Cal.diy user webhooks

Use these capabilities to list, read, create, update, or delete user-level Cal.diy webhook subscriptions. This page owns five management operations and the exact trigger catalog they accept.

SECTION	Connectors
TYPE	Connector
PROTOCOL	AIP 1.0
SOURCE	docs/connectors/cal-diy/capabilities/webhooks.md

Subscription management is distinct from inbound delivery. These operations configure where Cal.diy sends events; they do not verify signatures, enforce a replay fence, or prove that any delivery occurred.

All five operations use provider API version `2024-08-13`. Managed webhook payloads use the separate fixed version `2021-10-20`.

Choose an operation

Capability	Provider request	Risk	Approval
`cap:cal_diy:webhook.list`	`GET /v2/webhooks`	Low	No
`cap:cal_diy:webhook.get`	`GET /v2/webhooks/{webhook_id}`	Low	No
`cap:cal_diy:webhook.create`	`POST /v2/webhooks`	Medium	Required
`cap:cal_diy:webhook.update`	`PATCH /v2/webhooks/{webhook_id}`	Medium	Required
`cap:cal_diy:webhook.delete`	`DELETE /v2/webhooks/{webhook_id}`	High	Required

List or read subscriptions

List accepts optional query pagination:

```
JSON
{
  "skip": 0,
  "take": 50
}
```

`skip` is an integer of at least `0`. `take` is from `1` through `250`. Either field may be omitted. Additional fields are rejected.

Read one subscription with a non-empty provider id:

```
JSON
{
  "webhook_id": "provider-webhook-uid"
}
```

Both reads are low risk and approval-free at the capability level. Their descriptions require secrets to be redacted from returned provider data.

Supported triggers

Create and trigger-changing updates accept a non-empty array of unique values from this exact catalog:

```
TEXT
BOOKING_CREATED
BOOKING_PAYMENT_INITIATED
BOOKING_PAID
BOOKING_RESCHEDULED
BOOKING_REQUESTED
BOOKING_CANCELLED
BOOKING_REJECTED
BOOKING_NO_SHOW_UPDATED
FORM_SUBMITTED
MEETING_ENDED
MEETING_STARTED
RECORDING_READY
RECORDING_TRANSCRIPTION_GENERATED
OOO_CREATED
AFTER_HOSTS_CAL_VIDEO_NO_SHOW
AFTER_GUESTS_CAL_VIDEO_NO_SHOW
FORM_SUBMITTED_NO_EVENT
DELEGATION_CREDENTIAL_ERROR
WRONG_ASSIGNMENT_REPORT
```

Trigger names are case-sensitive. The catalog defines admitted subscription input; it does not promise that a given Cal.diy account will emit every event.

Create a subscription

Create requires active state, destination, triggers, and an opaque secret reference:

```
JSON
{
  "active": true,
  "subscriberUrl": "https://aip.example.com/connectors/cal-diy/webhooks/bookings",
  "triggers": ["BOOKING_CREATED", "BOOKING_RESCHEDULED"],
  "webhook_secret_ref": "bookings"
}
```

Field	Constraint
`active`	Required boolean
`subscriberUrl`	Required absolute HTTPS URL after connector checks
`triggers`	Required non-empty unique trigger array
`webhook_secret_ref`	Required opaque deployment secret selector
`version`	Optional input; when present, exactly `2021-10-20`

The connector always inserts `version: 2021-10-20`. Custom payload templates are not supported; managed subscriptions use the standard Cal.diy payload. Additional input fields, including raw `secret`, are rejected.

Bind the destination to the secret selector

The destination policy is deny-by-default. A trusted deployment configures one or more credential-free HTTPS prefixes without query or fragment. For a secret reference such as `bookings`, the candidate URL path must equal the configured prefix path followed by `bookings`, on the same scheme, host, and effective port.

The candidate URL must also contain no username, password, query, or fragment. This binds the public ingress selector to the deployment secret selector rather than accepting an arbitrary callback URL.

`webhook_secret_ref` must be 1 through 128 characters. Its first character must be alphanumeric; remaining characters may be alphanumeric, `.`, `_`, `~`, or `-`.

Keep the raw secret outside the action

The action carries only `webhook_secret_ref`. Before provider dispatch, the connector resolves that reference through the deployment secret provider, injects the resolved value as provider field `secret`, and removes `webhook_secret_ref` from the request body.

The raw resolved value is never an AIP input or returned application value. The common output scrubber replaces provider `secret` and token-like field values with `[REDACTED]`.

Create is medium risk and declares `webhook.delete` as best-effort compensation. Compensation requires separate approval and is not automatic.

