Bianco

Engineering Privacy Into Web3 with Bianco

WHITEPAPER

Abstract

Bianco is a cryptographic privacy infrastructure layer for EVM-compatible blockchains. It provides address-level unlinkability through a dual-key stealth address, enabling any wallet, application, or protocol to receive and send value without creating traceable onchain identity links between counterparties.

The core mechanism - an Elliptic Curve Diffle-Hellman(ECDH) stealth address derivation - is mathematically well established and deployed in production environments today. What Bianco contributes is a self-sovereign, non-custodial, EVM-native implementation of this primitive, with clean seperation between payment detection (viewing keys) and payment executoon (spending keys), and no dependence on user behaviour to pressure privacy guarantees.

This document describes the problem Bianco solves, its cryptographic construction, privacy guarantees, securtiy analysis, and development roadmap.

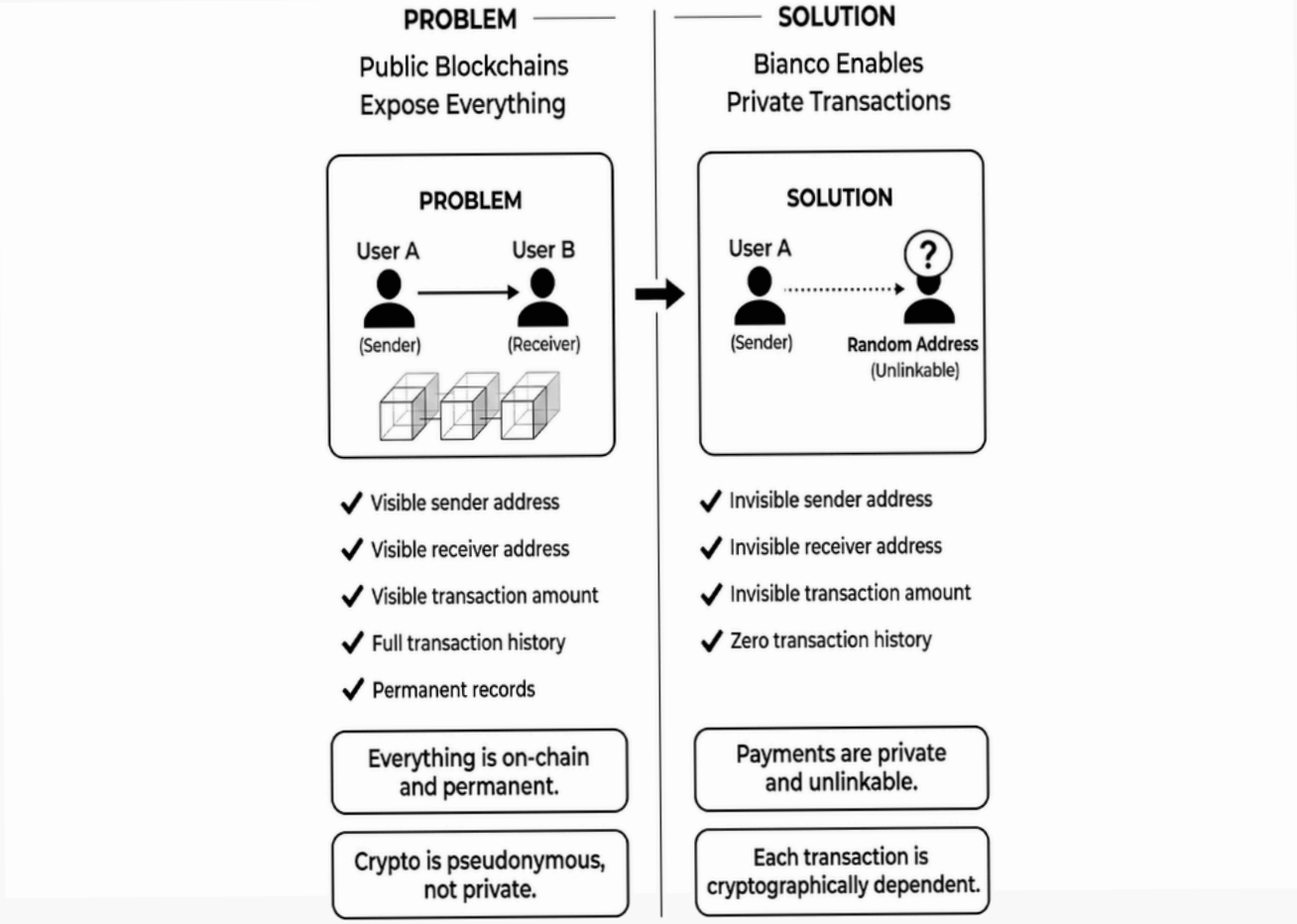
1. The Problem: Identity Leakage on Public Ledgers

