WP-039
June 2026
10 sections

PARTNERSHIPS, PILOTS & ELIGIBILITY VERIFICATION

Scan-to-Prove Partnerships: Verifying Eligibility Without Exporting Source Data

Partnership pages should not ask pilot applicants to email spreadsheets of customer records. This paper documents `affix-scan-prove.js`: one URL parameter, a yesno circuit verify, Merkle inclusion on `proof_digest`, and shareable `pr` / `mr` / `sid` references. Written for integrators and pilot teams wiring the flow today.

AffixIO Research | June 2026 | Try scan-to-prove live | Contact partnerships

ABSTRACT

Pilot programmes fail on data friction long before they fail on cryptography. A venue group wants to prove it meets AffixIO partnership criteria. A fintech wants to show eligibility for a co-marketed pilot. Neither should upload source CRM exports to a third-party form. AffixIO's scan-to-prove pattern attaches a 124-line browser script to any partnership landing page. A visitor arrives with `?affix_scan=1`. The script calls `/api/demo/circuit-verify` on the `yesno` circuit, fetches Merkle inclusion for the returned `proof_digest`, confirms it with `/v1/merkle/verify-proof`, strips the trigger parameter from the URL, and writes `pr`, `mr`, and `sid` query keys plus a

`sessionStorage` record at `affix_scan_proof`. Source data never leaves the integrator's control plane. This field report walks the full execution path in affix-scan-prove.js, the configuration surface, failure modes, and a reproduction checklist you can run in under ten minutes at partnerships-and-pilots/?affix_scan=1.

CONTENTS

SECTION 01

Why partnership pages need scan-to-prove

Partnership intake forms collect names, company size, use-case essays, and often sample data files. Legal reviews those uploads. Security teams flag the export. The pilot stalls for weeks while someone redacts columns.

Scan-to-prove inverts the ask. Instead of "send us your records," the page says "prove you meet the policy, here is a reference we can audit." The visitor's browser runs a zero-knowledge circuit verify against public AffixIO infrastructure. The output is a `proof_digest` anchored in the live Merkle tree, not a zip of customer rows.

We built this for the Partnerships & Pilots programme first. The same pattern applies to venue operators testing anti-scalping tickets, regulated firms

evaluating PII-free KYC, and any pilot where eligibility is yes/no but the underlying evidence must stay local. Integrators embed one script tag. Pilot teams test with a single URL parameter. AffixIO receives auditable proof references, not source payloads.

> Live demo: open partnerships-and-pilots/?affix_scan=1 and watch the URL rewrite after the prove cycle completes. Questions on production keys or custom circuits: contact/partnerships.

SECTION 02

What this is not

Scope boundaries keep integrators from over-promising to their own pilot applicants:

- **Not a full KYC upload.** The default `yesno` circuit uses three synthetic `condition_greater` fields. It demonstrates the prove path and Merkle anchoring. Production eligibility circuits (for example `kyc`, `eligibility`, sector-specific composites) are configured per partnership agreement.
- **Not silent background tracking.** The script does nothing unless `affix_scan` is present in the query string. Normal page visits incur zero API calls.
- **Not server-side session storage.** Proof state lives in the browser tab via `sessionStorage` and URL parameters. Reload in a new tab without `affix_scan` does not re-prove unless you trigger again.
- **Not a replacement for contract review.** Cryptographic proof of circuit satisfaction complements legal due diligence. It does not sign an MSA.

What it *is*: the smallest integration surface we could ship for "show me you can verify eligibility without exporting source data," observable in production JSON without a sales call.

SECTION 03

The affix_scan URL parameter

The trigger is a query-string key, default name `affix_scan`. Value is irrelevant: `?affix_scan=1`, `?affix_scan=true`, and `?affix_scan` (empty) all satisfy `params.has(cfg.scanParam)` in the script.

Early exit is explicit. Lines 13 and 14 of `affix-scan-prove.js` parse `window.location.search` and return immediately if the parameter is absent. No fetch, no DOM mutation, no console noise beyond normal page load.

