuance stops.

▪ Two details a casual reading gets wrong

First, the supply does not close at exactly 21,000,000. Rewards are whole numbers of atomic units, and halving by a right shift throws away the remainder each time, exactly as it does on Bitcoin. Summed across every era, the emitted total lands just short of the ceiling, at **20,999,999.9999853 BMR**. The constant named MONEY_SUPPLY is a limit the emission approaches, not a figure it reaches. Bitcoin behaves the same way for the same reason and never mints its full 21 million.

Second, there is no tail emission. FINAL_SUBSIDY is zero. Once the last halving drives the reward to zero, no new coins are created and miners are paid only by transaction fees. This is a deliberate break from Monero, which keeps a small permanent reward to fund security forever. Whether a fee-only security budget is enough is an open question for every fixed-cap chain, Bitcoin included. BitMonero inherits that question rather than answering it.

Property	BitMonero	Bitcoin	Monero
Supply cap	21,000,000	21,000,000	None (tail emission)
Reward schedule	Halving, right shift	Halving, right shift	Smooth exponential
Halving interval	1,050,000 blocks	210,000 blocks	n/a
Tail emission	No	No	Yes (0.6 XMR/block)
Block target	120 s	600 s	120 s
Proof of work	RandomX (CPU)	SHA-256 (ASIC)	RandomX (CPU)
Amounts on chain	Hidden	Public	Hidden
Decimals	12	8	12

Bitcoin and Monero figures are given for comparison and reflect their published parameters. BitMonero values are from its own source.

3 · EMISSION SCHEDULE

§ 03

At a 120-second target (DIFFICULTY_TARGET_V2 = 120), an interval of 1,050,000 blocks is close to four years of blocks. The first halving therefore arrives roughly four years after launch, and the schedule tracks Bitcoin's four-year rhythm without matching it exactly, because the block time differs.

> **DISCREPANCY // ANN VS SOURCE**
>
> The Bitcointalk announcement states the interval as 1,051,200 blocks. The code says 1,050,000. The 1,051,200 figure reads like a back-of-the-envelope estimate: 365 days × 4 years × 720 blocks per day at 120-second blocks equals 1,051,200. The value in the source, 1,050,000, is the round number the developer actually committed. The network runs the code. Where the announcement and the source disagree, the source wins.

Emission is front-loaded, as it is on any halving schedule. About half of all coins that will ever exist are created in the first era, half of the remainder in the second, and so on. The long tail of small rewards continues for decades, but the bulk of issuance happens early. Late participants receive proportionally less from mining and more of their supply, if any, from the market.

4 · FAIR LAUNCH § 04

A fair launch means no one held coins before everyone else had a chance to. BitMonero backs the claim in two ways that can be checked rather than taken on faith.

- **No premine, no developer allocation**

The genesis block pays no reward to any address. Emission begins with ordinary mining after genesis. This is verifiable twice over: by reading the genesis definition in the source, and by inspecting the earliest blocks in any explorer and confirming that no large balance predates open mining.

- **The start time is anchored to Bitcoin**

The genesis nonce is a fixed constant in the source:

```
GENESIS_NONCE = 2751356999
```

It was derived from the hash of a Bitcoin block near the launch moment on 16 January 2026. Because that Bitcoin block did not exist until Bitcoin miners produced it, the BitMonero genesis could not have been prepared far in advance. This is the blockchain version of holding up a dated newspaper in a photograph: a specific recent block can only be referenced after it exists, so its presence proves the work came later. Anyone can take the referenced Bitcoin block, apply the documented derivation, and confirm the nonce for themselves.

> **VERIFY // GENESIS**
> The project's own three claims are: no presale, no ICO, no developer allocation. These are easy to assert and harder to keep honest. The Bitcoin anchoring is what converts "trust us, we launched fairly" into a check you can run yourself. Run it before you weigh anything else in this document.

5 · PRIVACY § 05

BitMonero's privacy is Monero's, taken from the v0.18 codebase with no changes to the cryptography. Three mechanisms do the work.

- **Stealth addresses**

Every payment goes to a fresh one-time address derived from the recipient's public keys. The recipient's published address never appears on chain, and two payments to the same person cannot be linked by their addresses.

- **Ring signatures**

Each input is signed as one member of a set of possible spenders, so an observer cannot tell which member actually spent. The true origin of a transaction is hidden among decoys drawn from the chain's history.

- **RingCT (Ring Confidential Transactions)**

Amounts are hidden behind cryptographic commitments, and range proofs certify that each hidden amount is valid without revealing it. The network can confirm that inputs equal outputs while no one learns the figures involved.

Together these conceal sender, receiver, and amount by default. A block explorer built for BitMonero will show that transactions exist and when they occurred, but not who sent them or how much moved. That is protocol behavior, not a fault in the explorer.

> **SCOPE // WHAT PRIVACY DOES NOT COVER**
>
> On-chain privacy is not anonymity everywhere. Network metadata, such as the IP a transaction is broadcast from, sits outside the protocol; routing a node over Tor or i2p addresses that separately. Exchange accounts, reused off-chain identities, and careless habits can deanonymize a user no matter what the chain hides. The cryptography protects the ledger, not the rest of a person's decisions.

6 · PROOF OF WORK § 06

BitMonero uses RandomX, the proof-of-work algorithm Monero adopted in 2019. RandomX is built to run well on general-purpose CPUs and badly on specialized hardware. It executes a program of random instructions against a few gigabytes of memory, which is cheap for a commodity processor and expensive to freeze into an ASIC.

