PRIMARY-SOURCE EVIDENCE FOR THE ADJACENT-TOOLS ANALYSIS

CEED Tool Verification Matrix

Dated primary-source review of adjacent delivery-governance technologies

Author	Viktor Barakoski
Document status	Tool Verification Matrix — Public Review Draft, editorial revision November 5, 2026 Evidence review: September 9, 2026. First publication: October 3, 2026. Editorial revision: November 5, 2026.
Review date	September 9, 2026
Ownership notice	The CEED Framework™. Copyright © 2026 Viktor Barakoski.
Status	Dated snapshot; checked against current primary documentation on September 9, 2026.
Official resources	ceedframework.com

Publication information

Licensing

This Matrix is released under CC BY 4.0, an open license. The CEED Public Review Draft 2 documents, including the CEED Specification, schemas, CEED Path Pattern Profile 1.0, tool catalog, and conformance materials, are released under CC BY 4.0. CEED Router source code will be released as open source under the Apache License 2.0. These licenses do not grant endorsement, official status, trademark rights, or a right to claim conformance by the CEED Project.

Trademark notice

The marks CEED™ and CEED Framework™ are trademarks of Viktor Barakoski. Use of the CEED Specification, the CEED Router, or other components under the licenses described above does not grant any right to use these marks.

Author independence and personal views

CEED Framework™ is an independent framework authored by Viktor Barakoski and developed using personal time and personal resources. The views in this document are the author's personal views, and no confidential, proprietary, restricted, or non-public materials were used in its development.

Third-party names

Product, project and organization names used in this Matrix are trademarks or names of their respective owners. Their inclusion identifies technologies reviewed from public documentation and does not imply affiliation, sponsorship or endorsement.

Status of this document

This Tool Verification Matrix is a Public Review Draft and an explanatory evidence record; it is not normative. The CEED Specification and Change Passport Schema, Public Review Draft 2, published October 26, 2026, governs where this record differs from it. The Matrix is updated independently from CEED releases, whenever the tools and standards it covers change.

Recommended citation

Barakoski, Viktor. (2026). CEED Tool Verification Matrix: Dated Primary-Source Review of Adjacent Delivery-Governance Technologies. Tool Verification Matrix — Public Review Draft, editorial revision November 5, 2026 (review date September 9, 2026). Official resources: ceedframework.com.

Corrections

Primary evidence that may affect a verdict can be submitted through ceedframework.com. This Matrix is a dated evidence record and is intended to be corrected when a later source establishes a materially different capability.

Contents

1 Finding....4
2 Scope and method....4
3 Test definitions....5
4 Results....8
5 What the review shows....9
6 Row notes....11
7 Enforcement integrity: T8 assessed profiles....14
8 CEED self-check....16
9 Composed alternative....19
10 Limitations and unresolved items....21
11 Corrections....22
12 Evidence register....22

1. Finding

CEED gives delivery tools a shared, producer-neutral specification for authorizing each software change and preserving its history through production. The CEED Delivery Contract records the approved scope and required evidence, the CEED Router is designed to check the exact candidate and required deployment stages, and the CEED Change Passport connects the decisions and later production findings to the applicable authorization. Teams can reuse the CEED Specification, schemas, and conformance cases instead of designing, documenting, and maintaining a private authorization model.

The September 9, 2026 review assessed 23 technologies within their documented product scopes and configurations. The review did not establish one supplied model combining issued per-change modification authority, explicit rules for allowed, protected, and unlisted paths, exact candidate and deployment binding, and a continuing history connecting decisions and production findings to that authority.

The reviewed tools supply approval, protection, evidence, enforcement, and traceability capabilities that a delivery system can use. Section 9 identifies the additional specification and integration work needed to connect those capabilities in the proposed composition. CEED supplies a common specification and conformance cases for that work, giving implementers a shared model and a defined basis for comparing results.

On the review date, the assessed CEED rules and conformance materials were held in a working package. The CEED Public Review Draft 2 document package was published on October 26, 2026, followed by the corrected CEED Conformance Pack and Guide on October 27. The CEED Project's CEED Router remains in development, with release expected in January 2027. The published rules and cases provide a basis for implementation; operational verification, agreement between independent implementations, and field evaluation provide the evidence for production readiness and organizational outcomes.

This editorial revision updates the Finding, brings statements about the CEED Framework's publication status up to date, and includes the correction recorded on October 8, 2026. The comparator review remains dated September 9, 2026; the comparator scores, assessed configurations, method, and comparator evidence are unchanged.

Correction recorded October 8, 2026: the CEED self-check in §§8.1 and 8.4 displays Yes for T7a, published implementation-independent semantics. The correct result for September 9, 2026 is No: the semantics were complete in the working package but were not public on that date. This correction governs the displayed historical score.

2. Scope and method

This review covers 23 comparator technologies. Each was checked against first-party documentation current on September 9, 2026. A disclosed CEED self-check was scored separately and is not a twenty-fourth comparator.

- IBM Bob
- AWS Kiro
- Kosli
- GitHub Enterprise AI Controls
- GitHub Spec Kit
- Tessl
- in-toto layouts
- in-toto Attestation Framework
- SLSA provenance
- OPA / Conftest
- CDEvents
- AGENTS.md
- Google Binary Authorization
- CODEOWNERS + GitHub rulesets
- Buf
- CodeQL
- Playwright
- Argo Rollouts

- SLSA Source Track
- gittuf
- Cosign
- LaunchDarkly
- OpenSpec

A capability counts only when current first-party documentation shows that the product, specification or its documented intended integration supplies it. General extensibility, a generic programming surface or something an adopter could build does not count by itself.

The review was conducted by inspecting publicly available first-party documentation, specifications, repositories and release records.

The CEED Framework Whitepaper, Public Review Draft 2, §9.2 examines adjacent tools through six areas: approved intent, allowed and protected scope, evidence, risk, enforcement, and production outcomes. This Matrix scores those areas in finer detail through the twelve tests below.

Each comparator row assesses one named configuration. Alternative configurations are disclosed rather than averaged. Plan, level, beta and Enterprise qualifications sit with the affected verdict.

The scored verdicts are Yes, Partial and No. The symbol — is used only for T2c when T2a is No. Unresolved means that the evidence is insufficient; it is never a substitute for a demonstrated limitation and is not a synonym for Partial.

No aggregate score, ranking or total is computed. The tests are not equally weighted, and a count of Yes cells is not an overall assessment of a technology.

3. Test definitions

T1 — Change-bound authorization

Does the supplied model establish an approved requirement, specification or authorization for a specific change, with both approval state and change association represented as system state rather than only as a workflow convention? A recommendation to review does not qualify unless the approval or association is represented by the supplied model. Authorization does not have to precede implementation. A machine-managed, approval-dependent transition may qualify without durable storage, an external approval API or an immutable record; durability and binding limitations are described separately. The changed object need not be application source code. T8 separately tests whether the producer can alter its own authority.

T2a — Change-specific modification boundary

Does the supplied model define what this particular change may modify? The boundary can be expressed as paths, modules, services, packages, APIs, schemas, artifacts or another technically meaningful scope. Repository paths are not required for a comparator to pass.

T2b — Explicit protection

Does the supplied model distinguish permitted modifications from deliberately protected or prohibited modifications? The protection does not have to be change-specific; T2a tests that separately. The review records whether the protection applies to a change, repository, workspace, agent or profile, session, environment or another defined scope. Restricting automatic approval is not the same as prohibiting a modification, and the distinction is recorded where relevant.

T2c — Residual disposition

When a change-specific modification boundary exists, does the supplied model deterministically decide what happens to scope the boundary does not enumerate? Qualifying approaches include intrinsic boundary semantics, an explicitly referenced or inherited policy, default deny, default allow, escalation or another deterministic supplied rule. A generic engine or repository default is not enough by itself. T2c is recorded as — when T2a is No. When T2a is Partial, the residual rule is evaluated for the partially supported boundary and the limitation is stated.

