ENB Platform

Hash Commitments, Ledger Checkpoints and Bitcoin Anchoring for Companies and Countries

with sign-in through ENIGMA accounts

IAIC TECHNICAL REPORTS

IAIC-TR-2026-03 • November 2026

https://doi.org/10.66693/iaic.tr.2026.3

Andrey Nechesov, PhD

Chair, International AI Committee (IAIC)

IAIC AI Research & Trading FZCO
Dubai, United Arab Emirates
iaic.world • enigma.ist • enb.world

Reference software licensed under Apache 2.0.
This document is licensed under Creative Commons Attribution 4.0 (CC BY 4.0).

Abstract

enb.world is the platform through which companies, public authorities and countries use the Enigma Blockchain (ENB). An organisation writes to the Enigma Root the cryptographic fingerprints of its own records, or checkpoints of a blockchain it runs itself; the platform anchors the Root's history in the Bitcoin blockchain every day; and anyone can check a record, or a single transaction of an organisation's ledger, against the Root and the Bitcoin anchor without trusting the organisation or the platform. People sign in with their ENIGMA account from enigma.ist, and every account has keys on the ledger from the moment it is created, held by the platform or made by the holder. An account manages an organisation but cannot sign for it: commitments, checkpoints and key changes are valid only with the organisation's own threshold signature, made with keys that the platform never holds.

IAIC-TR-2026-01 specifies the ledger and IAIC-TR-2026-02 the commitment protocol. This report specifies the platform between them and their users: sign-in through ENIGMA accounts and the binding of an account to its holder's keys; organisations, roles and four verification levels; signer key sets with rotation, emergency suspension and recovery; record commitments; ledger checkpoints, which make an organisation's own blockchain a child of the Root, with on-chain reports of conflicting checkpoints and an algorithm that verifies one transaction of a child ledger down to a Bitcoin block; a launcher with which any organisation starts a child ledger of its own within minutes, free of charge, on its own computer or servers; notary receipts for files hashed in the browser; daily anchoring through OpenTimestamps, and why members write to an independent layer that is fast and free to write while Bitcoin protects its history; the proof bundle and its QR form; the Platform API; nested hierarchies for countries and company groups, with three deployment models for a country; security, privacy and operations; and the implementation status. No coin, token or fee is involved, and verification is free to everyone.

Keywords. blockchain platform; hash commitments; ledger checkpoints; Bitcoin anchoring; OpenTimestamps; multi-blockchain; single sign-on; threshold signatures; proof bundle; verifiable records; data sovereignty.

Suggested citation. Nechesov, A. (2026). *ENB Platform: Hash Commitments, Ledger Checkpoints and Bitcoin Anchoring for Companies and Countries.* IAIC Technical Reports, IAIC-TR-2026-03. IAIC AI Research & Trading FZCO, Dubai, UAE.
https://doi.org/10.66693/iaic.tr.2026.3

Contents

Abstract ii

1 Introduction 1

2 Terms 2

3 Design Principles 2

4 Architecture 3

4.1 Three levels . . . 3
4.2 Components . . . 4
4.3 What each component holds . . . 5

5 Identity, Organisations and Access 5

5.1 ENIGMA accounts and keys . . . 5
5.2 Organisations and roles . . . 6
5.3 Verification levels . . . 6
5.4 Signer keys . . . 7
5.5 API keys . . . 7

6 Services 7

6.1 Record commitments . . . 7
6.2 Ledger checkpoints . . . 8
6.3 Launching a child ledger . . . 10
6.4 Notary receipts . . . 11
6.5 Bitcoin anchoring . . . 11
6.6 An independent layer: fast, free to write, secured by Bitcoin . . . 12
6.7 Verification . . . 12
6.8 Corridors . . . 13

7 Platform API 13

7.1 Conventions . . . 13
7.2 Calls . . . 14
7.3 Webhooks . . . 14
7.4 Software . . . 14
7.5 Sandbox . . . 15

8 Hierarchies: Countries and Groups 15

8.1 National hierarchy . . . 15
8.2 Company groups and networks . . . 15
8.3 Composition of proofs . . . 15
8.4 Deployment models for a country . . . 16
8.5 Neutrality towards members . . . 16

9 Fields of Use 16

10 Security and Privacy 17

11 Operations 19

11.1 Deployment . . . 19
11.2 The platform as a member . . . 19
11.3 Service levels . . . 19
11.4 Data held and for how long . . . 19

11.5 Continuity 20
12 Governance, Law and Commercial Principles 20
13 Implementation Status and Plan 21
13.1 State in November 2026 21
13.2 Plan 21
References 22
A Example of a Proof Bundle 24
B Document History 25

1. Introduction

Organisations keep their records in their own databases and, more and more often, in permissioned blockchains of their own. Whoever runs such a system controls it: the operator of a database can change a row, and the validators of a private ledger, acting together, can rewrite its history. Customers, regulators and foreign counterparts receive the result and must take it on trust.

IAIC-TR-2026-02 [2] gives such organisations a place to write the fingerprints of their records, signed by themselves, on a ledger operated by a third party and anchored in Bitcoin. IAIC-TR-2026-01 [1] specifies that ledger, the Enigma Blockchain (ENB). Neither report says how an organisation joins, how its people sign in, how it manages its keys, how it connects a blockchain of its own, or how it obtains proofs and hands them to others. That layer is the platform at `enb.world`, and it is the subject of this report.

The platform does four things:

1. it lets people sign in with their ENIGMA account and lets organisations register, prove who they are and manage their keys;
2. it receives signed record commitments, ledger checkpoints and notary submissions, and writes them to the Enigma Root;
3. it anchors the Root in the Bitcoin blockchain once a day, through OpenTimestamps [4], without holding any cryptocurrency;
4. it serves verification to anyone, free of charge, through a web page, an API and open-source software that also runs offline.

This report cites the ledger and the commitment protocol and does not repeat them. Section 13 states which parts run today; every other part of this report is a specification.

2. Terms

Term	Meaning
Account	A person's account on enigma.ist, used to sign in to enb.world. It has keys on the ledger: those the platform holds for it, and any the holder made.
Organisation	A company, public authority or other body registered on the platform.
Member	An organisation's identity as a writer on the Enigma Root, as defined in IAIC-TR-2026-02.
Signer key set	The n public keys and the threshold t with which an organisation signs. The private keys stay with the organisation.
API key	A secret that lets an organisation's software call the Platform API. It authenticates the connection and cannot sign anything.
Commitment	A member's signed statement that the Merkle root of its records for a period is R (IAIC-TR-2026-02).
Child ledger	A blockchain run by an organisation or a country whose checkpoints are written to the Root.
Checkpoint	A signed statement that block h of a child ledger has hash b, together with a Merkle root of the block hashes since the previous checkpoint.
Notary receipt	A proof that a fingerprint was submitted to the platform's notary no later than a stated time.
Anchor	An OpenTimestamps proof that fixes one day of the Root in a Bitcoin block.
Proof bundle	Everything a verifier needs for one record or transaction, in one JSON object.
Witness	An organisation holding a full read-only replica of the Root.

