# Authenticate and isolate a Chatwoot account

Use this guide to bind one standalone Chatwoot host to one verified AIP tenant, one fixed provider origin, and one Chatwoot account. The API token stays in an owner-controlled file and is added to provider requests inside the connector.

| | |
|---|---|
| SECTION | Connectors |
| TYPE | Connector |
| PROTOCOL | AIP 1.0 |
| SOURCE | docs/connectors/chatwoot/getting-started/authentication-and-account-isolation.md |

The account is deployment identity, not Action input. A caller can provide operation-specific path, query, or body values, but it cannot select another Chatwoot origin, account, token, or tenant.

## Keep the credential layers separate

Five values protect different boundaries. Do not reuse one value for another layer.

| Layer | Protects | Secret |
|---|---|---|
| Caller credential | Application to `aipd` | Yes |
| Gateway signature | `aipd` to the connector host | The gateway signing key is secret |
| Host credential revision | Route authorization before provider work | No provider secret bytes |
| Chatwoot API token | Connector host to the fixed Chatwoot account | Yes |
| Webhook HMAC secret | Chatwoot ingress to the host | Yes, and separate from the API token |

`AIPCTL_NATIVE_BEARER_TOKEN_FILE` authenticates a caller to `aipd`. It must not point to the Chatwoot API-token file. Likewise, credential revision metadata can fence an admitted route, but it does not contain or replace the provider token.

## Prerequisites

Before changing the binding, require:

- one admitted Chatwoot connector version and instance;
- one verified AIP tenant and membership;
- one stable Chatwoot origin and account ID;
- one provider token with only the permissions needed by admitted operations;
- one owner-controlled secret mount readable by the host process;
- exact host, gateway, control-plane, artifact, and registry identities;
- a rollback window in which the old replica and token can remain valid.

Record the source revision, image digest, connector version, instance, replica, tenant, external account, provider account, and credential revision. Do not record the token value.

# 1. Provision the provider token file

Create the token through the deployment's secret manager and mount it as a regular owner-controlled file. On Unix, any group or world permission bit causes startup to fail.

A representative permission shape is:

```
TEXT
-rw------- 1 <host-uid> <host-gid> ... /run/secrets/chatwoot-api-token
```

The file must contain one non-empty UTF-8 value after surrounding ASCII whitespace is removed. The Chatwoot host reads at most 16 KiB. The common reader rejects a symlink, non-regular file, replacement race, empty file, or oversized file, and zeroizes its temporary string.

Do not place the token in:

- an Action, manifest, admission package, registry row, or capability input;
- a command-line argument or ordinary environment variable;
- a base URL, image layer, Compose file, log, or evidence bundle;
- the non-secret credential ID, issuer, revision, or secret-provider reference;
- the webhook-secret file.

# 2. Configure the fixed provider boundary

Configure one origin, one account, and the token-file path:

```
TEXT
AIP_CHATWOOT_BASE_URL=https://support.example.test
AIP_CHATWOOT_ACCOUNT_ID=account_primary
AIP_CHATWOOT_API_TOKEN_FILE=/run/secrets/chatwoot-api-token
```

The origin must:

- use HTTPS, except for explicit loopback HTTP;
- contain no username, password, path prefix, query, or fragment;
- identify the deployment-owned Chatwoot service rather than an

Action-selected destination.

The connector does not follow redirects. It uses a 10-second connection timeout, a 120-second request timeout, and adds `api_access_token` inside the host process.

The account ID must contain 1 to 256 bytes, must not be blank after trimming, and must not contain control characters. Treat it as immutable instance ownership metadata.

Every `{account_id}` segment in the frozen provider catalogue is replaced with this configured value. The connector intentionally omits `account_id` from those Action schemas.

# 3. Bind the host to the same tenant and external account

Configure fleet identity separately from the provider token:

```
TEXT
AIP_CONNECTOR_HOST_TENANT_ID=tenant-support-production
AIP_CONNECTOR_HOST_MEMBERSHIP_ID=membership-chatwoot-primary
AIP_CONNECTOR_HOST_EXTERNAL_ACCOUNT_ID=account_primary
AIP_CONNECTOR_HOST_EXTERNAL_ACCOUNT_SYSTEM=chatwoot
AIP_CONNECTOR_HOST_SECRET_PROVIDER_REF=secret://production/chatwoot/primary
```

The external-account ID and system must be present together or both absent. For a tenant-owned Chatwoot instance, set both and make the external-account ID equal to `AIP_CHATWOOT_ACCOUNT_ID` through deployment policy.

