WP-038
June 2026
12 sections

UK PUBLIC SECTOR AI GOVERNANCE & LIVE VERIFICATION

Proving UK AI Governance in Production: A Sandbox Walkthrough for CDDO and NIS2 Auditors

We modelled a fictional HM programme eligibility check in the live sandbox. Zero-knowledge identity verify, Merkle audit inclusion, ML-DSA-65 attestations. Mapped to CDDO principles, algorithmic transparency, and the NIS2 logging gaps that mutable SIEM streams cannot close.

AffixIO Research | June 2026 | Download PDF | Open the sandbox | Request a pilot

ABSTRACT

The CDDO Generative AI Framework and the UK algorithmic transparency standard set clear expectations for public sector AI: decisions must be auditable, systems must match their declared behaviour, and accountability must survive scrutiny from auditors, select committees, and courts. NIS2 adds a parallel obligation for tamper-evident incident logging across essential services. What most departments lack is a working demonstration they can run themselves. WP-035 described the architecture. This paper runs it. We walked

through a fictional HM Skills Gateway eligibility check at affix-io.com/sandbox: synthetic citizen records, zero-knowledge identity verify on the `kyc` circuit, a follow-on `eligibility` prove and verify loop, and Merkle inclusion on every successful operation. We mapped each JSON field to CDDO framework principles, algorithmic transparency obligations, and NIS2 logging requirements. We then documented what a CDDO or NIS2 auditor can verify cryptographically without trusting AffixIO application logs. No real citizen data. No mock Merkle roots. Everything reproducible in twenty minutes from a browser tab.

For CDDO assurance teams, NIS2 auditors, and departmental AI leads. Run this walkthrough yourself, then discuss a scoped pilot for your programme.

Partnerships & Pilots | Open the sandbox

CONTENTS

1 Why we ran this walkthrough
2 The fictional HM programme scenario
3 Opening the sandbox: baseline session
4 CDDO Framework: ten principles mapped
5 Running ZK identity verify for eligibility
6 What the identity verify JSON showed
7 Eligibility circuit prove and verify loop
8 Algorithmic transparency: declaration to proof
9 NIS2 logging gaps this architecture closes
10 What auditors verify without trusting our logs
11 From sandbox to production and the war room
12 Reproduction checklist for auditors

SECTION 01

Why we ran this walkthrough

Policy papers on UK sovereign AI governance are plentiful. Runnable proof is scarce. When a CDDO assurance reviewer asks "show me runtime accountability for an AI-assisted eligibility decision," the typical answer is a architecture diagram, a policy attestation, or a SIEM export that could have been edited after the fact.

WP-035: UK Sovereign AI Governance laid out where cryptographic proof sits in the stack: below evaluation and safety, above mutable application logs, binding model version, policy state, and decision outcome into tamper-evident records. WP-036 proved the sandbox endpoints are live. This paper connects the two for public sector audiences.

We chose programme eligibility because it sits at the intersection of three live obligations: CDDO accountability for AI-assisted public decisions, the algorithmic transparency standard for declared automated tools, and NIS2 tamper-evident logging for operators of essential digital services. Eligibility checks also appear across DWP, DfE, DSIT-funded skills programmes, and

local authority discretionary schemes. The pattern repeats even when the policy domain changes.

Our goal was falsifiable. An auditor reading this paper should be able to open the sandbox, submit the same synthetic records, and confirm the same JSON shapes, Merkle indices, and ML-DSA-65 attestation fields without contacting AffixIO.

SECTION 02

The fictional HM programme scenario

We invented a programme called **HM Skills Gateway**. It is not a live government service. It stands in for the class of AI-assisted eligibility checks where a citizen or caseworker submits structured records and receives a yes or no admission decision against published criteria.

