AIP MCP inside Hermes

Use this reference when a Hermes role must discover or call admitted AIP capabilities through aipd Streamable HTTP MCP. The configuration gives Hermes a bounded model-facing tool surface. Native AIP identity, policy, approval, and Action li

SECTION	Connectors
TYPE	Connector
PROTOCOL	AIP 1.0
SOURCE	docs/connectors/hermes-agent/reference/aip-mcp-inside-hermes.md

This path is deployment configuration. The Hermes connector does not create or change the Hermes MCP profile, deliver its access token, or grant the MCP principal permission to call a capability.

Keep the four execution roles separate

Role	Entry point	Authority
Direct Hermes capability	Native AIP Action routed to one fixed Hermes API route	The admitted capability and endpoint binding
Hermes operator	Native AIP Action routed to `cap:hermes_agent:<endpoint_id>:operator`	The operator binding and terminal provider state
First-class AIP delegation	AIP routes a child Action to `agent:hermes_operator:<endpoint_id>`	The native delegation graph and child lifecycle
AIP MCP inside Hermes	Hermes calls selected tools on protected `aipd /mcp`	The authenticated MCP principal and native AIP gateway

Configuring the fourth role does not enable the third. Enabling first-class delegation does not configure the fourth. The two paths meet only when the delegation connector gives Hermes one exact native child-Action call to make.

An ordinary Hermes operator can also call `aip_call`. That call creates or executes work as the Hermes MCP principal. It is not first-class delegation unless AIP already created the child, supplied its immutable contract, and routes the child to that Hermes delegate.

Follow the authority flow

The supported inside-Hermes path is:

1. A deployment-owned Hermes profile opens Streamable HTTP MCP to `aipd`.
2. The protected MCP endpoint authenticates the profile's access token.
3. Hermes discovers only the server-native tools selected by its allowlist.
4. The model sees Hermes-namespaced wrappers for those tools.
5. `aipd` converts a tool call to a native message and trusted execution

context.

6. The AIP gateway authorizes, admits, and routes the resulting Action.
7. A lifecycle facade reads or changes the native Action owned by that

authenticated context.

The MCP allowlist controls which tool schemas Hermes registers. It does not authorize the principal, restrict every possible `capability_id` argument to `aip_call`, or replace capability policy. Apply both controls.

Keep each identity distinct

Identity	Created or verified by	Purpose
Hermes role/profile	Hermes deployment	Selects configuration, model, and local tool exposure
MCP principal	`aipd` authentication	Owns the trusted actor and tenant context for MCP calls
MCP session ID	Streamable HTTP MCP	Binds protocol version, actor, and bounded replay state
Hermes transcript or `session_id`	Hermes	Retains provider conversation or structured-run context
AIP Action ID	AIP caller or `aip_call`	Identifies one native unit of requested work
AIP Session ID	Native AIP lifecycle	Groups native Actions when the caller explicitly uses one
Delegation and child Action IDs	Native delegation runtime	Bind parent, delegate, scope, and exact child execution

An MCP session is not an AIP Session or a Hermes transcript. Closing or replacing one does not close the others. Persist the Action ID before relying on a result that may outlive the MCP connection.

Use a distinct MCP principal for every independently governed Hermes role or instance. A role that only discovers and calls capabilities should not inherit approval-decision, audit-export, broad cross-principal read, or `delegation:create` authority from an external orchestrator.

Install the Hermes MCP client surface

The pinned Hermes source treats MCP support as an optional package extra. A minimal image intended for this path installs the checkout with `.[mcp]`. Without the MCP SDK and its Streamable HTTP client module, the configured server cannot connect or register tools.

The reviewed Hermes code reads `mcp_servers` from its active `~/.hermes/config.yaml`. A deployment may place that file under the effective `HERMES_HOME`, but the owning Hermes profile and filesystem permissions must remain explicit.

Configure the bounded server entry

The following profile uses the stable AIP facade and a token injected into the Hermes secret scope or environment:

```yaml
YAML
mcp_servers:
  aip:
    url: "https://aip.example.com/mcp"
    headers:
      Authorization: "Bearer ${AIP_MCP_TOKEN}"
    enabled: true
    timeout: 300
    connect_timeout: 15
    skip_preflight: true
    supports_parallel_tool_calls: false
    tools:
      include:
        - aip_capabilities
        - aip_call
        - aip_action_status
        - aip_action_result
        - aip_action_events
```

```
      - aip_action_cancel
    resources: false
    prompts: false
```

Use the private service URL only inside a controlled network. Use the public HTTPS resource URL when Hermes crosses a trust boundary. The global MCP guide owns the supported protocol versions and protected-resource configuration.