3. Design Principles

P1. Accounts manage, keys sign. An ENIGMA account manages an organisation, and only the organisation's keys sign its commitments, checkpoints and key changes.

P2. Records stay with their owner. The platform receives fingerprints, Merkle roots and checkpoints. It never receives records or files; a file for a notary receipt is hashed in the browser.

P3. Verification without the platform. A proof bundle verifies with the open-source verifier, offline if needed, given Bitcoin block headers. The platform's page and API make verification convenient, and a verifier does not depend on them.

P4. One hierarchy. Any ledger can be a child of the Root and the parent of ledgers of its own, as in the multi-blockchain of Nechesov and Ruponen [3]. The Root sees only what a parent writes to it.

P5. Anchoring in Bitcoin without holding cryptocurrency. The Root's history is anchored daily through OpenTimestamps. The platform holds no cryptocurrency and pays no transaction fee, as IAIC-TR-2026-01 requires.

P6. No asset. There is no coin, no token and no on-chain fee. Services are paid in ordinary

currency; reading and verification are free to everyone.

P7. Status stated plainly. Each component is described as running or as specified, and the validator set as operator-run or consortium, in the sense of IAIC-TR-2026-01.

P8. Free exit. An organisation can export all its entries and proofs and leave at any time. Its past proofs stay verifiable.

4. Architecture

4.1 Three levels

The platform follows the three levels of the architecture published in IEEE Access [3], which enb.world describes as L1, L2 and L3:

- **L1, where records are made:** members' own systems and child ledgers, and the ledgers below them.
- **L2, the Enigma Root:** the ENB ledger of IAIC-TR-2026-01, which accepts signed commitments and checkpoints, orders and seals them.
- **L3, Bitcoin:** the daily anchor, reached through OpenTimestamps [4, 5].

Records never leave L1. Fingerprints, Merkle roots and checkpoints move up; proofs come back down to whoever needs them.

4.2 Components

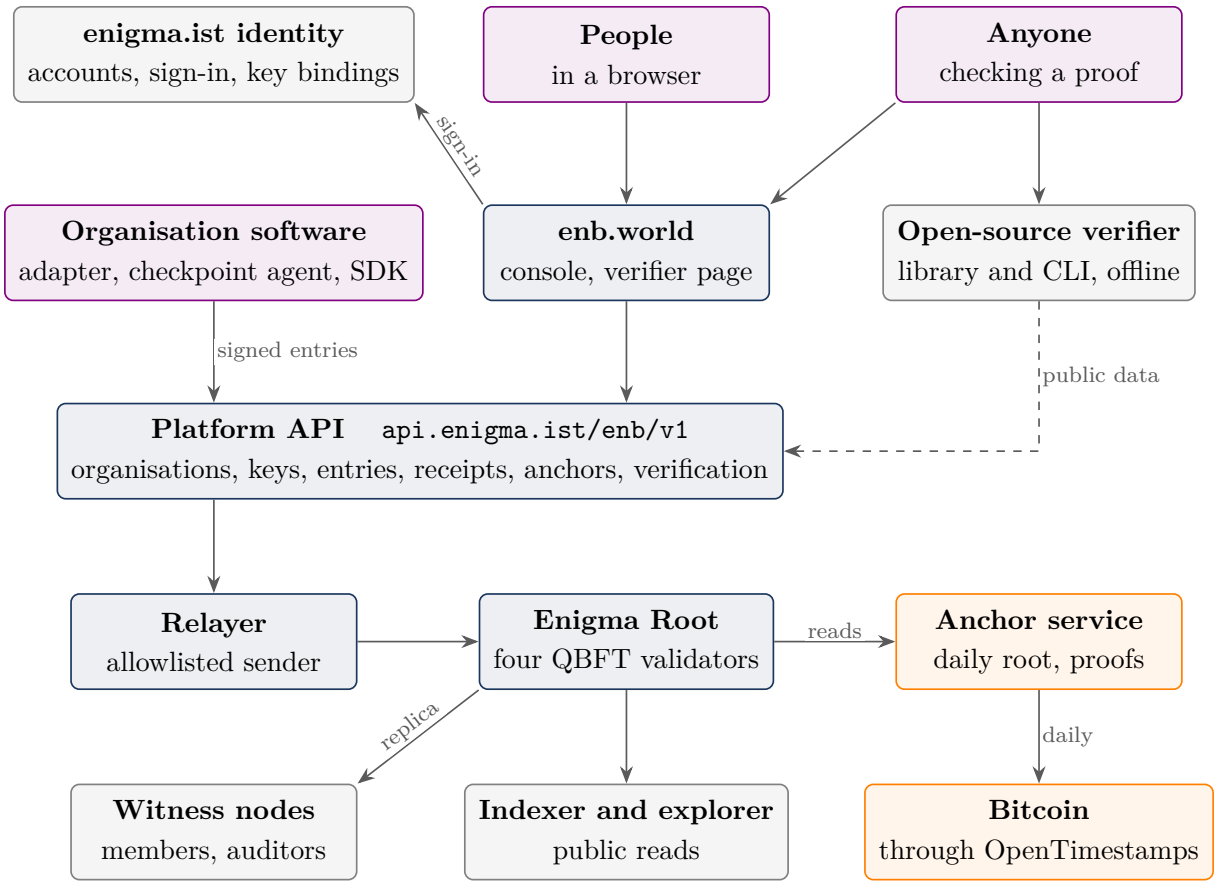

Figure 1: Components of the platform. Organisations sign their entries before they reach the API; the relayer only carries them to the Root.

4.3 What each component holds

Component	Holds	Cannot
enigma.ist identity	Accounts, sign-in methods, the keys it holds for accounts, the binding of every key to its account	Sign for an organisation; sign with a key that a holder made; see any record
Platform API	Organisation profiles and verification evidence; public key sets; hashes of API keys; webhook endpoints; submission logs	Sign for a member; change an accepted entry
Relayer	Its allowlisted sending address on the Root	Change a signed payload, whose signature would then fail
Root validators	The chain	Create a member's entry; rewrite an anchored day without detection
Anchor service	Day roots and their OpenTimestamps proofs	Anchor a set other than the day's entries without detection: anyone recomputes the day root from the chain
Indexer and explorer	A cache of the chain for fast reads	Change the chain; a verifier may read from a witness instead

5. Identity, Organisations and Access

5.1 ENIGMA accounts and keys

A person signs in to enb.world with an account on enigma.ist and the sign-in methods offered there. enb.world has no accounts and no passwords of its own.