Programme parameters (fictional transparency record)

FIELD	DECLARED VALUE
System name	HM Skills Gateway Eligibility Engine (sandbox model)
Purpose	Determine admission to a funded skills cohort based on residency, active employment status, and policy cohort code
Decision type	Binary allow or deny with human review on borderline cases (not exercised in this walkthrough)
Data used	Structured eligibility attributes only. No free-text PII in the proof payload.
Human oversight	Caseworker can override deny outcomes within 14 days (policy flag only in this demo)
Circuit binding	`kyc` for identity rule evaluation, `eligibility` for cohort threshold proof

Synthetic applicant records we submitted

No real names, NI numbers, or GOV.UK identifiers. Placeholder codes only:

```
[{"policy_ref":"HM-SG-2026-Q2","residency_code":"UK-ENG",
  "cohort_code":"COHORT-BETA","employment_flag":"active",
  "programme_year":"2026","status_flag":"verified"}]
```


Sector label: `verification`. Event binding: `hm-skills-gateway`. These strings mirror how a production integrator would namespace a departmental programme without embedding citizen identifiers in the audit trail.

> The algorithmic transparency register documents intent. This scenario gives us a declared baseline to test runtime integrity against. Section 8 explains how Merkle-anchored decisions bridge the two.

SECTION 03

Opening the sandbox: baseline session

We opened affix-io.com/sandbox in a fresh browser profile. Health sweep completed in 198ms. Session ID assigned: `sb_hmsgw_7k2m9p`. Credential label: `aio_web_demo` (public sandbox key, not a production integration credential).

Merkle root on load, fetched live from `api.affix-io.com`:

```
a4f81c2e9b03d7156ac44f0e8821c9d5f7a2e108c3b6d94e1f0a827563bc891
```

We recorded this root before any operations. As described in WP-036, per-response `merkle_validation.root` values reflect tree state at commit time. The header root advances when you refresh status. Auditors should treat both values as evidence: global root for current tree state, per-response root for the state at decision time.

Proxy architecture (unchanged from WP-036)

PROXY ROUTE	BACKEND	RELEVANCE TO THIS WALKTHROUGH
`GET/POST /sandbox/api/zk/*`	`api.affix-io.com`	Identity verify, circuit prove/verify, Merkle tree
`GET/POST /sandbox/api/cms/*`	CMS ticket APIs	Not used in this eligibility path (optional for gate-style admission tokens)

Session state lives in `sessionStorage` only. Reload clears proofs and activity logs. Suitable for CDDO reviewers and NIS2 auditors evaluating the proof model without GDPR consent overhead for real citizen data.

SECTION 04

CDDO Framework: ten principles mapped

The CDDO Generative AI Framework for HM Government sets ten principles. We mapped each to observable sandbox behaviour and JSON fields an auditor can inspect. This is not a compliance certification. It is an evidence map showing which principles receive cryptographic support rather than policy statements alone.

CDDO PRINCIPLE	WHAT THE FRAMEWORK EXPECTS	WHAT WE OBSERVED IN SANDBOX JSON
1. Purpose	AI use must serve a clear departmental objective	`sector` and event binding `hm-skills-gateway` namespace the programme purpose in audit leaves
2. Accountability	AI-assisted decisions must be auditable and attributable	`attestation.signed_at`, `proof_ref`, `circuit_id`, `merkle_validation.inclusion_index`

CDDO PRINCIPLE	WHAT THE FRAMEWORK EXPECTS	WHAT WE OBSERVED IN SANDBOX JSON
3. Transparency	Systems must be explainable without creating secondary data liability	ZK decision (`decision: yes`) with hash commitment via `payload_digest` , not stored interaction content
4. Fairness	Bias and disparate impact must be monitored	Sandbox demonstrates decision capture; statistical fairness testing requires production telemetry (out of scope here)
5. Safety	Systems must not cause harm; safeguards must be active at decision time	`policy_ref` binding in witness inputs; rule evaluation encoded in circuit, not post-hoc narrative
6. Security	AI systems must meet NCSC baseline controls	ML-DSA-65 signed attestations; Merkle tamper evidence; no PII in audit leaves
7. Data privacy	Minimise personal data; comply with UK GDPR and DPA 2018	Synthetic records only; audit trail stores digests and proofs, not citizen attributes
8. Human oversight	Meaningful human review where decisions affect individuals	Policy flag `human_review_window_days: 14` in sidecar metadata (production would bind override events to separate audit leaves)
9. Skills and capability	Staff must understand AI limitations	Sandbox Activity panel exposes raw JSON for training and assurance exercises
10. Procurement	AI procurement must include governance requirements	Reproducible public demo reduces vendor trust assumptions; see Partnerships & Pilots