Update a subscription

Update requires only `webhook_id`. It may change `active`, `subscriberUrl`, `triggers`, `webhook_secret_ref`, or the fixed `version`:

```
JSON
{
  "webhook_id": "provider-webhook-uid",
  "subscriberUrl": "https://aip.example.com/connectors/cal-diy/webhooks/bookings-v2",
  "webhook_secret_ref": "bookings-v2",
  "triggers": ["BOOKING_CREATED", "BOOKING_CANCELLED"]
}
```

Destination and secret changes are coupled:

- changing `subscriberUrl` requires a valid `webhook_secret_ref`;
- supplying `webhook_secret_ref` for rotation requires `subscriberUrl`;
- the destination must bind to that exact reference under the configured

prefix policy.

The connector resolves and injects the new secret with the same create-time boundary. It also inserts the fixed webhook version.

An identifier-only update can pass the published schema without expressing a useful change. Include at least one intentional patch field. The connector reads the exact subscription before update, but this preflight does not lock provider state or validate eventual delivery.

Update is medium risk and declares no compensating capability.

Delete a subscription

Delete accepts only the non-empty `webhook_id`. The connector reads that subscription before deletion.

Delete is high risk and destructive, with no rollback contract. Recreating a subscription later is a new provider action and does not restore the prior subscription or its delivery history.

Shared read contract

The two reads are synchronous, retry-safe, and eligible for connector retry. Their idempotency key is optional, action-scoped, and collision-checked against the input hash. No capability-level human approval is required.

Shared mutation contract

All three mutations require:

Contract field	Value
Human approval	Required
Approval selector	Tenant policy
Approval validity window	900,000 ms
Approval evidence	Reason, input snapshot, policy decision
Idempotency	Required; external-account scope
Collision behavior	Revalidate input hash
Connector retry	Not supported
Retry safety	Unsafe
Execution	Synchronous only
Transaction modes	`dry_run`, `plan`, `commit`, `reconcile`
Plan before commit	Not required
Dry-run fidelity	Policy and schema only

Create and update are medium risk. Delete is high risk, destructive, and uses the destructive approval reason. None carries the `send_message` side-effect marker.

Only create declares best-effort delete compensation. Update and delete declare rollback as not supported.

Data and result contract

All five operations are restricted, contain PII, and require redaction. URLs, trigger configuration, subscription identifiers, and provider metadata must remain within the authorized tenant and application boundary.

The connector returns the validated provider JSON body. A non-empty string `status` is required. `data`, `pagination`, and additional provider fields are optional and provider-owned. Secret-, key-, and token-named values are scrubbed recursively; ordinary PII is not removed by that targeted redactor.

Failures and recovery

Failure	Relevant cause	Safe response
`connector.cal_diy.invalid_action`	Invalid trigger, URL, selector binding, rotation pair, version, or unknown field	Correct input before dispatch
`connector.cal_diy.policy`	Missing mutation key or unresolved secret reference	Preserve approved input; fix the key or deployment secret
`connector.cal_diy.state`	Secret resolver or durable state failed	Repair the deployment dependency before retry
`connector.cal_diy.idempotency_collision`	One key owns different mutation input	Stop and inspect the original action
`connector.cal_diy.in_flight`	A matching webhook mutation is executing	Read durable state and wait
`connector.cal_diy.outcome_unknown`	Prior provider dispatch cannot be proved	Reconcile; do not create a replacement key
`connector.cal_diy.authentication`	Provider returned `401` or `403`	Verify account binding and provider credentials
`connector.cal_diy.state_conflict`	Provider returned `409` or `412`	Refresh subscription state and review the mutation
`connector.cal_diy.rate_limited`	Provider returned `429`	Retry reads within policy; reconcile mutations
`connector.cal_diy.transport`	Provider exchange failed	Retry reads only; treat dispatched mutations as uncertain
`connector.cal_diy.invalid_provider_output`	Successful response lacked valid `status`	Preserve evidence and reconcile a mutation

Other permanent provider rejections, temporary provider errors, oversized responses, and durable settlement failures use the common Cal.diy error family. Gateway and lifecycle errors are described in the global error

reference.

Verify subscription management

For a read, retain the capability, pagination or webhook id, actor, tenant, external account, response status, and provider request id when present.

For a mutation, also retain the exact destination, secret reference name, trigger set, approval evidence, and original idempotency key. Never retain the resolved raw secret. Wait for a completed durable result, then read the exact subscription.

For uncertain dispatch, reconcile the original operation before another mutation. A completed create or update confirms provider acceptance of the management request, not successful signed delivery to the application.

Related documentation

- Cal.diy capability index
- Event-type webhooks
- Authentication and tenant routing
- Cal.diy configuration reference
- Webhook security and replay