Every ENIGMA account has keys on the ledger from the moment it is created. The platform derives them for the account from a secret that it keeps apart from its databases, as many as are needed: one for the records of the account itself, and one for each application the person uses, so that applications cannot link a person by an address. These keys are held by the platform, which signs with them on the instruction of the signed-in account. A signature of such a key shows that the platform recorded the act for that account, and the daily anchor shows that the record was not changed afterwards. It does not show that the person signed independently of the platform. This differs from the hosted custody of IAIC-TR-2026-01 [1], where a password of the holder is needed for every signature.

A person who needs a signature that the platform cannot make binds a key of their own: a key made on their own device or hardware signs a challenge that names the account, and its private key never reaches the platform. Every binding states which of the two kinds the key is, and a verifier is shown the kind. A person who holds a key of an organisation's signer key set (Section 5.4) binds it in the same way, so that the console shows which account holds which key; the signature is still made by the holder, outside the platform. Binding gives a key no authority over an organisation: what may be signed for an organisation is decided only by its signer key

set, whose keys the platform never holds.

A console session ends after eight hours, and at once when the account signs out or is suspended on enigma.ist. Actions that change who may act for an organisation (inviting an administrator, creating an API key, proposing a key set) require a fresh confirmation with a second factor of the account.

5.2 Organisations and roles

An account that creates an organisation becomes its owner. Roles are held by accounts:

Role	May
Owner	Everything an administrator may; transfer ownership; close the organisation.
Administrator	Invite and remove people; register ledgers; create and revoke API keys; set webhooks; propose key sets; start verification.
Signer	Hold one key of the signer key set and sign with it outside the platform; trigger an emergency suspension (Section 5.4). The authority comes from the key that the person holds.
Developer	Use sandbox API keys; read submission logs.
Auditor	Read the organisation's entries, logs and administrative history.

5.3 Verification levels

A verifier must know who signed. Each organisation therefore holds one of four levels, and the verifier shows the level that applied when an entry was signed.

Level	How it is established	Shown to verifiers as	May write to
V0 Registered	An account created the organisation	Unverified	Sandbox only
V1 Domain	A DNS TXT record with a platform challenge under the organisation's domain, re-checked monthly	The name with its verified domain	The Root
V2 Legal entity	The registration checked by IAIC against the public register of the issuing jurisdiction, or certified documents where no public register exists; a named legal representative	The registered legal name and jurisdiction	The Root
V3 Public authority	Confirmation from the authority through an official channel: a letter, an official domain, or its diplomatic mission	Legal name, "public authority", country	The Root; may register a national hierarchy (Section 8)

Every change of level is committed by the platform's own member (Section 11), so the level in force at any past time can be proved.

5.4 Signer keys

Keys. Signer keys are secp256k1 keys, as on ENB, and every signature is an EIP-712 signature over typed data [6]. The recommended set is two of three keys, held in hardware security modules, hardware wallets or a national key-management service. The platform never generates, receives or stores a private key; it publishes an offline tool for the key ceremony.

Registration. Each key of a new set signs a challenge naming the organisation and the hash of the whole set (proof of possession). An owner submits the set. For levels V2 and V3, IAIC confirms the set with the legal representative through a separate channel before it is activated.

Rotation. A new key set takes effect only when it is signed by t keys of the set in force, and the Member Registry of IAIC-TR-2026-02 records the period from which it applies. The console can propose a key set; it cannot activate one.

Emergency suspension. Any single key of the set in force can suspend the member, a self-suspension that this report adds to the Member Registry of IAIC-TR-2026-02 beside suspension by governance. While it is suspended the Root accepts nothing for it. Lifting the suspension, or rotating the set, needs t keys. One honest key holder who suspects theft can therefore stop further writes at once.

Recovery. If fewer than t keys remain, an organisation at level V2 or V3 may apply through its legal representative. IAIC verifies identity and authority, publishes the request on enb.world, commits it through the platform's member, and waits fourteen days, during which any remaining key can block the request. The new set is then registered and marked as recovered, and verifiers see the mark. An organisation at V0 or V1 that loses its keys registers again.

5.5 API keys

An API key is a random 256-bit secret, shown once and stored only as a hash. It carries scopes (submit commitments, submit checkpoints, notary, read logs), belongs either to the sandbox or to production, and may be limited to stated network addresses. Two keys can be active at once, so that a key is replaced without downtime. A stolen API key lets its holder send traffic under the organisation's quota; it cannot produce an entry that the Root will accept.

6. Services

6.1 Record commitments

The commitment format, acceptance rules, fallback and verification algorithm are those of IAIC-TR-2026-02 [2]. The platform adds the reference *adapter*, a container installed in the organisation's environment, which:

1. reads newly finalised records from the organisation's database or message queue;
2. encodes each record canonically (RFC 8785 [11]) and salts low-entropy fields;

3. builds the period's Merkle tree, by default every ten minutes;
4. obtains the threshold signature from the organisation's key holders or hardware modules;
5. submits the commitment through the API;
6. stores, for each record, its leaf index and audit path, so that the organisation can issue a proof bundle for any record later.

The platform keeps the commitment and its anchor. It holds no record and no audit path of a record.

6.2 Ledger checkpoints

An organisation that runs its own blockchain (a national registry on a permissioned EVM ledger, a consortium of companies, a supply-chain network) registers the ledger as a *child ledger* of the Root and writes checkpoints of it. Outsiders can then verify a transaction of that ledger without trusting its validators, and its validators cannot rewrite the ledger's history without leaving evidence signed with their own keys.

```
struct LedgerCheckpoint {
    bytes32 ledgerId;          // registered child ledger
    uint64  chainId;           // the child ledger's own chain identifier
    uint64  height;            // number of the checkpointed block
    bytes32 blockHash;         // hash of that block on the child ledger
    bytes32 blocksRoot;        // RFC 6962 root of block hashes since the last checkpoint
    bytes32 stateRoot;         // state root of the block (EVM), or zero
    bytes32 validatorSetHash;  // hash of the child's validator set in force at this height
    bytes32 prevCheckpoint;    // id of the ledger's previous checkpoint
    uint64  blockTime;         // block time on the child ledger
}
// checkpointId = the EIP-712 typed-data hash of the struct
```

Who signs. A ledger is registered with one of two signer kinds, which verifiers are shown:

- *Validator-signed.* Each validator operator of the child ledger holds a checkpoint key, and a checkpoint needs signatures from a quorum, $\lceil 2n/3 \rceil$ of n for a BFT ledger. The checkpoint then shows that a quorum of the child's validators agreed on the block.
- *Organisation-signed.* The organisation's signer key set signs, for a ledger run by a single organisation.

