EMILIA Protocol — SOC 2 Evidence Map (Authorization Receipts → CC6.1 / CC6.2 / CC7.2 / CC7.3)

Version: 1.0 **Date:** 2026-06-13 **Artifact:** EMILIA authorization receipt (wire tag `EP-RECEIPT-v1`) and Class-A device signoff **Framework:** AICPA Trust Services Criteria (2017, rev. 2022) — Common Criteria CC6.1, CC6.2, CC7.2, CC7.3 **Status:** Experimental. This is a composition aid, not a certification.

How to read this document

This map states, for four SOC 2 Common Criteria, **what an authorization receipt can serve as evidence *for*** when an auditor independently re-verifies it. It is deliberately narrow.

- It does **not** claim any audit outcome, time-to-certification, cost saving, insurance benefit, or customer result. EMILIA has not measured those.
- It does **not** assert that a receipt *satisfies* a criterion. A receipt is one input an auditor weighs alongside the auditee's policies, consumption records, and revocation records.
- Every value listed is a fact the auditor can **re-derive offline** from the signed packet itself (browser at `/verify`, or `npx @emilia-protocol/verify receipt.json`) — never an EMILIA-vouched opinion, score, ranking, or reputation number. EMILIA issues no scores.

An advisory or observation layer (EMILIA "Eye") may tighten posture or inform a human reviewer, but it never authorizes an action on its own and is never the sole gate. The authorizing fact in every receipt is a named human's user-verified signature over the exact action.

A receipt is the same kind of artifact whether it is read by a person, a SIEM, or another machine: it is signed, canonical (RFC 8785) JSON that carries its own public key. "Authorization receipt" is the name in prose; `EP-RECEIPT-v1` is the wire tag.

The receipt fields referenced below

Field / check	What it carries	Re-derivable offline?
`payload.receipt_id`	Stable identifier for the receipt	Yes
`payload.action` / `action_hash`	The exact governed action (canonicalized) and its SHA-256 digest	Yes
`payload.authorization.approver_id`	The named human who approved	Yes (identity *binding* only — see boundary)
`payload.authorization.approver_key_class`	A = approver-held device key; C = platform-held	Yes
`payload.authorization.approved_at`	Timestamp asserted in the signed payload for the approval	Yes (the *assertion*; not wall-clock truth)
`payload.policy.id` / `policy.hash`	Which policy version governed the decision	Yes

Field / check	What it carries	Re-derivable offline?
`payload.policy.decision` / `enforcement_mode`	allow / allow_with_signoff / deny	Yes
`payload.authorization.consumed_at` / `execution_reference_id`	Server-state: that and when the authorization was spent	**No — server-state**
`challenge_binding` (check)	Signature bound to the exact action bytes; any parameter change invalidates	Yes
`user_verified` (check)	Biometric/PIN passed at the signing moment (WebAuthn UV)	Yes
`user_present` (check)	A human was present at the signing device	Yes
`signature` (check)	Signed by the enrolled device key (Class A: ECDSA P-256) / issuer key (Ed25519)	Yes
`anchor.merkle_root` / `merkle_proof`	Inclusion in the published append-only Merkle anchor	Yes (against a checkpoint)
`replay_attempts` (evidence packet)	Count of detected re-presentation attempts	Operator-reported

Privacy note: sensitive action parameters and per-entity transaction volumes are not required to be disclosed in evidence shared with an auditor. A receipt binds to the **action_hash**; the cleartext action and any volumes are disclosed only at the auditee's discretion, on a per-sample basis. Default sampling should use hashes, not bulk parameter dumps.

CC6.1 — Logical access is restricted to authorized users

> *The entity implements logical access security software, infrastructure, and architectures over protected information assets to protect them from security events to meet the entity's objectives.*

A receipt can serve as evidence that a **specific governed action** was authorized by a **named human** who **proved control of an enrolled credential at the moment of approval** — i.e., evidence about action-level authorization, not session-level access.

