# Hermes health, models, and discovery capabilities

Use these six endpoint-qualified reads to inspect one configured Hermes API server. They report transport health, runtime detail, model aliases, the provider API surface, enabled skills, and resolved toolsets.

| | |
|---|---|
| SECTION | Connectors |
| TYPE | Connector |
| PROTOCOL | AIP 1.0 |
| SOURCE | docs/connectors/hermes-agent/capabilities/health-models-and-discovery.md |

These results describe the selected Hermes endpoint. They do not change the tenant's admitted AIP capability catalogue or grant permission to invoke a newly observed model, skill, tool, or provider route.

## Operations

Replace `` with the normalized ID present in AIP discovery.

| Operation | Provider request | Auth | Stability | Result purpose |
|---|---|---|---|---|
| `health` | `GET /health` | None at endpoint | Stable | Basic process reachability and version |
| `health_detailed` | `GET /health/detailed` | Endpoint Bearer | Stable | Gateway state, activity, uptime, and process detail |
| `models` | `GET /v1/models` | Endpoint Bearer | Stable | Default model and configured route aliases |
| `capabilities` | `GET /v1/capabilities` | Endpoint Bearer | Stable | Hermes API-server self-description |
| `skills` | `GET /v1/skills` | Endpoint Bearer | Experimental | Enabled skills visible to the API-server agent |
| `toolsets` | `GET /v1/toolsets` | Endpoint Bearer | Experimental | Configurable toolsets and resolved tool names |

Every input is the same strict empty object:

```
JSON
{}
```

Additional input properties are rejected by the published schema. The endpoint comes from the capability ID, not from this object.

## Shared AIP contract

| Contract field | Value for all six operations |
| --- | --- |
| Kind | Tool |
| Risk | Low |
| Read-only | Yes |
| Human approval | Not required by the capability |
| Idempotency | Optional; capability scope; no published TTL |
| Collision behavior | Return the original result |
| Connector retry support | Yes |
| Retry safety | Safe |
| Side effects | `read`, `external_network` |
| Data | Internal; no PII marker; redaction not required by the contract |
| Execution | Synchronous; no async, stream, or cancellation support |
| Expected latency hint | 5000 ms |
| AIP contract timeout | 60000 ms |
| Provider request timeout | 30000 ms |
| Transaction | None |
| Compensation | Rollback not supported |

The contract publishes a `99.9%` availability target and 10000 ms maximum queue-delay hint. These are declared values, not observed availability or qualification evidence.

“No approval required” does not mean anonymous access. The caller, tenant, admitted route, host trust, and credential revision remain governed. Five of the six operations also require the endpoint's Bearer token.

## Keep three discovery sources separate

| Source | Authority | What it does not prove |
| --- | --- | --- |
| Tenant AIP capability catalogue | Which admitted capability version the caller may route | Provider health or a successful operation |
| Hermes `capabilities` result | What the selected Hermes API server reports about itself | AIP admission, binding, approval, or policy |
| Hermes models, skills, and toolsets | Current provider-side execution inventory | Model credentials, tool authorization, or compatibility with every operation |

Use AIP discovery to select an action. Use these provider reads to verify the selected endpoint's current state. Never synthesize a new AIP capability ID from a provider response.

## Read basic health

`cap:hermes_agent::health` is the only unauthenticated provider operation in the connector. At the pinned Hermes revision, the provider returns:

```
JSON
{
  "status": "ok",
  "platform": "hermes-agent",
  "version": "<hermes-version>"
}
```

The connector normalizes that response:

```
JSON
{
  "endpoint_id": "primary",
  "status": "ok",
  "platform": "hermes-agent",
  "version": "<hermes-version>",
  "http_status": 200,
  "raw": {
    "status": "ok",
    "platform": "hermes-agent",
    "version": "<hermes-version>"
  }
}
```

The AIP action completes only when the provider status is exactly `ok` and the HTTP status is successful. A successful HTTP response with another or missing status produces a failed action with the normalized output retained.

The connector's host-readiness probe uses a narrower reachability test. It calls basic health on every configured endpoint and treats any successfully read 2xx response as reachable; it does not inspect the raw `status` value. Consequently, host readiness does not prove endpoint authentication, model access, or semantic health of every Hermes subsystem.

## Read detailed health

`health_detailed` is authenticated. The pinned provider returns an object that can include:

- `status`, `platform`, and `version`;
- gateway state and connected-platform detail;
- active-agent count plus busy and drainable markers;
- exit reason and last update time;
- process ID.

The connector returns the provider object without normalizing these fields. Treat the result as internal operational data. Do not expose process or runtime detail merely because the capability is low risk.

Detailed health is a point-in-time provider report. It is not the connector host's `/ready` contract, registry lease, storage health, or durable action state.

## List models

`models` returns the OpenAI-compatible model list published by the selected Hermes API server. The pinned provider includes its default model and any configured model-route aliases:

```
JSON
{
  "object": "list",
  "data": [
    {
      "id": "hermes-agent",
      "object": "model",
      "created": 0,
      "owned_by": "hermes",
      "permission": [],
      "root": "hermes-agent",
      "parent": null
    }
  ]
}
```

The real `created` value is the provider's current Unix time. Route aliases can use their resolved model as `root` and the default model as `parent`. Provider credentials are not included.