For QBFT child ledgers [7] the commit seals in each block header are already a quorum's signatures over the block, and a later version may accept them directly. This version uses separate checkpoint keys, which keeps the registry independent of the child ledger's client software.

Acceptance. The Ledger Registry accepts a checkpoint only if the ledger is registered and active; at least t valid signatures come from the signer set in force at that height; `height` exceeds the last accepted height; `prevCheckpoint` equals the last accepted identifier; and `blockTime` is no later

than the time of acceptance plus sixty seconds. A change of the child's validators is announced by a checkpoint signed by the outgoing quorum that carries the new `validatorSetHash`; the new set signs from the next checkpoint.

Interval. By default a checkpoint is written every ten minutes. For a child ledger with probabilistic finality, only blocks deeper than its finality depth are checkpointed. At five-second blocks a ten-minute interval covers 120 blocks, so the path from a block hash to `blocksRoot` has seven hashes.

Conflicts. The registry keeps the block hash of every checkpointed height. Anyone who holds a checkpoint of ledger L, correctly signed by L's signers, that names a different block hash for a height already checkpointed may submit it to `reportConflict`. The registry checks the signatures, records the conflict, marks the ledger, and accepts no further checkpoints for it until its owner resolves the conflict, for example by replacing validators. In a BFT ledger with $n = 3f + 1$ validators, any two quorums share at least $f + 1$ validators, so two contradictory validator-signed checkpoints identify at least $f + 1$ validators that signed both, by their own signatures. That is more faults than the child ledger tolerates, and the evidence is public.

Algorithm 1 Verify a transaction of a child ledger against the Root and its anchor

Require: transaction x and its receipt ρ; header H_b of block b; trie proofs π_x, π_ρ; audit path π_b; checkpoint C with signatures Σ; audit path π_C; OpenTimestamps proof P

if $x \notin H_b.\mathit{transactionsRoot}$ by π_x **or** $\rho \notin H_b.\mathit{receiptsRoot}$ by π_ρ **or** ρ failed **then**
  return INVALID
end if
$h_b \leftarrow \mathrm{Keccak256}(\mathrm{RLP}(H_b))$ ▷ the block hash
if RootFromPath$(h_b, \pi_b) \neq C.\mathit{blocksRoot}$ **then**
  return INVALID
end if
if Σ holds fewer than t valid signatures of the ledger's signer set at $C.\mathit{height}$ **then**
  return INVALID
end if
if a conflict is recorded for the ledger at or below $C.\mathit{height}$ **then**
  return CONFLICT
end if
$A' \leftarrow$ RootFromPath(checkpointId$(C), \pi_C$)
if P attests A' in a Bitcoin block with at least six confirmations **then**
  return VALID: x was in the child ledger no later than the time of that block
end if
return PENDING ▷ accepted on the Root, anchor not yet complete

The trie proofs are those of the Ethereum specification [8]. RootFromPath is the audit-path computation of RFC 9162 [12], which needs the leaf index and the tree size.

State at a height. With `stateRoot` and an account and storage proof as returned by `eth_getProof` [9], a verifier can prove the value of a contract's storage at the checkpointed height. This answers questions of the form "what did the register say on that date", for example the holder of a licence recorded in a registry contract.

Other ledgers. For Hyperledger Fabric [10], `blockHash` is the hash of the block header, and `stateRoot` is zero. Fabric's header commits to the block's data as a whole, so a transaction is proved by presenting the block's data with it. Ledgers and databases of other kinds use record commitments.

What a checkpoint does not prove. It does not prove that the child ledger's transactions are lawful or true, nor that its validators are independent of each other. The registry shows the number of signers and the organisation that registered the ledger; the checkpoint proves that the stated signers signed this history no later than the stated time.

```
interface ILedgerRegistry {
    event CheckpointAccepted(bytes32 indexed ledgerId, uint64 height,
                             bytes32 id, bytes32 blockHash);
    event ConflictReported(bytes32 indexed ledgerId, uint64 height,
                           bytes32 acceptedId, bytes32 conflictingId);
    function registerLedger(bytes32 ledgerId, bytes32 memberId, uint64 chainId,
                            uint8 signerKind, address[] calldata signers,
                            uint8 threshold, bytes[] calldata memberSignatures)
        external;
    function submit(LedgerCheckpoint calldata c, bytes[] calldata signatures)
        external returns (bytes32 id);
    function reportConflict(LedgerCheckpoint calldata c, bytes[] calldata signatures)
        external;
    function lastOf(bytes32 ledgerId)
        external view returns (bytes32 id, uint64 height);
    function signersAt(bytes32 ledgerId, uint64 height)
        external view returns (address[] memory, uint8);
}
```

The Ledger Registry sits beside the registries of IAIC-TR-2026-02 and follows the governance of IAIC-TR-2026-01: upgrades through the governance multi-signature with timelocks.

6.3 Launching a child ledger

An organisation does not need a blockchain of its own to use the platform, but it can have one within minutes, as a child ledger (a daughter blockchain) of the Root. The *ledger launcher*, a page of the console and an open-source command-line tool, produces everything such a ledger needs:

1. the organisation chooses a name, the number of validators (one for a trial, four or more for production) and the block period;
2. the tool generates the validator keys and the checkpoint keys on the organisation's own machine; they never leave it;
3. it writes the genesis file (QBFT, no native asset, zero fees, as on ENB), the configuration of every node, and a container bundle holding the node software (Besu), a checkpoint agent already pointed at the Root, and a block explorer;
4. the organisation starts the bundle with one command, on its own computer or on servers

of its choice, and the ledger seals blocks at once;

5. the console registers the ledger in the Ledger Registry with the member's signatures, and its checkpoints begin.

The launcher and the software are free and open source. A trial ledger of one node runs on an ordinary computer and checkpoints to the sandbox at no charge. A production ledger is the same bundle on the organisation's servers: four validators in two sites, each with 4 vCPU, 16 GB of memory and 500 GB of SSD, as in IAIC-TR-2026-02. The ledger is EVM-compatible, so the organisation deploys its own contracts for its workflow on it.

