BoreLine Pay

Non-Custodial Bitcoin Payment Infrastructure

VERSION 1.0 · JUNE 2026 · GERMANY, EUROPE

0 Abstract

BoreLine Pay is non-custodial Bitcoin payment infrastructure. A merchant accepts Bitcoin payments that settle directly to an account only the merchant holds the keys to, while BoreLine never holds, touches, or is able to move the funds at any point.

Conventional payment processors stand between a merchant and their revenue. They hold balances, take a percentage of every transaction, and retain the power to freeze, reverse, or withhold settlement. BoreLine removes that intermediary by design. A merchant registers the extended public key, known as a ZPUB, of the native SegWit account on their own hardware signing device. From that key alone, BoreLine derives a unique Bitcoin address for each invoice, observes the public blockchain for payment, and notifies the merchant when funds confirm. The value never passes through BoreLine. It cannot, as a structural property of the system rather than a matter of policy.

This document sets out the model, the technical architecture, the security and trust assumptions, and the explicit limitations of the system, so that a technically literate reader can evaluate the claims and verify them independently.

1 The problem with custodial settlement

When a merchant accepts Bitcoin through a conventional processor, the Bitcoin does not arrive at the merchant. It arrives at the processor, who records a balance and settles later, often subject to fees, delays, and conditions. This reintroduces the very problems Bitcoin was created to remove:

- **Custody risk.** A third party holds the funds. If that party is breached, becomes insolvent, or chooses to freeze an account, the merchant's revenue is exposed.
- **Permissioned settlement.** The processor can refuse service, demand documentation, or reverse a payout. The merchant receives funds only at the processor's discretion.
- **Per-transaction fees.** A percentage of each sale is taken, scaling with revenue rather than with any real cost of providing the service.
- **Forfeited sovereignty.** The principle that holding the keys is the only true ownership applies to merchants as much as to individuals. A processor-held balance is a claim, not Bitcoin.

For a merchant who adopted Bitcoin precisely for its self-sovereign properties, routing revenue through a custodian defeats the purpose.

2 The BoreLine model

BoreLine is built on a single principle: the operator must never be able to reach merchant funds. This is not a promise to behave well. It is a structural consequence of how the system is constructed.

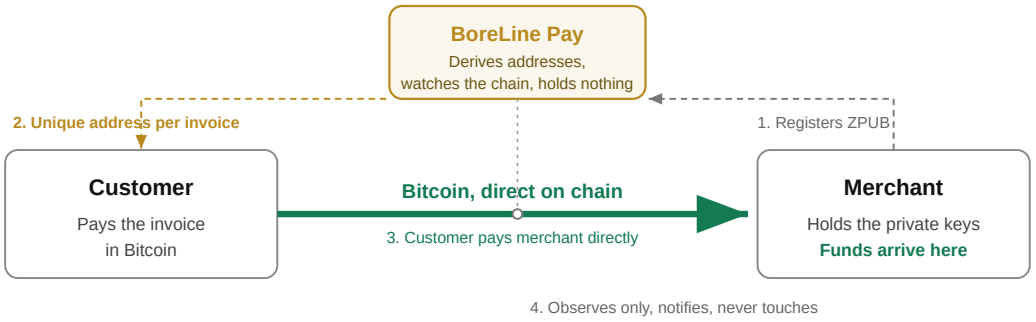
A merchant registers only a **ZPUB**, the extended public key of the native SegWit account on their hardware signing device. A ZPUB is a public key. It can derive receiving addresses and observe balances, but it carries no spending authority of any kind. The private keys that authorise spending never leave the merchant's signing device and are never transmitted to or seen by BoreLine.

When a customer pays, the transaction is broadcast to the Bitcoin network and arrives at an address derived from the merchant's own key. BoreLine learns that this happened by reading the public blockchain. At no point does value flow through, or rest in, anything BoreLine controls. The merchant receives every satoshi the customer sends, directly, on chain.

> ***The core guarantee.*** *BoreLine derives addresses and observes the chain for payment. It holds no private keys, takes no custody, and has no mechanism to move, freeze, or reverse a single satoshi. Merchant funds are always, and only, under the merchant's own keys.*

How a payment flows

The diagram below shows the path of a single payment. The funds move directly from the customer to the merchant's own account on the Bitcoin network. BoreLine sits to the side, deriving addresses from the public key and observing the chain, never on the path the money takes.



The money line never passes through BoreLine. BoreLine only reads the public chain.

3 Technical architecture

3.1 Address derivation (BIP84)

BoreLine follows the BIP84 standard for native SegWit pay-to-witness-public-key-hash addresses, the modern bc1q format. From the registered ZPUB at derivation path m/84'/0'/0', BoreLine derives external-chain receiving addresses at successive indexes, m/0/0, m/0/1, m/0/2, and onward. Each invoice is assigned its own freshly derived address, so payments never collide and each is tracked independently.

