ACAS-1

Operator Manual — Deployers

The emitting side: for operators deploying agents — Edition 1

Companion to	ACAS-1 Agent Compliance Accountability Standard v0.1 (acas-1.org)
Companion volume	ACAS-1 Operator Manual — Compliance Officers & MLROs, Edition 1 — a separate document covering the receiving side
Published by	Kadikoy Limited, Bermuda (Reg. 202302362)
Date	1 August 2026
Status	Draft — open for public comment
License	Creative Commons CC0 — no rights reserved

WHAT THIS MANUAL COVERS

The two sides	The Compliance Officers & MLROs manual is for the party *receiving* a Presentation and deciding whether to rely on it. This manual is for the party *emitting* one — the operator whose agent carries a Compliance Record and must satisfy a counterparty who did not have to take its word for anything.
Assumed position	You deploy or operate one or more agents. You hold, or your sponsor holds, a regulatory licence or is otherwise the accountable legal person. You want counterparties to clear your agents quickly and without a bilateral negotiation each time.
What you get from conforming	Your completed diligence becomes checkable rather than assertable. Your monitoring becomes a demonstrable commitment rather than a policy claim. Your counterparty's onboarding of you compresses from weeks to minutes — which is the commercial return on all of this.
What conforming costs you	Real execution cost, real integration work, and a public commitment to a cadence you must actually meet. §3 and §7 are honest about all three.

This manual is not normative. Where it differs from the specification, the specification governs.

Four terms that are new

The standard borrows most of its vocabulary from compliance practice as it stands — *disposition*, *hit*, *materiality*, *currency* and *determination* mean here what they mean in your existing programme. Four terms are genuinely new, because there was no existing word for the thing:

Term	What it means
Compliance Step	The evidentiary record of one due-diligence act. You already perform the acts; the Step is the signed, timestamped record that falls out of one
Carriage class	How far a given result can usefully travel to a counterparty. Some results are worth carrying, some must be re-run, some cannot travel at all
Detection latency	The interval between a fact becoming true in the world and your programme acting on it. To *declare a latency* is to commit publicly to a maximum for that interval
Presentation	The signed subset of a subject's record actually handed to one counterparty, for one purpose, for a stated period

1. What you are actually building

Four capabilities. Nothing else in the standard matters until these work.

Emission. Every due-diligence act your systems perform must produce a Compliance Step: signed, timestamped, carrying its carriage class and its determination level, chained to what came before. If your screening already runs, this is an adapter, not a rebuild.

A programme you can prove. A declared cadence, matching methodology and disposition rule, backed by an evidence chain that a stranger can recompute. This is the part with no conventional equivalent and it is where most of your integration effort will go.

Mandate management. A grantor must be able to authorise disclosure narrowly, and revoke it in a way a third party can independently see. If revocation only works when you cooperate, you are not conforming.

Presentation construction. On request, assemble a signed subset for one named counterparty, for one stated purpose, valid for a stated window, within the mandate — and log it.

An operator that can do these four things is conforming. Everything else in this manual is about doing them well enough to survive review.

2. Sequence of work

A realistic order, because doing these out of sequence wastes effort.

First, identity. Every subject, performer and determiner is an AIS-1 identity, and your signing keys are the verification methods published in it. A natural person — your MLRO, an independent verifier — holds one as an AIS-1 Sponsor Card with `entity_type: individual`, so a responsible person and a company are addressed the same way. Nothing else can be built until this is real. If your agents are not yet bonded, that is the prerequisite project.

Second, emission of the easy Steps. `identity_verification` and `purpose_and_nature` are portable-class, low-sensitivity, and usually already performed somewhere in your stack. Emit those first. They will surface every plumbing problem — canonicalisation, key handling, timestamping — on the least sensitive material you have.

Third, screening as a freshness signal. Lower stakes than it appears, because you are not asking anyone to rely on the result. You are declaring your list sets, your matching, and how recently you ran. Get the declaration honest before you get the cadence aggressive.

