Authenticate and bind Hermes endpoints

Use this guide to bind one or more owner-configured Hermes API servers to one AIP tenant. Each endpoint receives a stable ID, a fixed base URL, and its own owner-only APISERVERKEY file. The connector then publishes endpoint-qualified capabi

SECTION	Connectors
TYPE	Connector
PROTOCOL	AIP 1.0
SOURCE	docs/connectors/hermes-agent/getting-started/authentication-and-endpoints.md

The endpoint is selected by the admitted capability ID, never by action input. A caller cannot supply a different URL, token, or tenant at invocation time.

Keep the credential layers separate

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

Layer	Protects	Configured for	Secret
Caller credential	Application to `aipd`	Native HTTP or signed-native client	Yes
Gateway signature	`aipd` to connector host	Pinned gateway DID and trust domain	Private signing material stays with the gateway
Host credential revision	Route authorization before provider work	Opaque handle, scopes, revision, and revision policy	No provider secret bytes
Hermes endpoint token	Connector host to one Hermes API server	Endpoint `api_key_file`; sent as Bearer auth	Yes
Model-provider credential	Hermes to its configured model provider	Hermes deployment	Yes; never supplied to AIP

`AIPCTL_NATIVE_BEARER_TOKEN_FILE` authenticates the caller to `aipd`. It must not point to a Hermes endpoint token. Likewise, the host credential revision fences which admitted route may perform provider work; it does not load or replace `API_SERVER_KEY`.

Prerequisites

Before changing an endpoint set, require:

- one approved AIP tenant and standalone Hermes connector instance;
- one owner-controlled Hermes API server per endpoint;
- a valid `API_SERVER_KEY` for every protected endpoint;
- an HTTPS service identity or a controlled local-network exception;
- owner-controlled regular files for endpoint descriptors and tokens;
- the common host identity, gateway trust, database, admission, and credential

revision inputs described in the connector-host reference;

- a signed admission package for the exact manifest produced by the endpoint

set;

- a non-mutating Hermes read and access to provider audit evidence.

Changing endpoint IDs, URLs, tenant values, or membership changes the connector manifest. Plan a new immutable connector version and admission package before making any of those changes. The current manifest does not project `display_name` or token bytes.

1. Choose the endpoint and tenant boundary

Give each endpoint a stable operational ID. The host trims surrounding whitespace, accepts only ASCII letters, digits, hyphens, and underscores, and normalizes the result to lowercase.

For example, `Primary_US` becomes `primary_us`. Two configured values that normalize to the same ID make startup fail. Do not encode a rotating token, replica name, or mutable deployment timestamp in the ID.

Every endpoint in one standalone host belongs to the host's exact `AIP_CONNECTOR_HOST_TENANT_ID`. An omitted descriptor `tenant_id` inherits that value. An explicit different value fails startup, and every invocation fails closed unless the trusted runtime tenant matches the endpoint tenant.

Use separate connector instances for different tenants. `display_name` is retained in endpoint configuration at this revision, but it does not affect routing, capability names, or the discovered manifest.

2. Provision one token file per endpoint

Create the token through the owner-approved Hermes process and configure the same value as `API_SERVER_KEY` on that Hermes API server. Store only the connector-side copy in a dedicated file, for example:

```
TEXT
/run/secrets/hermes-primary-api-key
```

The connector accepts a non-empty UTF-8 value of at most 16 KiB after trimming leading and trailing ASCII whitespace. On Unix, the path must be a regular non-symlink file with no group or world permission bits. A typical mounted-file mode is `0600`.

Keep these invariants:

- one endpoint token file identifies one Hermes authentication boundary;
- the file is mounted read-only into only the replicas that need it;
- the token never appears in descriptor JSON, action input, admission data,

logs, shell history, or a source repository;

- the file path may be recorded, but its bytes may not be printed as evidence;
- model-provider, MCP, and webhook credentials remain outside this file.