Receipt field / check	What it can serve as evidence *for* under CC6.1	Boundary (what it does not establish)
`approver_id`	That a named, enrolled human — not an unattributed process — granted the authorization for this action	Binds a key to an enrolled name; does not by itself prove real-world identity proofing. Pair with the auditee's enrollment/IDM records.
`approver_key_class`	The assurance tier of the approver's credential. Class A = approver-held device key the operator never possesses	Class C is platform-held; it does not evidence approver-sole control of the key
`user_verified` (check)	That biometric or PIN verification was performed at the moment of approval	Verification ≠ absence of coercion; UV proves a factor was satisfied, not the approver's free will

Receipt field / check	What it can serve as evidence *for* under CC6.1	Boundary (what it does not establish)
`user_present` (check)	That a human was present at the signing device, not a headless replay	—
`challenge_binding` (check)	That the authorization is scoped to *this exact action*, not a broad session grant	Tampering with any parameter invalidates the receipt; this is parameter-binding, not policy adequacy
`policy.id` + `policy.hash`	Which access policy version governed this authorization	Binds to a policy reference; does not certify the policy is adequate for CC6.1
`signature` (check)	That the credential that produced the approval was the enrolled one	—

Auditor questions to pair with the receipt: "Show me the enrollment record binding `approver_id` to a real person." · "For actions your policy designates Class A, show me there are no Class C approvals in the sample." · "Show me the policy text at `policy.hash`."

CC6.2 — Access is authorized prior to issuance / before the transaction

> *Prior to issuing system credentials and granting system access … the entity registers and authorizes new internal and external users … access is removed when no longer required. (and the entity authorizes transactions before they take effect)*

A receipt can serve as evidence that authorization existed **before** the action took effect, and that the authorization is **single-purpose** (bound to one action), rather than a standing or after-the-fact grant.

Receipt field / check	What it can serve as evidence *for* under CC6.2	Boundary
`approved_at` (signed)	The timestamp the approval was asserted to occur, against which the auditor correlates the execution record	The receipt asserts a timestamp; it does not by itself prove the system *refused* to execute before approval — pair with the execution/consume log
`payload.action` (full, canonical)	That the approver signed *these exact parameters* — "what you sign is what you get"	Cryptography binds the bytes; it cannot prove the signing surface rendered them faithfully to the human
`enforcement_mode` / `policy.decision`	That the action reached `allow_with_signoff` (required a human) rather than silent allow	Proves the policy outcome recorded; does not prove the decision was wise or lawful
`challenge_binding` (check)	That this authorization cannot be reused for a different action	Single-action binding; *single-use* (one consumption) is server-state — see below
`consumed_at` / `execution_reference_id`	(When disclosed) the link from authorization to the one execution it was spent on	**Server-state, not offline-verifiable.** The receipt cannot prove the nonce was consumed exactly once

The honest boundary for CC6.2: Offline verification proves the authorization is authentic, intact, and bound to the exact action *before* execution if `approved_at` precedes the execution record. It does **not** prove **one-time**

use or **current revocation status** — both are server-state held by the auditee. Ask: "*Show me the consumption record proving this authorization was spent exactly once, and the revocation status of* `approver_id`*'s credential at* `approved_at`*.*"

CC7.2 — The entity monitors system components for anomalies

> *The entity monitors system components and the operation of those components for anomalies that are indicative of malicious acts, natural disasters, and errors … and analyzes them to determine whether they represent security events.*

A receipt is a structured, per-action record. The *population* of receipts can serve as evidence that governed actions are monitored at the level of individual human authorizations, and the fields support anomaly analysis.

Receipt field / mechanism	What it can serve as evidence *for* under CC7.2	Boundary
`receipt_id` + per-action issuance	That each governed action produces a discrete, attributable monitoring record	Completeness depends on the policy correctly designating which actions are governed; absence of a receipt for a sampled governed action is itself the finding
`action_hash` / action type	That actions can be classified for anomaly detection without disclosing parameters	Hash-level monitoring preserves privacy; cleartext is opt-in
`decision` / `enforcement_mode`	That approval vs. denial vs. deny outcomes are recorded and can be trended	Trends are inputs to analysis; the receipt does not itself flag anomalies
`replay_attempts` (evidence packet)	That re-presentation attempts against an authorization were detected and counted	Operator-reported count; corroborate against the auditee's logs
`approver_id`	That authorizations are attributable to named approvers for reviewer-level analysis	—
SIEM ingest fields (canonical JSON, RFC 8785)	That receipts can be forwarded to the auditee's SIEM so every governed action is correlated with its human approval	The SIEM copy is an index; the signed JSON is the evidence — verify signatures before relying on ingested copies

Auditor questions to pair: "For a sampled window, show me the receipt population vs. the governed-action population — are there governed actions with no receipt?" · "Show me how `replay_attempts > 0` is reviewed."