Fourth, the monitoring programme. This is the substantial build: continuous execution, the evidence chain, the performance figures, the subscription endpoint. Do not start here, however tempting, because it depends on emission working reliably.

Fifth, mandates and Presentations. Straightforward once the Steps exist, and genuinely dangerous if built before you have thought about who may authorise what. See §5.

Last, referenced disclosure. The `source_of_funds` endpoint and its authorisation flow. Deferrable — a Record incomplete for enhanced-diligence profiles is still useful for baseline ones.

---

3. Declaring a programme you can hold

Your programme declaration is a public commitment that a counterparty will test. Four fields carry it.

Field	What you are committing to
`cadence`	When the programme executes. `on_list_update` is the strong claim
`maxLatency`	The maximum interval between a fact arising and your programme acting on it
`matching`	Method, threshold, provider. **Required** — a cadence claim without it is unevidenced
`disposition`	What happens when screening produces a **hit** — a possible match against a list entry, in the ordinary screening sense. `human_required_on_hit` is the normal value

Declare a latency you can hold under load, not on a good day. `missedWindows` is a required field precisely so a gap is visible, and a counterparty recomputing your chain will find misses whether or not you report them. A four-hour commitment held for ninety days is a far stronger position than a one-hour commitment missed eleven times.

Set `maxLatency` from your worst dependency, not your average. If your list provider publishes on an unpredictable schedule, or your screening API rate-limits at volume, that is your constraint. Measure before you declare.

Do not declare a cadence tighter than your matching justifies. Screening hourly against a stale list with naive exact matching is a worse control than screening daily with tuned fuzzy matching. Competent counterparties read the two together, and a high cadence beside an absent `matching` object reads as theatre rather than diligence.

Change your programme on a schedule, never on an event. An abrupt change in a published programme is an inference channel — see §6.

---

4. Making cadence provable

The claim in §7.2 of the specification is that your evidence is *portable*, not merely that it exists. That distinction is the whole product, and it is easy to build something that fails it.

Emit per execution, not per period. A daily summary record is not an evidence chain. Every execution emits its own Step-level evidence entry, signed and timestamped, chained by `prevHash` to its predecessor.

Use an independent timestamp. A timestamp you generate is a timestamp you could have generated later. RFC 3161, a transparency log, or a ledger anchor — the specific mechanism is yours, but the authority must not be you.

Anchor the chain head periodically. Hash chaining detects omission, reordering and substitution *within* a chain. It does not detect wholesale reconstruction of the entire chain. Periodic anchoring of the head to an independent notary or public ledger closes that gap, and §15.2 tells verifiers to expect it.

Publish `expectedExecutions` alongside `executions`. A reviewer's first question is whether the programme silently stopped. Answer it before it is asked. If the numbers diverge, say why in the operator notes rather than letting a counterparty discover it.

Assume the chain will be recomputed. The companion manual instructs recipients to verify the chain rather than read your summary. Build as though every counterparty does, because the ones that matter will.

---

5. Mandates: the part to get right first time

The disclosure mandate is where an operator can do real harm to the subject, and where a mistake is hardest to unwind. Five properties are required of any mandate (§9.1). Three of them are where implementations fail.

Enumerated scope, no wildcards. The schema rejects `*`, `all` and `any` in `stepTypes`, `audiences` and `purposes`. Resist the temptation to work around this with a class identifier so broad it functions as a wildcard. `class:any-counterparty` is a wildcard wearing a coat.

Stated purpose. This is the property most often absent when operators map an existing authorisation scheme onto the mandate, because most capability formats carry scope but not purpose. If you are profiling OAuth, a verifiable credential, or an internal grant system, purpose is probably the field you will have to add.

Independently resolvable revocation. A verifier must be able to check status *without your cooperation*. A status endpoint you can decline to answer, or that requires the grantee's token, does not satisfy this. This is the second property that usually fails when profiling an existing scheme.

