DOCUMENTATION OAUTH2

pour les fournisseurs d'application tierces

Version: 1.4.7

Date: 31.05.2024

SOMMAIRE

1. INTRODUCTION

HIN permet l'utilisation d'applications raccordées à la plate-forme HIN via l'Access Control Service HIN (ACS) au moyen d'oAuth2. Pour ce faire, le fournisseur de l'application ACS doit autoriser l'utilisation d'oAuth auprès de HIN (e-mail à betrieb@hin.ch). HIN prend en charge les flux oAuth «Authorization Code» et «Client Credentials».

Authorization Code Flow:
L'application d'accès tierce (oAuth: client) qui veut accéder à une autre application au nom de l'utilisateur redirige ce dernier sur une application protégée par HIN (apps.hin.ch), dans laquelle il doit s'authentifier. Apps.hin.ch génère un token temporaire (oAuth: Auth Code) pour obtenir le token d'accès (oAuth: Access Token). Le code d'autorisation (Auth Code) est transféré à l'application tierce via le navigateur de l'utilisateur. Il existe pour cela deux variantes:
Variante a): Affichage du code d'autorisation dans l'application Web apps.hin.ch: le code d'autorisation est affiché dans l'application Web et est transféré vers l'application tierce par copier/coller.
Variante b): Transmission via paramètres de requête: lors de la redirection de l'utilisateur vers apps.hin.ch, l'application tierce spécifie déjà le point d'extrémité (oAuth: redirect_URI) pour lequel il veut le code d'autorisation.
Le code d'autorisation permet à l'application tierce d'obtenir le token d'accès. Le token d'accès permet à l'application tierce d'accéder à l'application correspondante via oauth2.*<URLexistante>*. Les tokens d'accès peuvent être valables pour une URL ou pour un groupe d'URL.

Client Credentials Flow:
Le Client Credentials Flow a été spécialement conçu pour les applications machine-to-machine. Dans le Client Credentials Flow, l'obtention du code d'authentification (Auth Code) et donc l'interaction avec un utilisateur final ne sont pas nécessaires. Contrairement à l'Authorization Code Flow, le token d'accès (Access Token) délivré n'est valable que pour un utilisateur spécifique et préconfiguré. Le Client Credentials Flow ne convient donc pas si un accès doit être effectué au nom d'un utilisateur final.

2. AUTHORIZATION CODE FLOW

2.1 Procédure

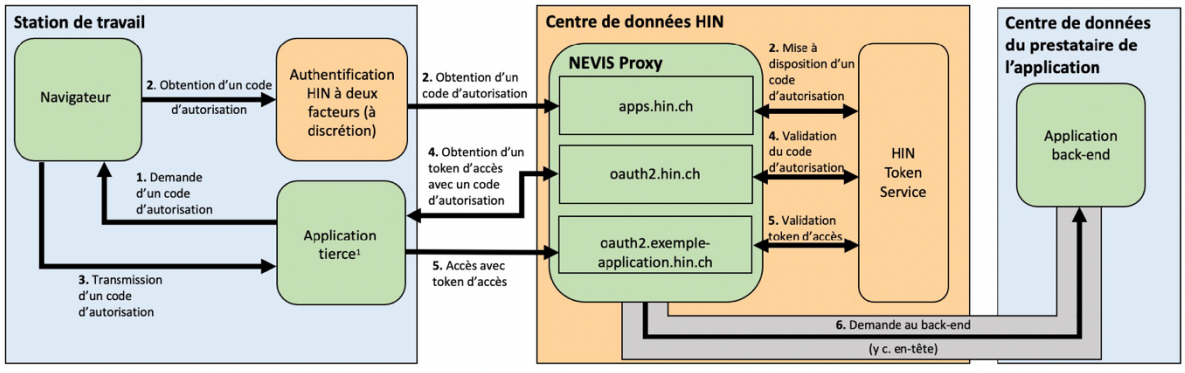
1. Une application tierce veut accéder à une ressource protégée par HIN (application ACS) au nom d'une identité HIN.
2. Si aucun token pour l'identité HIN correspondante n'est disponible dans l'application tierce, le navigateur doit être redirigé vers apps.hin.ch. Lors de l'accès à apps.hin.ch, l'utilisateur accédant doit se connecter avec une authentification à deux facteurs (p. ex. client HIN). Un code d'autorisation est présenté à l'utilisateur dans l'application Web.
3. Le code d'autorisation est transféré de l'utilisateur à l'application tierce. Différents mécanismes sont utilisés à cet effet (voir le chapitre «Obtention et utilisation du code d'autorisation») :
    a. Copier/coller le code d'autorisation ou photographier un code QR
    b. Transmission directe via https ou Protocol Handler
4. Le code d'autorisation permet à l'application tierce d'obtenir le token d'accès.
5. Le token d'accès permet d'accéder à la ressource protégée au nom de l'identité HIN de l'utilisateur.

