WaterChain Blockcerts

A Hybrid Cryptographic Trust Architecture for Privacy-Preserving Verifiable Credentials

Research Edition · Version 5.2 · September 2026

Production baseline: WaterChain Blockcerts v3.6.3 (CONTRACT_VERSION 30603) · Avalanche C-Chain

0x8e587876Aa16320D482E35C0ECf3f8744313fDE6

Research contribution: separating durable public proof from private credential evidence through a cryptographic bridge.

Abstract

Publicly verifiable credentials create a fundamental systems problem: the facts that benefit from durable public consensus are often not the facts that should be permanently disclosed. A credential may contain a document, personally identifiable information, operational metadata, and evidence that must remain mutable, access-controlled, or erasable, while its authenticity requires a durable and independently verifiable trust anchor.

WaterChain Blockcerts addresses this problem through a hybrid cryptographic architecture that separates evidence from proof. The protocol anchors compact commitments and credential lifecycle state on Avalanche C-Chain while retaining documents, personal information, salts, operational metadata, and storage infrastructure off-chain. SHA-256 provides deterministic evidence integrity; IPFS/CID provides content-addressed persistence; salted commitments bind private evidence to public credential state; EIP-712 and hashed Claim Codes provide alternative recipient-authorization mechanisms alongside a direct-to-recipient issuance path.

This paper formalizes the architecture, system model, threat model, protocol flows, lifecycle semantics, and security properties of WaterChain Blockcerts v3.6.3. We argue that the principal contribution is architectural: the blockchain is treated as a trust substrate rather than a general-purpose document database. The design preserves independent verification while minimizing unnecessary public disclosure and maintaining operational flexibility outside the chain.

1. Problem Statement

A conventional digital certificate system typically combines three functions: storage of the certificate, presentation of the certificate, and establishment of trust in the certificate. Centralization makes these functions operationally convenient, but it also creates a common trust dependency: the verifier must trust the issuer's database or service to remain available and correct.

A blockchain can remove part of this dependency by providing a replicated, tamper-evident state history. However, directly storing certificates or personal information on a public blockchain creates an opposing problem. Public ledgers are intentionally durable and replicated. Privacy regimes, enterprise data governance, and document-management requirements frequently demand controlled access, retention management, and deletion.

The design problem is therefore a separation problem:

Which credential facts require public durability, and which credential facts require private operational control?

WaterChain treats this as a systems boundary rather than a storage-choice question. The public layer records only the cryptographic and lifecycle state necessary to establish trust.

The private layer retains evidence and operational data. The two layers are coupled through cryptographic commitments and verifiable transaction state.

2. Design Goals and Non-Goals

The protocol has six primary goals: (1) evidence integrity, (2) issuer attribution, (3) lifecycle integrity, (4) privacy-aware public anchoring, (5) storage independence, and (6) independent verification.

Evidence integrity means that a verifier can detect modification of the evidence associated with a credential. Issuer attribution means that the protocol identifies the authorized blockchain identity responsible for issuance or lifecycle operations. Lifecycle integrity means that issuance, claiming, expiration, revocation, reinstatement, and erasure are represented as protocol state rather than inferred solely from a document.

The system explicitly does not attempt to prove the substantive truth of a real-world assertion. A valid blockchain commitment proves that a particular cryptographic representation was anchored by an authorized address; it does not prove that the underlying issuer's statement was factually correct. Likewise, WaterChain is not a general-purpose document database, cryptocurrency, replacement for enterprise identity management, or automatic legal-compliance mechanism.

3. System Model

We model the system as four principal actors: an Issuer I, a Recipient R, a Verifier V, and an Adversary A. The Issuer is authorized to create and manage credentials. The Recipient may claim a credential, or may receive it directly at issuance (§8). The Verifier evaluates the authenticity and lifecycle state of a credential. The Adversary may control arbitrary application components or observe public blockchain traffic, subject to the assumptions defined in the threat model.