A ledger with a single validator gives its owner an ordered history that can no longer be rewritten unnoticed once its checkpoints are anchored; it does not survive the failure of that validator. Tolerance of a faulty validator needs at least four. Verifiers are shown how many validators signed.

6.4 Notary receipts

A person or an organisation without keys of its own may still need to show that a document existed at a time: a contract draft, a design, a data set, a manuscript. The verifier page computes the SHA-256 of a file in the browser, after appending a random salt, so the file never leaves the device and a short text cannot be guessed from its fingerprint. The fingerprint is sent with the account's session or an API key. The platform's notary, a member operated by IAIC, builds a Merkle tree of the fingerprints it receives every ten minutes and commits it under its own key set. The receipt is a proof bundle that contains the salt.

A receipt proves that the fingerprint reached the notary no later than the anchored time. It does not prove who wrote the document or what it says. An organisation that needs its own name on a record writes its own commitment instead.

6.5 Bitcoin anchoring

After each UTC day d, the anchor service computes

$$A_d = \mathrm{MTH}(b_d,\ e_1, \ldots, e_m),$$

where b_d is the hash of the last Root block of day d and $e_1, \ldots, e_m$ are the identifiers of all entries accepted that day (commitments, checkpoints, key-set changes, conflict reports) in order of acceptance. Including b_d anchors the whole history of the Root up to that block, as IAIC-TR-2026-01 specifies; including the entries gives each of them a short path to the anchor. Identifiers are EIP-712 hashes whose type is part of the hash, so entries of different types cannot be confused. This extends the daily anchor of IAIC-TR-2026-02, which covers commitments, to the last block of the day and to every type of entry.

The service submits A_d to at least two independent OpenTimestamps calendar servers. The calendars aggregate many submissions into one Bitcoin transaction, and a proof becomes complete

when that transaction is confirmed, usually within hours. The service then upgrades the proof, checks it against Bitcoin block headers, publishes it with the list of the day's entries, and records its hash on the Root. Anyone can recompute A_d from the chain and compare.

A complete anchor shows that the entry existed no later than the time of the Bitcoin block. Bitcoin's rules allow a block's time to differ from real time by up to about two hours, which is small against a daily anchor. As a safeguard against delay by the operator, the protocol of IAIC-TR-2026-02 lets a member's commitment be anchored on its behalf outside the daily batch, and verifiers accept that path as well.

6.6 An independent layer: fast, free to write, secured by Bitcoin

The Root is a layer of its own between the members' systems and Bitcoin. Members write to the Root, which accepts their entries in seconds and at no cost and hands its history to Bitcoin once a day, so that the history keeps Bitcoin's permanence without each entry paying Bitcoin's price in time and money.

	Writing each entry to Bitcoin	**Writing to the Enigma Root**
Time to record	A block about every ten minutes [5]; about an hour for the six blocks usually awaited	A block every five seconds, final when sealed (QBFT) [1]
Cost	A fee for every transaction, set by demand, paid in bitcoin that the writer must hold	No fee and no cryptocurrency held by the member [1]
Capacity	Shared with every user of Bitcoin worldwide	Reserved for registered members; a day's entries of all members cost one anchor
Who wrote it	An address, anonymous unless announced	A key set of a verified organisation, with its rotations (Section 5.4)
Shared state	None between writers	Corridors, receipts and single use of a certificate across members
Protection of history	Proof of work	BFT consensus, member signatures, witnesses, and the daily anchor in Bitcoin

A member therefore gets speed and zero cost from the Root and the long-term protection of Bitcoin from the anchor. Once a day is anchored, any rewriting of it by the Root's validators is visible to anyone who holds a proof.

6.7 Verification

Verification is available three ways, all running the same algorithms (IAIC-TR-2026-02 for records, Algorithm 1 for transactions):

- the public verifier page at `enb.world/verify`;

- the open-source library and command-line tool, for Python and TypeScript, which also run offline;
- the `/verify` call of the Platform API, for systems that prefer not to embed the library.

The result is Valid, Pending, Invalid with its reason, or Conflict. A valid result names the organisation with the verification level it held when it signed, the key set, the Root block, and the Bitcoin block with its time. If the member was suspended after signing, the result stays valid and the suspension is shown beside it.

Offline verification needs Bitcoin block headers: 80 bytes each, about 4.2 MB for a year of blocks, which a phone can hold.

Proof bundle. A bundle is a JSON object with these parts: `kind` (record, transaction or notary); the record in canonical form, or the transaction, receipt and block header; the leaf index, tree size and audit path; the commitment or checkpoint with its signatures; the Root block and transaction that carry it; the path to the day root; and the OpenTimestamps proof. Appendix A gives an example.

QR form. A QR code carries the short form of a bundle: the record, its leaf index, tree size and audit path, and the identifier of the commitment, compressed and placed after the `#` of a link of the form `https://enb.world/v#...`. Browsers do not send the part after `#` to the server, so opening the link does not disclose the record to the platform. The verifier then fetches the commitment, its signatures, the path to the day root and the anchor proof by the commitment's identifier, from the Platform API or from any witness. These parts are public and the same for every record of the commitment, so fetching them shows the platform which commitment is being checked and nothing about the record.

6.8 Corridors

Corridors between members, receipts and the single-use rule are specified in IAIC-TR-2026-02. The platform adds the console steps for proposing, approving and closing a corridor, and API calls for writing receipts.

7. Platform API

7.1 Conventions

- Base address `https://api.enigma.ist/enb/v1`, the platform's single API host, which already serves the explorer data of enb.world.
- JSON in both directions; hexadecimal values with a `0x` prefix in lower case; times in UTC as in RFC 3339 [13].
- Calls that create entries accept an `Idempotency-Key` header, so that a retry never creates a second entry.
- Errors carry a stable machine-readable code and a message. Limits apply per organisation

and plan and are returned in response headers.

- The sandbox and production use the same paths and are told apart by the API key.