Server-entry fields

Field	Reviewed behavior	Deployment decision
`url`	Selects HTTP; the default non-SSE path is Streamable HTTP	Point to the exact protected `/mcp` resource
`headers`	Adds configured headers after secret interpolation	Inject a role-specific token; never put it in an Action
`auth: oauth`	Selects Hermes OAuth 2.1 PKCE handling	Use instead of a static header when the identity service supports it
`enabled`	`false` skips connection and registration	Keep disabled until identity and policy are ready
`timeout`	Per-tool call; default is 300 seconds	Keep within the AIP and provider deadline design
`connect_timeout`	Initial connection; default is 60 seconds	Use an explicit operational bound such as 15 seconds
`skip_preflight`	Skips the HTTP content-type probe	Enable only for a verified POST-capable MCP endpoint
`supports_parallel_tool_calls`	Opt-in; default is `false`	Keep false for governed state-changing work
`tools.include`	Original server-native name whitelist	List only the facade tools the role needs
`tools.resources`	Controls resource utility wrappers	Keep false when the role needs tools only
`tools.prompts`	Controls prompt utility wrappers	Keep false when the role needs tools only

`tools.include` takes precedence if both `include` and `exclude` are present. An empty or mismatched include list can leave the server connected with no useful registered tools.

Hermes recursively replaces `${VAR}` and `${env:VAR}` in string values. It first uses the active profile's secret scope and then the process environment. An unset value remains a literal placeholder. Treat that state as a failed deployment check; do not assume an unresolved token becomes empty or safe.

For `auth: oauth`, complete the deployment-specific authorization bootstrap and token storage before starting an unattended role. Do not configure both an unreviewed static Authorization header and OAuth for the same identity.

Use original names in the allowlist

The allowlist contains names advertised by `aipd`. Hermes then registers each selected tool under a namespaced model-facing name.

Server-native allowlist name	Hermes registered name when server is `aip`	Purpose
`aip_capabilities`	`mcp**aip**aip_capabilities`	Discover visible native capabilities and contracts
`aip_call`	`mcp**aip**aip_call`	Create or execute a native Action
`aip_action_status`	`mcp**aip**aip_action_status`	Read current Action state
`aip_action_result`	`mcp**aip**aip_action_result`	Read the authoritative durable result
`aip_action_events`	`mcp**aip**aip_action_events`	Read bounded event and chunk history
`aip_action_cancel`	`mcp**aip**aip_action_cancel`	Request native Action cancellation

Do not put the `mcp**aip**` form in `tools.include`. That name exists only after Hermes converts the discovered schema. Hermes sanitizes non-alphanumeric characters in the server and tool components, so keep the logical server name stable.

Generated per-capability MCP tools may change with the admitted local catalog and can disappear when a fleet catalog owns discovery. Use the stable `aip_capabilities` and `aip_call` pair for native capability identity.

Bound what `aip_call` can target

The tool allowlist admits one generic `aip_call` schema. Its arguments still contain a `capability_id`. Therefore, tool filtering alone does not stop a Hermes role from asking for another visible capability.

Bind the MCP principal to one tenant and apply AIP authorization and policy for the smallest allowed capability set. In particular, prevent a role from calling its own Hermes operator capability or another route that returns to the same model loop unless the topology has an explicit bounded recursion design.

Discovery is not authorization. A capability returned by `aip_capabilities` can still be rejected at invocation, approval, connector admission, or provider execution.

Preserve native Action identity

For an ordinary call, `aip_call` accepts a native `capability_id`, input, and optional Action fields. If `action_id` is omitted, the facade creates a new native ID. Persist the returned ID before retrying or switching to lifecycle tools.

For first-class delegation, omission is not allowed. The Hermes operator receives an exact `aip_call` argument object whose fields describe the child Action already created by AIP. The call must preserve:

- `action_id`;
- `capability_id` and input;
- mode, idempotency key, and timeout;
- conversation and memory context;
- delegation chain and federation context;
- callback, observability, compliance, approval, and transaction context.

The connector removes model-selected identity because the MCP transport must supply the authenticated principal. After Hermes completes, the result resolver compares the native child contract, trusted owner, tenant, queue, and lifecycle result. A replacement Action ID or model-written result fails that verification.

Exclude recursive delegation

Do not expose `aip_delegate` to the ordinary model-facing Hermes tool set. The external AIP runtime owns the original parent, child ID, delegate, scope, depth, cycle checks, approval, and durable outbox.

Giving the same model an unrestricted delegation-creation tool can start a new delegation graph instead of continuing the one it received. It can also route work back into the same operator. The reviewed profile excludes this tool and does not grant `delegation:create` to the normal internal operator identity.

If a separate broker is designed to create delegation, isolate its principal, propagate and validate the original chain, enforce depth and cycle bounds, and keep that broker out of the ordinary Hermes model tool surface.