T3a — Evidence obligations

Does the assessed configuration declare which evidence the governed action must have and check that evidence before progression? The product does not have to ship application-specific rule content; evidence content selected by an organization may qualify. General ability to run arbitrary checks is not enough without a supplied obligation model or a documented intended integration.

T3b — Supplied context scaling

Does the supplied model provide a named risk dimension or another first-class context dimension that changes evidence, approval or release obligations? Environment, reference, module, cluster, severity, release stage and plan level count on the same basis as a field named risk. A risk field invented solely by an adopter inside a generic policy language does not qualify.

T4 — Gate coverage

T4 is Yes when a real blocking mechanism is supplied directly or through a documented intended integration at W, M, D or X. Partial requires a named substantive limitation in the supplied enforcement capability; a workspace location is not a limitation. Missing evidence is unresolved rather than Partial. Each result names the blocked action and enforcement location. A verifier that only returns failure, without an identified blocking action, receives no gate credit. A documented required-check integration is credited even when an adopter configures it. Bypass concerns belong to T8 and do not reduce T4.

Location	Meaning
W	Workspace or tool action.
M	Repository admission or integration: a supplied enforcement point can reject a repository push, protected-reference update, merge or equivalent integration operation.
D	Deployment or admission.
X	Traffic or exposure control.

T5 — Lifecycle continuity

Does the supplied model keep later production observations or incident knowledge connected to the governed change under a persistent change identity or history? Generic telemetry without reliable change-level linkage does not qualify. Every row records a mandatory lifecycle join describing exactly what is connected to what; no particular join shape is itself a pass condition. Finite or configured retention is disclosed rather than penalized. A difference from the CEED Model's preferred record structure is not an adverse finding: T5 does not require the outcome to occupy an approval record or to link to an authorization version. When those linkages are absent, they are recorded as qualifications on the join, not as reasons to withhold a verdict.

T6 — Producer independence

Does the authorization or lifecycle model retain the same governing meaning whether the change comes from a human, one agent, several agents or a mixed pipeline, without recreating governance semantics inside each producer runtime? This test is not producer attribution. Producer-specific adapters are compatible with a Yes when the governing semantics remain unchanged.

T7a — Published implementation-independent semantics

Are the relevant lifecycle semantics defined independently of a proprietary runtime in versioned public form with enough detail for independent implementation? Documentation in a software repository can qualify. A formal conformance program and an explicit independent-implementation statement are informative but not mandatory. This is an interoperability-maturity test, not a product-quality score. A proprietary platform may legitimately score No because public independent implementation is not its purpose.

T7b — Demonstrated independent interoperability

Is there an identifiable, substantially separately authored implementation and evidence that it correctly exchanges or interprets the relevant records, or reaches the relevant decisions, under the published semantics? Participation in the specification's development or governance is not by itself disqualifying. A fork, wrapper, renamed distribution or execution of the original implementation does not establish independent interoperability. The evidence must identify the parties, codebases, versions, tested scope and demonstrated agreement or interchange. Interoperability at one semantic layer is not credited to another.

T8 — Enforcement integrity

Can the governed producer, using the credentials and privileges provided by the vendor's documented supported configuration, alter or bypass the governing control? Positive polarity applies: Yes means protected; Partial means meaningful protection with a material limitation inside the assessed profile; No means the producer can unilaterally alter or bypass the governing authority. An authorized administrator's ability to change policy is disclosed as an operational limitation, not scored as a producer bypass. Protection is not assumed when the supplied integration does not establish it. Configurations are not averaged: each row names one assessed profile and discloses alternatives. T1 Yes does not establish T8.

4. Results

All results in this section describe the documented scopes and assessed configurations checked on September 9, 2026. Qualifiers are part of the verdict and must be read with the tables.

4.1 Authorization and evidence

Comparator	T1	T2a	T2b	T2c	T3a	T3b
IBM Bob	No	No	Partial	—	No	No
AWS Kiro	Yes [g]	No	Yes	—	No	No
Kosli	Yes [β]	No	No	—	Yes	Yes
GitHub Enterprise AI Controls	No	No	Partial	—	No	No
GitHub Spec Kit	Yes	No	No	—	No	No
Tessl	Yes [R]	Partial	Partial	No	Yes [R]	Yes [R]
in-toto layouts	No	Partial	Yes	Yes	Yes	No
in-toto Attestation Framework	No	No	No	—	Partial	No
SLSA provenance	No	No	No	—	Yes	Partial
SLSA Source Track	Yes [L]	No	Yes	—	Yes	Yes
gittuf	Yes	Partial	Yes	Yes	Yes	Yes
Cosign	No	No	No	—	Yes	No
OPA / Conftest	No	No	No	—	Yes	No
CDEvents	No	No	No	—	No	No
AGENTS.md	No	No	No	—	No	No
Google Binary Authorization	No	No	Partial	—	Yes	Yes
CODEOWNERS + GitHub rulesets	Yes	No	Yes	—	Yes	Yes
Buf	Yes [E]	No	Partial	—	Yes	Yes
CodeQL	No	No	No	—	Yes	Yes
Playwright	No	No	No	—	Yes	No
Argo Rollouts	No	No	No	—	Yes	Yes
LaunchDarkly	Yes [f]	No	No	—	Yes	Yes
OpenSpec	Partial	No	No	—	Partial	No

[g] gated Spec workflow only, not Quick Spec · [β] beta controls and decisions · [R] Tessl Review App, beta, enforcing mode · [L] Source L4, specified · [E] Enterprise BSR · [f] flag or configuration change

4.2 Enforcement, lifecycle and interoperability

Comparator	T4	W	M	D	X	T5	T6	T7a	T7b	T8
IBM Bob	Yes	Yes	No	No	No	No	No	No	No	unresolved
AWS Kiro	Yes	Yes	No	No	No	No	No	No	No	Yes
Kosli	Yes	No	No	Yes	No	Yes	Yes	No	No	No
GitHub Enterprise AI Controls	Yes	Yes	Yes [p]	No	No	No	Partial	No	No	Yes
GitHub Spec Kit	Yes	Yes	No	No	No	No	Yes	Partial	No	No
Tessl	Yes [R]	No	Yes	No	No	No	Yes [R]	Partial	No	unresolved
in-toto layouts	No	No	No	No	No	No	Yes	Yes	unresolved	Partial
in-toto Attestation Framework	No	No	No	No	No	No	Yes	Yes	Yes [s]	Partial
SLSA provenance	No	No	No	No	No	No	Yes	Yes	Yes [s]	Partial
SLSA Source Track	No	No	No	No	No	No	Yes	Partial	unresolved	Yes
gittuf	Yes	No	Yes	No	No	No	Yes	Yes	No	Partial
Cosign	No	No	No	No	No	No	Yes	No	Partial	Partial
OPA / Conftest	Yes	No	Yes	No	No	No	Yes	Yes	Yes [s]	Yes [A]
CDEvents	No	No	No	No	No	Partial	Yes	Yes	unresolved	No
AGENTS.md	No	No	No	No	No	No	Partial	Partial	Partial	No
Google Binary Authorization	Yes	No	No	Yes	No	Partial	Yes	No	No	No [A]
CODEOWNERS + GitHub rulesets	Yes	No	Yes	No	No	No	Yes	No	No	Yes
Buf	Yes	No	Yes	No	No	No	Yes	Partial	No	Yes [A]
CodeQL	Yes	No	Yes	No	No	No	Yes	No	No	Yes [A]
Playwright	Yes	No	Yes	No	No	No	Yes	No	No	No
Argo Rollouts	Yes	No	No	Yes	Yes	Yes [v]	Yes	Partial	No	No [A]
LaunchDarkly	Yes	No	No	No	Yes	Yes [gr]	Yes	No	No	Partial
OpenSpec	Yes	Yes [x]	No	No	No	No	Yes	Partial	No	No

[p] push rejection at a protected path, not merge gating · [R] Tessl Review App, beta, enforcing mode · [s] scoped interoperability — see the evidence register and row note · [v] revision-linked analysis · [gr] guarded-rollout observations · [x] archive operation, not merge or deployment · [A] assessed T8 profile in §7

No aggregate score is implied. These cells describe different control properties and are not equally weighted.