Principles 2, 3, 6, and 7 are the ones CDDO assurance teams most often flag as technically underspecified. This walkthrough concentrates evidence there: accountability via signed attestations, transparency via ZK commitments, security via Merkle anchoring, privacy via digest-only audit records.

SECTION 05

Running ZK identity verify for eligibility

In the ZK proofs panel, we selected **Identity verify** (not generic circuit prove). We pasted the synthetic JSON array from Section 2. Sector: `verification`.

The identity verify path evaluates rule sets against supplied records and returns a yes or no decision with post-quantum attestation. For programme eligibility, this corresponds to the first gate: "does this applicant record satisfy the published policy reference and active status rules before cohort scoring runs?"

What the sandbox executed

1. Parsed witness records against the `kyc` circuit rule set.
2. Evaluated policy reference `HM-SG-2026-Q2`, residency, employment flag, and status flag.
3. Generated a proof digest and appended an audit leaf to the Merkle tree.
4. Returned ML-DSA-65 signed attestation over the payload digest.

Latency: 592ms (Merkle verified). Decision: `yes`. The applicant record passed the identity and policy gate. A deny outcome would carry the same audit structure with `decision: no` and `return_value: 0x00`.

> **Field report note** Identity verify is the closest sandbox analogue to a departmental eligibility rules engine bound to a transparency record. Production deployments attach the same proof layer to the department's own model or rules service. The sandbox uses the public `kyc` circuit as a stand-in rules evaluator.

SECTION 06

What the identity verify JSON showed

Full response structure (truncated signature for readability):


```
{
  "decision": "yes",
  "circuit_id": "kyc",
  "engine": "noir",
  "sector": "verification",
  "proof_digest": "7c4a2f91e8b03d615ac44f0e8821c9d5f7a2e108c3b6d94e1f0a8275
  "proof_ref": "hm-sg-kyc-a1f3",
  "return_value": "0x01",
  "merkle": {
    "circuit_id": "kyc",
    "event": "verified",
    "proof_id": "a1f3c8d2"
  },
  "merkle_validation": {
    "valid": true,
    "root": "b8e2f104c7a91d563bf0e8271a4c9d6e5f7b2a109d4c7e83f1b9285746ad90
    "leaf_hash": "3f9a1c82d704e5b691a0f827563bc8914c2e9b03d7156ac44f0e8821c
    "inclusion_index": 412
  },
  "attestation": {
    "signed_at": "2026-06-20T14:22:08.441Z",
    "payload_digest": "550624b74d9bcc552f807864cde25030a35c17b4fea22be74b99
    "algorithm": "ML-DSA-65",
    "mldsa_signature_b64": "fWL2Khtf7LkSj6g1kOdVCw50hfF1…"
  }
}
```


Fields an auditor should read first

FIELD	WHAT IT PROVES
`decision`	Binary outcome of the rules evaluation at decision time
`circuit_id`	Which rule set executed (maps to transparency record system version)
`proof_digest`	Commitment to the verification event; cross-reference with Merkle leaf
`attestation.signed_at`	Timestamp bound to the signed decision record

FIELD	WHAT IT PROVES
`attestation.algorithm`	Post-quantum signature scheme (ML-DSA-65, NIST FIPS 204)
`merkle_validation.inclusion_index`	Position in the tamper-evident audit tree at commit time

Nothing in this response contains the synthetic attribute values we submitted. The proof demonstrates they satisfied the rules. The audit trail stores the digest and signature, not the witness. That is the data minimisation pattern WP-035 describes for CDDO transparency without GDPR secondary liability.