The system has three computational domains: the Off-Chain Domain O, the Cryptographic Bridge B, and the On-Chain Domain C.

O contains documents, personal data, certificate numbers, salts, CIDs, databases, APIs, authentication systems, and user interfaces.

B consists of deterministic transformations that allow private evidence to be represented by compact public commitments. These transformations include SHA-256 and the protocol's commitment construction.

C contains the deployed WaterChain Blockcerts contract, blockchain state, events, issuer authorization, commitment values, timestamps, lifecycle flags, and claim authorization state.

The critical invariant is that C does not need to contain the complete contents of O. Instead, C must contain sufficient cryptographic state to detect inconsistency between authorized off-chain evidence and the anchored credential representation.

4. Threat Model

We consider an adversary capable of modifying or replacing off-chain documents, compromising a web presentation layer, manipulating application database records, observing public blockchain transactions, attempting to replay claim authorizations, obtaining a Claim Code, or compromising individual infrastructure components.

We also consider compromise of an issuer key and compromise of an administrative key as separate threats because their authority and blast radius differ.

The adversary is not assumed to break SHA-256, the underlying cryptographic hash used by the commitment construction, or ECDSA signature security at the parameter sizes used by the platform. If these primitives fail cryptographically, the security arguments in this paper no longer hold.

The threat model does not claim protection against malicious issuers. An issuer with legitimate authorization can create a credential containing a false real-world assertion. Blockchain integrity cannot correct an invalid input assertion. WaterChain therefore establishes provenance and consistency rather than truthfulness of the underlying statement.

5. Architecture

WaterChain is composed of an On-Chain Trust Layer, an Off-Chain Data and Privacy Layer, and a Cryptographic Bridge.

The On-Chain Trust Layer is WaterChain Blockcerts v3.6.3 on Avalanche C-Chain. The production deployment is 0x8e587876Aa16320D482E35C0ECf3f8744313fDE6. The contract implements credential identity commitments, document commitments, issuer authorization, lifecycle state, EIP-712 claims, Claim Code claims, direct-to-recipient issuance, soulbound semantics, and lifecycle events.

The Off-Chain Layer contains the website, Workspace, API, Cloud/Storage, document gateway, IPFS/CID infrastructure, authentication, operational databases, and privacy controls.

The Cryptographic Bridge is the protocol boundary. A document becomes a SHA-256 digest; the digest and a secret salt become a document commitment; the commitment is anchored on-chain. Similarly, a human-readable certificate number and certificate salt become a certificate commitment. This design permits the public chain to establish a durable relationship without storing the underlying document or plaintext certificate number.

Figure 1 — WaterChain Trust Architecture. Public proof is anchored on-chain; private evidence remains operationally controlled off-chain.

6. Formal Credential Representation

Let a credential be represented by the tuple:

$$K = (cid, I, T, C, D, X, S)$$

where cid is the on-chain token identity, I is the issuer, T is the recipient or claim target, C is the certificate identifier, D is the document representation, X is lifecycle metadata, and S is secret verification material retained off-chain.

The protocol does not publish K in full. Instead it publishes a projection P(K) containing the values required for public trust: issuer identity, compact commitments, lifecycle state, timestamps, and authorization state.

For a certificate identifier C and salt s_c, the certificate commitment is:

$$C^* = keccak256(abi.encode(C, s_c))$$

For a document digest d = SHA256(D) and document salt s_d:

$$D^* = keccak256(abi.encode(d, s_d))$$

The commitment construction uses Solidity's abi.encode rather than packed concatenation (abi.encodePacked). abi.encode applies ABI-standard type-length and padding rules, which avoids the collision class possible when packed-concatenating variable-length arguments (such as a certificate-number string) with fixed-length arguments — a distinction that matters for the binding property claimed in P2 (§14).

The protocol therefore establishes a cryptographic relation between the private credential and its public representation without making the private representation itself public.

The security interpretation of these equations is conditional: verification requires access to the appropriate off-chain pre-image material. A commitment is not encryption and should not be treated as an encrypted database field.