5. What the review shows

The findings below refer only to the configurations and documentation checked on September 9, 2026. They describe supplied capabilities; they do not rank the technologies.

Change-bound authorization in some form is supplied by gittuf, CODEOWNERS + GitHub rulesets, GitHub Spec Kit, Kosli, Buf, LaunchDarkly, AWS Kiro, Tessl and SLSA Source Track at Source Level 4. The form and qualifier differ by product: reference transition, pull-request state, workflow gate, beta control decision, schema commit, flag or configuration change, gated Spec workflow, beta Review App, or a specified Source L4 requirement.

Production observations connect to a governed change or release in several reviewed models. Kosli links runtime presence to change identity and versioned control decisions. Argo Rollouts and LaunchDarkly connect observations to the governed release. Google Binary Authorization and CDEvents connect post-deployment or incident information at the artifact layer.

AI coding tools can protect parts of their own governing configuration. Kiro hardcodes denial of writes to its own permission files. IBM Bob supplies administrator-enforced hooks and managed policy delivery. GitHub Enterprise AI Controls can use push-rule protection for agent-configuration paths. The exact integrity verdict still depends on the assessed profile and any documented bypass.

Producer independence is common across the reviewed landscape. It therefore does not distinguish one technology or the CEED Model by itself. The important comparison is which object is governed and whether the governing meaning survives a change of producer.

Published implementation-independent semantics with demonstrated independent interoperability already exist in parts of the landscape: the in-toto Attestation Framework, SLSA provenance and OPA have scoped evidence of independent interpretation or interchange. The CEED self-check has not established independent implementation interoperability.

The governed object differs across technologies. Kiro, IBM Bob, GitHub Enterprise AI Controls and AGENTS.md govern the producer. Cosign, SLSA provenance and Google Binary Authorization govern the artifact. gittuf and CODEOWNERS + GitHub rulesets govern the reference transition. Kosli governs process execution. LaunchDarkly governs the configuration change. The CEED Model governs what the change may modify. Multi-producer authorization and lifecycle traceability describes the CEED Model; it is not an exclusivity claim.

Argo Rollouts and LaunchDarkly provide the strongest T5 results in the assessed release-control rows: each joins production observations to a governed release. Neither establishes a join from those observations to the applicable authorization. That narrower missing link is the clause on which the bounded Finding depends.

6. Row notes

Each note below summarizes the assessed scope, version or configuration and the basis for its Yes and Partial cells as checked on September 9, 2026.

IBM Bob	IBM Bob IDE 2.1.0 and Bob Shell 2.0.2 provide blocking workspace hooks: PreToolUse and UserPromptSubmit can stop an action before it runs. `.bobignore` supplies the T2b Partial by excluding repository content. `DisabledAutoApprovalGroups` instead removes automatic approval and forces a human decision; it does not prohibit the operation. T8 remains unresolved for the managed profile because hook-code protection and enforced-hook startup or timeout failure behavior are not established.
AWS Kiro	AWS Kiro was assessed across IDE 1.x and CLI 3.x. The gated Spec workflow represents phase approval for a feature-specific specification, while Quick Spec is excluded from that T1 qualification. Kiro supplies standing permission protections and workspace blocking, including hardcoded denial of writes to its own permission files. The assessed managed policy is protected from the governed producer; local administrative power remains an operational limitation rather than a producer bypass.
Kosli	Kosli was assessed with its current controls and decisions capability, both documented as beta. Decisions can be bound to Trails, artifacts and control definitions, and environment policies can require them before progression. Runtime presence links through artifact fingerprints to a Trail and associated control decisions, supporting T5. The supported Member service-account role can also alter policies and flows, so the assessed producer can change governing authority and T8 is No.
GitHub Enterprise AI Controls	GitHub Enterprise AI Controls were assessed in the generally available control plane, with enterprise MCP allowlists noted as preview. The T2b Partial records explicit protection for the custom-agent configuration paths under the enterprise push rule; the product supplies no wider distinction between permitted and protected modifications. The model also blocks agent tool activity. T6 is Partial because the controls govern agent activity rather than all producers; the assessed push-rule configuration is protected from the governed repository producer.
GitHub Spec Kit	GitHub Spec Kit was assessed at v1.0.4. The supplied workflow persists run state and includes approval gates that can abort the run, supporting change-bound authorization and workspace gating. Its workflow semantics apply regardless of the producer, while published semantics remain only partially implementation-independent for this Matrix. The workflow and run-control files remain writable in the working repository, so T8 is No. PR 2699 was Open on the review date.
Tessl	Tessl was assessed on the Review App beta in enforcing mode. It reviews the exact pull-request head, represents approval or requested changes, declares standards to evaluate, and can hold merge when GitHub requires the resulting check. The Review App's `requestChangesAt` context threshold changes which findings can block, supporting T3b Yes [R]. Scope and protection are partial. T7a Partial reflects the publicly documented `.spec.md` format and MIT-licensed tile without a complete implementation-independent lifecycle specification. T8 remains unresolved for the hardened profile because effective policy content comes from the pull-request head and the author-accessible `approve` intent is not fully characterized.
in-toto layouts	in-toto layouts were assessed against Specification 1.0. Artifact rules provide step-scoped modification constraints, explicit ALLOW and DISALLOW protection, and deterministic residual handling through sequential rule consumption with implicit ALLOW *. The layout also declares required materials and products. The model is producer-independent and publicly specified. No supplied merge or deployment gate was established; layout-level independent interoperability remains unresolved, and signed layouts do not prevent a producer from simply skipping verification.

in-toto Attestation Framework	The in-toto Attestation Framework was assessed at specification v1.2. It supplies authenticated, typed Statements, Envelopes and Bundles, but leaves the declaration of required claims to a policy layer, so T3a is Partial. The semantics are public and producer-independent. Demonstrated interoperability is scoped to Statement v1 exchange in the recorded npm producer-to-slsa-verifier path. T8 is Partial because DSSE authenticates the attester while key management remains outside the framework.
SLSA provenance	SLSA provenance was assessed against v1.2. The Build track defines provenance requirements and scales platform trustworthiness by level, which produces a Partial T3b rather than per-change context scaling. It is producer-independent and publicly specified. The recorded npm interoperability path demonstrates scoped Statement v1 and SLSA provenance interchange. The specification requires higher-level provenance to resist falsification by the build process, but enforcement is supplied by ecosystems rather than SLSA itself, so T8 is Partial.
SLSA Source Track	SLSA Source Track was assessed at v1.2 Source Level 4. That profile specifies final-revision and context-specific approvals, protected named references, evidence requirements and level-dependent obligations, supporting T1, T2b, T3a and T3b. The model is producer-independent. T7a is Partial because some source-provenance formats remain source-control-system specific. T7b remains unresolved. Separately, T8 is Yes for the stated Source L4 trust and protection assumptions.
gittuf	gittuf was assessed at v0.16.0 with the metadata-version update applied. Reference authorizations bind approval to a specific reference-state transition; policy metadata supplies protection, deterministic residual handling, evidence requirements and context through delegations. RSLPush can reject repository admission, giving T4 at M. The semantics are public and producer-independent. T2a is Partial because an exact transition is not a modification envelope, while T8 is Partial because enforcement is client-side and freshness has stated limits.
Cosign	Cosign was assessed at v3.1.3. Attestation verification can require predicate types and evaluate CUE or Rego policy, supporting T3a. The tool is producer-independent, supporting T6. The row does not supply a blocking delivery action by itself. T7b Partial records interoperability through the shared in-toto attestation envelope, without a complete independently authored Cosign-specific interoperability demonstration. Keyless signing binds records to an OIDC identity with Fulcio and Rekor; control of that signing identity remains outside Cosign, leaving T8 Partial.
OPA / Conftest	OPA / Conftest were assessed with an independently administered OPA service as the T8 profile. Policy obligations can be declared and checked in required CI, and the documented integration can block repository progression. Rego provides public implementation-independent semantics, with Regorus establishing scoped independent evaluation interoperability. Signed bundle verification plus API authentication and authorization can protect the governing service, giving T8 Yes for that configuration. Local Conftest policy stored in the producer repository is disclosed as a No alternative.
CDEvents	CDEvents was assessed at v0.5.1. Its public event vocabulary is producer-independent and can represent incident, artifact and change references, including artifact events that identify a building revision. That supports T5 only partially because the specification does not establish that deployments populate and retain the relationships needed for a reliable lifecycle history. Independent producer-to-consumer interchange beyond first-party SDKs remains unresolved. The vocabulary supplies neither a delivery gate nor protected governing configuration.
AGENTS.md	AGENTS.md was assessed as an open Markdown convention with no versioned specification release identified. It provides cross-agent instruction discovery and precedence, which gives partial producer independence and partial interoperability maturity rather than delivery authorization. Independent agents interpret the same convention, but that is weaker than lifecycle-record interchange. It supplies no requirement approval, scope boundary, evidence gate or production history, and the file itself has no integrity mechanism beyond repository controls supplied elsewhere.