Three operational rules on top:

- **Revocation is prospective only.** It stops further Presentations; it cannot recall issued ones. The Presentation `expiresAt` window is the outer bound of exposure a revocation cannot reach. Set it short — days, not months — and let counterparties re-request.
- **The grantor signs, not the agent.** A mandate signed by the grantee is circular and is not a grant.
- **Log every Presentation to the subject, not only to yourself.** The subject's protection is that they can see what went where. If that log is only visible to you, you have built surveillance rather than consent.

---

6. What you must never emit

These are conformance failures, not style preferences.

Never declare auto-clear on a reserved determination. The `disposition` enum has no auto-clear value. If your system would clear its own hit where a profile reserves that determination, you are outside the standard, and the schema will not let you say otherwise.

Never emit anything from which a suspicious transaction report could be inferred. Three channels, all of which an operator can create by accident:

- *Persistent outcome states.* A `review` outcome that never resolves is a signal. Resolve it or let the Step lapse.
- *Silent absence.* A Step that disappears from a Presentation it previously appeared in is a signal. Withdrawal must be indistinguishable from expiry or a scope change — which in practice means scope changes should not be rare events.
- *Cadence anomalies.* A programme that changes abruptly is a signal. Change on a schedule.

Never define an extension type for suspicion, escalation to an FIU, or internal report status. Not under any namespace, not for internal use, not with access controls.

Never claim Proved carriage. The class name and enum value are reserved (§5.4) and unspecified in v0.1.

Never emit a Step without `validUntil`. It is required, and a Step without an expiry makes a claim of permanence that no due-diligence result supports.

---

7. Cost, and being honest about it

Continuous execution costs money that periodic execution does not. Three things bear on whether your commitment is credible.

Negotiate on the right unit. Conventional screening is priced per customer per period, on a book of known size. Continuous execution inverts that into many small machine-initiated queries. Priced per query, high cadence is expensive; priced per list delta evaluated, it usually is not, because an update touching none of your monitored subjects is nearly free to evaluate. Push for the second.

Sort out how the agent pays before you declare the cadence. Metered screening requires either a funded operator account or, machine-to-machine, the agent's own settlement capability under a bounded, revocable spending authority. This is a settlement and delegated-authority problem rather than an ACAS-1 one, but an undeclared dependency here is what causes `missedWindows` to appear in month three.

Verify provider interfaces before committing. You can only screen at machine cadence against providers exposing an API, CLI or Model Context Protocol server. Coverage across the major list-set providers is uneven and changing. Check yours, at your volume, before publishing a `maxLatency`.

---

8. Pre-publication checklist

Before a counterparty sees your first Presentation.

Identity and keys Agents bonded under AIS-1 · signing keys in appropriate custody · determiner identities resolvable for any human sign-off

Emission Every Step carries `carriage`, `validUntil` and `aas1RecordRef` · canonicalisation is JCS and produces identical bytes on an independent implementation · signatures verify against the published verification method

Programme `matching` populated, not defaulted · `maxLatency` measured rather than aspirational · `disposition` is not auto-clear · `expectedExecutions` computed from the actual publication schedule · chain head anchored independently

Mandates No wildcards, including disguised ones · purpose enumerated · status endpoint resolves without your cooperation · grantor signature, not grantee · subject can see the Presentation log

Presentations `audienceRef` set and enforced · `expiresAt` short · mandate status checked at construction · every issue emits a record

Hygiene No field from which an STR could be inferred · no Proved carriage claimed · no extension type without a declared carriage class

Documentation Published policy URIs resolve · your programme's change schedule is stated · your notes explain any divergence between `executions` and `expectedExecutions`

---

9. How your Presentation will be tested

Worth knowing, because it tells you where to spend effort.

They will check the envelope before the contents — audience, window, mandate status — and stop at the first failure. They will verify each Step's signature, evidence and currency. They will read `carriage` before `result`, so your screening results will not be relied on and your monitoring subscription will be. They will recompute your evidence chain rather than reading your performance summary. They will check `missedWindows` and the gap between `executions` and `expectedExecutions`. And they will apply their own minimum determination levels, which may be higher than yours.