7. Evidence Integrity and Persistence

SHA-256 and IPFS/CID are intentionally retained as separate primitives.

SHA-256 establishes a deterministic digest of the evidence. If the evidence bytes change, the digest changes with overwhelming probability under the assumed security of the hash function.

IPFS/CID provides content-addressed persistence and retrieval. It answers an availability and addressing problem rather than a consensus problem.

The blockchain commitment provides the public trust anchor. It establishes that a particular representation of the evidence was incorporated into the credential state.

The resulting architecture can be written as two paths:

Integrity path: DOCUMENT → SHA-256 → salted commitment → AVALANCHE.

Persistence path: DOCUMENT → IPFS / STORAGE → CID → DOCUMENT GATEWAY.

A storage provider may change without changing the historical trust state. Conversely, a valid CID is not itself evidence that an issuer authorized the associated credential. The protocol deliberately prevents these concepts from being conflated.

8. Issuance Protocol

Issuance is a two-domain protocol.

Off-chain preparation computes the evidence digest, generates or retrieves required salts, constructs certificate and document commitments, persists the document, records the CID where applicable, and prepares claim authorization where applicable.

On-chain issuance validates the parameters, rejects duplicate certificate commitments, establishes the certificate record, assigns the initial owner or issuer custody, and emits the corresponding blockchain events.

The protocol supports two issuance paths, selected by whether a recipient address is supplied at issuance time. If no recipient address is supplied, the credential is minted to the issuer's own custody in the UNCLAIMED state and requires a subsequent claim — via EIP-712 signature or Claim Code (§9) — to reach CLAIMED. If a recipient address is supplied directly at issuance, the credential is minted immediately to that address and recorded as CLAIMED at issuance; no separate claim transaction occurs. This direct path is appropriate when the recipient's wallet address is already known and verified at issuance time.

The production contract supports batch issuance with MAX_BATCH_SIZE = 100. Batch issuance changes transaction economics but does not change the credential semantics: each credential receives its own identity and lifecycle state, and each entry in a batch independently selects between the two issuance paths above.

The application layer must maintain provenance between the internal certificate record and the on-chain transaction hash, token ID, commitment values, issuer address, evidence location, and lifecycle state. This mapping is operational state; the blockchain remains the independent trust anchor for the public protocol facts.

9. Recipient Authorization Protocols

The two mechanisms below apply when a credential is issued without a predetermined recipient (§8), i.e. while it is in the UNCLAIMED state. A credential issued directly to a known recipient address skips the claim step entirely and is marked CLAIMED at issuance.

EIP-712 Signature Claim. The authorization message is bound to token ID, recipient address, nonce, and deadline. The issuer signs the typed data. The contract recovers the signer and verifies issuer authorization, recipient equality, nonce freshness, and deadline validity before completing the claim. Consumed nonces prevent replay.

Claim Code. The issuer creates a code and sends it through an off-chain channel. The contract stores a hash of the code rather than the plaintext value. The recipient connects a wallet and submits the code; the contract verifies the hash and completes the claim.

These mechanisms solve different operational problems. EIP-712 is stronger when the intended recipient address is known at authorization time. Claim Code is useful when the issuer needs to distribute a credential before learning the recipient's wallet address.

A given credential's claim path is mutually exclusive at the parameter level: a credential issued with a direct recipient address cannot also carry a Claim Code, and vice versa. This prevents an issued-but-unclaimed credential from being claimable through more than one authorization channel at once.

Claim Codes are not confidentiality mechanisms. A plaintext code submitted in a public blockchain transaction may be observable in calldata. Therefore, codes must be generated with sufficient entropy, treated as one-time authorization material, and protected against reuse.

10. Credential State Machine

Credential validity is modeled as a lifecycle state machine rather than a static property.