Google Binary Authorization	Google Binary Authorization was assessed on the current GKE admission model. It can require named attestations, vary requirements by cluster and block deployment admission; Continuous Validation extends artifact-linked checks after deployment, giving Partial T5. Image-name allowlist patterns provide partial protection semantics. T8 is No for the assessed direct-deployer profile because breakglass is enabled through a Pod-spec label without policy-administration rights. A separately controlled deployment path is disclosed as the alternative configuration.
CODEOWNERS + GitHub rulesets	CODEOWNERS + GitHub rulesets were assessed on the current ruleset model. Review approval can remain bound to the current head through stale-review dismissal and last-push approval rules. CODEOWNERS, push rules and required checks supply standing path or reference protection, evidence obligations and repository gating; reference scoping supplies context scaling. The model is producer-independent. T8 is Yes where organization-level rulesets are active and the governed producer is not listed as a bypass actor.
Buf	Buf was assessed on Enterprise Buf Schema Registry. Per-commit review states provide change-bound approval, registry checks and breaking-change rules supply evidence obligations, and policy scopes change which checks apply. Registry enforcement can reject a schema push or hold default-label advancement. T2b is Partial because the prohibition concerns schema compatibility, not which parts a change may modify. T7a Partial reflects publicly documented BSR checks and policy semantics without a complete versioned implementation-independent registry specification. T8 is Yes for the Enterprise profile; CLI-only local configuration is No.
CodeQL	CodeQL was assessed on GitHub code scanning with default setup and merge protection. Named tools plus alert and security-severity thresholds define required evidence and context-dependent obligations. Merge protection can block when qualifying alerts exist, analysis is incomplete or the required tool is absent. The model is producer-independent. T8 is Yes for default setup because configuration and merge protection are administered outside repository content; advanced setup with a producer-editable workflow is disclosed as the No alternative.
Playwright	Playwright was assessed on its documented CI integration. The generated GitHub Actions workflow declares and executes the selected test evidence, and when GitHub requires that check it can block merge, supporting T3a and T4 at M. The test framework itself remains producer-independent. It supplies no change authorization, scope model, context scaling or lifecycle history. T8 is No because the tests and workflow that produce the evidence are files the governed repository producer can modify.
Argo Rollouts	Argo Rollouts was assessed on the stable controller, Rollout, AnalysisTemplate and AnalysisRun model. Analysis declares metrics by release stage and can gate deployment and traffic exposure. T5 joins production observations → AnalysisRun → Rollout revision through pod-template hashes; configured retention is disclosed, not penalized. No source-candidate link, authorization-version link or general late-incident path was established. T7a Partial reflects public CRD and controller semantics without a complete implementation-independent lifecycle specification. T8 is No for producer-controlled GitOps manifests; ClusterAnalysisTemplate separation is the disclosed alternative.
LaunchDarkly	LaunchDarkly was assessed with approval controls and Guardian-plan guarded rollouts. Approval requests create change-bound state for a flag or configuration change; environment-specific requirements and rollout metrics supply evidence and context scaling. T5 joins production metric observation → guarded rollout → flag variation and targeting change. No source-candidate or approval-version link and no general late-incident path were established. T8 is Partial because bypassRequiredApproval exists and some changes fall outside the approval mechanism.

OpenSpec	OpenSpec was assessed on current main; the corresponding release tag was not established, and PR #1277 was Closed. Per-change artifacts provide change association but no represented approval, so T1 is Partial. Supplied validation rules are machine-checkable but partly nonblocking, giving T3a Partial. Archive validation provides a narrow workspace gate, supporting T4 Yes with the archive qualifier. The artifact and gate semantics are producer-independent, supporting T6 Yes. T7a Partial reflects publicly documented repository artifacts and CLI/archive semantics without a versioned implementation-independent lifecycle specification. `--no-validate` and producer-writable files make T8 No.

7. Enforcement integrity: T8 assessed profiles

The profiles below are the configurations used for T8 on September 9, 2026. Each profile follows the same sequence: governed principal → permissions held → protected object → enforcement configuration → supported limitation → verdict. Alternative supported configurations are disclosed in the profile rather than averaged into a single score.

Comparator	Assessed profile	Verdict
IBM Bob	Bob agent under the developer's user account → ordinary user rights, no local administrative access → hook configuration, enforced hook set, and enterprise policy source → managed fleet with MDM, GPO or a root-owned `/etc/bob/policy.json`; `EnforcedHooks` documented as non-overridable by users → **two open questions**: protection of the enforced hook's script and dependencies, and enforced-hook startup or timeout failure handling. No bypass established	**unresolved**
AWS Kiro	Kiro agent, IDE and CLI → same filesystem rights as the user, no local administrative access → admin permission policy at an OS-protected path, and the agent's own permission files → managed policy requiring administrator or root to modify, applying to both surfaces from one file; hardcoded Kiro-scope deny on `~/.kiro/settings/`, `.kiro/settings/`, `~/.kiro/workspace-roots/`; admin rules may only deny or ask, never allow; malformed JSON or an invalid `allow` effect fails closed, while unknown capabilities are skipped with a warning and valid rules still load → vendor-stated: client-level enforcement can be circumvented via local administrative access; agent profiles and hooks are protected by *ask*, not deny	**Yes**
Kosli	Producer's CI system holding a **Member** service-account credential, per Kosli's own recommendation for CI/CD systems → report attestations, create and update policies, update and delete flows, attach and detach environment policies → flow templates and environment policies → no documented role reports attestations without policy write; a snapshotter token can additionally modify an existing environment's policies by re-PUTting a full payload → *alternative disclosed:* an independently controlled reporting service holding the Member credential is not the governed producer, and would change this verdict	**No**
GitHub Enterprise AI Controls	Agent operating in a repository on behalf of a user → repository token; not enterprise AI-control permissions → `.github/agents/*.md`, enterprise policies, `.github-private/agents/*.md` → enterprise-wide push rule protecting the agent path, plus enterprise-scoped policy users cannot override locally → GitHub-stated: policy settings can drift if access is too widely held	**Yes**
GitHub Spec Kit	Coding agent or developer → filesystem write → `workflow.yml`, `.specify/workflows/runs/`, `.specify/workflow-catalogs.yml` → none supplied → the agent can edit the workflow, resume a run with `specify workflow resume`, or invoke `speckit.implement` directly	**No**
Tessl	Pull-request author, human or agent → repository write on the PR head → Review App effective policy content, repository-local skills, registry skills → **hardened**: required code-owner approval over every trusted local source; registry RBAC with install and publish policies → author-accessible `approve` intent, interaction with blocking findings and required checks not established	**unresolved**
in-toto layouts	Step functionary → functionary key only → the layout → layout signed with project-owner keys, held separately from functionary keys → nothing in the supplied model prevents a producer from not running verification	**Partial**
in-toto Attestation Framework	Attester → signing key → the envelope → DSSE authentication separating attester from consumer → key management is deliberately outside the framework's scope	**Partial**
SLSA provenance	Build process → build-platform execution context → the provenance record → higher build levels require the platform to make provenance non-falsifiable by the build process → specified, not supplied	**Partial**