Keep calls sequential unless proved otherwise

`supports_parallel_tool_calls: false` keeps the server out of Hermes's parallel-safe MCP set. This reduces concurrent model-issued calls from the same role, but it is not an exactly-once guarantee.

A network failure can occur after `aipd` accepted an Action. For a mutation, reuse the same Action ID or required idempotency key only according to the native capability contract. Query `aip_action_status` or `aip_action_result` before issuing another call. Never infer that a timeout means the provider did nothing.

Protect credentials and returned data

- Store the MCP token in the Hermes profile's secret scope, environment, or an

equivalent deployment secret, not in source or Action input.

- Use one token and principal per independently governed Hermes role.
- Give the principal only the transport and native operation scopes it needs.
- Keep endpoint, issuer, audience, expiry, tenant, and scope checks at `aipd`.
- Keep `ssl_verify` enabled and use a trusted CA at network boundaries.
- Do not give Hermes database or system-of-record credentials merely to prove

an AIP-mediated workflow.

- Treat MCP tool output as application data. Apply the capability's declared

classification and redaction rules before adding it to a transcript.

The current Hermes Streamable HTTP client strips Authorization when a redirect crosses origin. The deployment should still use the final MCP URL and avoid relying on redirects as a routing mechanism.

Verify configuration and discovery

The pinned Hermes CLI exposes two useful static-to-connection checks:

```
SH
hermes mcp list
hermes mcp test aip
```

`hermes mcp list` reports configured servers and selected-tool count. It does not connect. `hermes mcp test aip` resolves the profile, attempts MCP initialization, and lists discovered server-native tools. It does not prove that the principal can invoke every listed capability or that a connector is qualified.

After changing a running Hermes profile, `/reload-mcp` reconnects configured servers, rebuilds the registered tool set, and invalidates the provider prompt cache. Treat that as an operational change and confirm that the next execution receives the intended schemas.

Retain evidence that:

1. the effective profile belongs to the intended Hermes role;
2. the URL is the intended protected `aipd` MCP resource;
3. the token resolves without appearing in logs or retained output;
4. authentication resolves the expected principal and tenant;
5. exactly the six original stable facade names are discovered;
6. Hermes registers the corresponding six `mcp**aip**...` names;
7. resources, prompts, generated tools, and `aip_delegate` are absent;
8. a permitted read succeeds and a forbidden capability fails closed;
9. an accepted Action can be found by the same stable Action ID;
10. a timeout is reconciled through lifecycle reads rather than blind replay;
11. a delegated child rejects changed identity, contract, or result authority;
12. restart or reload preserves the intended least-privilege surface.

These checks qualify only the recorded profile, token, `aipd`, connector catalog, Hermes artifact, and time window.

Resolve failures by boundary

Symptom	Boundary	Safe response
Hermes reports no MCP SDK	Image/package	Install the pinned Hermes `mcp` extra and rebuild the exact artifact
Server is configured but absent	Profile or `enabled`	Confirm the active profile path and that `enabled` is true
Preflight reports a non-MCP page	URL or proxy	Point to `/mcp`; bypass preflight only after verifying POST behavior
`401 Unauthorized`	MCP authentication	Check token resolution, issuer, audience, expiry, and required transport scopes
Connection works but selected tools are missing	Discovery or filter	Compare original names with `tools.include` and the effective catalog
Model requests an unknown single-underscore name	Hermes namespace	Use the registered `mcp__aip__<tool>` name; keep original names in the allowlist
A listed capability is denied	Native AIP authorization	Keep the denial; correct principal policy rather than widening the tool set
Call times out after submission	Action lifecycle	Query by Action ID; do not repeat an unsafe mutation blindly
Delegated result reports an integrity mismatch	Delegation contract	Fail closed and retain the supplied and observed native records
The same Hermes role invokes itself	Capability policy	Disable the route, preserve current Actions, and remove the recursive target
OAuth works interactively but not at startup	Token bootstrap	Establish the role's approved token flow before unattended startup
Reload changes the visible tool surface	Hermes registry	Reconcile config, discovered names, and the owning execution context

Do not solve an AIP policy denial by exposing more MCP tools or granting a broad orchestrator token to Hermes. Do not solve an MCP transport failure by bypassing native Action identity or calling the provider directly.

Evidence boundary

The reviewed sources define the profile, tool projection, stable facade, and delegated-result checks. They do not prove that a particular Hermes artifact connected, that an identity provider issued the intended token, that a model used the correct tool, or that a provider completed an Action.

Record exact AIP and Hermes revisions, image digests, profile hash, endpoint, principal, tenant, allowlist, policy revision, and retained Action evidence before making an interoperability or readiness claim.

Related documentation

- Use AIP through MCP
- Hermes operator and delegation capability
- Hermes authentication and endpoints
- Identity and trust
- Delegation
- Errors and recovery