The connector excludes token bytes from serialized endpoint configuration and redacts its endpoint collection in debug output. Deployment logging and secret mount policy must preserve that boundary too.

3. Write the endpoint descriptor file

Create a bounded JSON array in an owner-controlled configuration file:

```
JSON
[
  {
    "id": "primary",
    "base_url": "https://hermes-primary.example.internal:8642",
    "api_key_file": "/run/secrets/hermes-primary-api-key",
    "display_name": "Primary Hermes",
    "tenant_id": "tenant-acme"
```

```
  }
]
```

Point `AIP_HERMES_ENDPOINTS_FILE` at this file. Each object has two required fields and three optional fields:

Field	Required	Meaning
`id`	Yes	Stable endpoint segment in every generated capability ID
`base_url`	Yes	Owner-selected Hermes API-server origin
`api_key_file`	No	File containing the endpoint Bearer token
`display_name`	No	Human-readable label; never a routing key
`tenant_id`	No	Explicit tenant assertion; otherwise inherits the host tenant

Unknown fields are rejected. The descriptor file must be a regular non-symlink JSON file of 1 byte through 4 MiB. On Unix it may be group- or world-readable, but it must not be group- or world-writable. It is non-secret only because it contains file references rather than token bytes.

The parser accepts 1 through 1000 descriptors. That is not the deployable limit. Each endpoint creates 35 capabilities, while the current standalone host accepts at most 512. The effective manifest limit is therefore 14 endpoints: 14 produce 490 capabilities, while 15 produce 525 and fail host preparation. This is a source limit, not a scale qualification.

4. Validate endpoint identities and destinations

Review the descriptor set before admission:

1. Normalize every endpoint ID and reject collisions.
2. Confirm that every explicit tenant equals the host tenant exactly.
3. Confirm that every URL parses with the `http` or `https` scheme.
4. Resolve each URL through owner-controlled service discovery.
5. Confirm that every token file belongs to the intended endpoint.
6. Count `35 × endpoints` and keep the result at or below 512.

The connector accepts both HTTP and HTTPS but does not prove destination ownership. Use HTTPS outside a controlled loopback or private test network. The default client follows no redirects, so a Hermes response cannot forward the Bearer token to another origin. It uses a 10-second connection timeout and a 900-second client timeout; individual operations can impose narrower bounds.

Record the resolved service identity privately. Do not add a caller-controlled proxy, URL suffix, or method to action input.

5. Bind host trust and credential revision

Configure the standalone host's common identity and trust inputs independently from the endpoint file. At minimum, the admitted record and running host must agree on:

- connector type, immutable version, instance, and replica IDs;
- tenant and optional external-account identity;
- public endpoint, host signing identity, and artifact digest;
- central gateway principal, DID, and trust domain;

- opaque credential ID, issuer, verified scopes, and revision reference;
- credential-revision policy;
- durable database and control-plane membership.

The host validates the route-pinned credential revision immediately before dispatching provider work. A revision-policy change can therefore stop an old route without changing the Hermes token. Conversely, replacing a Hermes token does not authorize a different route revision.

Use the shared connector-host configuration reference for exact fields and bounds. Keep this page's endpoint descriptor focused on Hermes only.

6. Start through an admitted deployment

Generate the manifest from the exact endpoint set, artifact, and host configuration that will run. Admission must bind that immutable manifest to the intended tenant-owned connector version before the replica becomes discoverable.

Start the host with the descriptor and secret files already mounted. Startup must fail if:

- the array is empty, oversized, or invalid JSON;
- an object has an unknown field;
- an endpoint ID is invalid or duplicated after normalization;
- a URL is invalid or uses another scheme;
- a token file is missing, unsafe, empty, invalid UTF-8, or oversized;
- an explicit endpoint tenant differs from the host tenant;
- 35 capabilities per endpoint exceed the host manifest ceiling;
- the common trust, storage, admission, or credential-revision checks fail.