Comparator	Assessed profile	Verdict
SLSA Source Track	Trusted person or trusted robot, not an Administrator → contribution and review rights → change history, protected named references, control continuity → two-party review with final-revision approval, force-push prohibition, continuity tracked from a start revision, and the requirement that a trusted robot's identity and codebase cannot be unilaterally influenced → the Administrator role may modify change history and security policies; enforcement is delegated to the source-control system	**Yes**
gittuf	Party with push access to the repository and the Reference State Log → push credentials, not root keys → gittuf policy metadata → root-key threshold signing; assessed at **v0.16.0 with the metadata-version update applied** → verification is client-side, so a non-conforming client can push invalid entries and the model recovers rather than prevents; a privileged forge may branch the RSL; no timestamp role, so RSL freshness is not guaranteed	**Partial**
Cosign	Signer → OIDC identity → attestation authenticity → keyless signing with Fulcio certificates and Rekor transparency-log inclusion → whether the governed producer holds the signing identity is a key-management question outside Cosign	**Partial**
OPA / Conftest	Producer's CI requesting a decision → decision-request rights only, not policy write → policy bundles and the decision API → **independently administered OPA service** with bundle signature verification required and API authorization enabled → both authentication and authorization are **off by default**, so this is a configured rather than a default posture. *Alternative disclosed:* local Conftest with `policy/` in the producer's repository → No	**Yes**
CDEvents	— → — → — → none supplied → an event vocabulary supplies no protected configuration	**No**
AGENTS.md	— → — → — → none supplied → the convention supplies no integrity mechanism of its own; whether the file is producer-writable depends on repository controls belonging to the CODEOWNERS row	**No**
Google Binary Authorization	Party authoring or submitting the Pod specification → deployment submission rights; **no policy-administration rights required** → the admission policy → `defaultAdmissionRule` and `clusterAdmissionRules` under project IAM → **breakglass is enabled by a label in the Pod specification and does not require altering the policy**; audit logging is detection, not prevention. *Alternative disclosed:* a separately controlled deployment path preventing the producer from setting that label would change this verdict, and is the corresponding “required integration” in §9	**No**
CODEOWNERS + GitHub rulesets	Producer pushing to a matching ref → repository write; not organization administration; not listed in `bypass_actors` → branch and push rulesets → organization-level ruleset with `enforcement: active` and an empty bypass list → `enforcement: evaluate` does not block; an over-broad bypass list defeats the control; organization owners may change rulesets	**Yes**
Buf	Producer pushing a schema → BSR write rights; not administration or member-role rights → registry policies and checks → **Enterprise BSR** with registry-side checks prevailing over local configuration, under a role model separating writing resources from administering them and changing member roles → holding the default label does not make a pending commit inaccessible through every explicitly pinned reference. *Alternative disclosed:* CLI-only with `buf.yaml` in the producer's repository → No	**Yes**
CodeQL	Producer with repository write → workflow and code write; not repository or organization settings → analysis configuration and merge protection → **default setup**, whose configuration is created through repository settings and can supersede an existing workflow configuration, with merge protection administered outside the repository → merge-queue groups and certain Dependabot pull requests are documented coverage exceptions. *Alternative disclosed:* advanced setup with a repository-authored workflow → No	**Yes**
Playwright	Producer with repository write → test and workflow write → tests and the generated workflow → none supplied → the requirement to run the check is administered in the CODEOWNERS + rulesets row, not here	**No**
Argo Rollouts	Producer with write access to the GitOps manifests → manifest write → `Rollout` and `AnalysisTemplate` objects → the assessed profile places these in the producer's own manifests → the producer can weaken or remove the analysis. *Alternative disclosed:* `ClusterAnalysisTemplate` with cluster-scoped RBAC separating it from the producer's namespace	**No**
LaunchDarkly	Producer requesting a flag change → change-request rights; not `bypassRequiredApproval`; self-approval disallowed → approval settings and required approvals → hosted service with environment-scoped approval requirements administered outside the repository → some changes fall outside the approval mechanism entirely; `bypassRequiredApproval` exists as a grantable permission	**Partial**
OpenSpec	Coding agent or developer → filesystem write → change artifacts and the CLI invocation → none supplied → `--no-validate`, nonblocking proposal-check classes, and direct bypass of the CLI	**No**

8. CEED self-check

The CEED self-check applies the same tests, evidence thresholds and assessed-profile method used for the comparators. It is disclosed and is not an independent review. It assessed the CEED working package as it stood on September 9, 2026, and the scores below remain those of that date. The CEED Specification and Change Passport Schema and the CEED Conformance Pack were published as Public Review Draft 2 on October 26, 2026, with the corrected Conformance Pack and Guide following on October 27. The descriptions and Specification references in this section, such as §4.2, §4.3 and §9.2–§9.5, record that working package. The published Draft 2 Specification sets the current rules: authorization in §§4.2–4.3 and deployment in §§9.2–9.5. Two of those rules changed: Draft 2 recommends issuing the CEED Delivery Contract before implementation begins instead of requiring it, and it prohibits producer self-approval. The published Conformance Pack Guide records passing runs of 1,473 schema-example checks and 387 behavioral vectors and expected reports for its identified Pack snapshot. Independent scoring of the CEED Model is invited.

8.1 Self-check row

T1	T2a	T2b	T2c	T3a	T3b	T4	W	M	D	X	T5	T6	T7a	T7b	T8
Yes [sp]	Yes [sp]	Yes [sp]	Yes [sp]	Yes [sp]	Yes [sp]	Spec Yes / Impl No	No	Yes [sp]	Yes [sp]	Yes [sp]	Yes [sp]	Yes [sp]	Yes [pub]	No	Yes [sp] [A]

[sp] specified in the assessed Version 1.0 working package with executable packaged cases; no implemented integration and no operational evidence · [pub] material complete in the working package; not yet public on September 9, 2026 · [A] assessed profile in §8.5

Correction recorded October 8, 2026: the correct T7a result for September 9, 2026 is No; see the correction notice after §1.

8.2 Evidence states

State	Status in the assessed working package, September 9, 2026
Specified — normative requirement	T1, T2a, T2b, T2c, T3a, T3b, T4 (M, D, X), T5, T6, T8
Executable cases in the working package	Path classification, issuance, evidence binding, batch, release, append order, outcome, context band and reporting: 14 positive examples accepted, 15 named negative examples rejected, 80 behavioral vectors, 5 golden reports and SHA-256 integrity.
Implemented integration	None.
Operational evidence	None.

8.3 Specified versus implemented gate coverage

The CEED Specification in the assessed working package defines decisions that can gate release checkpoints. CEED Specification §9.2–§9.5 defines ordered checkpoints, strategy verification at the checkpoint governing production exposure, and a requirement for the deployment system to wait for each decision before continuing. The deployment platform executes the rollout. The assessed working package does not demonstrate an operational integration that implements those gates. The CEED Project's CEED Router for Git and GitHub, which makes these decisions, is in development, with release expected in January 2027.

Coverage	W	M	D	X
Specified	No	Yes	Yes	Yes

Coverage	W	M	D	X
Implemented	No	No	No	No

Both rows are required whenever the CEED Model's T4 result is cited.