3.2 A unique address for every invoice

Address reuse harms privacy and complicates accounting. BoreLine derives a new address for each invoice and enforces non-reuse at the database level. There is no practical ceiling. Derivation behaves identically at index five or index five million, so a heavily used account is never a constraint.

3.3 On-chain monitoring

BoreLine observes each invoice address for incoming transactions by querying public blockchain data. A payment is recognised only when a transaction of sufficient value arrives **after** the invoice was created. A historical balance on a previously used address can never be mistaken for a new payment. This is an important correctness property that prevents a stale transaction from falsely marking an invoice as paid.

3.4 Confirmation and reorganisation handling

An invoice is marked confirmed once the payment receives its first on-chain confirmation. BoreLine continues to observe for several further blocks to detect a chain reorganisation. If a reorganisation removes the confirming block, the merchant is alerted and any access granted on the strength of that payment can be revoked. This is a safeguard that simpler tools omit.

4 Security architecture

4.1 What is stored, and what is never seen

✓ STORED, AND HARMLESS	✕ NEVER STORED, NEVER SEEN
✓ The registered ZPUB, a public key	✕ The recovery phrase
✓ A SHA-256 fingerprint of the ZPUB for tamper detection	✕ Private keys
✓ Invoice records and derived addresses	✕ Any spending authority
✓ API keys, stored only as SHA-256 hashes	✕ Customer card or bank details, of which there are none

Everything on the left is public or one-way hashed. None of it can move a single satoshi.

The ZPUB is retained so that addresses can be derived server side. It is a public key. Possessing it reveals a merchant's addresses and balances, but confers **no ability to spend.** An exposed ZPUB is therefore a privacy consideration, not a theft risk, which is why BoreLine recommends a dedicated business account and regular sweeping of received funds into cold storage.

The fingerprint is enforced, not merely available. Before deriving any payment address, the server recomputes the SHA-256 of the stored ZPUB and compares it to the stored fingerprint. If the two ever diverge, the telltale sign of the ZPUB having been altered outside the signed change flow, for example by direct tampering with the database, the server refuses to derive, locks the account, and raises an alert. No payment can be routed to a substituted key, because a key that fails this check is never used. The merchant can verify the same fingerprint independently at any time.

4.2 Authentication without passwords

There are no passwords to be leaked. A merchant authenticates by signing a challenge message with their key, cryptographically proving control of the registered account. API keys are stored only as SHA-256 hashes and are shown in full only once, at the moment of creation.

Because the key itself is never retained, it can never be shown again or recovered. This is deliberate. A database that only holds hashes cannot leak usable keys. If a merchant loses a key, the answer is not recovery but replacement: they generate a fresh key, and the lost one stops working at once. Every key action, rotating all keys, revoking a single key, or replacing one key in place, requires a fresh signature from the registered hardware device. A merchant on a plan with several active keys can replace a single lost key without disturbing the others, so one compromised integration never forces the rest to be rebuilt.

4.3 Protected key changes

Changing the registered ZPUB is a sensitive operation and is deliberately resistant to abuse. It requires two signatures, one from the current hardware device authorising the change and one from the incoming device proving control of the new ZPUB, followed by a security hold during which the previous ZPUB continues to receive. An attacker who compromised a merchant's session alone could not silently redirect funds, because the merchant's physical signing device is still required.

4.4 Operational hardening

The server employs atomic writes, an append-only audit log, rate limiting, login lockouts, threading locks for safe concurrency, and standard HTTP security headers. Administrative access is gated and can be restricted to an explicit IP allowlist.

4.5 Scoped read-only access

A merchant often wants to glance at payments from a phone without carrying out any sensitive action. For this, BoreLine issues a separate class of access token, created only from an already authenticated session, that grants read-only visibility and nothing more. A paired device can see invoices, balances, payment history, and the ZPUB fingerprint, but it can never change settings, rotate or reveal an API key, alter the registered ZPUB, or move funds. Those actions remain gated behind a hardware signature.

The distinction is enforced on the server, not merely hidden in the interface. A read-only token is presented in its own request header and can reach only a fixed set of read endpoints; it is structurally incapable of authenticating any operation that changes state. Each pairing carries an expiry and can be revoked at any time from the desktop dashboard, and rotating the registered ZPUB revokes every paired device automatically. A leaked read-only token therefore exposes only the ability to look, never to touch.

5 The trust model: do not trust, verify

BoreLine does not ask to be trusted. The system is designed so that every claim in this document can be confirmed by the merchant, independently, before any real money is involved.

Each account includes a verification view showing the derivation path, the SHA-256 fingerprint of the registered ZPUB, and the derived addresses. The merchant compares these against the same key in their own software, whether Trezor Suite, Ledger Live, Sparrow, or any BIP84-compatible tool. If the addresses match, the merchant has proven, without trusting BoreLine, that payments will arrive at their account and nowhere else.