An observed model ID proves only that Hermes advertised the alias. It does not prove that upstream credentials, quota, model policy, or a chat request will succeed.

## Read the Hermes API self-description

`capabilities` returns the provider's structured API-server document. At the pinned revision it reports:

- platform, model, and Bearer-auth state;
- server-side tool-execution mode;
- feature flags for chat, Responses API, runs, sessions, skills, streaming,

approvals, and related surfaces;

- provider method and path entries for its published endpoints.

This document is not the AIP manifest. For example, the pinned connector publishes scheduled-job AIP capabilities even though the provider document's `endpoints` object does not enumerate the job routes. The admitted connector manifest remains authoritative for AIP routing.

Retain the provider document with the exact endpoint and Hermes revision when investigating compatibility drift. Do not let it enable an unadmitted binding.

## List enabled skills

`skills` enumerates skills visible to the API-server agent. The pinned provider sorts entries and returns their bounded metadata:

```
JSON
{
  "object": "list",
  "data": [
    {
      "name": "example-skill",
      "description": "Example skill description",
      "category": "example"
    }
  ]
}
```

The provider scans local and configured external skill directories, applies platform and environment filters, gives local names precedence, and filters disabled skills. A malformed or unreadable individual skill can be skipped; an enumeration failure returns a provider server error.

This operation is experimental in the AIP manifest. Skill presence does not authorize an operator, model, or delegated action to use it.

## List resolved toolsets

`toolsets` returns the configurable toolset surface for the API-server platform. Each entry can contain:

| Field | Meaning |
| --- | --- |
| `name` | Stable toolset name |
| `label` | Human-readable label |
| `description` | Provider description |
| `enabled` | Whether the API-server platform enables the toolset |
| `configured` | Whether provider configuration satisfies its key check |
| `tools` | Sorted, de-duplicated concrete tool names |

The top-level object also identifies `platform` as `api_server`. A failed toolset resolution becomes an empty `tools` list for that entry; a broader enumeration failure returns a provider server error.

`enabled` and `configured` are provider observations, not AIP authorization or proof that a tool call will complete. This operation is experimental.

## Interpret raw results safely

The published output schema for the five authenticated reads is an object without required provider-specific properties. The connector parses a JSON body to its actual JSON value, wraps non-JSON text as `{"text": "..."}`, and turns an empty successful body into `null`.

The execution path does not reject an array or `null` after parsing, even though the published schema says object. Require the expected object shape in application code and treat a mismatch as provider compatibility drift.

The connector exposes no provider response-byte limit for these reads. Apply a reviewed response bound at the Hermes server, ingress, or transport boundary; do not assume the 4 MiB AIP request limit bounds provider responses.

## Call one protected discovery capability

Use a configured signed-native caller context and an endpoint ID returned by the tenant catalogue:

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

aipctl action call \
  "$AIP_URL" \
  "cap:hermes_agent:${HERMES_ENDPOINT_ID}:models" \
  --action-id act_hermes_models_001 \
  --input '{}'
```

Expected result: the action completes with an object-shaped provider model list. Correlate the action, tenant, admitted connector version, endpoint, and provider audit time. Do not record endpoint token bytes.

If you call `health` instead, a completed result cannot prove the token path because that provider route deliberately omits Bearer auth.

## Failures and retry decisions

| Failure code | Typical boundary | Safe response |
|---|---|---|
| `connector.hermes_agent.configuration_or_input` | Unknown endpoint or unsupported capability | Refresh admitted discovery and correct the ID |
| `connector.hermes_agent.tenant_mismatch` | Trusted runtime tenant does not own the endpoint | Repair admission and routing; never override tenant in input |
| `connector.hermes_agent.missing_credential` | Protected read has no usable endpoint key | Replace the host with the correct owner-only key file |
| `connector.hermes_agent.authentication` | Hermes returned `401` or `403` | Verify endpoint selection and rotate or correct the token |
| `connector.hermes_agent.remote_temporary` | Hermes returned `429` or a server error | Retry this read with bounded backoff and the same action contract |
| `connector.hermes_agent.remote_rejected` | Hermes returned another non-success status | Correct provider state or compatibility before retrying |
| `connector.hermes_agent.transport` | Request or response-body read failed | Retry the read within deadline and policy; inspect network evidence |
| `connector.hermes_agent.deadline_exceeded` | Trusted AIP deadline elapsed | Preserve the action and issue a new read only when policy permits |

Invalid JSON is not a decode failure in this path; it becomes a text object. There is no `response_too_large` connector error for these operations.

## Verify an integration

Require all of the following:

- AIP discovery returns the exact endpoint-qualified capability and admitted

version;

- basic health is treated as reachability, not credential evidence;
- a protected read proves the endpoint token path separately;
- result handling rejects an unexpected non-object shape;
- provider capabilities never add or enable an AIP route;
- model, skill, and toolset observations are not confused with authorization;
- retry preserves the original read contract and respects deadline and policy;
- internal discovery output is not exposed to an untrusted tenant or model.

These checks validate integration behavior. They do not qualify a particular Hermes deployment, model provider, skill, or toolset.

## Related documentation

- Hermes capability index
- Authenticate and bind Hermes endpoints
- Hermes configuration
- Capabilities and contracts
- Error reference