8.4 Cell results

Test	Verdict	Basis and limitation
T1	Yes (specified)	The authorization decision MUST be made before code generation. Each Delivery Contract version is proposed in a Contract-only issuance commit modifying only `.ceed/contracts/<change-id>.yaml`; the Git platform MUST bind the configured protected approval to that exact commit; the Change Passport writer appends a `contract-issued` entry with canonical path, immutable commit and SHA-256 content digest. A draft has no authorization effect.
T2a	Yes (specified)	`allowed_scope` declares the files and directories the change may modify. Scope is repository paths, not semantic effects; see §8.6.
T2b	Yes (specified)	`protected_scope`, together with policy-file protections that no Delivery Contract may override.
T2c	Yes (specified)	If neither scope set matches, classification is out-of-allowed-scope and the candidate blocks. Literal-prefix specificity is deterministic; ties resolve to protected; negated patterns are disallowed.
T3a	Yes (specified)	`required_merge_evidence` is read at the merge check. Every item MUST carry `candidate_ref` with both `base_sha` and `head_sha`, and both must equal the candidate under evaluation. Head-only binding is non-conformant.
T3b	Yes (specified)	Risk tier determines required evidence, minimum approvals, reviewer groups, whether an approver must differ from the issuer, release baselines and observation-window length.
T4	Specified Yes (M, D, X); implemented No	See §8.3 for the specified-versus-implemented gate table. Mandatory merge or deployment gates fail closed in the assessed rules. The Git platform MUST treat the CEED Router result as a required status check where the commit exists before landing; the deployment gate holds the artifact until the final SHA has its own approved entry. No CEED Router release and no demonstrated blocking integration are included in the assessed working package.
T5	Yes (specified)	Lifecycle join: production entry → Change Passport (`change_id`, UUID) → `contract_version` → `candidate_ref` / `artifact_ref` / `deployment_ref` → `incident_ref`. Entries are append-only and immutable with contiguous numbering. Multiple lifecycle cycles are supported for late-surfacing incidents. Observation windows vary by risk tier. Retention duration is left to organizational policy.
T6	Yes (specified)	The Delivery Contract governs the change rather than the producer. In the assessed working package, the AI coding agent has no authority to approve scope, issue a Delivery Contract version or override a decision. Multi-agent pipelines read the same Delivery Contract.
T7a	Yes — material complete; not yet public on September 9, 2026	The working package contains normative semantics, schemas, CEED Path Pattern Profile 1.0, positive and negative examples, 80 behavioral vectors, 5 golden reports, runners, a manifest and digests. This was the self-check's strongest cell, but on September 9, 2026 the working package was not yet public. Correction recorded October 8, 2026: the correct result for September 9, 2026 is No; see the correction notice after §1. The material was published as Public Review Draft 2 on October 26, 2026.
T7b	No	No independently authored implementation has been demonstrated. The highest-value independent test remains a separately written implementation that consumes the same manifest and produces the same decisions without importing the reference runner.
T8	Yes (specified), assessed profile in §8.5	The repository MUST protect `.ceed/policy.yaml` and the complete contracts directory; no Delivery Contract may authorize those control-plane paths; the own-Contract transport rule never applies to repository policy or another change's Delivery Contract; approval is bound to the exact issuance commit.

8.5 CEED T8 assessed profile

Governed principal: the party producing the candidate — an AI coding agent, a human engineer or a mixed pipeline.

Permissions held: ordinary repository write; not the protected-approval role; not repository administration.

Protected object: .ceed/policy.yaml and the complete contracts directory.

Enforcement configuration: the repository protects those paths; no Delivery Contract may authorize control-plane paths; the own-Contract transport rule never applies to repository policy or another change's Delivery Contract; the Git platform binds the configured protected approval to the exact issuance commit.

> **Required qualification for this profile's Yes.** “Approver differs from issuer” is not equivalent to “the producer cannot self-authorize.” For example, a bot could issue the Delivery Contract while the human who produced the candidate supplies the approval; the two identities differ, but the governed producer still participates in its own authorization. For this assessed profile: the required approval must come from an authorized principal outside the governed producer's control, and the producer cannot supply or impersonate that approval or alter the relevant enforcement configuration. This qualifies the assessed profile only. It does not impose universal separation of duties on every CEED configuration.

- CEED Specification §4.3 prohibits an AI agent from issuing or approving its own Delivery Contract version in the assessed working package.
- CEED Specification §4.2 leaves approval requirements, including whether an approver must differ from the issuer, to organizational policy. A risk tier whose policy does not meet the condition above can allow a human producer to participate in its own authorization.
- A platform administrator may bypass native controls. A bypassed action MUST NOT be represented as CEED-approved.

T8 verdict: Yes within the stated profile; specified, with no implemented integration and no operational evidence.

8.6 Disclosed limitations

- Effect boundary. Repository-path authorization classifies paths in the Git candidate. It does not prove that code in an allowed path cannot create, modify or influence a protected build-time or runtime resource. Build isolation, sandboxing, testing, security analysis, deployment policy and runtime controls remain responsible for those effects.
- Contract quality. The CEED Model cannot detect a permissive Delivery Contract merely because the candidate stays within it. Risk-tier distribution and local review remain necessary.
- Submodules. Authorizing a parent-repository submodule pointer is not authorizing changes inside the referenced repository.
- Provider trust. The CEED Router consumes authenticated metadata from Git, CI, deployment, observability and incident providers. It does not independently prove provider or runner integrity.
- Evidence replay. Deployment-checkpoint evidence binds to one `deployment_ref` and checkpoint; artifact-bound evidence may be reused only for the identical artifact.
- Administrative bypass. A platform administrator may bypass native controls. A bypassed action MUST NOT be represented as CEED-approved.

Tooling the Whitepaper, Public Review Draft 2, proposes for Version 2 — an AI Contract Writer and editor or Model Context Protocol (MCP) integrations that show the issued boundary before an agent edits (Whitepaper §8.2) — is outside the assessed working package and is not scored.

9. Composed alternative

This section proposes one concrete GitHub-hosted stack with a GKE target, using capabilities documented in the September 9, 2026 review. It illustrates possible implementation roles for adjacent technologies, as described in the Whitepaper. It is a composition, not a claim that the integration already exists. Three components are explicitly marked Required integration — not yet established.

Layer	Named component and configuration	Status
Requirement	Jira issue, referenced by key in the pull-request title	Supplied
Change-bound approval	Branch ruleset targeting `main`: `required_approving_review_count: 2`, `requires_code_owner_review`, `dismisses_stale_reviews`, require-last-push approval, `bypass_actors` empty	Supplied
Control-plane protection	Push ruleset, separate from the branch ruleset and applying to pushes across the repository: restricted file paths covering `.scope/**`, `CODEOWNERS` and `.github/workflows/**`. A restricted path prevents any commit changing that path from being pushed.	Supplied
Scope-artifact issuance	A scope-issuance GitHub App is the sole bypass actor on the push ruleset. It creates or revises `.scope/<pr-number>.yaml` through the API after recording the approval it enforces. The producer is not a bypass actor.	Required integration — not yet established
Change-specific scope	`.scope/<pr-number>.yaml` listing allowed and protected paths	Private integration
Scope evaluation	An independently administered OPA service consumes signed policy bundles with signature verification required and API authorization enabled. A required CI check calls the service and consumes its decision. The organization supplies the change-specific Rego policy comparing the pull request changed-file list to the scope file.	Supplied engine; private policy
Source integrity	GitHub as source control, configured for the SLSA Source controls the organization has actually enabled: protected `main`, force-push prohibited, two-party review. This is not a claim of full L3 or L4 conformance.	Supplied, partially
Evidence	CodeQL default setup with merge protection at the selected severity threshold; Buf Enterprise BSR policies at organization level; Playwright via its generated workflow, made a required check.	Supplied
Provenance	GitHub Actions artifact attestations generating SLSA provenance, verified with `gh attestation verify` against repository identity expectations. `slsa-verifier` is retained only as historical interoperability evidence and is no longer actively maintained.	Supplied
Deployment admission	Google Binary Authorization on GKE: `requireAttestationsBy` a QA attestor and a security attestor; `clusterAdmissionRules` stricter for production.	Supplied
Breakglass containment	Normal deployment requests must pass through a protected deployment or admission control that prevents the governed producer from enabling breakglass. Emergency use follows a separately authorized path. The enforcing component can be a deployment service the producer cannot alter or an additional admission controller rejecting Pod specifications carrying the breakglass label.	Required integration — not yet established
Exposure	Argo Rollouts canary with an `AnalysisTemplate` on error rate and latency, auto-rollback enabled.	Supplied
Lifecycle record	Kosli: a Flow per service, Trails keyed by commit SHA, `kosli attest decision` against named controls, environment snapshots.	Supplied
Cross-component join	—	Required integration — not yet established

9.1 Three integration claims made explicit

Breakglass. Google documents GKE breakglass as a label in the Pod specification. A deployment can therefore proceed despite a denying Binary Authorization policy without changing that policy. Restricting policy administration does not stop an otherwise authorized deployer from submitting the label. The reviewed sources did not establish a breakglass-specific IAM permission, so none is assumed in this stack.

