Twenty provider boundaries and exclusions

Use this reference to decide whether an intended integration belongs inside the reviewed Twenty connector. The connector is a fixed, workspace-aware mapping; it is not a generic transport for the complete provider API.

SECTION	Connectors
TYPE	Connector
PROTOCOL	AIP 1.0
SOURCE	docs/connectors/twenty/reference/provider-boundaries-and-exclusions.md

An integration is in scope only when its operation, route, identity, data, governance, execution, and evidence needs fit the boundaries below.

Version boundary

Identity	Reviewed value
AIP source	`97be86e9efedf07ecf1783b03800f683f107fb04`
Provider mapping source	`96a24563674313a3071d359bfccaf33d5e130ab8`
Connector ID	`twenty`
Connector profile	`aip.connector.twenty.v1`
AIP workspace version	`1.0.0`

The provider revision is the source identity from which the mapping was derived. It is not a runtime version negotiation, deployed-provider digest, or proof that a later provider release remains compatible.

The connector does not fetch, compile, or inspect the provider source at startup. Live compatibility requires a separately identified deployment and evidence campaign.

Supported operation boundary

The closed catalogue contains:

Family	Operations
Record query and aggregation	4
Record create and update	4
Record delete, restore, and merge	7
Workspace metadata	5
OpenAPI documents	2
Total	**22**

An optional host file can admit any non-empty subset. It cannot define a new suffix, route, method, risk, schema, or contract.

Standard and custom record objects share the same bounded object-name grammar. That dynamic object selector does not make provider paths dynamic: every route still comes from a fixed operation template.

The metadata selector is narrower. It accepts exactly twelve source-owned resource names.

Explicit API exclusions

The connector does not provide:

- arbitrary REST methods or paths;
- arbitrary GraphQL documents, variables, or operation names;
- provider API-token creation, listing, rotation, or revocation;
- arbitrary webhook endpoint forwarding;
- caller-selected origins, workspaces, provider accounts, or credentials;
- capability generation from an observed OpenAPI document;
- every operation present in the upstream provider source;
- a generic passthrough for unsupported record or metadata endpoints.

Do not encode a missing path into `object`, `resource`, `id`, query, or body. Those fields are validated or appended within one fixed request template.

Add an operation through reviewed connector source, schema, tests, admission, documentation, and qualification. Do not weaken route validation to bypass the catalogue.

Provider origin and authentication boundary

The host fixes one provider origin, one workspace UUID, and one API-token file when its process starts.

The origin:

- contains no credentials, path prefix, query, or fragment;
- requires HTTPS except for controlled loopback or explicitly allowed private HTTP;
- must contain a host;
- never follows redirects.

The connector sends the provider credential only as a Bearer header to a route joined to that origin. Action input cannot replace the token or destination.

The host validates AIP tenant and external-account metadata independently from the provider workspace. Equality between external-account ID and workspace UUID is a deployment policy, not an automatic connector check.

Route and query boundary

Every record object must match:

```
TEXT
^[A-Za-z][A-Za-z0-9_]{0,127}$
```

Every ID selected by the connector must be an RFC 4122 UUID version 1 through

5. These validators exclude path separators and traversal but do not establish

object or record ownership.

Each operation has a closed query-key set. Unknown keys and `null` values fail. Arrays and objects are compact-JSON encoded into one provider query value; the connector does not interpret provider filter or order syntax.

The connector does not provide automatic pagination, cursor iteration, field projection, target-count estimation, provider query planning, or relationship expansion beyond the admitted `depth` value.

Provider schema boundary

Outer AIP schemas validate object, UUID, query, body shape, batch counts, duplicate alternatives, merge bounds, and selected metadata dependencies.

The connector does not validate arbitrary record or metadata body fields against the configured workspace schema. It also does not normalize provider record, metadata, OpenAPI, or error bodies into a stable cross-version model.

The two OpenAPI Resources can expose current provider descriptions. The connector does not:

- validate those documents as OpenAPI;
- resolve references or calculate canonical digests;
- reconcile them with the fixed mapping;
- execute operations found in them;
- turn observed fields into authorization policy.

Applications must pin an observed schema digest and maintain object and field allowlists when workspace-specific bodies are consequential.

Execution boundary

All 22 operations are synchronous and non-streaming. The connector advertises no:

- connector-managed asynchronous execution;
- result or event streaming;
- Action cancellation;
- transaction planning or commit;
- reconciliation operation;
- compensation capability;
- provider rollback.

Fourteen mutations declare `rollback_not_supported`. A separate provider operation, such as restore after soft delete, is a new governed intent rather than an AIP compensation contract.