Public blockchains achieve consensus on transaction state through full transparency — every transfer, balance, and interaction is permanently recorded and globally visible. This is a feature from an integrity standpoint, but it creates a structural privacy problem for users: every on-chain action links wallet addresses in ways that accumulate into identity profiles.

1.1 What the Chain Reveals

On any standard EVM chain, a single public address exposes:

- Complete inbound and outbound transaction history
- Current and historical token balances
- Counterparty relationships - who pays whom, how often
- interaction patterns with protocols - lending, trading and staking



This is not a theoretical concern. Blockchain analytics firms build full economic graphs of wallet behaviour from public data alone. A single address disclosure, whether through a KYC process, a public ENS name, or a leaked payment receipt, can unravel years of transaction history.

1.2. Why Pseudonymity Is Not Privacy

Addresses are pseudonymous, not anonymous. They are consistent identifiers across time. Once linked to an identity through any vector — exchange withdrawal, public disclosure, on-chain social graph — the pseudonymous protection is permanently eliminated for all historical and future activity.

The solution is not obfuscation or mixing — it is preventing the linkage from forming in the first place. This is what stealth address infrastructure achieves.

2. The Bianco Architecture

Bianco implements a stealth address protocol on EVM chains. It allows a sender to generate a unique, one-time receiving address for each transfer — derived from the recipient's public keys but unlinkable to any known address — without requiring the recipient to generate or share a new address in advance.

2.1 Dual-Key Design

Bianco users generate two independent key pairs:

Key Pair	Components	Function	Custody Model
Spending Key	(sk_spend, PK_spend)	Authorizes fund movement from stealth addresses	Must remain local and secured - analogous to a wallet private key
Viewing key	(sk_view, PK_view)	Scans the blockchain to detect incoming payments	Can be selective disclosed to auditors or compliance authorities without granting spending rights

This separation is architecturally significant. A user can grant a third party full visibility into their incoming transaction history — for tax reporting, compliance, or audit purposes — while maintaining exclusive control over their funds. The viewing key cannot move assets.

2.2 The Stealth Address Protocol - Step by Step

Bianco uses Elliptic Curve Cryptography on the secp256k1 curve — the same curve underpinning Ethereum key pairs — and an ECDH-based key derivation. Below is the full cryptographic flow.

Step 1 - Registration

A user connects their wallet and registers their stealth meta-address on-chain: a pair of public keys (PK_spend, PK_view) stored in Bianco's Stealth Key Registry smart contract. This is a one-time action and does not expose any private key material.

Step 2 - Ephemeral Randomness Generation (Sender Side)

When initiating a transfer, the sender's device generates a cryptographically random ephemeral scalar r. This value is generated fresh for every single transaction, ensuring that no two transfers produce the same derived address.