Do not bypass one of these failures by launching the product connector inside `aipd` or by admitting a manifest from a different descriptor set.

7. Verify selection and authenticated access

Use an already configured signed-native client context. Set the exact tenant gateway URL and normalized endpoint ID:

```
SH
export AIP_URL=https://aip.example
export HERMES_ENDPOINT_ID=primary
```

First require the endpoint-qualified protected capability in the tenant catalog:

```
SH
aipctl capability list "$AIP_URL" \
  --signed-native \
  --capability-id "cap:hermes_agent:${HERMES_ENDPOINT_ID}:models"
```

Expected result: the bounded catalog page contains exactly the requested capability for the admitted Hermes version. A catalog result proves routing metadata, not token validity.

Basic health is deliberately unauthenticated and cannot prove the endpoint token. Call the protected models read instead:

```
SH
aipctl action call \
  "$AIP_URL" \
  "cap:hermes_agent:${HERMES_ENDPOINT_ID}:models" \
  --action-id act_hermes_endpoint_auth_001 \
  --input '{}'
```

Expected result: the action reaches a terminal completed state and returns the Hermes model catalog. Correlate the action ID, connector version, instance, replica, endpoint ID, tenant, and provider audit record without retaining the Bearer token.

A completed protected read demonstrates that the selected endpoint possessed a usable token at that time. It does not prove that every model-provider credential, mutation, stream, operator policy, or delegation path works.

8. Rotate an endpoint token

The standalone binary reads each API-key file once during startup. Replacing the mounted file alone does not update a running process.

For a routine rotation:

1. Create a new Hermes token and a new immutable connector-side secret file.
2. Keep the old token valid while replacement replicas start.
3. Start new replicas with the same admitted endpoint manifest and new token

bytes, using new replica identities.

4. Require host readiness and repeat the protected `models` read through a

controlled route.

5. Stop new assignments to old replicas and let their pinned actions settle.
6. Drain and take the old replicas offline.
7. Revoke the old Hermes token.
8. Retain secret-free action, host, registry, and provider evidence.

The endpoint manifest need not change when only secret bytes change. The process still must be replaced or restarted so it loads the new bytes. If the credential revision also changes, use the full blue/green instance procedure and preserve old pinned assignments under their original revision.

For a suspected compromise, first stop new routing and revoke the affected revision or token according to the incident boundary. Do not prioritize a graceful overlap over containment.

Resolve authentication and endpoint failures

Symptom	Likely boundary	Safe decision
Host rejects descriptor JSON	Shape, size, unknown field, or file permissions	Correct the owner-controlled file; do not relax parsing
Duplicate endpoint error	IDs collide after trimming and lowercase normalization	Assign distinct stable IDs and create a new manifest
Tenant mismatch at startup	Descriptor tenant differs from host tenant	Move the endpoint to the correct tenant instance
`connector.hermes_agent.tenant_mismatch`	Trusted runtime tenant is absent or different	Repair admission and routing; never override tenant in input
`connector.hermes_agent.missing_credential`	Protected operation has no usable endpoint key	Mount the correct owner-only file and replace the process
Provider returns `401` or `403`	Token is invalid, revoked, or insufficient	Check the exact endpoint and provider audit; rotate or scope correctly
Basic health passes but models fails	`/health` does not require endpoint auth	Diagnose the protected credential path; do not accept health as proof
Catalog lacks the endpoint capability	Admitted manifest or binding differs	Reconcile version, instance, binding, and endpoint set
Old token remains in use after file replacement	Running process retained startup bytes	Replace or restart the affected replica safely
Request tries another origin after redirect	Provider returned redirect	Correct the configured base URL; redirects remain disabled

Do not retry a state-changing action merely because an authentication or transport error appears retryable. Preserve the original action ID and use the operation-specific reconciliation guidance.

Related documentation

- Hermes Agent connector
- Hermes Agent quickstart
- Rotate connector credentials
- Connector registry and routing
- Security model