2.2 Obtention de codes d'autorisation (Auth Codes)

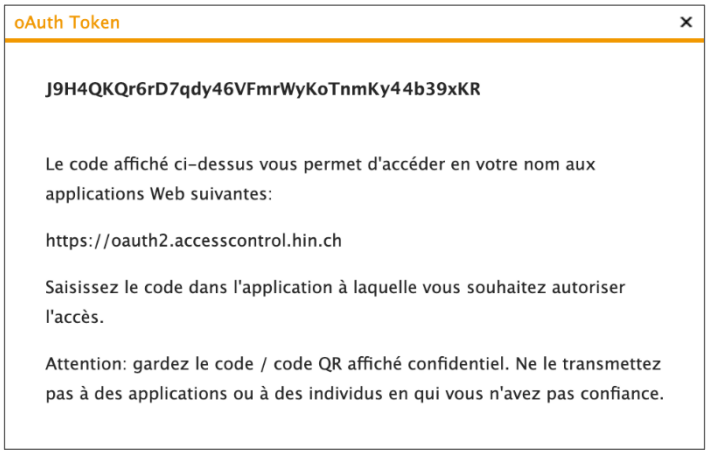
Le code d'autorisation (Auth Code) est un OTP (One Time Password) utilisé pour obtenir le token d'accès et est valable dix minutes. Il y a deux façons de l'obtenir :

2.2.1 Variante a): Affichage du code d'autorisation dans l'application Web

Sur apps.hin.ch, des codes d'autorisation peuvent être générés pour obtenir le token d'autorisation. Un registre distinct est fourni pour le service oAuth.



[1] p. ex. système de gestion de cabinet



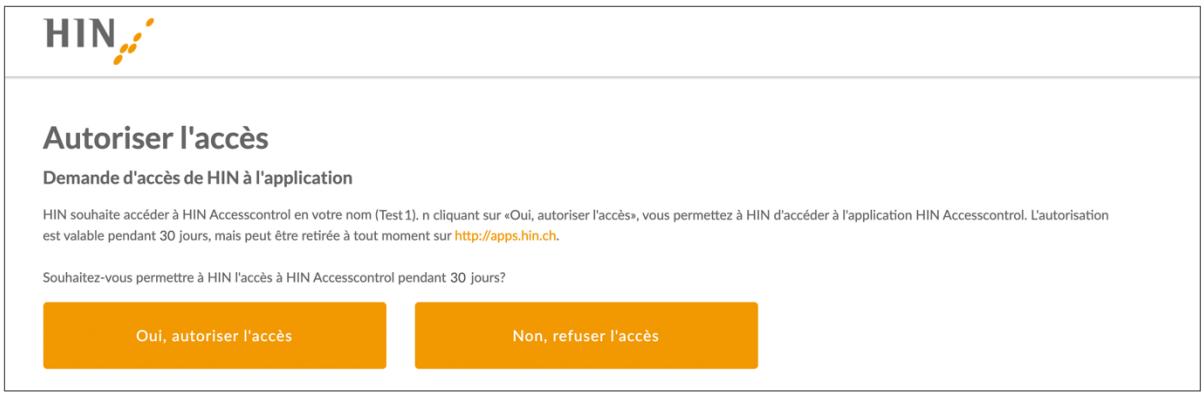
Lors de l'accès à apps.hin.ch , le premier registre («HIN Mail») s'affiche par défaut. Afin de simplifier le processus pour l'utilisateur, les codes d'autorisation peuvent être obtenus via un lien direct:

```
http://apps.hin.ch/#app=HinCredMgrOAuth;tokenGroup=<TokenGroupe>
```

La valeur après tokenGroup= diffère selon l'application cible et peut être demandée à HIN.

2.2.2 Variante b): Transmission via des paramètres de requête

Dans cette variante, le code d'autorisation est transmis à un redirect_URI défini. La personne utilisatrice le confirme en cliquant sur «Oui, autoriser l'accès».