Step 3 - Shared Secret Derivation (ECDH)

The sender computes a shared secret S using the recipient's public viewing key and the ephemeral scalar:

$$S = r \times Pk\_view$$

Due to the properties of elliptic curve multiplication, the recipient can later reconstruct S independently using their private viewing key sk_view and the published ephemeral public key R = r × G:

$$S = sk\_view \times R$$

Both computations yield the same point on the curve. This is the Diffie-Hellman key exchange property applied to ECC.

Step 4 - Stealth Address Derivation

The sender uses S to 'blind' the recipient's public spending key, producing the one-time stealth address:

$$P\_stealth = H(S) \times G + PK\_spend$$

Where G is the secp256k1 generator point and H is a cryptographic hash function. The result P_stealth is a valid Ethereum address — standard-looking, entirely unique, and mathematically controlled only by the holder of sk_spend.

Step 5 - On-Chain Transaction

The sender broadcasts a transaction to the Bianco smart contract, sending funds to P_stealth and including the ephemeral public key R = r × G as transaction metadata. On-chain, the transaction appears as a payment to a never-before-seen address. No data reveals the recipient's identity..

Step 6 - Payment Detection (Recipient Side)

The recipient's local client (or a private scanning service) iterates over Bianco transactions. For each transaction with ephemeral key R, it:

- Computes the candidate shared secret: S = sk_view × R
- Derives the candidate stealth address: P_candidate = H(S) × G + PK_spend
- Checks whether P_candidate matches the on-chain destination

A match confirms the payment belongs to this user. No third party can perform this check without sk_view.

Step 7 - Spending Key Derivation

To spend the received funds, the recipient derives the stealth private key:

sk_stealth = H(S) + sk_spend

This key controls the stealth address. It is computed on-demand, locally, and never persisted. The user signs the withdrawal transaction using sk_stealth, moving funds to any destination without creating any on-chain link to their registered identity or other transactions.

3. Privacy Guarantees

3.1 Address Unlinkability

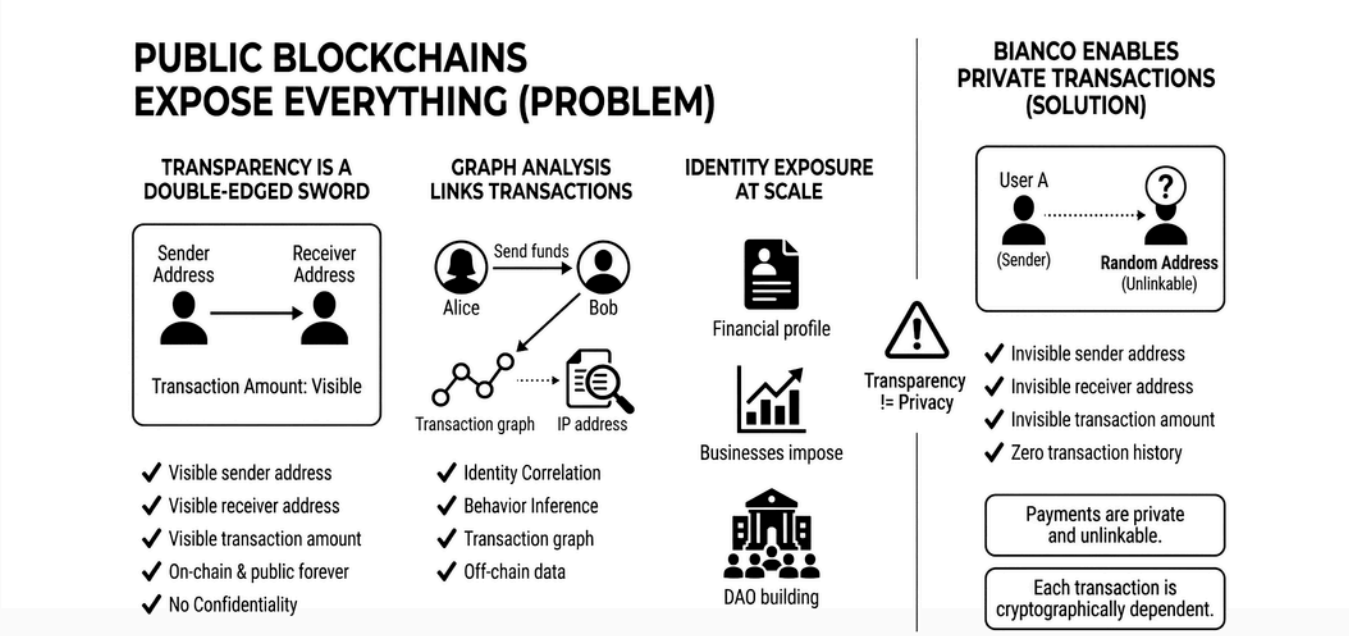
Because each transaction derives a unique stealth address from a fresh ephemeral scalar r, no two payment addresses generated for the same recipient are mathematically related — even to each other. One hundred payments to the same user produce one hundred unrelated addresses on-chain. An observer cannot determine they share a common owner.