The principal states are:

```
ISSUED → UNCLAIMED → CLAIMED
```

and from applicable states:

```
CLAIMED → REVOKED → REINSTATED
```

A credential issued directly to a known recipient address (§8) enters CLAIMED immediately at issuance, bypassing the intermediate UNCLAIMED state. Time can move an otherwise valid credential into EXPIRED, subject to its expiry configuration. ERASED represents a terminal privacy-oriented state, reachable from any non-erased state.

The state machine provides historical semantics. Revocation does not erase the historical fact that a credential was issued. Reinstatement does not erase the revocation event. Expiration is distinguishable from revocation. Erasure is distinguishable from ordinary invalidation.

This distinction allows a verifier to answer both historical and present-state questions: “Was this credential anchored?” and “Is it currently valid?”

```
ISSUED → [UNCLAIMED →] CLAIMED → REVOKED → REINSTATED

■ EXPIRED / ERASED
```

11. Soulbound Semantics

WaterChain credentials are designed to represent a credential relationship, not a tradable asset. Ordinary ERC-721 transfer and approval semantics are therefore restricted. Controlled issuer-to-recipient transitions are performed through the protocol's issuance and claim mechanisms; all other transfer and approval calls revert.

The use of ERC-721-compatible identity does not imply economic transferability. Token identity is used as a durable identifier and integration primitive, while the application semantics define the credential as soulbound.

This distinction is important because unrestricted transfer could separate a credential from the recipient for whom the issuer intended it. The protocol therefore makes recipient assignment an authorization event rather than a market operation.

The contract additionally implements EIP-4906 (the Metadata Update Extension), emitting MetadataUpdate and BatchMetadataUpdate events whenever a certificate's on-chain state changes — for example on issuance, claim, or a base-URI update. This allows off-chain indexers, marketplaces, and verifier front-ends to detect that a token's metadata should be refreshed without re-scanning full contract history.

12. Issuer Trust and Governance

Issuer trust has two dimensions: cryptographic authorization and organizational legitimacy.

The blockchain establishes which address performed an operation and whether that address was authorized by the protocol. It does not establish that the real-world organization behind the address is legitimate, accredited, or truthful.

WaterChain therefore separates issuer registry state from certificate state. Issuer verification can change over time without rewriting the historical provenance of previously issued credentials.

Issuer authorization can further be toggled between an open mode and an enforced mode via an issuer-registry-enforcement flag. In enforced mode, only addresses explicitly marked as verified issuers may call the issuance functions; in open mode, any address may issue, though each issuer's verification status remains visible to verifiers regardless of mode. This allows a deployment to run permissively during early rollout and tighten to a verified-issuer-only model later without a contract redeployment.

Administrative authority is also separated from routine issuer activity. Ownable2Step reduces accidental ownership transfer risk, while production governance should use multisignature controls for high-impact administrative actions. Issuer keys and administrative keys should not be treated as interchangeable credentials.

13. Public Verification Protocol

Public verification is intentionally walletless. The verifier does not need a blockchain wallet merely to inspect a credential.

A verification request may contain a certificate identifier, token ID, document, CID, or verification URL. The public service resolves the request into on-chain state and, where authorized and available, corresponding off-chain evidence.

A complete verification may perform the following checks:

- Resolve the certificate identity.
- Read issuer and lifecycle state from the blockchain.
- Recompute or validate the certificate commitment when the required pre-image material is available.
- Compute SHA-256 of the presented evidence.
- Reconstruct the document commitment using the appropriate verification material.
- Compare the reconstructed commitment with the on-chain commitment.
- Evaluate issuer authorization, claim state, expiration, revocation, reinstatement, and erasure.
- Present the result with sufficient provenance for independent reproduction.

The public website is therefore a verifier interface, not the root of trust.

14. Security Properties and Proof Sketches

P1 — Evidence Integrity. Under the collision-resistance assumption of SHA-256, changing the document changes its digest except with negligible probability. If the commitment construction is binding under the assumed hash function, a changed digest cannot reproduce the original document commitment without breaking the commitment assumption.