Ce redirect_URI est fourni via l'URL appelée:

```
http://apps.hin.ch/REST/v1/OAuth/GetAuthCode/<TokenGruppe>?response_type=code&client_id=<client_id>&redirect_uri=<Redirect_URI>&state=<state>
```

Valeur	Description
<GroupeToken>	Groupe d'application pour lequel un token doit être obtenu (attention: le nom est sensible à la casse)
<redirect_URI>	Redirect_URI vers lequel le navigateur est redirigé après confirmation du dialogue avec le code d'autorisation. La valeur doit être encodée en url. Le redirect_URI doit par ailleurs être enregistré pour le client_ID associé. L'utilisation d'un redirect_URI non enregistré conduit à des messages d'erreurs. Ceci est exécuté par HIN. Plusieurs URI de redirection peuvent être définis par client_ID.
<state>	Une valeur statique qui demeure lorsque le navigateur est redirigé
<client_id>	oAuth client_ID: ID attribué par HIN à l'application tierce. Il ne s'agit pas d'une identité HIN au sens traditionnel du terme.

Exemple de demande pour l'application «ACS»:

```
http://apps.hin.ch/REST/v1/OAuth/GetAuthCode/application-ACS?res-
ponse_type=code&client_id=ch.hin&redi-
rect_uri=https%3A%2F%2Fwww.hin.ch&state=teststate
```

Le code d'autorisation est transmis au Redirect_URI désiré en utilisant les paramètres URI. Le statut (state) est également fourni.

```
https://www.hin.ch/?state=teststate&code=qdoWMwRNHnn9wDNynbMxytwa-
hEGNXBqtipQhZXLF
```

2.3 Obtention de tokens d'accès

Le token d'accès est le token utilisé pour un accès efficace à l'application.

2.3.1 Demande de token d'accès

Le token d'accès peut être obtenu avec le code d'autorisation reçu. L'obtention s'effectue au moyen d'un POST à oauth2.hin.ch/REST/v1/OAuth/GetAccessToken, et les paramètres sont transmis sous forme de données de formulaire (form-data) (`Content-Type: application/x-www-form-urlencoded`):

```
POST https://oauth2.hin.ch/REST/v1/OAuth/GetAccessToken
  grant type=authorization code&
  code=AUTH_CODE&
  redirect_uri=REDIRECT_URI&
  client_id=CLIENT_ID&
  client_secret=CLIENT_SECRET
```

Le redirect_URI doit concorder avec celui utilisé lors de l'obtention du code d'autorisation. Veuillez noter que, selon la configuration, le paramètre «client_secret» n'est pas forcément requis.

Obtention d'un token d'accès avec Curl (AuthCode apps.hin.ch)

```
curl -H 'Content-Type: application/x-www-form-urlencoded' -H 'Accept:applica-
tion/json' --data 'grant_type=authorization_code&redi-
rect_uri=&code=<CODE>&client_id=<CLIENT_ID>&client_secret=<CLIENT_SECRET>'
https://oauth2.hin.ch/REST/v1/OAuth/GetAccessToken
```

Obtention d'un token d'accès avec Curl (code d'autorisation avec URI de redirection)

```
curl -H 'Content-Type: application/x-www-form-urlencoded' -H 'Accept:applica-
tion/json' --data 'grant_type=authorization_code&redirect_uri=<REDI-
RECT_URI>&code=<CODE>&client_id=<CLIENT_ID>&client_secret=<CLIENT_SECRET>'
https://oauth2.hin.ch/REST/v1/OAuth/GetAccessToken
```

Veuillez noter que le corps doit être encodé en url. Tout caractère spécial contenu dans le client_secret doit être pris en compte. Par ailleurs, dans les anciennes versions de documents, l'URL pour obtenir les tokens d'accès était «https://oauth2.hin.ch/REST/v1/getoAuthToken». Cette URL sera remplacée à long terme. Les systèmes qui utilisent l'ancienne URL doivent être convertis.

Élément	Valeur	Description
grant_type	authorization_code	Le type est invariablement «authorization_code»
code	auth_code	Code d'autorisation qui a été copié par l'utilisateur
redirect_uri	<"vide">	Point d'extrémité auquel la réponse est délivrée. Doit correspondre à la valeur définie lors de l'obtention du code d'autorisation <"vide"> uniquement si le code d'autorisation a été obtenu via apps.hin.ch. Sinon, <REDIRECT_URI> qui a été défini lors de l'obtention du code d'autorisation.
client_id	<client_id>	oAuth client_ID: ID attribuée par HIN à l'application tierce. Il ne s'agit pas d'une identité HIN au sens traditionnel du terme.
client_secret	<mot de passe>	oAuth client_secret: un mot de passe défini par HIN

2.3.2 Réponse token d'accès

Le token d'accès est retourné au format JSON:

```
{
  "access_token":"RsT5OjbzRn430zqMLgV3Ia",
  "expires_in":3600,
  "hin_id": "cmuster",
  "token_type": "Bearer"
}
```

«expires_in» définit en secondes le moment auquel le token expire. Veuillez noter qu'un token peut être supprimé par le propriétaire de l'identité à tout moment. Ainsi, un token peut être invalide avant même que le délai «expires_in» n'expire.