3.2 Sender-Recipient Unlinkability

The on-chain record of a Bianco transaction contains only: the stealth address (random-looking), the ephemeral public key R (random-looking), and the transaction amount. There is no field that links the sender's public address to the recipient's registered identity. The Announcement event functions purely as an encrypted notification and is opaque to all parties without the recipient's sk_view.

3.3 Transaction Graph Resistance

Blockchain analytics techniques — heuristic clustering, co-spend analysis, graph traversal — rely on consistent address reuse or on-chain linkage between inputs and outputs. Bianco severs these links at the protocol level:



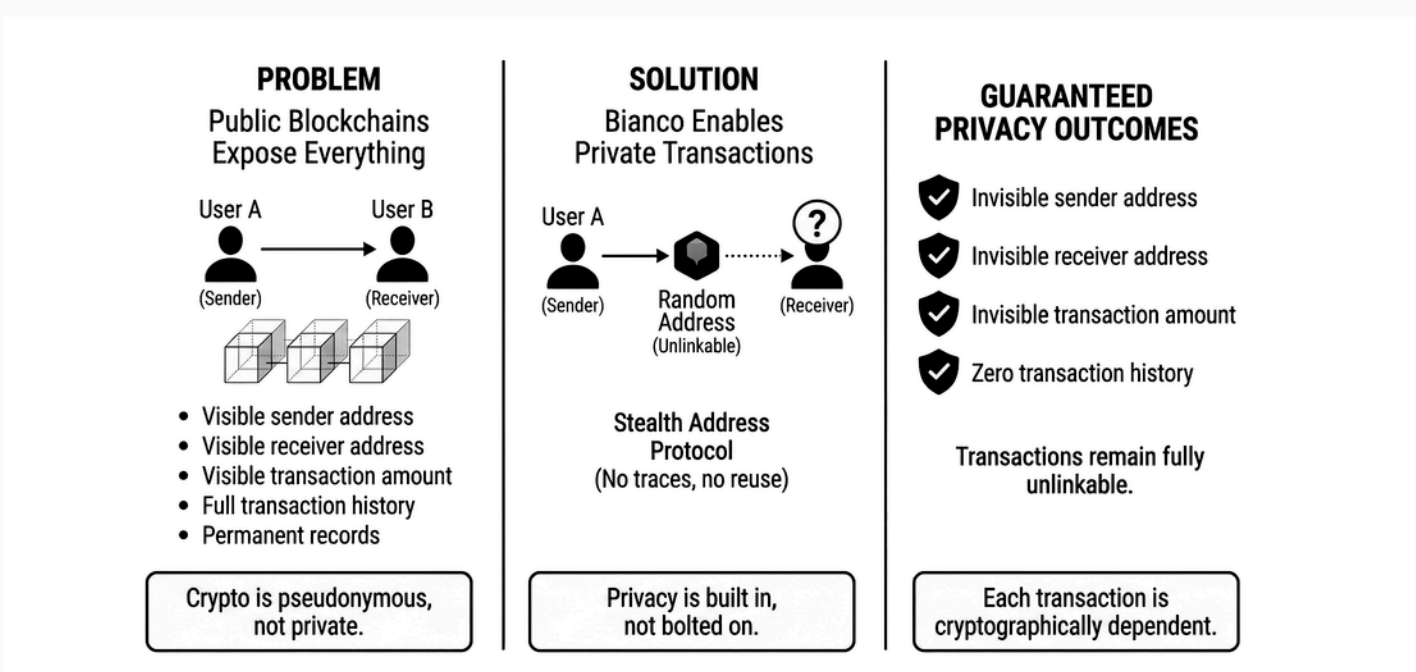
3.4 Non-Custodial by Construction

The Bianco backend is an indexer and interface layer. It has no access to any private key material. Stealth private keys are derived locally, on-demand, and never transmitted. The smart contracts are non-custodial and permissionless — no party can freeze, redirect, or access user funds.

Analysis Technique	Standard Wallet	Bianco
Address reuse clustering	Effective — same address appears repeatedly	Ineffective — every transaction uses a new address
Co-spend heuristic	Effective — linked inputs imply common ownership	Ineffective — stealth addresses are isolated
Transaction graph traversal	Effective — A → B → C chain is visible	Ineffective — A → ? → ? chain is broken
Balance Inference	Effective — public address balance is known	Ineffective — balance distributed across unlinked addresses

4. Security Analysis

Attack Vector	Description	Bianco Defence
Address Linkage	Attempt to connect stealth addresses to known identity	Mathematically infeasible without sk_view. Addresses are statistically indistinguishable from random keys.
Man-in-the-Middle	Attacker substitutes recipient keys during lookup	Keys are fetched from an on-chain registry. Alice encrypts to the key registered on-chain — MITM requires control of the registry contract.
Collector Pattern	Recipient withdraws from multiple stealth addresses to one known address, enabling heuristic linking	Bianco's protocol does not require withdrawal to a known address. Users withdraw to any fresh destination. No protocol-level enforcement of linkage.