Optional scan_id

Integrators may pass `scan_id` to correlate a prove run with their CRM or pilot ticket:

```
https://www.affix-io.com/partnerships-and-pilots/?affix_scan=1&scan_id=pilo
```

If omitted, the script generates one: `Date.now().toString(36)` concatenated with a random base-36 suffix. That value becomes the `sid` URL parameter after prove and is included in the sector label sent to the API.

Lifecycle

1. User lands with `affix_scan` present.
2. Script runs prove, verify, and Merkle check asynchronously.
3. On completion, `affix_scan` is deleted from the URL via `history.replaceState`.
4. `pr`, `mr`, and `sid` are appended. The address bar becomes shareable proof metadata without re-triggering the script on refresh.

This design lets marketing emails link to `?affix_scan=1` once, then lets the recipient forward a clean URL with proof references to an internal reviewer.

SECTION 04

Script configuration and load model

The partnerships page loads the script at the bottom of the document:

```
<script src="/partnerships-and-pilots/affix-scan-prove.js"
        data-affix-scan-prove defer></script>
```

Configuration is read from `data-*` attributes on the script element, with production-safe defaults:

DATASET KEY	DEFAULT	PURPOSE
`data-api-base`	`https://api.affix-io.com`	API host, trailing slash stripped
`data-api-key`	`aio_web_demo`	`X-API-Key` header on every request
`data-circuit-id`	`yesno`	Circuit passed to circuit-verify
`data-sector`	`partnerships_pilot`	Base sector label; suffixed with `:{scan_id}`
`data-scan-param`	`affix_scan`	Query key that triggers execution

The script resolves itself via `document.querySelector('script[data-affix-scan-prove]')` or `document.currentScript`, so defer loading is safe.

apiFetch helper

All HTTP calls go through a thin wrapper that sets `Content-Type: application/json`, attaches the API key, parses JSON responses, and throws on non-OK status with the server message when available. Integrators debugging pilot failures should watch the network tab for three sequential calls after trigger (circuit verify, Merkle proof fetch, Merkle verify-proof).

SECTION 05

yesno circuit verify on page load

The core operation is `proveVerifyAndMerkle()`. Step one posts to `/api/demo/circuit-verify`:

```
POST https://api.affix-io.com/api/demo/circuit-verify
X-API-Key: aio_web_demo
Content-Type: application/json

{
  "circuit_id": "yesno",
  "fields": {
    "condition_greater1": "1",
    "condition_greater2": "1",
    "condition_greater3": "1"
  },
  "sector": "partnerships_pilot:m7k2x9abc4f",
  "requestAttestation": false
}
```

The `yesno` circuit is the same minimal loop documented in WP-036 Live PQC API Sandbox. All three conditions set to `1` satisfy the greater-than checks. The sector string scopes the audit leaf to the partnerships pilot namespace and the specific scan instance.

`requestAttestation: false` skips ML-DSA-65 attestation on this path, keeping latency down for a lightweight page-load prove. Production partnership flows may set attestation true where signed outcomes are required; see WP-002 Post-Quantum Attestation.

Invalid circuit response

If `verify.valid` is false, the function returns early with `ok: false`, `stage: 'circuit'`, and the raw verify payload. Merkle steps are skipped. `attachProof` still runs, setting `data-affix-proof="failed"` on `document.documentElement`.

Valid circuit response

On success, the script reads `verify.proof_digest` and `verify.proof_ref`. These become the anchor for Merkle inclusion and the shareable `pr` URL

parameter respectively.

SECTION 06

proof_digest and proof_ref

Two fields do different jobs. Confusing them breaks audit trails.

FIELD	ROLE	WHERE IT APPEARS AFTER PROVE
`proof_digest`	32-byte hash identity of the proof outcome; Merkle leaf key	`sessionStorage` `affix_scan_proof` ; used in `GET /v1/merkle/proof/{digest}`
`proof_ref`	Short human-portable reference for support and CRM correlation	URL param `pr` ; `sessionStorage` ; `affix:scan-proved` event detail

The digest is what auditors verify against the published Merkle root. The ref is what a partnerships manager types into a ticket when an applicant says "I completed scan-to-prove yesterday."