Si un token d'accès avec un code d'autorisation invalide est demandé, une erreur est renvoyée:

```
{
  "error":"invalid_request"
}
```

Une fois que l'application tierce a reçu le token, l'accès au serveur de ressources est possible. Pour cela, le token est donné comme «Basic Auth Header». Selon le standard oAuth, «Bearer» est ajouté à l'en-tête d'autorisation de base (basic auth header).

Codes de statut

Code de statut	Description
400	Requête erronée, p. ex. paramètres manquants
403	client_secret ou GroupeToken invalide.
404	Le client_id n'est pas autorisé pour le GroupeToken correspondant. L'autorisation est accordée par le support HIN. Le GroupeToken sélectionné n'existe pas

3. CLIENT CREDENTIALS FLOW

3.1 Procédure

1. Les fournisseurs d'application qui souhaitent utiliser le Client Credentials Flow doivent contacter le support HIN (support@hin.ch).
2. Le support HIN crée un ID HIN de type «Device» pour le fournisseur d'application.
3. Le fournisseur d'application reçoit les Credentials pour l'ID HIN et les active (https://servicecenter.hin.ch/id-activation).
4. Pour obtenir le client_id et le client_secret, le fournisseur d'applications se connecte avec l'ID HIN sur https://apps.hin.ch/#app=ClientCredentials. Dans le même temps, une adresse e-mail de notification doit être définie pour l'expiration des Credentials.
5. Le fournisseur d'application peut ensuite obtenir un token d'accès (AccessToken) par le biais des appels définis dans le chapitre Appels

3.2 Obtention de Client Credentials

Le fournisseur d'application se connecte à apps.hin.ch avec son ID HIN (voir étape 2 de la procédure): https://apps.hin.ch/#app=ClientCredentials

Fonctions de l'application Web:

Fonction	Description
Définir un e-mail de notification	Une notification est envoyée à cette adresse avant l'expiration du client_secret. Si aucune adresse n'est enregistrée, la notification est envoyée à l'ID HIN associé.
Générer un nouveau client_secret	Un nouveau client_secret peut être généré via l'icône de la clé. Le client_secret affiché est valable 365 jours. Il devient actif lors de la première utilisation. Pour permettre une transition fluide d'un secret à l'autre, il peut y en avoir deux en parallèle. En cliquant une deuxième fois sur l'icône de la clé, un deuxième secret est généré. Le secret initial reste valable pour le reste de la période de 365 jours. Si le nouveau secret est activé (par la première utilisation), le secret initial n'est alors plus valable.
Supprimer un client_secret	L'icône de la corbeille permet de supprimer un client_secret. Cela est nécessaire, par exemple, si le secret a été compromis. Veuillez noter que l'entrée reste dans le tableau.

3.3 Obtention de tokens d'accès

Lors de l'obtention du token d'accès, il convient de définir pour quel GroupeToken le token doit être valable. Le client_id à utiliser a été défini à l'étape 3 de la procédure (3.1). Le client_secret a été obtenu à l'étape 4. Les tokens d'accès obtenus sont toujours valables pour l'ID HIN qui a été généré à l'étape 2 de la procédure (3.1).

3.3.1 Demande de token d'accès

```
POST https://oauth2.hin.ch/REST/v1/OAuth/GetAc-
cessToken/ACS-Applikation
  grant_type=client_credentials&
  client id=CLIENT ID&
  client_secret=CLIENT_SECRET
```

	Valeur	Description
URL	outh2.hin.ch/REST/v1/OAuth/Get-AccessToken/<GroupeToken>	Le dernier «/» est suivi du groupe de token pour lequel le token d'accès doit être obtenu
Grant_type	client_credentials	Valeur fixe. client_credentials"
client_id		client_id qui a été attribué par HIN
Client_secret		Secret qui a été généré via apps.hin.ch