Conftest. Conftest is not the evaluator in this baseline. Its documentation establishes policy distribution over HTTPS, Git and OCI. OPA documentation separately establishes configured bundle-signature verification. Those two facts do not establish that a Conftest invocation verifies a signature before evaluation, so the baseline uses an independently administered OPA service for that protection.

Rulesets. GitHub distinguishes branch and tag rulesets from push rulesets, and push restrictions apply across the repository rather than to a branch target. The baseline keeps them separate. An unconditional push restriction over `.scope/**` would block authorized scope issuance too, so the design needs an independently controlled scope-issuance App rather than a blanket producer bypass.

9.2 What the composition would reproduce

If the required integrations are implemented and tested, the composition can provide change-bound approval as system state, deterministic allowed and protected path policy with an explicit residual rule, context-scaled obligations by reference, severity and cluster, blocking at merge, deployment admission and traffic exposure, signed provenance checked against declared identity expectations, protected control-plane paths, production observation driving rollback, and a record that joins runtime state to an artifact, change identity and versioned control decisions.

> **This is a proposed composition of available controls.** Its custom policies, authorization-update path and cross-component bindings still require implementation and testing.

9.3 What must additionally be specified and integrated

1. **The scope object.** `.scope/<pr-number>.yaml` has no shared schema. The organization must define field names, allowed/protected/unlisted semantics and the precedence rule when a path matches more than one list. in-toto layouts and gittuf each supply deterministic residual disposition in their own models; a shared per-change residual profile across this selected stack was not established.
2. **Agreement between implementations.** Private conventions created independently are not guaranteed to agree. Two organizations using this architecture would need shared integration-level tests if they want to verify consistent interpretation.
3. **The issuance path.** The scope-issuance App described above must be implemented.
4. **The carry-forward.** The merge-gate scope decision does not automatically travel to deployment or into the production record. Kosli carries decisions and control-definition versions but no scope object; Binary Authorization binds attestations to a digest; CDEvents can represent incident → artifact → change but does not require population. Storage, retention, population and correlation still need an identified implementation.
5. **The authorization-version join.** Kosli supplies a documented route from runtime presence to decisions and their control-definition versions. This stack does not supply a link from a production incident to the version of the change-specific authorization under which that change was approved.

Assembling these components does not by itself make the result portable; the five items above are the work that connects them. The CEED Specification and Conformance Pack, published as Public Review Draft 2, specify those pieces as one shared model with executable checks: the CEED Delivery Contract format and issuance rules, the CEED Change Passport record that carries decisions into production, and conformance cases that let separate implementations compare results. Implementers reuse that model instead of writing and maintaining a private equivalent.

10. Limitations and unresolved items

Vendor documentation changes frequently, so every verdict in this Matrix is dated. Each row assesses one named configuration rather than an average across configurations. Plan, level, beta and Enterprise qualifications remain attached to the affected verdict. Unresolved cells remain visible. The CEED self-check is disclosed rather than independent; external scoring of the CEED Model would provide stronger evidence and is invited.

10.1 Remaining uncertainties

These unresolved items remain visible and do not block this dated publication.

Item	Question	What it could change
IBM Bob T8	Whether an administrator-enforced hook script and its dependencies are required to reside where the governed producer cannot write, and how enforced-hook startup or timeout failure behaves. The sample ADMX and policy JSON can confirm configuration keys but not runtime behavior; runtime documentation or testing is needed.	The Bob integrity verdict, in either direction.
Tessl T8	Whether the author-accessible approve intent can resolve or supersede outstanding blocking findings, and whether the resulting check state satisfies a required merge check.	The Tessl integrity verdict, in either direction.
in-toto layouts T7b	Demonstrated layout-level interoperability between separately authored implementations.	Maturity claims for the in-toto row and its use in the composed alternative.
SLSA Source T7b	Independently authored Source-track implementations.	Whether SLSA Source can be presented as an interoperable source-integrity profile.
CDEvents T7b	Independent producer-to-consumer interchange distinct from the first-party Go, Python and Java SDKs.	CDEvents' standing as a lifecycle transport in the composed alternative.
CDEvents T5	Whether real deployments populate and retain the `change` and `artifactId` references defined by the vocabulary.	The strength of CDEvents lifecycle continuity.
Kosli T5	Whether the runtime → decision → control-definition-version route is a supplied query or an architectural inference from two documented relationships.	The strength of the narrower lifecycle distinction in the Finding.

11. Corrections

This Matrix is a dated evidence record, not a permanent negative claim. A new capability, specification revision or contrary primary source will be recorded and the affected verdict corrected.

When a later review changes a verdict, the later version will preserve the fact that the dated assessment changed rather than silently rewriting the historical record.

Product marketing alone is not sufficient evidence where technical documentation or a specification exists. Vendors and maintainers are invited to submit primary evidence through ceedframework.com.

This Matrix records what adjacent tools supply on the review date. Evidence about the CEED Model itself comes from the published CEED Specification and Conformance Pack and, as implementations operate, from their operational results. The October 8, 2026 correction to the T7a self-check score is recorded after §1.

12. Evidence register

Every comparator claim in this Matrix is tied to the following first-party source register. Inspection depth describes how the source was read: full page, excerpt, repository or secondary. All verification dates are September 9, 2026 unless the row states otherwise.

Comparator	References	Version / state	Inspection depth	Verification date
IBM Bob	https://bob.ibm.com/docs/ide/changelog · https://bob.ibm.com/docs/ide/configuration/lifecycle-hooks · https://bob.ibm.com/docs/ide/security/group-policies · https://bob.ibm.com/docs/ide/configuration/bobignore	Bob 2.1.0, August 2026; Bob Shell 2.0.2	full page (all listed sources)	September 9, 2026
AWS Kiro	https://kiro.dev/docs/permissions/ · https://kiro.dev/docs/custom-agents/configuration-reference/ · https://kiro.dev/docs/enterprise/governance/permissions/ · https://kiro.dev/docs/specs/ · https://kiro.dev/docs/specs/feature-specs/requirements-first/ · https://kiro.dev/docs/custom-agents/built-in/	IDE 1.x, CLI 3.x; permissions page updated September 2, 2026; configuration reference August 4, 2026	full page (permissions, configuration, enterprise governance, specs pages); excerpt (built-in agents)	September 9, 2026