Address Reuse	Recipient reuses a stealth address, breaking unlinkability	Each stealth address is protocol-generated and single-use. The system never re-derives the same address for two transactions.
Quantum Computing	ECDLP broken, private keys recoverable from public keys	This affects all ECC-based Ethereum infrastructure equally. Bianco's security is equivalent to Ethereum wallet security.
Backend Compromise	Bianco server is breached, exposing user data	No private keys are held server-side. A backend compromise exposes scanning history only — not the ability to move funds.

5. Scope and Design Boundaries

Bianco is not a mixer, tumbler, or fund-pooling protocol. Funds are never combined with other users' assets. There is no shared pool, no counterparty, and no trust assumption between users. The privacy guarantee is cryptographic, not statistical.

Bianco does not obscure transaction amounts on-chain. The value transferred to a stealth address is visible in the transaction. What is protected is the identity of the recipient — their known address is never the destination of any transaction.

The viewing key architecture explicitly supports compliance use cases. A user can disclose their viewing key to any authorised party — tax authority, regulator, institutional counterparty — to provide a complete, verifiable record of inbound transactions without granting spending access.

6. How Bianco Works

Bianco's workflow is designed to make strong cryptographic privacy accessible without requiring users to understand the underlying mathematics. The three core flows — registration, sending, and receiving — are described below.

6.1 Registration

Registration is a one-time action. The user connects their Ethereum wallet and Bianco generates two key pairs locally on their device: a spending key pair and a viewing key pair. The corresponding public keys are registered on-chain in the Stealth Key Registry smart contract, making the user discoverable as a Bianco recipient. No private key material ever leaves the device.