Obtention d'un token d'accès avec Curl

```
curl -H 'Content-Type: application/x-www-form-urlencoded' --data-urlencode
"grant_type=client_credentials" --data-urlencode "client_id=<CLIENT ID>" --
data-urlencode "client_secret=<SECRET>"
https://oauth2.hin.ch/REST/v1/OAuth/GetAccessToken/ACS-Applikation
```

3.3.2 Réponse token d'accès

```
{
    "access_token": "<ACCESS TOKEN>",
    "expires_in": 2592000,
    "hin_id": "aakeret",
    "refresh_token": "<REFRESH TOKEN>",
    "token_type": "Bearer"
}
```

Codes de statut

Code de statut	Description
400	Requête erronée, p. ex. paramètres manquants
403	client_secret ou GroupeToken invalide.
404	Le client_id n'est pas autorisé pour le GroupeToken correspondant. L'autorisation est accordée par le support HIN. Le GroupeToken sélectionné n'existe pas

4. ACCÈS À L'APPLICATION PROTÉGÉE PAR HIN AVEC UN TOKEN D'ACCÈS

```
AUTHORIZATION: Bearer RsT5OjbzRn430zqMLgV3Ia
```

L'URL pour l'accès avec le token oAuth2 est différente de l'URL habituelle utilisée p. ex. pour l'accès via le client HIN. L'URL habituelle est précédée de «oauth2»: <application.hin.ch> → <oauth2.application.hin.ch>.

Accès avec token d'accès via Curl

```
curl --header 'Authorization: Bearer <token d'accès>' https://<oauth2.application.hin.ch>
```

4.1 Codes de statut et réponses

Codes de statut	Description
200	La requête a réussi.
400	Basic-Auth-Header manquant
401	Le token d'accès n'est pas valide ou a expiré. Obtention d'un nouveau token nécessaire.
403	En règle générale: autorisations non définies sur l'ID HIN pour l'application.

Les codes de statut peuvent également être définis par l'application concernée. Un code 403 peut donc aussi être généré par le backend.

5. REFRESH TOKEN

HIN peut contrôler la disponibilité du «refresh token» (token de rafraîchissement) sur la base du client_ID. Un refresh token est utilisé pour obtenir un nouveau token d'accès sans interaction utilisateur. Si le refresh token est activé pour un client_ID, la réponse à l'obtention initiale du token d'accès se présente ainsi:


```
{
      "access_token":"RsT5OjbzRn430zqMLgV3Ia",
      "expires_in":3600,
      "hin_id": "cmuster",
      "refresh_token": "rz6diRgWa5cqTrR8JY",
      "token_type": "Bearer"
}
```


Après expiration du token d'accès, le refresh token peut être utilisé pour le renouvellement du token pendant 7 jours supplémentaires. Au-delà, il expire et un nouveau token d'accès doit être généré par l'utilisateur:

```
POST https://oauth2.hin.ch/REST/v1/OAuth/GetAccessToken
  grant type=refresh token&
  refresh_token=REFRESH_TOKEN&
  client id=CLIENT ID&
  client_secret=CLIENT_SECRET
```

Élément	Valeur	Description
grant_type	refresh_token	Le type est invariablement «refresh_token»
refresh_token	<refresh_token>	refresh token émis lors de l'obtention du token d'accès initial (p. ex., «dSYBoT99PISaGvT5jqK3rrzoUtb»).
client_id	<client_id>	oAuth client_ID: ID attribuée par HIN à l'application tierce. Il ne s'agit pas d'une identité HIN au sens traditionnel du terme.
client_secret	<mot de passe>	oAuth client_secret: un mot de passe défini par HIN

En plus du nouveau token d'accès, l'utilisateur reçoit également un refresh token, lequel peut être réutilisé pour le renouvellement du token.

6. GLOSSAIRE

Terme	Description
Auth Code	Code pour obtenir un «token d'accès». Le code d'autorisation est un «mot de passe à usage unique».
Token d'accès	Token pouvant être utilisé pour accéder à une ressource protégée
client_ID	oAuth client_ID: ID attribuée par HIN à l'application tierce. Il ne s'agit pas d'une identité HIN au sens traditionnel du terme. Nécessaire pour pouvoir interroger le token d'accès avec le code d'autorisation
client_secret	Mot de passe défini par HIN pour l'application tierce. Doit être fourni en plus du client_ID lors de l'obtention du token d'accès.
Refresh token	Token utilisé pour obtenir un nouveau token d'accès sans interaction utilisateur