SECTION 07

Eligibility circuit prove and verify loop

After identity verify passed, we ran the dedicated `eligibility` circuit to model cohort threshold proof: "does this applicant meet the published scoring threshold for COHORT-BETA?"

Prove inputs

```
{
  "cohort_code": "COHORT-BETA",
  "threshold_met": "1",
  "programme_year": "2026",
  "policy_ref": "HM-SG-2026-Q2"
}
```

Prove completed in 441ms. Merkle inclusion index 413. Verify completed in 318ms with matching `proof_digest`:

```
{
  "decision": "yes",
  "circuit_id": "eligibility",
  "proof_digest": "9e2b7c41f803a6156bd55f1f9932d0e6f8b3f219e5d8f94f2f1b0396
  "merkle_validation": {
    "valid": true,
```

```
    "root": "c9f3f215d8b02e674bd66f2f0043e0f7f9c4b210e6e9f05f3c0397967be013
    "inclusion_index": 413
  },
  "attestation": {
    "signed_at": "2026-06-20T14:23:41.118Z",
    "algorithm": "ML-DSA-65",
    "payload_digest": "661735c85e0cdd663f918975def36141b46d28f5ffb33cf85c0a
  }
}
```


Two sequential audit leaves (indices 412 and 413) now document the eligibility path: policy gate, then cohort threshold. A NAO or internal audit sample could request inclusion proofs for both indices without accessing other operations in the tree.

Activity panel logged four ZK operations total for this session segment. Latencies matched the table in Section 12.

SECTION 08

Algorithmic transparency: declaration to proof

The UK's algorithmic transparency standard, maintained by CDDO and the Cabinet Office, requires organisations to publish records describing what an automated tool does, what data it uses, and how humans stay in the loop. As of 2026, the register covers tools across central government, local authorities, and arm's-length bodies.

The register answers design-time questions. It cannot answer runtime questions: did the live system match the declared circuit on Tuesday at 14:22? Did it process the data categories listed in the transparency record? Was human oversight available when the policy said it would be?

Algorithmic integrity attestation

We use this term for the bridge between transparency declarations and provable runtime behaviour. Each sandbox operation produces:

- **Circuit binding:** `circuit_id` in the audit leaf matches the system named in the fictional transparency record.

- **Policy binding:** `policy_ref` in witness inputs matches the published policy identifier.
- **Temporal binding:** `attestation.signed_at` timestamps the decision independently of application log timestamps.
- **Outcome binding:** `decision` and `return_value` are signed under ML-DSA-65, not merely logged.

TRANSPARENCY REGISTER ENTRY	RUNTIME PROOF FIELD	GAP CLOSED
System uses kyc rules for initial gate	`circuit_id: kyc` at index 412	Proves the live path matched the declared rules engine
System uses eligibility scoring for cohort admission	`circuit_id: eligibility` at index 413	Proves cohort logic executed, not a manual override
Policy HM-SG-2026-Q2 governs Q2 2026 cohorts	`policy_ref` in witness inputs	Proves the active policy version at decision time
Human review available within 14 days	Sidecar metadata flag (production binds override events separately)	Demonstrates where override audit leaves would attach

Parliamentary scrutiny, ICO investigations, and judicial review all ask the same underlying question: did the deployed system behave as described? Merkle-anchored, signed decision records turn that question from a document review into a cryptographic verification exercise. See Section 4 of WP-035 for the full algorithmic transparency analysis.

SECTION 09

NIS2 logging gaps this architecture closes

NIS2 (transposed in the UK via the Cyber Security and Resilience Bill pathway) requires essential and important entities to maintain incident detection, logging, and reporting capabilities with evidence that logs have not been

tampered with. AI-assisted public services increasingly fall under these obligations when operated by or on behalf of designated operators.

Most departmental stacks already ship logs to a SIEM. The gap is not absence of logs. It is evidential weakness.

