OAUTH2-DOKUMENTATION

für Anbieter von Drittapplikationen

Version: 1.4.6

Datum: 02.05.2023

INHALTSVERZEICHNIS

1. EINLEITUNG

HIN ermöglicht die Nutzung von Applikationen, die über den HIN Access Control Service (ACS) mittels oAuth2 an die HIN Plattform angebunden sind. Dazu muss der ACS-Applikationsanbieter die Nutzung von oAuth bei HIN freigeben (E-Mail an betrieb@hin.ch). HIN unterstützt die oAuth-Flows «Authorization Code» und «Client Credentials».

Authorization Code Flow:
Die zugreifende Drittapplikation (oAuth: Client), die im Namen des Anwenders auf eine andere Applikation zugreifen möchte, leitet den Anwender auf eine geschützte HIN Applikation (apps.hin.ch) weiter, an welcher er sich authentisieren muss. Apps.hin.ch generiert ein temporäres Token (oAuth: Auth Code) für den Bezug des Zugriff-Tokens (oAuth: Access Token). Der Auth Code wird über den Browser des Anwenders an die Drittapplikation weitergegeben. Hierfür gibt es zwei Varianten:
Variante a): Anzeigen des Auth Code in der apps.hin.ch-Webapplikation: Der Auth Code wird in der Webapplikation angezeigt und per Copy/Paste in die Drittapplikation übertragen.
Variante b): Übermittlung über Query Parameter: Bei der Weiterleitung des Anwenders an apps.hin.ch gibt die Drittapplikation bereits mit, auf welchen Endpunkt (oAuth: Redirect_URI) er den Auth Code möchte.
Mit dem Auth Code kann die Drittapplikation das Access Token beziehen. Mit dem Access Token kann die Drittapplikation dann auf die entsprechende Applikation über oauth2.*<bestehendeURL>* zugreifen. Access Token können entweder für eine URL oder für eine Gruppe von URLs gültig sein.

Client Credentials Flow:
Der Client Credentials Flow wurde speziell für Machine-to-Machine-Anwendungsfälle konstruiert. Im Client Credentials Flow entfällt der Bezug des Auth Codes und somit die Interatkion mit einem Endbenutzer. Im Gegensatz zum Authorization Code Flow ist das ausgestellte Access Token jedoch auch nur für eine spezifischen, vorkonfigurierten Benutzer gültig. Der Client Credentials Flow eignet sich somit nicht, wenn ein Zugriff im Namen eines Endbenutzers erfolgen soll.

2. AUTHORIZATION CODE FLOW

2.1 Ablauf

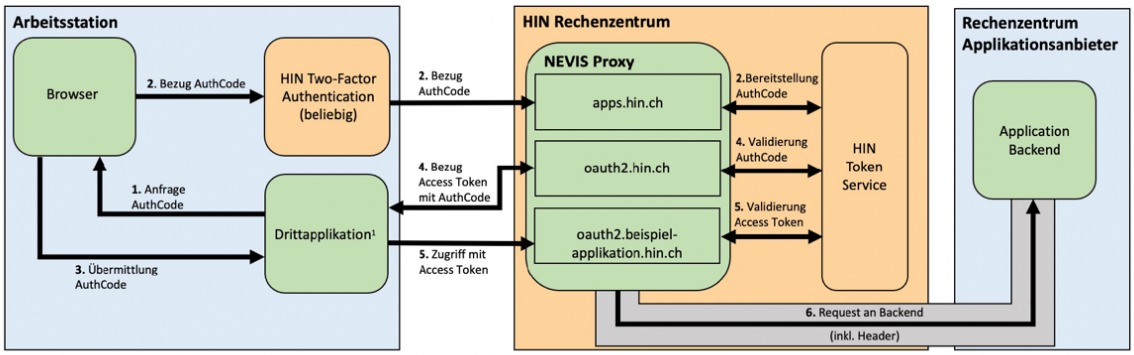
1. Eine Drittapplikation möchte im Namen einer HIN Identität auf eine HIN geschützte Ressource (ACS-Applikation) zugreifen.
2. Wenn in der Drittapplikation kein Token für die entsprechende HIN Identität vorhanden ist, muss der Browser an apps.hin.ch weitergeleitet werden. Der zugreifende Anwender muss sich beim Zugriff auf apps.hin.ch mittels Zwei-Faktor-Authentisierung (bspw. HIN Client) anmelden. Dem Anwender wird in der Webapplikation ein Auth Code dargestellt.
3. Der Auth Code wird vom Anwender in die Drittapplikation übertragen. Dazu werden verschiedene Mechanismen verwendet (siehe Kapitel «Bezug und Verwendung Auth Code»):
    a. Copy/Paste des Auth Codes oder Fotografieren eines QR-Codes
    b. direkte Übermittlung über https oder Protocol Handler
4. Mit dem Auth Code kann die Drittapplikation das Access Token beziehen.
5. Mit dem Access Token ist der Zugriff auf die geschützte Ressource im Namen der HIN Identität des Anwenders möglich.