The reviewed host validates the two settings independently. It does not compare the external-account ID with the Chatwoot account ID automatically. Admission review and deployment validation must detect a mismatch before routing is enabled.

The registry binding must associate only the intended tenant and admitted Chatwoot capabilities with this instance. A signed Action cannot override the verified tenant or redirect the host to another configured account.

## 4. Configure credential identity and revision fencing

The common host can attach a non-secret credential handle to its verified execution context. Configure all four handle fields together:

```
TEXT
AIP_CONNECTOR_HOST_CREDENTIAL_ID=chatwoot-primary-provider
AIP_CONNECTOR_HOST_CREDENTIAL_ISSUER=chatwoot-deployment
AIP_CONNECTOR_HOST_CREDENTIAL_SCOPES=chatwoot:account.get,chatwoot:conversation.list
AIP_CONNECTOR_HOST_CREDENTIAL_REVISION=chatwoot-primary-2026-07-27-01
AIP_CONNECTOR_HOST_CREDENTIAL_POLICY_FILE=/run/config/chatwoot-credential-policy.json
```

The ID, issuer, revision, and at least one non-empty scope are a unit. A partial handle fails startup. Use exact admitted scopes; do not copy a wildcard from a controlled fixture into a real deployment.

The reloadable policy can deny the configured or pinned revision before a provider side effect. An unreadable, invalid, or non-authorizing policy fails closed. This mechanism does not hot-reload `AIP_CHATWOOT_API_TOKEN_FILE` and does not let an Action select a different token.

## 5. Verify isolation before enabling mutations

Keep new mutation bindings disabled while admitting and checking the host. Verify in this order:

1. the host registers with the expected connector type, version, instance,

replica, artifact digest, tenant, and external account;

2. the configured provider-account health read succeeds without printing the

token;

3. discovery for the intended tenant contains the expected Chatwoot reads;
4. `cap:chatwoot:account.get` completes through the normal signed gateway path;
5. the returned account identity matches the deployment record;
6. an unauthorized tenant has no eligible Chatwoot route;
7. an Action cannot provide `account_id`, a provider origin, or credential

material;

8. a revoked credential revision is denied before provider work.

Use the deterministic quickstart for the source-owned read path. Do not begin verification with a message, handoff, delete, or other provider mutation.

If the account returned by the read differs from the deployment record, disable the binding. Correct the provider account, external-account metadata, admission material, or secret mount without weakening the equality check.

## 6. Rotate the provider token

The Chatwoot token is read when the host process starts. Replacing the mounted file under a running process is not a documented token reload.

Rotate with a bounded replica rollout:

1. create a least-privilege replacement token for the same Chatwoot account;
2. mount it at a new owner-controlled path without changing the old replica;
3. allocate a new credential revision and host configuration revision;
4. update the revision policy and admission material through their controlled

paths;

5. start a replacement replica with the new token file;
6. verify the account read and unauthorized-tenant isolation;
7. route new assignments to the replacement and drain the old replica;
8. revoke the old token only after in-flight work and rollback needs settle;
9. retain identities, timestamps, checks, drain evidence, and revocation

evidence without retaining token bytes.

Do not rotate by editing an Action, changing the account behind an unchanged instance, or reusing one replica ID for two running processes.

## Resolve authentication and isolation failures

| Symptom | Decision |
| --- | --- |
| Token file is rejected | Check file type, ownership, mode, UTF-8 content, and 16 KiB bound without printing it |
| Provider origin is rejected | Use an HTTPS origin with no credentials, path, query, or fragment |
| Account ID is rejected | Supply a non-blank 1-to-256-byte value without control characters |
| External-account configuration is rejected | Set ID and system together, then compare the ID with `AIP_CHATWOOT_ACCOUNT_ID` |
| Credential handle is rejected | Configure ID, issuer, revision, and non-empty scopes together |
| Revision policy denies the route | Keep mutations disabled and repair the policy or pinned revision |
| Host registers but discovery is empty | Compare tenant, membership, admission, capability, and route policy before changing the token |
| Account read returns `401` or `403` | Check token ownership, expiry, permissions, origin, and account; do not retry a mutation |
| Provider returns a redirect | Correct the fixed origin or provider routing; credentials are not forwarded |
| Returned account is unexpected | Disable the binding and reconcile provider account with external-account metadata |

## Related documentation

- Chatwoot connector overview
- Chatwoot connector quickstart
- Rotate connector credentials
- Connector registry and routing

- Security model