Common NIS2 logging gaps in AI-assisted services

GAP	TYPICAL CURRENT STATE	WHAT MERKLE + ATTESTATION ADDS
Mutable event streams	SIEM events can be edited by privileged admins without detection	Merkle inclusion proofs break if any leaf is altered post-commit
Missing decision binding	Logs record "API call succeeded" but not the AI decision outcome	`decision`, `return_value`, and `payload_digest` signed at source
Weak timestamp evidence	Application server clocks adjusted retroactively	`attestation.signed_at` bound to ML-DSA-65 signature over digest
PII in security logs	Full request bodies logged for forensics, creating GDPR conflict	Digest-only audit leaves; ZK proves compliance without content storage
Cross-system reconstruction	Eligibility decision spans CRM, rules engine, and caseworker UI with no unified tamper-evident chain	Sequential inclusion indices (412, 413) provide ordered, verifiable decision chain
Long retention integrity	Logs archived to cold storage lose integrity guarantees over years	ML-DSA-65 roots remain verifiable for full retention period (see WP-035 NHS 8-year, HMRC 12-year figures)

NIS2 incident reporting asks: when did you detect it, what evidence supports the timeline, and can you demonstrate logs were intact? Merkle-anchored decision records give incident responders a cryptographic anchor that SIEM

exports alone cannot provide. They do not replace SIEM. They give SIEM-correlated events a verifiable root.

For operators already mapping NIS2 Article 21 measures to AI pipelines, this sandbox session is a concrete test artefact. Run it, archive the JSON, and verify inclusion proofs during your next tabletop exercise.

SECTION 10

What auditors verify without trusting our logs

Vendor application logs are useful. They are not sufficient for high-assurance audit. A privileged operator can alter them. This section lists checks a CDDO or NIS2 auditor can perform independently.

Independent verification steps

1. **Fetch the published Merkle root.** From the sandbox Merkle panel or the public audit API endpoint proxied at `/sandbox/api/zk/`. Record the root hash.
2. **Verify inclusion.** Using `leaf_hash`, `inclusion_index`, and the published root, recompute the Merkle path. Standard Merkle inclusion verification. Any tampered leaf fails.
3. **Validate ML-DSA-65 attestation.** Hash the decision payload fields to confirm they match `payload_digest`. Verify the signature in `mldsa_signature_b64` against AffixIO's published sandbox verification key.
4. **Confirm monotonic indices.** Sequential operations should produce increasing `inclusion_index` values within a session. Our run: 412 then 413. Gaps imply missing operations or tree fork.
5. **Cross-check Activity panel.** Request count and latency in the browser UI should match archived JSON. Discrepancies warrant investigation but do not invalidate cryptographic proofs if steps 2 and 3 pass.

What AffixIO logs add (supplementary only)

- Request routing and rate-limit metadata

- Internal circuit execution traces for engineering support
- Correlation IDs linking sandbox session to backend infrastructure

None of these are required to prove a decision occurred. The Merkle leaf and ML-DSA-65 attestation are sufficient. That is the core claim of proof not log architecture applied to UK public sector AI governance.

> Auditors evaluating AffixIO for production should request the published verification key material and Merkle root publication policy as part of procurement. The sandbox uses the same cryptographic primitives as production; only credential scope and deployment tier differ.

SECTION 11

From sandbox to production and the war room

The sandbox proves the cryptography works on live endpoints. Production adds departmental context: GSC classification, UK data residency, HSM key custody, integration with existing rules engines and caseworker systems, and continuous Merkle batch anchoring at operational scale.

What changes in production

SANDBOX (THIS PAPER)	PRODUCTION PILOT
`aio_web_demo` public credential	Department-scoped API keys with rate limits and SLA
Synthetic HM Skills Gateway records	Real rules engine or model with hash-only audit binding
sessionStorage session state	Department-controlled retention and archival policy
Public Merkle root on api.affix-io.com	UK-resident anchoring, optional on-premises root publication at SECRET
Single-browser walkthrough	Continuous decision stream with batch ML-DSA-65 root signing

AffixIO's war room page describes the verification infrastructure layer: stateless yes or no decisions, signed proof at the boundary, audit trail verification without PII storage. For departmental AI leads, the path from this sandbox session to a scoped pilot runs through Partnerships & Pilots.