The intent is distribution. When mining is efficient only on custom chips, hashpower concentrates with whoever can fund and fabricate them. When an ordinary laptop or server can contribute, mining stays closer to the people running the software. This does not make concentration impossible. Large CPU farms and pools still form. It lowers the barrier to entry; it does not erase the advantage of scale.

The block target is 120 seconds. Difficulty adjusts block to block to hold that target as hashpower rises and falls.

7 · NETWORK PARAMETERS § 07

The constants an operator needs are listed below, read from the source.

Parameter	Value
Default P2P port	48080
Default RPC port	48081
Default ZMQ port	48082
Mainnet address prefix	11456
Block target	120 seconds
Decimal places	12
Atomic unit	10^-12 BMR

Two operational notes belong here. The RPC port, 48081, exposes wallet and daemon control and must stay bound to localhost; it should never be reachable from the public internet. Separately, the seed nodes compiled into early builds may be offline, so a fresh node can need explicit peers before it will begin syncing. Neither is a property of the protocol. Both are the kind of detail a young network accumulates, and the kind an operator has to know before the first sync succeeds.

8 · READING THE CODE § 08

BitMonero asks to be verified rather than trusted, which is the correct posture toward any new chain. A short list is worth working through before committing time or money.

- **Read the emission constants yourself**

Everything in the monetary-policy section above is a handful of lines in the source. Confirm them against the repository, not against this document and not against the announcement.

- **Check the genesis**

Confirm the nonce derivation, and confirm the genesis block carries no premine.

- **Understand the license**

The code is BSD-licensed, inherited from Monero, not GPL. That governs how it may be reused and redistributed.

- **Know what you are joining**

This is a young fork maintained by a small group. Forking a large privacy codebase cleanly is hard, and rough edges exist. During community deployment of public infrastructure, one such edge surfaced in the wallet serialization code and is documented in Appendix A. It does not touch consensus or the coins, but it is the kind of finding that tells you how finished a project is. Weigh that honestly.

BitMonero is not affiliated with Bitcoin or with Monero. Any "BMR" token on another chain is unrelated to this one. Verify everything.

APPENDIX A · BUILDING A BLOCK EXPLORER § A

Anyone running public infrastructure for BitMonero will want a block explorer. The community deployment uses the well-known `onion-monero-blockchain-explorer`, compiled against the BitMonero source rather than upstream Monero, so that the network ID and address formats match. That build does not succeed unmodified. This appendix records why, and what to change, so the next operator does not lose a day to it.

The failure

Compilation runs to roughly 95 percent and stops while building `main.cpp`:

```
error: 'struct tools::wallet2::signed_tx_set' has no member named 'serialize'
```

The message names the explorer, but the missing piece is in BitMonero. `src/wallet/wallet2.h` defines the structures used for offline (cold) transaction signing but does not ship the `boost::serialization` functions for them, nor the matching `BOOST_CLASS_VERSION` declarations. Upstream Monero ships both. Any program that serializes those structures through boost, the explorer included, fails to compile against the fork until the functions are present.

What is missing

BitMonero's `wallet2.h` otherwise tracks Monero v0.18.4.0: the same wallet class version (31) and the same background-sync structures. Measured against that baseline, five serializers are absent, together with their version declarations: `unsigned_tx_set`, `signed_tx_set`, `tx_construction_data`, `pending_tx`, and `multisig_sig`. Boost serialization is recursive, so all five are required to read one `signed_tx_set`, which holds a `pending_tx`, which in turn holds a `tx_construction_data` and a `multisig_sig`.

Only the boost layer is gone. The structures themselves remain, with their internal (epee) serialization. The supporting machinery also remains: `cryptonote_boost_serialization.h` in the fork is identical to upstream apart from a copyright year, and the unordered-container serialization header is present. The gap is exactly these five functions.

The fix

Restore the five serializers. They are header-only inline templates. They add no fields and change no memory layout, so they leave the prebuilt libraries and the running node untouched; only the consuming code is recompiled. One detail differs from a plain copy of upstream: BitMonero keeps `signed_tx_set::tx_key_images` as a plain `std::unordered_map`, where later Monero uses a wrapper type, so the serializer reads `x.tx_key_images` directly. The full patch is published with this document.

> **SCOPE // IMPACT**
> This is a tooling gap, not a protocol one. It has no bearing on consensus, emission, proof of work, or ordinary transactions; a node built from the same source syncs and validates normally. The affected path is offline transaction signing. It is recorded here because a

> public explorer is part of what makes a chain legible, and because the correction belongs in the open, where the next operator can find it.

REFERENCES

§ R

SOURCE	github.com/BitMoneroNet/bitmonero — the daemon and wallet code; the authority for every parameter in this document.
SITE	bitmonero.net — project landing page.
ANNOUNCEMENT	Bitcointalk ANN, topic 5571370 — launch thread; superseded by the source where the two disagree.
COMMUNITY	Discord — discussion and coordination.
EXPLORER	github.com/moneroexamples/onion-monero-blockchain-explorer — compiled against the fork; see Appendix A.
UPSTREAM	getmonero.org — Monero, the source of the inherited privacy stack and RandomX.

This document is a community reconstruction of BitMonero's design from its source, its announcement, and community records. It is not an official publication of any entity. Read the code. Trust nothing.