Chatwoot connector changelog

Use this page to decide whether a Chatwoot connector change requires a new artifact, admission record, migration, caller update, or qualification result. It records only changes established from the reviewed Git range.

SECTION	Connectors
TYPE	Connector
PROTOCOL	AIP 1.0
SOURCE	docs/connectors/chatwoot/changelog.md

Release status: the reviewed snapshot is not a tagged release. The repository tags `034a5206` as `v1.0.0`; the documentation revision is five commits later and still declares workspace version `1.0.0`. Use full source and artifact identities to distinguish these states.

Identify the exact version

Identity	Published `v1.0.0` baseline	Reviewed source snapshot
Source commit	`034a520608eea44e6b14e61c34734bd402d989c0`	`97be86e9efedf07ecf1783b03800f683f107fb04`
Source position	Tagged	`v1.0.0-5-g97be86e`
Publication status	Published tag in the reviewed repository	Unreleased source in the reviewed ancestry
Workspace version	`1.0.0`	`1.0.0`
Native AIP version	`1.0`	`1.0`
Chatwoot capabilities	3 composites	3 composites plus 150 catalogue operations
Target placement	Bundled connector	Separately built and admitted standalone host

An operator record for the reviewed snapshot needs the full source commit, tree state, immutable image digest, manifest digest, connector version ID, release ID, and upstream identity. The string `1.0.0` cannot replace that tuple.

Reviewed source snapshot

The entries below describe Chatwoot-relevant differences between tagged `v1.0.0` and `97be86e9`. They are source changes, not evidence that an artifact was built, published, deployed, or qualified.

Added

- A frozen catalogue of 150 account-scoped Chatwoot operations, pinned to upstream commit `8818d276b954ac4f84cffd8915c99f40e43804ed`.
- A dedicated `aip-host-chatwoot` binary and the common standalone-host image path.

- Registry admission, tenant-scoped routing, replica leases, health, drain,

runtime database, and recovery integration.

- Allowed-operation filtering for provider edition and account policy.
- Bounded JSON and multipart request handling plus bounded provider responses.
- A standalone `/webhooks/chatwoot` route with durable local Event and outbox

handling.

- Product-image and controlled-fleet qualification procedures.

Changed

The public surface expands from three composite capabilities to 153 total capabilities. The three original IDs remain available:

Composite	Purpose
`cap:chatwoot:message:create`	Send one conversation message
`cap:chatwoot:conversation:status`	Read or set the normalized conversation status
`cap:chatwoot:conversation:handoff`	Assign an agent or team and optionally add a private note

Catalogue IDs use the dotted `cap:chatwoot:` suffixes from the pinned operation table. They do not rename the colon-separated composite IDs. Callers should discover the intended contract instead of transforming an ID by string convention.

All 88 catalogue mutations now require an AIP idempotency key and publish approval support. Provider requests receive the Action identity, and catalogue requests receive the stable idempotency marker. The three composites retain their own orchestration and policy contracts.

Provider transport now rejects credentials, path prefixes, query strings, and fragments in the base URL. It requires HTTPS except for explicit loopback, disables redirects, applies connection and request deadlines, and bounds accepted response bytes.

Webhook replay and delivery

Standalone webhook replay fencing moves to instance-scoped durable profile state. An ordinary duplicate no longer returns a replay error from the normal ingress path. It reconstructs the same deterministic Event ID so durable Event storage can repair a crash between replay observation and Event append.

The host persists mapped Events and queues central publication before returning success. A successful webhook response can therefore represent local durable acceptance while central delivery remains queued. Recover that outbox instead of asking Chatwoot to create a replacement event.

Security boundary made explicit

The reviewed standalone topology fixes the account ID, token file, provider origin, allowed-operation set, connector identity, artifact, manifest, and credential revision outside Action input. Catalogue mutations require approval and idempotency, and multipart names and byte counts are validated before provider dispatch.

These controls describe the reviewed connector source. They do not qualify an arbitrary Chatwoot deployment or make direct provider calls part of AIP.

Deprecated, removed, or renamed

No Chatwoot capability is removed or renamed in the reviewed range. The source declares no connector-specific deprecation or removal date. The catalogue is an addition to the three composites, not a replacement announcement.

Future upstream route removals or aliases must not be inferred from the pinned operation names. Record them only when a reviewed source or release policy defines the change and migration.

Apply the required migration

Change encountered	Required action	Do not do
Bundled connector to standalone host	Preserve account, tenant, credential, Action, approval, key, webhook, Event, and recovery evidence through the bounded fleet migration	Run both mutation paths without an explicit fence
New snapshot with the same version string	Admit a new immutable connector version with exact source, image, manifest, and release identities	Replace an existing record or identify it only as `1.0.0`
Catalogue expands from 3 to 153	Refresh discovery, allowlists, generated clients, and expected catalogue digests	Infer dotted catalogue IDs from composite strings
Catalogue mutation is adopted	Add reviewed approval, stable idempotency, provider-state verification, and incident recovery	Treat a transport failure as proof of no effect
Standalone webhook ingress is enabled	Provision shared state and outbox storage, exact secrets, authenticated routing, and duplicate recovery	Use process memory as the production replay boundary
Upstream revision changes	Re-audit all affected methods, paths, schemas, bounds, and exclusions, then requalify them	Keep the old upstream pin while promoting new provider bytes

Stabilize the standalone topology before adopting new catalogue mutations or a provider upgrade. Separate changes make rollback and evidence ownership unambiguous.

Retain the compatibility pins

Surface	Reviewed pin
Connector ID	`chatwoot`
Connector profile	`aip.connector.chatwoot.v1`
Manifest format	`aip-manifest/v1`
Workspace and host version	`1.0.0`
Native AIP version	`1.0`
Chatwoot upstream commit	`8818d276b954ac4f84cffd8915c99f40e43804ed`
Published capability count	153
Default provider-response limit	16 MiB
Maximum configurable response limit	128 MiB
Standalone host	`aip-host-chatwoot`
Standalone version-ID shape	`cver_chatwoot_1_0_0_<release-id>`

The upstream commit identifies reviewed source, not a built Chatwoot image or deployment. Changing provider bytes without changing connector source does not establish compatibility with the new artifact.

Decide when a new connector version is required

Create and admit a new immutable connector version when a change affects:

- capability IDs, schemas, contracts, risk, approval, retry, or idempotency;
- provider method, path, query, body, multipart, response, or status mapping;
- account binding, credential scope, secret handling, redirect policy, or

operation allowlisting;

- webhook HMAC, timestamp, replay scope, loop prevention, Event mapping, or

delivery durability;

- artifact bytes, dependencies, manifest digest, upstream baseline, host

lifecycle, or storage behavior.

A documentation clarification alone does not require a connector version. It still needs the source revision that proves the corrected statement.

Record future entries completely

Each future entry should state:

1. release or explicit unreleased status and UTC review date;
2. full source commit, tree, and clean or reconstructable dirty state;
3. connector, image, manifest, catalogue, release, and upstream identities;
4. added, changed, fixed, security-relevant, deprecated, and removed surfaces;
5. caller, operator, storage, webhook, and provider migration actions;
6. rollback and compatibility boundary;
7. exact connector, image, fleet, and live qualification result, or `not_run`;
8. deprecation replacement and removal date only when source defines them.

Do not reconstruct release history from filenames, mutable tags, or source presence. Mark unknown history as unknown.

Related documentation

- AIP release notes
- Migrate bundled connectors to the fleet
- Chatwoot configuration
- Chatwoot capability index
- Qualify the Chatwoot connector