P2 — Certificate Binding. The certificate identifier is bound to the public credential through a salted, ABI-encoded commitment (§6). An observer lacking the salt cannot directly derive the identifier from the commitment except through guessing attacks against the identifier and salt space.

P3 — Claim Authorization. An EIP-712 claim is bound to a recipient, token, nonce, and deadline. A signature valid for one recipient is therefore not valid for another recipient under the same authorization tuple. A consumed nonce prevents replay of the same authorization.

P4 — Claim-Code Authorization. The contract verifies the hash of the supplied code against stored claim authorization material. The plaintext code is not required to be stored as persistent contract state.

P5 — Lifecycle Safety. Claim operations are conditioned on credential state and timing. A credential that is already claimed, revoked, erased, or expired cannot be treated as an ordinary claimable credential.

P6 — Trust-Layer Independence. Modification of an off-chain presentation component does not alter historical blockchain state. Modification of blockchain state cannot be performed by merely changing an off-chain database.

These are proof sketches, not formal machine-checked proofs. Their validity depends on the implementation, the cryptographic assumptions, and the correctness of the surrounding operational controls.

15. Failure and Compromise Analysis

The system is designed so that individual component failure does not imply universal trust failure.

Website compromise can alter presentation but should not rewrite blockchain state.

API compromise can alter application responses but should not authorize arbitrary blockchain state transitions unless the attacker also obtains an authorized signing capability.

IPFS or document-gateway failure can reduce evidence availability while leaving the public commitment intact.

Database compromise can expose off-chain information depending on deployment controls, but should not permit alteration of historical on-chain commitments.

Issuer-key compromise can permit unauthorized actions within issuer authority. Administrative-key compromise can have a larger blast radius. Multisignature governance reduces the probability that one compromised administrative key becomes a protocol-level single point of failure.

This is the principle of compromise separation: no single presentation or storage component should simultaneously control both the evidence and the public trust anchor.

16. Privacy and GDPR/KVKK Considerations

The architecture follows a data-minimization principle: the public ledger should contain the smallest representation that is sufficient for durable verification.

Plaintext certificate numbers, names, email addresses, documents, and similar personal information are intended to remain off-chain. Commitments reduce direct disclosure but do not automatically eliminate the possibility that a value can be linked to a person.

The privacy boundary therefore includes salts, databases, logs, backups, analytics, browser storage, document gateways, and IPFS persistence. A system that removes personal data from the smart contract but retains it indefinitely in backups has not achieved meaningful operational erasure.

The erasure operation may be triggered by either the issuing address or the current token holder, which allows the data subject — when in possession of the credential — to initiate erasure independently of the issuer. The operation is irreversible and clears the on-chain commitment and claim-related fields; it does not itself delete off-chain records.

The protocol's erasure operation is consequently only one component of a broader process. The off-chain platform must execute corresponding deletion and retention-policy procedures.

The architecture can support GDPR/KVKK-aligned deployment, but legal compliance remains deployment-specific and requires assessment of roles, lawful basis, data categories, retention, data-subject rights, security controls, and jurisdiction.

17. Storage, Gas, and Scalability

The design minimizes on-chain storage because blockchain state is the least flexible storage environment in the architecture.

A credential stores compact commitments, addresses, timestamps, flags, claim authorization state, and other values needed for protocol verification. Large documents and operational metadata remain outside the contract.

Batch issuance supports up to 100 credentials per transaction. The actual gas profile depends on deployed bytecode, optimizer configuration, storage writes, event emission, network conditions, and transaction composition. A benchmark should therefore be performed against the deployed production bytecode rather than inferred from source-level operation counts alone.

The architecture scales by keeping the expensive and mutable data plane off-chain while preserving a compact public trust plane on-chain.

18. Implementation and Production Baseline

The production contract is WaterChain Blockcerts v3.6.3 with CONTRACT_VERSION 30603 on Avalanche C-Chain. Production address:

`0x8e587876Aa16320D482E35C0ECf3f8744313fDE6`

The contract is treated as the current trust baseline. Website, Workspace, API, document gateway, storage, IPFS infrastructure, and interface improvements do not inherently require a new contract deployment.

