api0

GUIDE 11

# Outbound Authentication: Which Credential api0 Uses to Call Your Application

Every mode api0 uses to call your application or a third-party API: the identity token signed by api0, service keys and tokens, each person's own credential, named connections, and the decision tree for choosing.

Updated 11 November 2026 · Read online: api0.ai/guides/outbound-authentication

Once the caller is identified (inbound authentication, guide 10), the gateway calls your application or a third-party API. This guide describes **which credential** it uses: a token api0 signs itself, a service key, or each person's own credential.

In every case, the credential stays in api0. The AI model never sees it, and it is never sent back to the dashboard.

## The modes

| Mode | What the application receives | When to choose it |
|---|---|---|
| `none` | Nothing | Public API |
| **api0 identity token** ( `google_sa` , no key) | A Google identity token signed by api0's service account | **Your own applications.** No secret to store |
| `google_sa` with your key | An identity token signed by **your** service account | Google Cloud applications (Cloud Run…) with your own IAM |
| `static_bearer` | `Authorization: Bearer <token>` | Third-party API with a single key |
| `header_injection` | The headers you define | APIs with proprietary headers |
| `client_credentials` | A token obtained via OAuth client credentials | APIs that issue service tokens |
| `per_user` — personal secret | **Each** user's own credential: `Basic` (Azure DevOps PAT), `Bearer` , or a raw header | Per-person attribution and permissions |
| `per_user` — OAuth | **Each** user's OAuth token, renewed by api0 | Systems with OAuth sign-in (e.g. Microsoft) |

The authentication header always wins over a fixed tool header with the same name (see the header order in guide 04).

## Default and named connections

Each tenant has a **default connection**, and can add **named** ones when some tools target a service that needs a different sign-in. For example: Azure DevOps on the default connection, Microsoft Graph on a `graph` connection, a reporting database on a service identity.

On app.api0.ai, the **Connections** card on the **API** page creates and edits connections, and **Which connection each tool uses** sets which one each tool uses. A tool that names none uses `default`.

With Microsoft, you need one connection per **audience**: a token for Microsoft Graph is refused by Azure DevOps, and the other way round (guide 06, section 3).

## The api0 identity token in detail

For an application you run, the simplest option is often to store **no** secret at all: the gateway signs every call with api0's service account.

One signer for every tenant means the **audience** is what separates them, and the tenant doesn't choose it. The gateway builds it from the application URL's origin and the calling tenant's ID:

```
https://api.example.com/orders  →  aud = https://api.example.com/api0/tenant/<tenant id>
```

Only the origin is kept: no path or query can spell out another tenant's audience. Your application checks two values, once:

| Check | Value |
|---|---|
| `aud` | `https://<your origin>/api0/tenant/<your tenant>` |
| `email` (signer) | api0's service account |

Pinning the signer is mandatory: any Google service account can mint a token for any audience.

## Per-person credentials

This is the model to prefer whenever attribution matters.

- **Personal secret.** The user adds their token once (an Azure DevOps PAT, for example). It's encrypted and **can't be read back**, only replaced. An optional verification URL can check who the token belongs to.
- **OAuth.** The user connects their account once (**Settings** → **My access tokens** → **Connect**). api0 keeps the refresh token encrypted and renews access. The OAuth flow's `state` value is single-use and short-lived (CSRF protection).
- **No credential yet**: the user gets a tool result explaining how to add one. **Never** a call made as someone else.

Result: the target system sees the real person, applies their permissions and attributes the action to them.

Step-by-step examples with Microsoft: Azure DevOps with each person's Microsoft sign-in (guide 05), then SharePoint, Azure SQL and Logic Apps in guide 06.

## Choosing a mode: the decision tree

1. **Is the application yours?** → api0 identity token. Nothing to store.
2. **Must actions be attributed to the person, or their permissions applied?** → `per_user` (OAuth if the system supports it, otherwise a personal secret).
3. **Third-party API with a service key?** → `static_bearer`, `header_injection` or `client_credentials`, with identity forwarding off.
4. **Tools targeting different services?** → one named connection per service.