CC7.3 — The entity evaluates security events and the integrity of records

> *The entity evaluates security events to determine whether they could or have resulted in a failure … to meet its objectives (security incidents) and, if so, takes actions to prevent or address such failures. (record/log integrity)*

A receipt's anchoring and signature material can serve as evidence about the **integrity of the authorization record itself** — that the history of authorizations has not been silently rewritten — which supports evaluating whether a security event affected the record.

Receipt field / mechanism	What it can serve as evidence *for* under CC7.3	Boundary
`anchor.merkle_root` + `merkle_proof`	That this receipt is included in the published append-only anchor; the authorization history cannot be silently altered after the fact	Requires checking the proof against an independently obtained checkpoint root; integrity ≠ correctness of the underlying decision
`signature` + `key_id` / `key_class`	Which signing key produced each artifact, enabling detection of unexpected key usage or stale/rotated keys	Surfaces key material for review; rotation/revocation status is server-state
`key_class` distribution across a sample	That the mix of Class A vs. Class C can be reviewed for unexpected shifts (e.g., a surge of platform-held approvals)	A shift is a signal to investigate, not a verdict
Append-only, chain-linked events (`parent_event_hash`)	That the per-receipt lifecycle log (issued → approved → consumed/expired) is tamper-evident	Chain integrity is offline-checkable within the packet; cross-system corroboration is the auditee's
`replay_attempts`	That detected replay activity is available as an input when evaluating a suspected event	—

Auditor questions to pair: "Re-verify the Merkle inclusion proof for this `receipt_id` against your published checkpoint." · "Show me the key-rotation record for any `key_id` flagged in the sample."

Cross-reference: which receipt facts touch which criterion

Receipt fact (all offline re-derivable unless noted)	CC6.1	CC6.2	CC7.2	CC7.3
`approver_id` (named approver)	●	○	○	○
`approver_key_class` (A/C assurance tier)	●		○	○
`action_hash` / exact action binding	●	●	○	
`approved_at` (signed timestamp)		●	○	○
`policy.id` + `policy.hash`	○	○	○	○
`challenge_binding` (check)	●	●	○	
`user_verified` / `user_present` (checks)	●		○	
`anchor.merkle_root` (log integrity)				●
`replay_attempts`			●	○
`consumed_at` (**server-state**)		◐		○

● primary evidence the auditor can re-derive · ○ supporting · ◐ disclosed but server-state, not offline-verifiable · blank = not applicable.

What an authorization receipt does *not* evidence (carry into every workpaper)

1. **One-time consumption.** Offline verification cannot prove a nonce was spent exactly once — audit the auditee's consumption record.
2. **Current revocation status.** A receipt evidences the credential was valid as asserted at `approved_at`; live revocation is server-state.
3. **Policy adequacy.** A receipt binds to a policy reference; it does not certify the policy meets the criterion.
4. **Real-world identity proofing.** A receipt binds a key to an enrolled name; pair with the auditee's identity-proofing records.
5. **Rendering faithfulness.** Cryptography binds the action bytes; it cannot prove the signing surface displayed them honestly to the human.
6. **Absence of coercion.** User-verification proves a factor was satisfied; separation of duties defeats unilateral self-approval, not a coerced approver.
7. **Any outcome or result.** A receipt is evidence about an authorization event, never a claim about audit timelines, savings, or downstream consequences.

Workpaper fields to record per sampled receipt

`receipt_id` · `action_hash` (not cleartext, unless the auditee opts in) · `approver_id` · `approver_key_class` · `approved_at` · `policy.id` + `policy.hash` · `decision` / `enforcement_mode` · each verifier check (`challenge_binding`, `user_verified`, `user_present`, `signature`, `anchor`) with pass/fail · Merkle inclusion result · verifier name + version (e.g. `@emilia-protocol/verify 1.2.0`) · verified-on / verified-by. Then, from the auditee: consumption record and revocation status for the same `receipt_id`.

EMILIA Protocol is composition infrastructure, not a certification body, and issues no scores or reputation numbers. This document maps what an independently re-verifiable artifact can serve as evidence for; it makes no claim about audit results. Apache-2.0.