This separation is operationally significant. A platform should not redeploy its root trust layer merely because its presentation layer changes. New deployments should be reserved for genuine protocol changes, security remediation, or architectural migrations that cannot be implemented around the current contract.

19. Verification and Auditability

Auditability is achieved through the combination of blockchain events, deterministic commitments, application provenance, and independently retrievable evidence.

For each certificate, the operational system should preserve the relationship between certificate number, token ID, transaction hash, issuer, document SHA-256 digest, document commitment, CID, lifecycle state, and relevant authorization events.

An auditor should be able to move in both directions: from an operational certificate to its blockchain transaction, and from blockchain state to the corresponding evidence record. Neither direction should depend exclusively on an undocumented database convention.

This bidirectional traceability is a practical requirement for production trust systems because correctness is not only about cryptographic validity; it is also about being able to reconstruct how a record came to exist.

20. Related Work and Positioning

WaterChain overlaps with several established technical areas but makes a different architectural combination.

Blockcerts demonstrates blockchain-anchored academic and professional credentials. W3C Verifiable Credentials define interoperable credential semantics and presentation models. ERC-721 provides a widely understood token identity primitive, while EIP-5192 describes minimal soulbound semantics. EIP-712 provides typed structured-data signing. EIP-4906 provides a standard signal for off-chain metadata refresh (§11). IPFS provides content-addressed storage.

WaterChain does not attempt to replace these primitives. Instead, it composes them around a specific trust-boundary decision: the blockchain should anchor compact proof while the evidence and privacy-sensitive data remain outside the chain.

The resulting system should therefore be evaluated as a hybrid trust architecture rather than as another token standard.

21. Limitations

The system has several limitations.

First, cryptographic anchoring does not prove the truthfulness of the issuer's real-world assertion.

Second, commitment privacy depends on secret material and identifier entropy. Predictable identifiers with weak salts can remain vulnerable to guessing or correlation.

Third, complete document verification requires evidence availability. Blockchain durability does not automatically provide document availability.

Fourth, compromised issuer credentials remain a material risk because an authorized malicious issuer can produce correctly signed but substantively false credentials — including via the direct-to-recipient issuance path, which carries the same issuer-trust assumptions as the claim-based paths.

Fifth, the architecture cannot by itself guarantee GDPR or KVKK compliance. Compliance depends on the complete deployment.

Sixth, the security arguments presented here are proof sketches. Independent smart-contract audit, adversarial testing, formal verification of critical invariants, and operational security review remain appropriate for high-assurance deployment.

22. Future Research

The architecture suggests several research directions.

Formal verification could encode credential lifecycle invariants and prove that unauthorized transitions are unreachable under the contract's assumptions.

Selective disclosure and zero-knowledge mechanisms could allow a verifier to establish a credential property without receiving the entire underlying document.

Interoperability with W3C Verifiable Credentials could allow WaterChain commitments to operate as a trust and anchoring layer beneath broader credential ecosystems.

A dedicated Avalanche L1 could provide specialized governance, validator participation, institutional controls, and ecosystem-level infrastructure without changing the core separation between proof and data.

Long-term research should also examine cryptographic agility, migration between commitment schemes, post-quantum signature options, evidence availability guarantees, and multi-network anchoring.

23. Conclusion

WaterChain Blockcerts is a hybrid cryptographic trust architecture designed around a simple observation: public trust and private data have different technical requirements.

The protocol therefore does not attempt to make the blockchain the database. It makes the blockchain the durable trust substrate. Documents, personal information, operational metadata, and storage remain in the off-chain data plane. SHA-256 establishes evidence integrity; IPFS/CID provides content-addressed persistence; salted commitments bind private evidence to public state; EIP-712, Claim Code, and direct-to-recipient issuance authorize recipient assignment; and the credential lifecycle preserves the historical and current validity of the record.