In our test session against the live partnerships page, circuit verify completed in roughly 400 to 500ms. Exact latency varies with region and load. The script records round-trip time in milliseconds and exposes it as `data-affix-scan-ms` on the root element and in `affix_scan_proof.ms` .

> **Integrator rule:** persist `proof_digest` for cryptographic audit. Display `proof_ref` (`pr`) to end users. Never treat `pr` alone as a Merkle lookup key.

SECTION 07

Merkle inclusion and verify-proof

Circuit validity alone does not prove the operation entered AffixIO's audit tree. The script performs two Merkle steps, matching the architecture in WP-011 Merkle Tree Audit Architecture.

Step 1: fetch inclusion material

```
GET https://api.affix-io.com/v1/merkle/proof/{proof_digest}
```

Response includes `root`, `proof` (sibling hash array), and `leaf` with `leaf_hash`. This is the inclusion proof for the digest returned by circuit verify.

Step 2: verify locally via API

```
POST https://api.affix-io.com/v1/merkle/verify-proof
Content-Type: application/json

{
  "leaf_hash": "<from inclusion.leaf.leaf_hash>",
  "root": "<from inclusion.root>",
  "proof": [<sibling hashes>]
}
```

The script sets `ok: true` only when `verify.valid` from the circuit step and `merkleCheck.valid === true` from this step both hold. Partial success (circuit yes, Merkle no) surfaces as `data-affix-proof="failed"`.

Why both steps

Fetching the proof demonstrates the leaf exists at query time. Calling verify-proof confirms the sibling path recomputes to the published root. Partnership reviewers can independently repeat verify-proof with the same three fields without access to the applicant's source systems.

The returned `inclusion.root` becomes `merkle_root` in the result object. The first 16 hexadecimal characters are written to the `mr` URL parameter for compact sharing. Full root remains in `sessionStorage`.

SECTION 08

URL rewrite: pr, mr, sid

Function `attachProof(result, ms)` runs after prove completes. It mutates browser state in four places: URL, `documentElement` datasets, `sessionStorage`, and a custom event.

URL parameters

PARAM	SOURCE	EXAMPLE
`pr`	`verify.proof_ref`	Short proof reference string from API
`mr`	`inclusion.root.slice(0, 16)`	First 16 hex chars of Merkle root
`sid`	`scan_id` query param or generated ID	`pilot-acme-2026-q2` or `m7k2x9abc4f`

Before rewrite:

```
https://www.affix-io.com/partnerships-and-pilots/?affix_scan=1&scan_id=pilo
```

After rewrite (illustrative):

```
https://www.affix-io.com/partnerships-and-pilots/?pr=a1b2c3d4&mr=89c0225ccj
```

`affix_scan` is always removed. Refreshing this URL does not re-run prove because the trigger parameter is gone. To force a new prove cycle, append `affix_scan` again (optionally with a fresh `scan_id`).

DOM datasets

- `data-affix-proof`: `verified`, `failed`, or `error` (on uncaught exception)
- `data-affix-scan-ms`: round-trip latency in milliseconds
- `data-affix-scan-id`: the scan_id used for this run

CSS can target `html[data-affix-proof="verified"]` to show a confirmation banner without JavaScript framework dependencies.

SECTION 09

sessionStorage and DOM hooks

Key `affix_scan_proof` stores a JSON serialisation of the outcome:

```
{
  "ok": true,
  "proof_ref": "a1b2c3d4",
  "merkle_root": "89c0225ccf7d1b9ea0a396327877fdd02037bd19b756b47e3d225b3f3
  "proof_digest": "4e18200fd99be31d0c1e9329d692ee756e30c59137df03b1762f7db2
  "scan_id": "pilot-acme-2026-q2",
  "ms": 487,
  "at": "2026-06-20T14:32:01.204Z"
}
```

Storage is wrapped in try/catch for private browsing modes where `sessionStorage` may throw. In those cases URL parameters and DOM datasets remain the fallback read path.

affix:scan-proved event

The script dispatches:

```
window.dispatchEvent(new CustomEvent('affix:scan-proved', { detail: result
```

The `detail` object includes `ok`, `verify`, `inclusion`, `merkleCheck`, `proof_ref`, `merkle_root`, `proof_digest`, and `scan_id`. React, Vue, or vanilla integrators can listen once and update UI state without polling `sessionStorage`.