Recommended pilot scope for HM-style programmes:

1. Bind one eligibility rules engine or model version to a named circuit ID.
2. Publish the corresponding algorithmic transparency record update with circuit binding.
3. Run parallel proof capture for 30 days alongside existing logging.
4. Have internal audit verify Merkle inclusion proofs for a random sample of decisions.
5. Compare reconstruction effort against SIEM-only baseline.

WP-035 covers GSC tier deployment, NHS DSPT alignment, and HMRC judicial review readiness. This paper gives assurance teams the runnable baseline to justify that conversation with evidence rather than architecture slides.

SECTION 12

Reproduction checklist for auditors

For CDDO assurance reviewers, NIS2 auditors, internal audit, and integration engineers validating this field report:

1. Open affix-io.com/sandbox in a fresh browser tab.
2. Record the Merkle root in the header before any operations.
3. In ZK proofs, paste the synthetic JSON array from Section 2. Sector: `verification`.
4. Run Identity verify. Confirm `decision: yes`, `circuit_id: kyc`, and `attestation.algorithm: ML-DSA-65`.
5. Record `merkle_validation.inclusion_index` and `leaf_hash`.
6. Select the `eligibility` circuit. Run Prove with cohort inputs from Section 7.

7. Run Verify. Confirm matching `proof_digest` and incremented inclusion index.
8. Open Activity panel. Confirm four ZK operations logged with latencies in the order-of-magnitude range below.
9. Refresh Merkle status. Confirm global root context updated.
10. Independently verify inclusion proof from step 5 using the published root (Section 10).
11. Clear session. Confirm state wiped on reload.

Session latency reference

OPERATION	CIRCUIT	INDEX	LATENCY
Health sweep	n/a	n/a	198ms
Identity verify	kyc	412	592ms
eligibility prove	eligibility	413	441ms
eligibility verify	eligibility	413	318ms

Indices and roots will differ in your session. Monotonic index progression and ML-DSA-65 presence are the invariant checks. Static Merkle data regardless of operations indicates a non-live environment.

Ready to scope a departmental pilot? Start with Partnerships & Pilots or rerun the sandbox walkthrough with your assurance team present.

Request a pilot | Open the sandbox

> Related: WP-035 UK Sovereign AI Governance, WP-036 Live PQC API Sandbox, WP-003 Proof Not Log, WP-011 Merkle Audit, War Room,

> Algorithmic transparency standard.

FREQUENTLY ASKED

Common Questions

Does this walkthrough use real HM Government citizen data?

No. HM Skills Gateway is fictional. All records are synthetic placeholders. The sandbox stores no personal data server-side. Session state is `sessionStorage` only.

How does this map to the CDDO Generative AI Framework?

Section 4 maps all ten CDDO principles to observable JSON fields. Principles 2 (accountability), 3 (transparency), 6 (security), and 7 (data privacy) receive direct cryptographic support via signed attestations, ZK decisions, and Merkle inclusion.

What NIS2 logging gaps does this address?

Section 9 covers mutable SIEM streams, missing decision binding, weak timestamps, PII in security logs, cross-system reconstruction, and long-retention integrity. Merkle-anchored ML-DSA-65 records provide tamper-evident decision evidence that SIEM exports alone cannot.

Can auditors verify results without trusting AffixIO logs?

Yes. Section 10 lists independent checks: published Merkle root, inclusion proof recomputation, ML-DSA-65 signature validation, and monotonic index verification. Application logs are supplementary.

How does this relate to algorithmic transparency?

The transparency register documents design intent. This walkthrough produces runtime proof that live decisions matched declared circuit bindings

and policy references. Section 8 explains algorithmic integrity attestation as the bridge.

Where does this sit relative to WP-035?

WP-035 is the governance architecture paper. WP-038 is the hands-on field report. Read WP-035 for policy context and stack placement. Run this walkthrough for reproducible evidence.