The architecture's central property is controlled coupling. The layers are cryptographically connected but operationally separated. A compromised website should not be able to rewrite blockchain truth. A blockchain commitment should not disclose the underlying document. A storage migration should not require rewriting historical trust state.

The blockchain does not need to know everything in order to prove something important. It needs to preserve the right cryptographic facts. The surrounding system then manages the information and workflows required to use those facts responsibly.

That boundary between proof and data is the central design principle of WaterChain Blockcerts.

Certificate Identifier Standard

WaterChain certificate identifiers are opaque, date-independent unique identifiers using the canonical format:

```
WCB-WWWW-XXXX-YYYY-ZZZZ
```

The identifier contains no issuance date, geographic code, issuer code, recipient information, vessel information, or other semantic metadata. The character alphabet should exclude visually ambiguous characters such as 0/O and 1/I/L.

Uniqueness is enforced operationally by the Workspace/API issuance layer and cryptographically bound to the on-chain certificate commitment (§6). The blockchain timestamp remains the authoritative temporal record.

Note: This paper describes the current WaterChain production architecture and intended security properties. Security claims are proof sketches under stated assumptions and are not a substitute for independent audit or formal verification.

Appendix A — Production Parameters

Parameter	Value
Protocol	WaterChain Blockcerts
Production contract	v3.6.3 / CONTRACT_VERSION 30603
Network	Avalanche C-Chain
Contract address	0x8e587876Aa16320D482E35C0ECf3f8744313fDE6
MAX_BATCH_SIZE	100
Certificate format	WCB-WWWW-XXXX-YYYY-ZZZZ
Identifier properties	Opaque, unique, date-independent; no semantic metadata
Integrity primitive	SHA-256
Persistence	IPFS/CID + Cloud/Storage
Commitment	Salted keccak256(abi.encode(...))
Issuance paths	Direct-to-recipient · EIP-712 Signature · Claim Code
Credential semantics	Soulbound (ERC-5192), EIP-4906 metadata updates
Governance	Ownable2Step; issuer-registry-enforcement toggle
Verification	Walletless

Appendix B — Threat / Control Matrix

Threat	Primary control	Residual consideration
Document tampering	SHA-256 + document commitment	Requires correct verification material
Certificate enumeration	Salted certificate commitment	Depends on salt secrecy / entropy
Claim replay	EIP-712 nonce + deadline	Issuer key security remains critical
Wrong recipient	Recipient binding in EIP-712	Claim Code flow has different trust properties

Duplicate/ambiguous claim path	Direct-recipient and Claim Code mutually exclusive at issuance	Applies only within a single credential
Off-chain database tampering	Independent on-chain commitment	Evidence availability may be affected
Website compromise	Separation from blockchain authority	Presentation can still be misleading
Admin key compromise	Ownable2Step + recommended multisig	Governance remains an operational risk
IPFS/storage outage	Independent blockchain anchoring	Availability of evidence may degrade

Appendix C — Selected References

- S. Nakamoto, Bitcoin: A Peer-to-Peer Electronic Cash System, 2008.
- V. Buterin, A Next-Generation Smart Contract and Decentralized Application Platform, 2014.
- W3C, Verifiable Credentials Data Model.
- W3C, Data Integrity 1.0.
- IETF RFC 8785, JSON Canonicalization Scheme.
- EIP-712, Typed Structured Data Hashing and Signing.
- EIP-5192, Minimal Soulbound NFTs.
- EIP-4906, Metadata Update Extension.
- OpenZeppelin Contracts Documentation.
- Avalanche Developer Documentation.
- Blockcerts Open Standard.
- European Union, Regulation (EU) 2016/679 (GDPR).
- Republic of Türkiye, Law No. 6698 on the Protection of Personal Data (KVKK).

Note: This paper describes the current WaterChain production architecture and its intended security properties. The security claims are proof sketches under stated assumptions, not a substitute for independent audit or formal verification.