The host may receive lifecycle drain or shutdown controls. Those process controls do not make a dispatched provider mutation cancellable.

Approval, idempotency, and retry boundary

All fourteen mutations require tenant-policy approval and an external-account-scoped key. The approval window is 900,000 ms; the published key TTL is 604,800,000 ms.

The connector advertises retry support for eight reads plus:

- `record.create`;
- `record.create_many`;

- `record.update`.

Those three writes are safe only with the original key and unchanged canonical input. The other eleven mutations advertise unsafe retry.

Approval does not validate provider schema or make an effect reversible. A key does not prove that every provider endpoint deduplicates requests. A retryable flag does not authorize a replacement Action.

Data and result boundary

Every capability declares confidential data, possible personal information, and required redaction. The connector returns decoded provider JSON inside:

```
JSON
{
  "http_status": 200,
  "provider_request_id": null,
  "body": {}
}
```

It does not apply a general field-redaction policy to successful record, metadata, or OpenAPI bodies. The special `webhooks` metadata path recursively removes fields named `secret`.

Provider error projection retains only bounded `statusCode`, `error`, `message`, `messages`, and `code` fields. It omits other provider error data.

The connector does not own tenant retention, data residency, legal hold, field-level authorization, application output minimization, or evidence access policy.

Size and timeout boundary

Boundary	Reviewed value
Connect timeout	10 seconds
Total provider request timeout	120 seconds
Provider request JSON	16 MiB
Provider response default	64 MiB
Provider response maximum	256 MiB
Encoded query value	32 KiB
Encoded provider URL	128 KiB
Record batch	200 records
Webhook body	4 MiB

Raising the configurable response ceiling does not change other limits. The connector does not split oversized requests or return partial oversized responses.

Webhook boundary

The host optionally exposes one fixed route, `/webhooks/twenty`, backed by one startup-loaded HMAC secret and one configured workspace.

It requires exact timestamp, signature, and nonce headers; validates HMAC over timestamp, colon, and raw body; fences the nonce; matches payload workspace; maps one deterministic event; and durably queues central publication.

The ingress does not:

- accept an Action-provided secret;

- accept another workspace under the same binding;
- define a source-owned event-name enum;
- redact the complete provider event payload;
- guarantee central acknowledgement before every HTTP `200`;
- make same-nonce replay safe after a late local failure.

Provider webhook registration is a separate critical metadata mutation. A matching registration does not prove ingress, replay, storage, or publication.

Error and recovery boundary

The connector categorizes authentication, remote temporary, remote rejection, transport, storage, invalid response, and invalid request failures.

Mutation uncertainty can arise after dispatch even when a concrete provider result is unavailable. The connector exposes no automatic provider-state lookup or reconciliation contract.

Recovery must retain the original Action, key, approval, provider request identity, output, and receipts. Use a bounded authorized read before another mutation. Do not erase uncertainty by assigning a new identity.

Deployment and topology boundary

The repository contains a standalone `aip-host-twenty` binary and generic host image support. The reviewed five-product connector-fleet Compose topology does not contain a Twenty service.

Therefore:

- the connector source is deployable through the shared host architecture;
- the pinned topology does not provide a source-owned Twenty local quickstart;
- a separate deployment must define origin, workspace, secrets, database,

identities, registry admission, ingress, and operations;

- another topology requires its own immutable artifact and evidence identity.

Do not infer deployment or qualification from compilation support alone.

Evidence boundary

Source review can establish implemented operations and guards. It cannot by itself establish:

- a deployed provider version or workspace schema;
- exact host image digest or admitted registry row;
- current route eligibility and credential status;
- successful record, metadata, OpenAPI, or webhook execution;
- multi-replica replay or outbox recovery;
- production readiness or live compatibility.

Qualification claims must name the exact source, artifact, topology, provider, workspace, test procedure, results, retained evidence, and limitations.

Decide how to handle an out-of-scope need

Need	Correct response
A missing fixed provider operation	Add and review a typed connector operation
Arbitrary GraphQL	Build a separately governed connector; do not tunnel it through this mapping
Dynamic provider path	Define a bounded route and schema in source
API-token lifecycle	Use provider and deployment secret-management procedures
Asynchronous or streaming workflow	Design an explicit AIP contract and implementation before advertising it
Cancellation or transaction semantics	Add source-owned protocol behavior and provider evidence; do not infer it
Automatic schema adaptation	Build a reviewed generator and compatibility gate outside current claims
Unsupported live provider revision	Re-pin, review drift, test, qualify, and update evidence

Related documentation

- Twenty connector overview
- Twenty capability index
- Twenty OpenAPI documents
- Twenty webhook security, replay, and events
- Connector upstream baselines