7.2 Calls

Call	Purpose	Access
`POST /commitments`	Submit a signed record commitment	API key
`GET /commitments/{id}`	Status, Root block, anchor state	Public
`POST /ledgers`	Register a child ledger, with the member's signatures	Console
`POST /checkpoints`	Submit a signed ledger checkpoint	API key
`GET /ledgers/{id}/checkpoints`	Checkpoints of a ledger over a range of heights	Public
`POST /conflicts`	Report a conflicting signed checkpoint	Public
`POST /notary`	Submit up to 1,000 fingerprints	Session or API key
`GET /notary/{fingerprint}`	Notary receipt as a proof bundle	Public
`GET /anchors/{day}`	Day root, list of entries, OpenTimestamps proof	Public
`GET /members/{id}`	Name, verification level history, key sets with their periods	Public
`POST /verify`	Verify a proof bundle	Public
`POST /orgs`, `/orgs/{id}/domain`, `/orgs/{id}/keysets`, `/orgs/{id}/api-keys`	Create an organisation, prove a domain, propose a key set, manage API keys	Console

7.3 Webhooks

An organisation may register endpoints for these events: an accepted or rejected commitment (`commitment.accepted`, `commitment.rejected`); an accepted checkpoint or a conflict (`checkpoint.accepted`, `checkpoint.conflict`); a completed anchor (`anchor.completed`); and a changed key set (`keyset.changed`). Each delivery is signed with HMAC-SHA-256 under a secret per endpoint and is retried with growing intervals for 24 hours. An organisation should raise an alarm on any accepted entry that its own software did not submit: that is how theft of its keys is noticed (Section 10).

7.4 Software

Published under Apache 2.0: software development kits for Python and TypeScript (canonical encoding, salting, trees, signature requests, submission, bundles, verification); the adapter; the ledger launcher (Section 6.3); a checkpoint agent for EVM ledgers on Besu and go-ethereum; the witness node package; and the offline key-ceremony tool.

7.5 Sandbox

The sandbox runs on the test network (chain 31416, and its QBFT successor of IAIC-TR-2026-01). It is free and open to organisations at level V0. Sandbox entries are anchored as well, and their proofs are marked as test proofs.

8. Hierarchies: Countries and Groups

8.1 National hierarchy

A public authority at level V3 may register a national ledger as a child of the Root and admit its own members to it (agencies, districts, licensed companies) under its own rules and with validators it chooses. The Root receives only the national ledger's checkpoints. Which national members exist and what they do stays inside the country unless the country publishes it. In the corridor of IAIC-TR-2026-02, Ghana's member ledger is such a national ledger.

8.2 Company groups and networks

A group of companies can run a group ledger to which its subsidiaries write, and which checkpoints to the Root. A supply-chain network can do the same for its participants. The depth of the hierarchy is not limited.

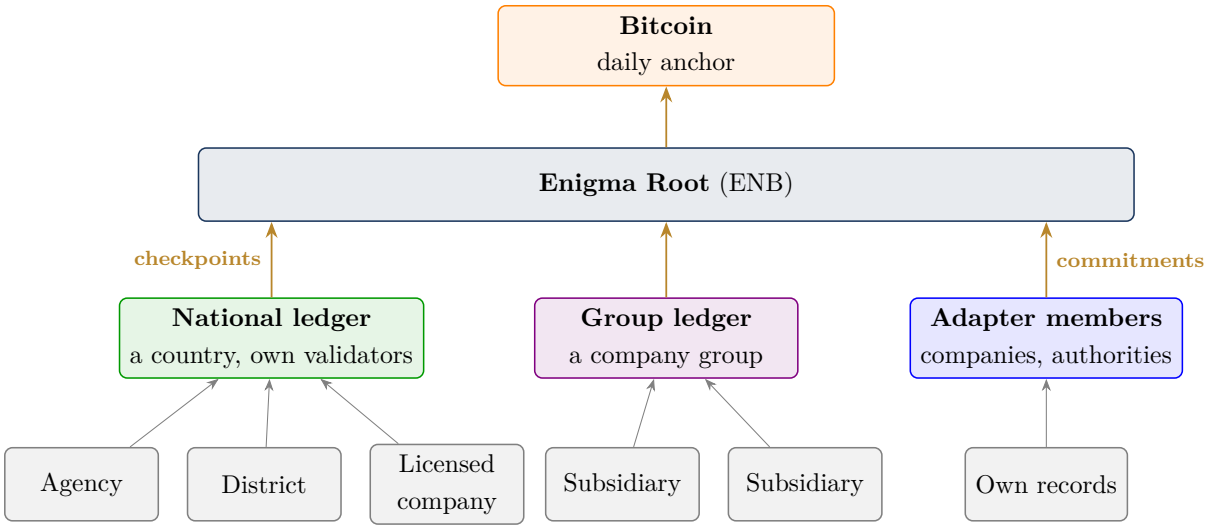

Figure 2: A hierarchy of ledgers. Each level writes only checkpoints or commitments to the level above; the Root sees no record and no member of a national ledger.

8.3 Composition of proofs

A record of a licensed company that is a member of a national ledger is proved in steps: the record against the company's commitment; the commitment as a transaction of the national ledger (Algorithm 1); the national block against a checkpoint on the Root; the checkpoint against the Bitcoin anchor. A bundle nests these proofs. Each further level adds a block header, an

inclusion proof and a checkpoint with its signatures, about one to two kilobytes.

8.4 Deployment models for a country

A public authority chooses how much of the system it runs itself. In every model the software is open source, records stay in the country, and IAIC is paid in ordinary currency for the work it does.

Model	Who runs what	What IAIC provides
Member of the Root	The authority's systems stay as they are; an adapter in its data centre writes signed commitments to the Enigma Root; corridors with partner authorities abroad	The adapter, integration with the authority's systems, verification app and QR format, operation of the Root, training
Operated member	As above, with IAIC running the adapter and a witness node inside the country under a service contract	Operation and monitoring, service levels, reports to the authority
National root	The country runs its own root ledger with validators it chooses, for its agencies and companies, and checkpoints it to the Enigma Root	Software, set-up of validators and keys, training of operators

A country can start as a member and move to a national root as more of its agencies join, or the reverse; past proofs stay valid in every case.

8.5 Neutrality towards members

The Root treats a national ledger like any other child ledger: it cannot read it, cannot write to it, and suspends it only for a reason stated in the published membership policy, which is written on chain with the suspension. A country may run a witness node of the Root to hold its own copy.

9. Fields of Use

- **Cross-border trade in minerals and goods:** certificates of one country checked in another, as in the Ghana–UAE gold corridor of IAIC-TR-2026-02.
- **Public registers:** licences, land titles, company and professional registers, diplomas. A registry kept on a ledger writes checkpoints; one kept in a database writes commitments. In both cases the content of the register at a past date can be proved.
- **Company records:** audit logs, quality and test records, supplier declarations, sustainability reports.
- **AI systems:** the decision logs of autonomous AI systems, committed each period, so that a later review sees the log as it was written. This supports the record-keeping that the IAIC 101:2026 standard [14] asks of such systems.
- **Research and publishing:** the records of the ENIGMA platform specified in IAIC-TR-

2026-01, whose daily digest is written to the test network today.