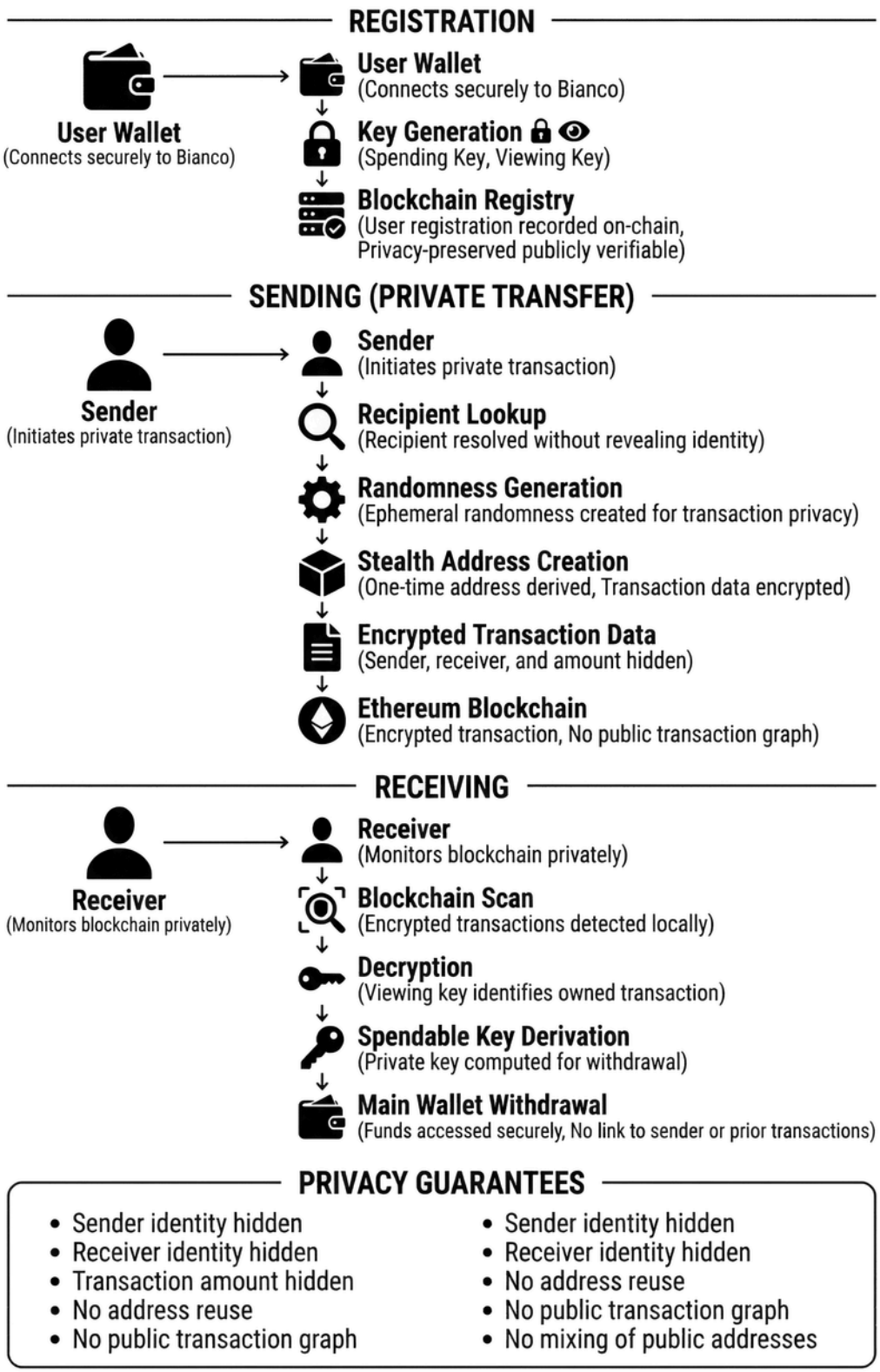
6.2 Sending a Private Transfer

To send funds privately, the sender enters the recipient's public Ethereum address or ENS name. Bianco resolves their stealth meta-address from the on-chain registry and performs the following automatically:

