Cal.diy operator-only and excluded routes

Use this reference when a Cal.diy provider route has no corresponding AIP capability. It distinguishes deliberate control-plane exclusions from the 81 published business operations and maps two deprecated aliases to their canonical capabili

SECTION	Connectors
TYPE	Connector
PROTOCOL	AIP 1.0
SOURCE	docs/connectors/cal-diy/reference/operator-only-and-excluded-routes.md

An excluded route is not a hidden capability. The connector host does not mount an operator API for it, and the business-operation client has no catch-all path. Do not call the provider directly with connector credentials to bypass this boundary.

Pinned inventory

Inventory	Entries	Published as AIP capabilities
Business operations	81	Yes
Operator-only excluded method-path pairs	38	No
Deprecated alias method-path pairs	2	No
Audited total	**121**	**81**

At source revision `97be86e9efedf07ecf1783b03800f683f107fb04`, tests require all 40 excluded pairs to be unique and disjoint from every published method-path pair.

The source-pinned Cal.diy upstream revision is `f00434927386c9ecdcbd7e6c5f82d22044a245bc`. A later upstream or connector revision requires a fresh inventory; this page is not a claim about every route in a current provider deployment.

Interpret the boundary

Term	Meaning here
Published	Has a fixed `CalDiyOperation`, capability id, schema, contract, route, version, and governed dispatch path
Operator-only	Deliberately omitted because it provisions or rotates credentials, establishes an external connection or browser session, administers OAuth clients, or verifies resource ownership
Deprecated alias	Older provider path omitted in favor of one canonical published route with the same intended provider semantics
Unknown	Neither published nor in this pinned exclusion inventory; do not infer support or safety

`Operator-only` describes the trust plane that should own the provider action. It does not mean `aip-host-cal-diy` exposes these paths to a privileged caller.

The excluded entries have no connector-owned input schema, risk level, approval policy, idempotency scope, API-version selection, request partition, response schema, reconciliation path, or qualification claim. Do not copy

those details from a neighboring public capability.

Decide what to do when a capability is absent

1. Search the admitted capability catalog for the intended business outcome,

not for a provider URL fragment.

2. If a published capability exists, use its exact schema and contract.
3. If the route appears below, stop the agent workflow and hand the task to a

separately authenticated provider or product operator process.

4. If the route is a deprecated alias, migrate to the named canonical

capability.

5. If it appears nowhere, treat it as unsupported at this revision.
6. Propose a connector change only after deciding that the outcome belongs in

the business trust plane rather than the credential or ownership plane.

An unknown capability id fails before provider dispatch as `connector.cal_diy.invalid_action`. Changing the capability id to resemble an upstream path cannot create a route.

Why HTTP methods do not define safety

The operator-only inventory contains 18 `GET`, 13 `POST`, 4 `DELETE`, and 3 `PATCH` pairs. Several `GET` paths contain `connect`, `save`, `callback`, or similar session-establishment language. Treat the route purpose and trust boundary as authoritative; do not assume an excluded `GET` is a harmless read.

The category labels below explain why a route is grouped. The method-path pairs remain the exact source inventory.

OAuth-client administration routes

These 18 routes create, inspect, change, refresh, or delete OAuth clients and their delegated users or webhooks. They can alter broad credential and delegation state and remain outside agent-callable business operations.

Method	Provider path
`DELETE`	`/v2/oauth-clients/{client_id}`
`DELETE`	`/v2/oauth-clients/{client_id}/users/{user_id}`
`DELETE`	`/v2/oauth-clients/{client_id}/webhooks`
`DELETE`	`/v2/oauth-clients/{client_id}/webhooks/{webhook_id}`
`GET`	`/v2/auth/oauth2/clients/{client_id}`
`GET`	`/v2/oauth-clients`
`GET`	`/v2/oauth-clients/{client_id}`
`GET`	`/v2/oauth-clients/{client_id}/users`
`GET`	`/v2/oauth-clients/{client_id}/users/{user_id}`
`GET`	`/v2/oauth-clients/{client_id}/webhooks`
`GET`	`/v2/oauth-clients/{client_id}/webhooks/{webhook_id}`
`PATCH`	`/v2/oauth-clients/{client_id}`
`PATCH`	`/v2/oauth-clients/{client_id}/users/{user_id}`
`PATCH`	`/v2/oauth-clients/{client_id}/webhooks/{webhook_id}`

Method	Provider path
POST	/v2/oauth-clients
POST	/v2/oauth-clients/{client_id}/users
POST	/v2/oauth-clients/{client_id}/users/{user_id}/force-refresh
POST	/v2/oauth-clients/{client_id}/webhooks

The standalone host can authenticate outbound business requests with one deployment-selected Bearer value or Cal platform client pair. That does not make OAuth-client administration a public connector capability.

Credential, session, and connection routes

These 12 routes mint or refresh credentials, establish calendar, conferencing, or Stripe connections, or save connection state. Browser redirects, callbacks, credential custody, and product administration belong outside the business action plane.