Privacy model

Nothing in `affix_scan_proof` contains applicant PII. It holds cryptographic references and timing metadata only. Tab close clears `sessionStorage`. This aligns with the proof-not-log posture in WP-003 Proof Not Log: the Merkle

tree records that a verification occurred; it does not need exported source rows.

Error path

Network failures and API errors hit the `.catch` handler. `data-affix-proof` becomes `error`. A console warning logs the message. No URL rewrite occurs on hard failure. Integrators should surface a retry link with `?affix_scan=1`.

SECTION 10

Integration checklist for pilot teams

For engineering leads, security reviewers, and partnerships managers validating this paper without AffixIO on a call:

1. Open a fresh browser tab (or incognito) to partnerships-and-pilots/?affix_scan=1.
2. Open DevTools Network. Confirm three requests: `circuit-verify`, `merkle/proof/{digest}`, `merkle/verify-proof`.
3. Confirm `affix_scan` disappears from the address bar after completion.
4. Confirm `pr`, `mr`, and `sid` appear in the URL.
5. Run `JSON.parse(sessionStorage.getItem('affix_scan_proof'))` in the console. Confirm `ok: true` and matching `proof_digest`.
6. Inspect `document.documentElement.dataset.affixProof` equals `verified`.
7. Repeat with `&scan_id=your-pilot-test-001`. Confirm `sid` matches your value.
8. Load the page without `affix_scan`. Confirm zero API calls to `api.affix-io.com`.
9. Compare yesno behaviour with Section 14 of WP-036 in the sandbox ZK panel.
10. For production partnership keys, custom circuits, or SLA terms: contact/partnerships.

To embed on your own pilot landing page, copy the script tag from the partnerships page, set `data-sector` to your programme slug, and link applicants to `https://your-domain.example/pilot?affix_scan=1`. Use `affix:scan-proved` or read `pr` from the URL to gate the next onboarding step.

Production considerations

- Replace `aio_web_demo` with your issued API key via `data-api-key`.
- Point `data-circuit-id` at your agreed eligibility circuit when moving beyond the default yesno demo.
- Log `proof_digest` server-side only after the applicant consents to share audit references, not before.
- Do not embed witness data or PII in `scan_id`; it appears in URL and sector labels.

> Related: WP-036 Live PQC Sandbox, WP-006 PII-Free KYC, WP-011 Merkle Audit, WP-003 Proof Not Log, WP-030 Stateless PQ ZK.

FREQUENTLY ASKED

Common Questions

What does the affix_scan URL parameter do?

It is the sole trigger for `affix-scan-prove.js`. Present in the query string: the script runs circuit verify, Merkle inclusion, and URL rewrite. Absent: the script exits immediately with no API traffic.

Does scan-to-prove export source records to AffixIO?

No. The default flow sends `circuit_id`, three synthetic condition fields, and a sector string. No CRM exports, identity documents, or holder PII are transmitted.

What is stored in sessionStorage affix_scan_proof?

A JSON object with `ok`, `proof_ref`, `merkle_root`, `proof_digest`, `scan_id`, `ms`, and `at`. It lasts for the browser tab session only.

What do pr, mr, and sid mean after prove?

`pr` is `proof_ref` for support correlation. `mr` is the first 16 hex characters of the Merkle root. `sid` is the scan identifier from the `scan_id` parameter or auto-generated. `affix_scan` is removed from the URL.

Which API endpoints does the script call?

`POST /api/demo/circuit-verify`, `GET /v1/merkle/proof/{proof_digest}`, and `POST /v1/merkle/verify-proof`. All use `X-API-Key` (default `aio_web_demo`).

How do pilot teams test without a full integration?

Visit partnerships-and-pilots/?affix_scan=1, inspect network calls, URL params, and `sessionStorage`. Cross-check yesno circuit behaviour in the sandbox. For partnership enquiries: contact/partnerships.

WP-036 | WP-006 | WP-011

▶ About

▶ Solutions

▶ Legal

▶ Trust & Security

Contact

truth layer | yes | no | proof