api0

GUIDE 11

Authentification sortante : avec quel identifiant api0 appelle ton application

Chaque mode qu'api0 utilise pour appeler ton application ou une API tierce : le jeton d'identité signé par api0, les clés et jetons de service, l'identifiant de chaque personne, les connexions nommées, et l'arbre de décision pour choisir.

Mis à jour le 11 novembre 2026 · Lire en ligne : api0.ai/fr/guides/authentification-sortante

Une fois l'appelant identifié (l'authentification entrante, guide 10), la passerelle appelle ton application ou une API tierce. Ce guide décrit **avec quel identifiant** : un jeton qu'api0 signe lui-même, une clé de service, ou l'identifiant de chaque personne.

Dans tous les cas, l'identifiant reste dans api0. Le modèle d'IA ne le voit jamais, et il n'est jamais renvoyé au tableau de bord.

Les modes

Mode	Ce que reçoit l'application	Quand le choisir
`none`	Rien	API publique
Jeton d'identité api0 (`google_sa` , sans clé)	Un jeton d'identité Google signé par le compte de service d'api0	**Tes propres applications.** Aucun secret à stocker
`google_sa` avec ta clé	Un jeton d'identité signé par **ton** compte de service	Applications Google Cloud (Cloud Run…) avec ton propre IAM
`static_bearer`	`Authorization: Bearer <jeton>`	API tierce à clé unique
`header_injection`	Les en-têtes que tu définis	APIs à en-têtes propriétaires
`client_credentials`	Un jeton obtenu par OAuth client credentials	APIs qui délivrent des jetons de service
`per_user` — secret personnel	L'identifiant de **chaque** utilisateur : `Basic` (PAT Azure DevOps), `Bearer` , ou en-tête brut	Attribution et droits par personne
`per_user` — OAuth	Le jeton OAuth de **chaque** utilisateur, renouvelé par api0	Systèmes à connexion OAuth (ex. Microsoft)

L'en-tête d'authentification gagne toujours sur un en-tête fixe de l'outil qui porterait le même nom (voir l'ordre des en-têtes dans le guide 04).

Connexion par défaut et connexions nommées

Chaque tenant a une **connexion par défaut**, et peut en ajouter d'autres, **nommées**, quand certains outils visent un service qui demande une autre authentification. Par exemple : Azure DevOps sur la connexion par défaut, Microsoft Graph sur une connexion `graph`, une base de reporting sur une identité de service.

Sur app.api0.ai, la carte **Connections** de la page **API** crée et modifie les connexions, et **Which connection each tool uses** dit laquelle chaque outil utilise. Un outil qui n'en désigne aucune utilise `default`.

Avec Microsoft, il faut une connexion par **audience** : un jeton pour Microsoft Graph est refusé par Azure DevOps, et inversement (guide 06, section 3).

Le jeton d'identité api0 en détail

Pour une application que tu opères, le plus simple est souvent de ne stocker **aucun** secret : la passerelle signe chaque appel avec le compte de service d'api0.

Un seul signataire pour tous les tenants : c'est donc l'**audience** qui les sépare, et le tenant ne la choisit pas. La passerelle la construit à partir de l'origine de l'URL de l'application et de l'identifiant du tenant appelant :

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

Seule l'origine est gardée : aucun chemin, aucune requête ne peut fabriquer l'audience d'un autre tenant. Ton application vérifie deux valeurs, une fois pour toutes :

Vérification	Valeur
`aud`	`https://<ton origine>/api0/tenant/<ton tenant>`
`email` (signataire)	le compte de service d'api0

Épingler le signataire est obligatoire : n'importe quel compte de service Google peut émettre un jeton pour n'importe quelle audience.

Identifiants par personne

C'est le modèle à préférer dès que l'attribution compte.

- **Secret personnel.** L'utilisateur ajoute une fois son jeton (par exemple un PAT Azure DevOps). Il est chiffré et **ne peut plus être relu**, seulement remplacé. Une URL de vérification optionnelle permet de contrôler à qui appartient le jeton.
- **OAuth.** L'utilisateur connecte une fois son compte (**Settings → My access tokens → Connect**). api0 garde le jeton de rafraîchissement chiffré et renouvelle l'accès. Le paramètre `state` du flux OAuth est à usage unique et court (protection CSRF).

- **Sans identifiant** : l'utilisateur reçoit un résultat d'outil qui lui explique comment l'ajouter. **Jamais** d'appel fait au nom de quelqu'un d'autre.

Résultat : le système cible voit la vraie personne, applique ses droits et lui attribue l'action.

Exemples pas à pas avec Microsoft : Azure DevOps avec la connexion Microsoft de chacun (guide 05), puis SharePoint, Azure SQL et Logic Apps dans le guide 06.

Choisir son mode : l'arbre de décision

1. **L'application est à toi ?** → jeton d'identité api0. Rien à stocker.
2. **Les actions doivent être attribuées à la personne, ou ses droits appliqués ?** → `per_user` (OAuth si le système le permet, sinon secret personnel).
3. **API tierce avec une clé de service ?** → `static_bearer`, `header_injection` ou `client_credentials`, et transmission d'identité désactivée.
4. **Des outils visent des services différents ?** → une connexion nommée par service.