Comparator	References	Version / state	Inspection depth	Verification date
Kosli	https://docs.kosli.com/understand_kosli/how_kosli_works · https://docs.kosli.com/understand_kosli/controls · https://docs.kosli.com/getting_started/enforce_policies · https://docs.kosli.com/getting_started/policies/ · https://docs.kosli.com/client_reference/kosli_attest_decision · https://docs.kosli.com/tutorials/working_with_controls · https://docs.kosli.com/administration/managing_users/roles_in_kosli	Current; controls and decisions in beta, enabled per organization	full page (how it works, controls, enforcement, decision, controls tutorial, roles); excerpt (policies)	September 9, 2026
GitHub Enterprise AI Controls	https://github.blog/changelog/2026-02-26-enterprise-ai-controls-agent-control-plane-now-generally-available/ · https://docs.github.com/en/copilot/concepts/policies	GA February 26, 2026; enterprise MCP allowlists in public preview	full page (GA announcement); excerpt (Copilot policies)	September 9, 2026
GitHub Spec Kit	https://github.com/github/spec-kit/blob/main/workflows/README.md · https://github.com/github/spec-kit/blob/main/workflows/speckit/workflow.yml · https://github.com/github/spec-kit/blob/v1.0.4/workflows/speckit/workflow.yml · https://github.com/github/spec-kit/pull/2699	v1.0.4; schema_version 1.0; workflow v1.0.1; requires speckit_version >=0.8.5; PR 2699 Open	full page (workflow README, main workflow, v1.0.4 workflow); repository (PR 2699)	September 9, 2026
Tessl	https://docs.tessl.io/ · https://tessl.io/registry/tessl-labs/spec-driven-development · https://docs.tessl.io/tutorials/setting-up-agentic-code-review	Platform current; SDD tile v2.0.1, MIT; Review App beta	full page (platform documentation, registry tile, Review App tutorial)	September 9, 2026
in-toto layouts	https://github.com/in-toto/specification/blob/master/in-toto-spec.md §4.3.3.1 “Rule processing” · https://github.com/in-toto/community	Specification 1.0; ITE amendment process; CNCF	full page (§4.3.3.1); repository (community material)	September 9, 2026
in-toto Attestation Framework	https://github.com/in-toto/attestation spec README · https://github.com/in-toto/attestation/blob/main/docs/validation.md · https://raw.githubusercontent.com/slsa-framework/slsa-verifier/main/docs/npm.md	Spec v1.2; tagged release 1.2.0; SemVer MAJOR.MINOR matches spec version	repository (spec README); full page (validation, npm interoperability record)	September 9, 2026
SLSA provenance	https://slsa.dev/spec/v1.2/ · https://github.com/slsa-framework/slsa-verifier · https://raw.githubusercontent.com/slsa-framework/slsa-verifier/main/docs/npm.md	v1.2 current approved version; slsa-verifier is no longer actively maintained	excerpt (SLSA v1.2 track index); full page (verifier repository and npm interoperability record)	September 9, 2026
SLSA Source Track	https://slsa.dev/spec/v1.2/source-requirements · https://slsa.dev/spec/v1.2/verifying-source	v1.2, status Approved	full page (source requirements); excerpt (verifying source)	September 9, 2026

Comparator	References	Version / state	Inspection depth	Verification date
gittuf	https://github.com/gittuf/gittuf/blob/v0.16.0/docs/design-document.md · https://github.com/gittuf/gittuf/releases · https://github.com/gittuf/gittuf/security/advisories/GHSA-qc8j-pcph-6q9x · https://github.com/gittuf/gittuf/security/advisories/GHSA-vxvc-cg7j-rwqj	v0.16.0, released September 4, 2026, plus metadata-version update; design document last modified March 30, 2026	full page (design document and advisories); repository (release record)	September 9, 2026
Cosign	https://docs.sigstore.dev/cosign/attestation/ (moved; current location https://docs.sigstore.dev/cosign/verifying/attestation/, checked November 1, 2026) · https://raw.githubusercontent.com/sigstore/cosign/v3.1.3/doc/cosign_verify-attestation.md · https://github.com/sigstore/cosign/security/advisories/GHSA-fx35-mq7g-6g98 · https://github.com/sigstore/cosign/releases/tag/v3.1.3	Assessed: v3.1.3	excerpt (attestation documentation); full page (v3.1.3 verification documentation and advisories)	September 9, 2026
OPA / Conftest	https://www.openpolicyagent.org/docs/management-bundles · https://www.openpolicyagent.org/docs/security · https://www.conftest.dev/sharing/ · https://github.com/open-policy-agent/conftest · https://github.com/microsoft/regorus	OPA current; Regorus conformance against OPA v1.2.0	full page (OPA bundles and security, Conftest sharing, Regorus); repository (Conftest)	September 9, 2026
CDEvents	https://github.com/cdevents/spec/releases · https://raw.githubusercontent.com/cdevents/spec/v0.5.1/spec.md · https://raw.githubusercontent.com/cdevents/spec/v0.5.1/continuous-operations.md · https://raw.githubusercontent.com/cdevents/spec/v0.5.1/continuous-integration.md	v0.5.1 latest (supersedes the v0.4.0 figure in the repository README)	repository and full page	September 9, 2026
AGENTS.md	https://agents.md/	No versioned specification release identified; AAIF stewardship under the Linux Foundation	full page	September 9, 2026
Google Binary Authorization	https://docs.cloud.google.com/binary-authorization/docs/using-breakglass · Binary Authorization policy-model and continuous-validation documentation on `cloud.google.com`	Current; CV with platform policies supersedes legacy continuous validation	full page (breakglass); excerpt (policy model and Continuous Validation)	September 9, 2026
CODEOWNERS + GitHub rulesets	https://docs.github.com/en/repositories/configuring-branches-and-merges-in-your-repository/managing-rulesets/about-rulesets · https://docs.github.com/en/repositories/configuring-branches-and-merges-in-your-repository/managing-protected-branches/about-protected-branches · GitHub ruleset property reference · first-party `github/gh-stack` discussion of `require_last_push_approval`, August 2026	Current ruleset model	full page (rulesets and protected branches); excerpt (property reference and first-party discussion)	September 9, 2026

Comparator	References	Version / state	Inspection depth	Verification date
Buf	https://buf.build/docs/bsr/checks/ · https://buf.build/docs/bsr/checks/review-commits/ · https://buf.build/docs/bsr/checks/policies/ · https://buf.build/docs/bsr/admin/roles/	Enterprise BSR	full page	September 9, 2026
CodeQL	https://docs.github.com/en/code-security/concepts/code-scanning/merge-protection · https://docs.github.com/en/code-security/how-tos/find-and-fix-code-vulnerabilities/manage-your-configuration/set-merge-protection · https://docs.github.com/en/code-security/how-tos/find-and-fix-code-vulnerabilities/configure-code-scanning/configure-code-scanning	Current	full page	September 9, 2026
Playwright	https://playwright.dev/docs/ci-intro · https://playwright.dev/docs/ci · https://docs.github.com/en/repositories/configuring-branches-and-merges-in-your-repository/managing-protected-branches/about-protected-branches	Current	full page	September 9, 2026
Argo Rollouts	https://argo-rollouts.readthedocs.io/en/stable/features/analysis/ · https://argo-rollouts.readthedocs.io/en/stable/architecture/	Stable	full page (analysis); excerpt (architecture)	September 9, 2026
LaunchDarkly	https://launchdarkly.com/docs/home/releases/approvals-settings · https://launchdarkly.com/docs/home/releases/approval-config · https://launchdarkly.com/docs/home/releases/approval-requests · LaunchDarkly guarded-rollout documentation and changelog	Guardian plan required for guarded rollouts; sequential testing since January 2026	full page (approval documentation); excerpt (guarded rollouts and changelog)	September 9, 2026
OpenSpec	https://github.com/Fission-AI/OpenSpec · https://github.com/Fission-AI/OpenSpec/pull/1277 (**Closed**) · https://raw.githubusercontent.com/Fission-AI/OpenSpec/main/docs/cli.md · https://raw.githubusercontent.com/Fission-AI/OpenSpec/main/src/core/archive.ts	main; corresponding release tag not established	repository (README and PR); full page (CLI and archive implementation)	September 9, 2026
CEED self-check	Assessed CEED working package (Specification and Change Passport Schema; Conformance Pack Guide) as of September 9, 2026; published in revised form as Public Review Draft 2 on October 26, 2026 at ceedframework.com/publications	Assessed Version 1.0 working package; not yet publicly released	full page	September 9, 2026

Closing note. The opening Finding is the controlling interpretation of this dated review. The Matrix records how the assessed technologies distribute authorization, protection, evidence, gating, lifecycle continuity and interoperability; it does not convert those observations into an overall product ranking or an exclusivity claim for the CEED Model.