Step	What Happens	What the Sender Does
Recipient Lookup	Bianco fetches the recipient's public spending and viewing keys from the registry	Enters recipient address — no extra input needed
Ephemeral Key Generation	A random scalar r is generated fresh for this transaction	Automatic - happens client-side
Stealth Address Derivation	ECDH shared secret computed; one-time stealth address derived from recipient's keys	Automatic
Encryption	The ephemeral public key R is embedded in transaction data alongside the encrypted hint	Automatic
Transaction Broadcast	Funds sent to the stealth address on-chain; appears as payment to a random, unseen address	Confirms wallet transaction — same as any other send

BIANCO PRIVACY PROTOCOL

Advanced Privacy & Security for Decentralizeed Finance



6.3 Receiving a Private Transfer

The recipient does not need to take any action to receive funds. Bianco's scanner continuously monitors the blockchain for Announcement events from the smart contract. For each event, the client attempts decryption using the user's viewing key. A successful decryption reveals:

- The stealth address where funds are held
- The amount received
- The transaction hash and timestamp
- The sender’s address (for reference - not linked on chain)

This scanning process is private. No one observing the blockchain can determine which users are scanning or which transactions they have identified as theirs.

6.4 Accessing Received Funds

What a user wants to move received fund, Bianco derives the stealth private key locally:

sk_stealth = H(S) + sk_spend

This key is computed on-demand from the shared secret and the user's spending key. It is never stored by Bianco's backend or transmitted over the network. The user signs a withdrawal transaction with sk_stealth, moving funds to any destination address — with no on-chain link to their registered identity or any prior transaction.

7. User Experience

Bianco is designed so that the complexity of the cryptographic operations is fully abstracted. From a user perspective, the workflow mirrors a standard wallet send/receive flow.

7.1 Simple Workflow

Step	Action	Notes
1. Connect Wallet	Connect via MetaMask, WalletConnect, or any EVM wallet	Standard wallet connection — no new seed phrase or account
2. Register Keys	One-time stealth key registration (automatic on first use)	Keys generated locally; only public keys go on-chain
3. Send Privately	Enter recipient address and amount, confirm transaction	Stealth address derivation is fully automatic
4. Receive Automatically	Incoming payments detected by background scanner	No action required from recipient
5.Access Funds	Claim to any destination address with one confirmation	Stealth key derived locally; no custody handover

7.2 Key Properties

- No new seed phrase (works with any existing Ethereum wallet)
- Transactions confirm at Standard Ethereum block speed (~12 seconds)
- Cross-chain ready — the same architecture deploys on any EVM-compatible network
- No KYC required — the protocol is permissionless; access is determined by wallet ownership alone
- Gas-optimised — smart contract interactions are designed for minimal on-chain footprint

8. Use Cases

Bianco's privacy guarantees are relevant across a wide range of users and contexts. The common thread is a legitimate need to conduct on-chain activity without permanent, public exposure of identity and transaction history.

8.1 Individual Financial Confidentiality

- Prevent profiling of spending habits, income sources, and balances from a single address disclosure
- Maintain separation between different financial contexts — personal, professional, investment — without managing multiple wallets
- Protect counterparty information in peer-to-peer transactions

8.2 Business and Institutional Transactions

- Conceal commercially sensitive payment flows — vendor relationships, payroll, procurement — from competitors and market observers
- Enable confidential treasury operations without broadcasting fund movements to the market
- Conduct institutional transfers with the privacy expectation of traditional finance

8.3 Compliance-Compatible Privacy

- The viewing key model allows full transaction auditability on request — selective disclosure to regulators, auditors, or counterparties without granting spending access
- Compatible with GDPR data minimisation principles — on-chain data does not expose personal identifiers
- Provides a legally distinguishable privacy mechanism from mixers or obfuscation tools — no fund pooling, no counterparty trust, cryptographically verifiable audit trail