None of that is adversarial. It is what makes the Presentation worth anything — a claim nobody can check is a claim nobody values. Build for the reviewer who verifies everything, and the ones who verify less will be satisfied automatically.

---

10. A transaction, end to end

A single worked example, from onboarding through clearing to a status change a month later. Identifiers are illustrative. The point of the example is the last part: what the standard buys is not a faster onboarding but a shorter interval before a problem is known.

10.1 Before the transaction

Your agent is `did:ais1:base:agent-047`; its sponsor is a licensed payment institution. Over the ninety days before any counterparty asks, your systems have emitted four Steps.

When	Step	Carriage	Determination
Day −90	`identity_verification`	portable	independently determined by a regulated verifier
Day −88	`beneficial_ownership` of the sponsor entity	portable + referenced	prepared by the agent, then determined by your MLRO
Day −88	`purpose_and_nature`	portable	agent determined
Day −90 onward	`ongoing_monitoring`	subscription	agent determined; programme declared at `maxLatency` PT4H

Nothing has been sent anywhere. The Steps sit in the Compliance Record, and the monitoring programme runs, emitting a signed record on every list publication and chaining each to the last.

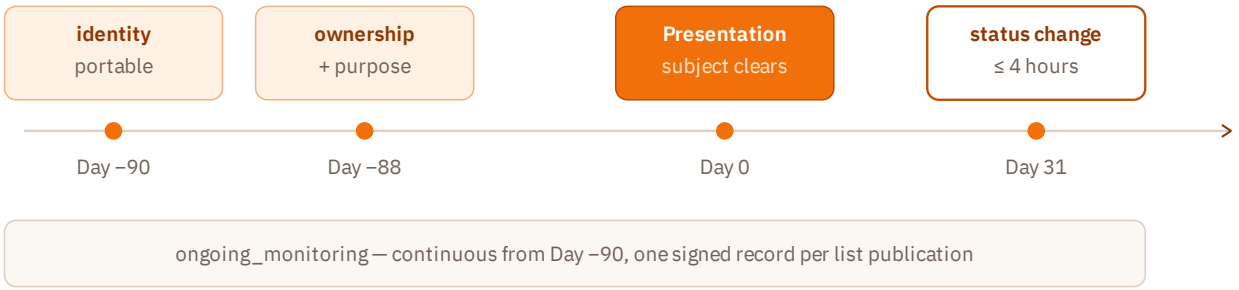

Figure 1 — The Record accumulates for ninety days before anything is disclosed.

10.2 The request

A counterparty's assessing agent proposes a payment and asks for a Presentation against the agent-to-agent payment profile.

Your agent checks the mandate first. The sponsor granted a disclosure mandate enumerating five step types, this counterparty's class, one purpose, and an expiry. `source_of_funds` is not in it — that Step exists in the Record, but the sponsor has not authorised its disclosure to this class of counterparty, so it does not leave.

Your agent constructs the Presentation. Audience set to the assessing agent, purpose set to counterparty assessment for payment, expiry set to seven days, the five permitted Steps referenced by carriage class and determination level, signed, and logged — so the sponsor can see precisely what was disclosed and to whom.

Elapsed: under a second. Nothing here required a human.

10.3 What the counterparty does with it

The assessing agent verifies the envelope, then each Step, then acts on carriage class:

- The identity Step is **portable** — it takes the result into its assessment and does not repeat the verification.
- The beneficial-ownership Step is **portable and referenced** — it accepts the determination, and because its own policy requires the underlying ownership data for a first-time counterparty, it requests the material at the disclosure endpoint under its own authorisation, receives it, and confirms it hashes to the carried value.
- The screening Steps are **freshness signals** — it reads which list sets you use, your matching methodology and how recently you ran, treats that as a risk signal about your programme, and then screens you itself. Two seconds.
- The monitoring Step is a **subscription** — it recomputes your ninety-day evidence chain, confirms 847 executions against 841 expected with zero missed windows, checks that your maximum observed latency of PT3H12M sits inside its own policy threshold, and subscribes to your change endpoint.