Method	Provider path
GET	/v2/calendars/{calendar}/connect
GET	/v2/calendars/{calendar}/save
GET	/v2/conferencing/{app}/oauth/auth-url
GET	/v2/conferencing/{app}/oauth/callback
GET	/v2/stripe/check
GET	/v2/stripe/connect
GET	/v2/stripe/save
POST	/v2/api-keys/refresh
POST	/v2/auth/oauth2/token
POST	/v2/calendars/ics-feed/save
POST	/v2/calendars/{calendar}/credentials
POST	/v2/oauth/{provider}/refresh

Public calendar and conferencing capabilities can inspect or use connections that already exist. They do not grant permission to create browser sessions, save credentials, refresh provider tokens, or connect a financial account.

Verified-resource ownership routes

These eight routes list verified email or phone resources or request and submit ownership codes. Verification can establish identity and authority outside the connector's tenant-bound business input, so it remains operator-owned.

Method	Provider path
GET	/v2/verified-resources/emails
GET	/v2/verified-resources/emails/{resource_id}
GET	/v2/verified-resources/phones
GET	/v2/verified-resources/phones/{resource_id}
POST	/v2/verified-resources/emails/verification-code/request
POST	/v2/verified-resources/emails/verification-code/verify
POST	/v2/verified-resources/phones/verification-code/request
POST	/v2/verified-resources/phones/verification-code/verify

Do not place verification codes, ownership proofs, or provider-administration credentials in an AIP business action merely to reach these routes.

Deprecated calendar-event aliases

The pinned upstream inventory included two singular `/event/` aliases. The connector publishes only the plural `/events/` routes.

Excluded alias	Canonical capability	Canonical provider path
`GET /v2/calendars/{calendar}/event/{event_uid}`	`cap:cal_diy:calendar.event.get`	`GET /v2/calendars/{calendar}/events/{event_uid}`
`PATCH /v2/calendars/{calendar}/event/{event_uid}`	`cap:cal_diy:calendar.event.update`	`PATCH /v2/calendars/{calendar}/events/{event_uid}`

Applications should migrate by selecting the canonical capability and its published input schema. Do not rewrite a direct provider request by changing only `event` to `events`: AIP identity, approval, idempotency, tenant routing, and durable result handling still apply. The update capability is a governed mutation; the get capability is a governed read.

Handle an operator workflow separately

If a deployment legitimately needs an excluded provider function, keep it outside the business connector and define a separate operator-owned workflow with its own:

- authenticated human or administrative service identity;
- least-privilege provider credential and explicit secret custody;
- browser state and callback validation where a session is involved;
- ownership-proof handling and restricted audit evidence;
- change approval, rotation, incident, and revocation procedure;
- provider-specific versioning, validation, and recovery behavior.

This page does not define such an operator API or authorize a particular direct provider call. The provider's current administrative documentation and the deployment's security policy own that separate workflow.

Review a proposed connector expansion

Moving a route into the business surface is a protocol-facing security change, not an inventory edit. Review all of the following together:

1. explain the user outcome and why it belongs outside the operator plane;
2. pin the exact upstream revision, method, version, and canonical path;
3. define one bounded capability id and closed input and output schemas;
4. partition path, query, and body fields without arbitrary pass-through;
5. classify data, PII, side effects, risk, and credential scope;
6. define approval, idempotency, transaction, compensation, and reconciliation;
7. keep provider credentials, verification codes, and raw secrets out of input;
8. add positive, negative, boundary, restart, and provider-contract tests;
9. remove the exact pair from the exclusion inventory and preserve uniqueness

and disjointness checks;

10. regenerate capability metadata, update public documentation, and qualify the

exact artifact before making a live claim.

Do not add a generic method-plus-path capability. That would erase the fixed schema, policy, credential, and audit boundary established by the connector.

Audit the exclusion boundary

Check	Expected result at this revision
Published capability ids	81 unique ids
Operator-only inventory	38 unique method-path pairs
Deprecated alias inventory	2 unique method-path pairs
Combined excluded inventory	40 unique pairs
Published versus excluded pairs	Disjoint
Manifest compatibility metadata	`operator_only_routes_excluded: 38`; `deprecated_aliases_excluded: 2`
Unknown capability invocation	Rejected before provider dispatch
Arbitrary provider proxy	Absent

These checks establish an implementation boundary. They do not show that a current provider deployment exposes, removes, or preserves any excluded route.

Security and evidence boundaries

- Provider-route discovery is not permission to invoke the route.
- A provider `GET` verb is not evidence that a control-plane action is read-only.
- Connector credentials must not be reused as a general provider admin token.
- An excluded route has no inherited AIP approval or idempotency guarantee.
- A deprecated alias has no compatibility promise from this connector.
- Source tests prove inventory separation, not live provider behavior.

Related documentation

- API versions and public provider routes
- Cal.diy capability index
- Pinned upstream baselines
- Connector contract
- Build a connector