8.3 Compliance-Compatible Privacy

- Any EVM application can integrate Bianco's stealth address primitive via the SDK to offer private receive flows to their users
- DeFi protocols can use Bianco to offer confidential position entry and exit
- Wallet providers can embed Bianco as a native feature, making privacy opt-in at the application layer

9. Competitive Positioning

Comparison	Approach	Bianco Distinction
Standard EVM Wallets	Single reusable address; fully public transaction history	Bianco provides per-transaction address unlinkability with zero UX overhead for the recipient
Privacy Coins (Monero, Zcash)	Native privacy on a separate blockchain with mandatory or opt-in shielding	Bianco operates natively on Ethereum and all EVM chains — full access to existing DeFi, NFT, and token ecosystems. No separate chain required.
Mixers / Tumblers	Pool-based obfuscation; statistical unlinkability through fund mixing	Bianco involves no fund pooling, no counterparty, and no trust assumption. Privacy is cryptographic and deterministic, not probabilistic. Legally distinguishable from mixing.
Other Stealth Address Implementations	Stealth address protocols with limited UX enforcement, creating deanonymisation risk through user behaviour patterns	Bianco enforces single-use address discipline at the protocol level and derives stealth keys on-demand — the attack vectors identified in published academic literature against naive implementations do not apply.

10. Technical Architecture

Layer	Components	Role
Smart Contracts	Stealth Key Registry, Send/Receive contracts	On-chain state: key registration, fund settlement, announcement events
Indexer / Backend	Blockchain event scanner, payment database	Scans Announcement events, maintains encrypted payment metadata for authenticated users. Holds no private keys.
Client SDK	Key generation, ECDH derivation, signing	All private key operations execute locally. SDK is open-source and auditable.
User Interface	React application, wallet connectors	Wallet connection, send/receive flows, payment history dashboard

The trust hierarchy is layered such that backend unavailability degrades user experience — payment scanning becomes manual — but never compromises fund security or user privacy. Smart contracts are the source of truth; the backend is a convenience layer.

11. Roadmap and Future Development

Phase 1: Core Protocol (Current)

- ETH transfers on Ethereum mainnet
- Stealth key Registry and dual-key architecture
- Client-side key generation and ECDH derivation
- Payment scanning and stealth key derivation for withdrawal

Phase 2: Asset Expansion

- ERC-20 token support - (USDC, USDT, DAI and others
- Multi-chain deployment - Polygon, Arbitrum, Optimism, Base
- Batch transaction support and gas optimisation

Phase 3: Advanced Privacy Primitives

- Zero-Knowledge proof integration for enhanced on-chain privacy
- Decentralised key recovery mechanisms
- Privacy-preserving scanning - recipient scans without exposing scanning activity

Phase 4: Ecosystem Integration

- Defi protocol integrations (private positions, confidential swaps)
- Enteprise API for institutional and compliance-oriented deployments
- Mobile SDKs - IOS and Andriod
- Browser extensions for seamless wallet-level integration

12. Conclusion

The default transparency of public blockchains imposes a privacy cost that has no analogue in traditional financial systems. Bank account numbers are not public. Transaction histories are not searchable by anyone with an internet connection. Bianco restores the baseline expectation of financial confidentiality to EVM-native activity, not through obfuscation, but through mathematically rigorous address unlinkabilitiy.

The ECDH stealth address primitive ensures that every transaction produces a unique, unlinked destination address. the dual-key architecture ensures that privacy and compliance are not in conflict, view keys provide full auditability without compromising custody. Non-custodial smart contracts ensure Bianco cannot be a chokepoint for censorship or fund seizure.

Bianco is infrastructure. It does not restrict who uses it or for what purpose - it provides a cryptographic primitive that any user, wallet, or application can leverage to conduct on-chain activity without permanent identity exposure