> ***Verify with a free trial, with nothing at stake.*** *Register a fresh ZPUB from a newly created account that holds no funds. Check the derived addresses against your own software and approve them. Create a small invoice, for example five dollars. Pay it yourself and observe*

> *exactly where the funds arrive, in your own account. Repeat as many times as you wish, waiting for each payment to confirm. The flow does not change, because it is fixed in code. Only once you have seen it work with your own coins do you register the key for your real business account.*

6 Payment lifecycle

1. **Invoice created.** A unique address is derived from the merchant's ZPUB and assigned to the invoice, together with an amount and an expiry window.
2. **Customer pays.** The customer sends Bitcoin to that address, and the transaction is broadcast to the network.
3. **Detection.** BoreLine observes the incoming transaction on chain, requiring the full amount or more and a timestamp later than the invoice creation time.
4. **Confirmation.** On the first confirmation the invoice is marked paid and the merchant's site is notified through its webhook. Observation continues to guard against a reorganisation.
5. **Late payments.** If a payment arrives after expiry, for instance during network congestion, BoreLine keeps observing the address for a grace period and alerts the merchant if funds arrive, so that nothing is silently lost.

Integration is flexible. A merchant can share a no-code hosted payment link, one for each product, or call a single API endpoint for a custom checkout with automatic fulfilment by way of a signed webhook.

7 Pricing philosophy

BoreLine charges a flat monthly subscription, paid in Bitcoin. It takes **no percentage of any transaction.** Whether a merchant settles one invoice or ten thousand in a month, the fee is identical, because the cost of deriving an address and observing the chain does not scale with the value being moved.

This aligns incentives honestly. A percentage-based processor earns more as the merchant grows, extracting rent from that growth. A flat fee charges for the service actually rendered. Every satoshi a customer pays reaches the merchant, and the subscription is the merchant's only cost, known in advance.

A new merchant can begin with a seven-day free trial that carries the full Starter feature set with no payment, then select any plan from the dashboard when ready.

8 Limitations and honest disclosures

A credible whitepaper states what a system is *not*, as plainly as what it is.

- **BoreLine is infrastructure, not custody and not insurance.** Because it never holds funds, it cannot recover, refund, or reverse a payment. Bitcoin transactions are final.
- **Key security is the merchant's responsibility.** BoreLine never sees the recovery phrase or the private keys and therefore can never restore them. A lost recovery phrase means lost funds, exactly as in any self-custody arrangement.
- **Correct registration matters.** Funds derived from a key the merchant does not in fact control cannot be recovered by BoreLine. The verification step exists precisely to rule this out before going live.
- **Blockchain conditions apply.** Confirmation times depend on the state of the network, and detection relies on public blockchain data sources. BoreLine cannot guarantee a specific confirmation time.
- **Not legal or tax advice.** Merchants remain responsible for their own compliance, licensing, consumer-protection, and tax obligations in their jurisdiction.
- **Provided as is.** BoreLine targets high reliability but provides the service without warranty, as set out in its Terms of Service.

9 Roadmap

BoreLine's direction follows its founding constraint: strengthen self-sovereignty without ever introducing custody. Planned and exploratory directions include the following.

Area	Direction
Nostr integration	Optional, opt-in payment notifications and a decentralised coordination channel, reducing reliance on any single messaging platform.
Encryption at rest	Encrypting the stored ZPUB so that even the public key is protected at rest, beyond the existing tamper-detection fingerprint.
Hardware status display	A small local-network device that shows incoming payment status at a glance, for merchants who prefer a physical indicator.
Self-host friendliness	Continued reduction of external dependencies so the most sovereignty-minded merchants can run their own instance.
Broader device guidance	Expanded and verified setup paths for additional BIP84-compatible signing devices and software.

Roadmap items describe intent rather than commitment and may change. None of them introduces custody of merchant funds. That constraint is permanent.

10 Conclusion

Bitcoin gave individuals the ability to hold money that no other party can touch. BoreLine extends that same property to the act of getting paid. By deriving addresses from a merchant's own public key and never holding funds, it offers the convenience of a payment processor without the custody, the percentage cuts, or the permission.

The design is deliberately verifiable. A merchant need not take any of this on faith. Every claim can be checked against the merchant's own key, with the merchant's own coins, before the system is trusted with a single satoshi. That is the standard self-custody Bitcoiners apply to everything else, and it is the standard BoreLine was built to meet.

> ***Accept Bitcoin. Own every sat.*** *Not a slogan, but a structural fact of how BoreLine operates.*

BoreLine Pay · Germany, Europe · boreline.app · Contact: Telegram @orion_veyr or borelineapp@proton.me. This document is informational and does not constitute legal, financial, or tax advice. Bitcoin payments are irreversible. Review the Terms of Service and Privacy Policy before use. Version 1.0, June 2026.