- **Education and qualifications:** results of courses and examinations, signed by the examiners, held by the learners and checked by employers, with no name written on the chain; specified in IAIC-TR-2026-04 of this series.
- **Software:** hashes of released builds, so that users can check that what they install is what was released.

10. Security and Privacy

Threat	Control	Residual risk
Takeover of an ENIGMA account	Accounts manage, keys sign (P1); second factor for administrative actions; every administrative action notified and committed	Settings changed until noticed; no entry of an organisation can be forged
Keys that the platform holds for accounts	Derived from one secret kept apart from the databases; every binding marked as held by the platform; verifiers shown the kind of key	Such a signature is the platform's record of an act and rests on trust in the operator until the day is anchored
Leak of an API key	Keys authenticate traffic only; scopes; address limits; rotation	Quota used by others
Theft of one signer key	Below the threshold nothing can be signed; any key can suspend; rotation by t keys	None while $t \geq 2$
Theft of t signer keys	Webhook alarm on every accepted entry; suspension; rotation	Entries signed between theft and suspension are valid on their face and must be disputed by the member
Compromise of the Platform API or relayer	Entries are signed before they reach it; the member's own anchoring (fallback)	Delay or refusal, bounded by the fallback
Collusion of Root validators while operator-run	Witness replicas; daily anchor; recomputable day roots	Rewrite before the anchor, detectable by witnesses
Equivocation by a child ledger's validators	On-chain conflict reports (Section 6.2)	Detected after the fact; the child ledger itself must recover
False QR code or verifier page	Open-source verifier; key sets published on chain; any copy of the verifier gives the same result	People who trust an unknown page
Flooding with free requests	Write allowlist; per-organisation limits; size limits; block gas limit	Slower service for a time

What the platform learns. Organisation profiles and verification evidence; the accounts that act for them; the time, size and frequency of submissions; fingerprints and roots, which reveal

nothing about salted records; and the network addresses of API calls, kept for ninety days. It does not learn records, files, salts or audit paths, and the QR link keeps the record in the browser (Section 6.7).

Personal data. The Root holds no personal data. Account data is held by enigma.ist under its privacy policy. Records about people are committed only as salted fingerprints, under the data policy of IAIC-TR-2026-01, and stay in the member's jurisdiction under its data-protection law.

11. Operations

11.1 Deployment

- **Root:** four QBFT validators on Hyperledger Besu in at least two jurisdictions, as in IAIC-TR-2026-02, and two read nodes behind the API.
- **Services:** relayer, anchor service, Platform API, indexer and explorer, console and verifier page; daily copies of the chain data to a second provider.
- **Witnesses:** the witness package lets any member, auditor or regulator hold a full replica and re-check every block.

11.2 The platform as a member

The platform's own administrative acts (verification levels granted or withdrawn, key sets approved, recoveries, suspensions and their reasons) are committed by the platform's member every period and anchored with everything else. The platform's administration is therefore as tamper-evident as the entries it carries.

11.3 Service levels

Measure	Target
Acceptance of a commitment or checkpoint	95% within two minutes
Availability of public reads and verification	99.5% each month
Daily anchor submitted	By 01:00 UTC
Anchor proof complete and published	Within 24 hours of submission
First webhook delivery	Within one minute of the event
Incident report published on enb.world	Within 72 hours

11.4 Data held and for how long

Data	Kept
Entries on the Root and anchor proofs	Permanently; they are public
Organisation profile and verification evidence	While the organisation is registered, then as the law requires
Administrative history of an organisation	Committed permanently as fingerprints; details with the profile
Link between an account and a notary fingerprint	Ninety days; the receipt itself stays valid
API request logs with network addresses	Ninety days

11.5 Continuity

If the platform stopped, proofs already issued would stay verifiable: bundles are self-contained, the anchors are in Bitcoin, witnesses hold the Root, and the verifier is open source. Any witness could serve the Root for reading. An organisation can export all its entries, checkpoints and proofs at any time.

12. Governance, Law and Commercial Principles

Governance. The Ledger Registry and the registries of IAIC-TR-2026-02 follow the governance of IAIC-TR-2026-01: a governance multi-signature, upgrades through timelocks, and validators added by vote. The Root is described as operator-run until at least four validators are operated by at least three independent institutions.

Membership policy. The policy is published on enb.world. A member or a ledger is suspended only for a reason it lists (a reported key compromise, an order of a competent court, or a breach of the policy), and the reason is written on chain. If production service ends, for non-payment or otherwise, new writes stop after notice; past entries stay verifiable.

No asset. No coin, token or on-chain fee exists. Production writes are provided under service plans paid in ordinary currency to IAIC FZCO. Reading, verification and the sandbox are free to everyone.

Independence. Where IAIC assesses or certifies a system, it does not also supply that system.

Legal review. A written legal opinion in the United Arab Emirates precedes production service, as in phase 5 of the roadmap of IAIC-TR-2026-01.

13. Implementation Status and Plan

13.1 State in November 2026

Component	Status
ENB test network, chain identifier 31416, a block every five seconds, operator-run; explorer at enb.world with its JSON interface on `api.enigma.ist`	RUNNING
The ENIGMA platform as the first writer: the messenger's pulse every minute and its daily digest since 5 November 2026, checkable at enb.world/enigma.html	RUNNING
ENIGMA accounts on enigma.ist, and the sign-in widget on enb.world, which shows the signed-in account	RUNNING
Root on QBFT with zero fees and a write allowlist; daily anchoring through OpenTimestamps (phases 2 and 4 of IAIC-TR-2026-01)	SPECIFIED
Member, Root and Corridor registries (IAIC-TR-2026-02); Ledger Registry with conflict reports (this report)	SPECIFIED
Console sign-in with ENIGMA accounts; binding of accounts to keys; organisations, roles, verification levels; key sets, suspension and recovery; API keys	SPECIFIED
Platform API, webhooks, software development kits, adapter, ledger launcher, checkpoint agent, witness package; sandbox	SPECIFIED
Notary receipts; verifier page, library and command-line tool; proof bundle and QR form	SPECIFIED

13.2 Plan

The phases are ordered by dependency. Dates are published on enb.world as each phase is committed.