Its policy is satisfied. The subject clears, the payment proceeds, and no human was involved on either side — because no determination reserved to a person arose.

10.4 Day 31: the part that matters

A sanctions authority publishes an update. A beneficial owner of your sponsor entity appears on it.

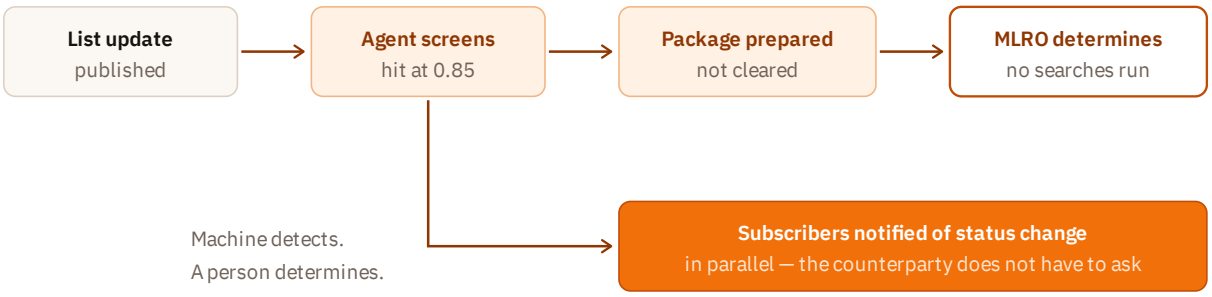

Figure 2 — Detection, determination and notification on a hit.

Your programme executes on the publication, as declared. Fuzzy matching at threshold 0.85 returns a possible match. Because `disposition` is `human_required_on_hit`, **your agent does not clear it and does not resolve it.** It assembles: the search performed, the list set and version, the matched entry, the evidence, and the identity of the MLRO whose determination is required. A Step is emitted at `prepared_for_determination`.

Your MLRO opens a complete package. They perform no searches. They make the determination the rules reserve to them, and a superseding Step is issued at `person_determined` carrying their signature.

In parallel, your subscription endpoint notifies every counterparty holding a live subscription that the subject's monitoring status has changed. The assessing agent from Day 0 receives that notice and applies its own policy — which in its case means holding further transactions pending its own determination.

Elapsed from publication to the counterparty knowing: under four hours. Under a quarterly review cycle the same fact would have surfaced somewhere between a day and three months later, and the counterparty would not have been told at all — it would have had to ask.

That interval is the entire product. Everything else in this manual exists to make that number small, and to make it something a stranger can verify rather than something you assert.

10.5 What the example did not do

Worth stating, because the example reads smoothly and the boundaries are what a reviewer will test.

No obligation moved: the assessing agent remained fully responsible for its own due diligence throughout, and it screened you itself. No reserved determination was automated — the one that arose went to a person, and the record shows what was put before them. Nothing disclosed suspicion: the record shows a screening hit and its determination, and carries nothing about what was or was not reported to anyone. And nothing was published — one Presentation went to one counterparty, for one purpose, for seven days, under a mandate the sponsor can revoke.

---

11. Quick reference

The four capabilities: emit Steps · prove the programme · manage mandates · construct Presentations.

Declare only what you can hold. Measured latency, populated matching, honest `expectedExecutions`.

The three mandate properties that usually fail: enumerated scope with no disguised wildcards · stated purpose · revocation resolvable without you.

Never: auto-clear a reserved determination · emit anything implying a report · claim Proved carriage · omit `validUntil` · change a programme on an event.

Always: emit per execution · timestamp independently · anchor the chain head · log every Presentation to the subject · assume the chain gets recomputed.

ACAS-1 Operator Manual — Deployers, Edition 1. Companion to ACAS-1 v0.1; paired with the Compliance Officers & MLROs manual. Published 1 August 2026 by Kadikoy Limited, Bermuda.