Phase	Work	Complete when
1. Root	QBFT network from a new genesis; Member and Root registries	A commitment signed 2 of 3 is accepted; a forged one is rejected
2. Anchor and verifier	Anchor service; verifier library, command-line tool and page; proof bundle	A day's proof is published; a bundle verifies offline
3. Console and API	Sign-in with ENIGMA accounts and binding of keys; organisations, levels V0–V2; key sets; API keys; sandbox	An organisation registers, sets keys and submits from the kit on the sandbox
4. Ledger checkpoints	Ledger Registry; ledger launcher; checkpoint agent for Besu and go-ethereum	A ledger started from the launcher on one computer checkpoints every ten minutes; a transaction verifies end to end; a planted conflict is recorded
5. Notary and corridors	Notary receipts; corridor steps in the console	A file receipt verifies; a corridor receipt is single-use
6. First external members	At least one organisation at level V2 and one proof of concept, such as the six-week plan of IAIC-TR-2026-02	Partner sign-off
7. Audit and legal review	Independent audit of the registries and the anchor service; legal opinion	Audit report published; opinion received
8. Production	Production service on the operator-run Root; consortium validators as institutions join	Four validators of three institutions for the consortium state

References

[1] Nechesov, A. *Enigma Blockchain (ENB): A Tokenless Smart-Contract Ledger for Verifiable Records.* IAIC Technical Reports, IAIC-TR-2026-01. IAIC AI Research & Trading FZCO, Dubai (2026). doi:10.66693/iaic.tr.2026.1

[2] Nechesov, A. *Enigma Root of Trust: A Third-Party Notary for National and Corporate Ledgers.* IAIC Technical Reports, IAIC-TR-2026-02. IAIC AI Research & Trading FZCO, Dubai (2026). doi:10.66693/iaic.tr.2026.2

[3] Nechesov, A. and Ruponen, J. "The AGI-Powered Metaverse: A Decentralized Task-Based Cognitive Architecture." *IEEE Access* 14, 81234–81256 (2026). doi:10.1109/ACCESS.2026.3697396

[4] Todd, P. *OpenTimestamps: Scalable, Trust-Minimized, Distributed Timestamping with Bitcoin* (2016). `https://petertodd.org/2016/opentimestamps-announcement`

[5] Nakamoto, S. *Bitcoin: A Peer-to-Peer Electronic Cash System* (2008). `https://bitcoin.org/bitcoin.pdf`

[6] Bloemen, R., Logvinov, L. and Evans, J. *EIP-712: Typed structured data hashing and signing* (2017). `https://eips.ethereum.org/EIPS/eip-712`

[7] Enterprise Ethereum Alliance. *QBFT Blockchain Consensus Protocol Specification v1* (2021). https://hub.entethalliance.org/client-spec/qbft_spec.html

[8] Wood, G. *Ethereum: A Secure Decentralised Generalised Transaction Ledger* (Yellow Paper). https://ethereum.github.io/yellowpaper/paper.pdf

[9] Jentzsch, S. and Jentzsch, C. *EIP-1186: RPC-Method to get Merkle Proofs, eth_getProof* (2018). https://eips.ethereum.org/EIPS/eip-1186

[10] Androulaki, E., Barger, A., Bortnikov, V. et al. "Hyperledger Fabric: A Distributed Operating System for Permissioned Blockchains." In *Proceedings of the Thirteenth EuroSys Conference*, ACM, 1–15 (2018). doi:10.1145/3190508.3190538

[11] Rundgren, A., Jordan, B. and Erdtman, S. *JSON Canonicalization Scheme (JCS)*. RFC 8785, IETF (2020). doi:10.17487/RFC8785

[12] Laurie, B., Messeri, E. and Stradling, R. *Certificate Transparency Version 2.0*. RFC 9162, IETF (2021). doi:10.17487/RFC9162

[13] Klyne, G. and Newman, C. *Date and Time on the Internet: Timestamps*. RFC 3339, IETF (2002). doi:10.17487/RFC3339

[14] International Artificial Intelligence Committee (IAIC). *IAIC 101:2026, International Standard for Autonomous AI Systems*. Dubai (7 September 2026). doi:10.66693/iaic.101.2026

A. Example of a Proof Bundle

The record of the worked example of IAIC-TR-2026-02 (Appendix A of that report), written as a bundle. The record, its fingerprint, the audit path and `recordsRoot` are the real values of that example; parts that are assigned only when a commitment is accepted and anchored are shown in angle brackets.

```
{
  "v": 1,
  "kind": "record",
  "record": {"exported": "2026-11-13", "fineness": "930.2", "grams_in": "2105.3",
             "grams_out": "2099.0", "lot": "GH-DEMO-0413"},
  "leaf": {
    "index": 2,
    "size": 4,
    "hash": "06d3ff226c3665786edcacca6355e6ddab24ebace7fd6c11fbbe0e47eed367c4",
    "path": [
      "bc4fb4d5ef5377356020091088846f4980d1ea1f4783d7b5376068a697c9a86a",
      "40cef276b9589de8743ec2ea0e3899289a938d544177f7c1ef6642e9ca6c8187"
    ]
  },
  "commitment": {
    "memberId": "<member>",
    "period": "<n>",
    "recordsRoot":
      "0dd9a76ebb6db8b4e96226f194169799762a2d482479043ac61c565ca283e049",
    "signatures": ["<signature 1 of 2>", "<signature 2 of 2>"]
  },
  "root": {"chainId": 31415, "block": "<n>", "tx": "<transaction hash>"},
  "anchor": {"day": "<YYYY-MM-DD>", "index": "<k>", "path": ["<hashes>"],
             "ots": "<OpenTimestamps proof, base64>"}
}
```

The remaining fields of the commitment (period start and end, record count, previous commitment, policy hash) are those of the `Commitment` structure of IAIC-TR-2026-02 and are omitted here. The leaf hash is $\mathrm{SHA256}(\texttt{0x00} \,\|\, \mathrm{JCS}(r))$ of the 106-byte canonical record. From it and the two hashes of the path, the verifier obtains `recordsRoot`; the other three records of the period are never seen. The short QR form carries `record`, `leaf` and the identifier of the commitment, and fetches the rest.

B. Document History

Report	Date	Notes
IAIC-TR-2026-01	2026-11-14	Enigma Blockchain (ENB): the tokenless ledger.
IAIC-TR-2026-02	2026-11-18	Enigma Root of Trust: the commitment protocol, corridors and the Ghana–UAE reference deployment.
IAIC-TR-2026-03	2026-11-23	ENB Platform: sign-in through ENIGMA accounts, organisations and keys, ledger checkpoints, notary receipts, Bitcoin anchoring, verification and the Platform API.
	2026-11-23	Revised on the day of publication: Section 5.1 describes the keys of an ENIGMA account, held by the platform in the first version or made by the holder; wording made plainer in several places.

Contact: nechesov@iaic.world • iaic.world

© 2026 IAIC AI Research & Trading FZCO. Text licensed under CC BY 4.0.