2.2 Bezug von Auth Codes

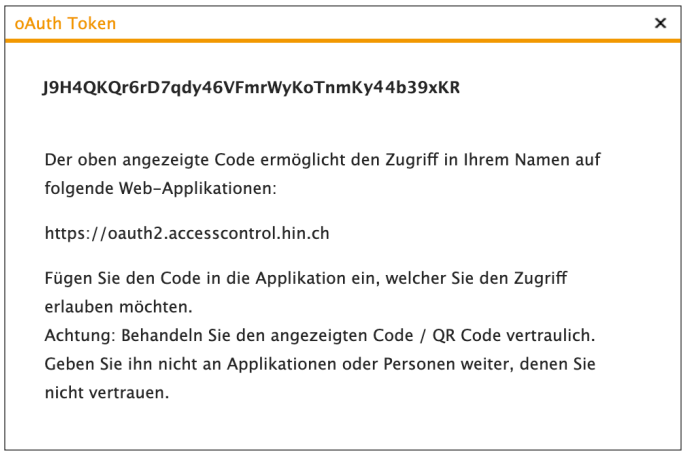
Der Auth Code ist ein OTP (One Time Password), welches für den Bezug des Access Tokens verwendet wird und ist zehn Minuten gültig. Der Bezug kann über zwei Varianten erfolgen:

2.2.1 Variante a): Anzeigen des Auth Code in der Webapplikation

Auf apps.hin.ch können Auth Codes für den Bezug des Access Tokens generiert werden.
Für den oAuth-Service wird ein separates Register bereitgestellt.



[1] z.B. PIS



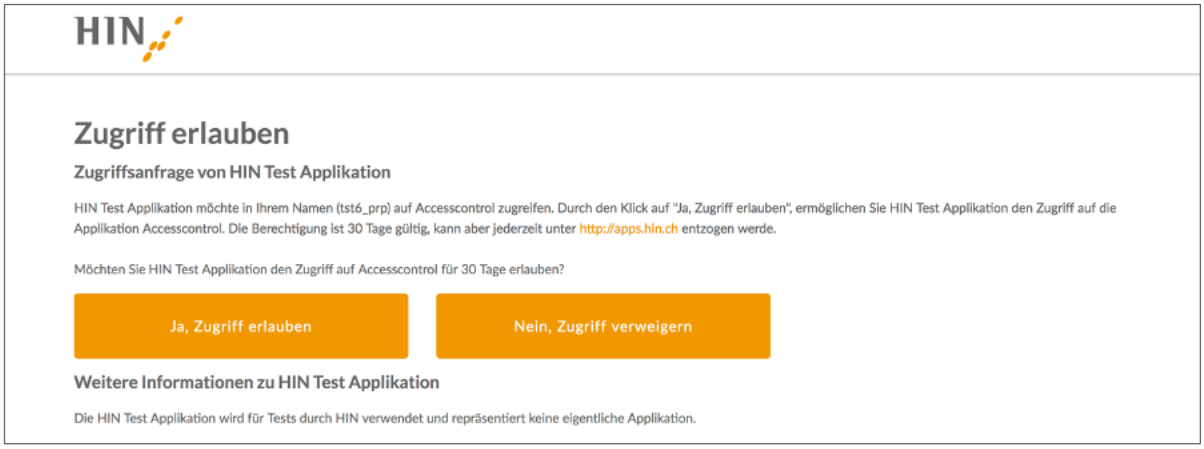
Beim Zugriff auf apps.hin.ch wird standardmässig das erste Register («HIN Mail») angezeigt. Um das Ganze für den Anwender zu vereinfachen, können die Auth Codes über einen Direktlink bezogen werden:

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

Der Wert nach tokenGroup= ist je nach Ziel-Applikation unterschiedlich und kann bei HIN angefragt werden.

2.2.2 Variante b): Übermittlung über Query Parameter

Bei dieser Variante wird der Auth Code an eine definierte Redirect_URI übermittelt. Die anwendende Person bestätigt dies durch den Klick auf «Ja, Zugriff erlauben».



Die Redirect_URI wird über die aufgerufene URL mitgegeben:

```
http://apps.hin.ch/REST/v1/OAuth/GetAuthCode/<Token-
Gruppe>?response_type=code&client_id=<client_id>&redirect_uri=<Redi-
rect_URI>&state=<state>
```

Wert	Beschreibung
<TokenGruppe>	Applikationsgruppe, für welche ein Token bezogen werden soll (Achtung: Der Name ist Case-sensitive)
<Redirect_URI>	Redirect_URI, an welcher der Browser nach dem Bestätigen des Dialoges mit dem Auth Code weitergeleitet wird. Der Wert muss URL encoded sein. Zudem muss die Redirect_URI für die entsprechende Client_ID hinterlegt werden. Das Verwenden einer nicht hinterlegten Redirect_URI führt zu Fehlermeldungen. Dies wird durch HIN vorgenommen. Es können pro Client_ID mehrere Redirect_URIs definiert werden.
<state>	Ein statischer Wert, welcher bei der Weiterleitung des Browsers bestehen bleibt
<client_id>	oAuth-Client_ID: HIN vergebene ID für die Drittapplikation. Es handelt sich hierbei nicht um eine HIN Identität im herkömmlichen Sinne.

Beispiel Request für die Applikation «ACS-Applikation»:

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

Der Auth Code wird mittels URI Parameter an die gewünschte Redirect_URI übermittelt. Zudem wird der State mitgeliefert.

```
https://www.hin.ch/?state=teststate&code=qdoWMwRNHnn9wDNynbMxytwahE-
GNXBqtipQhZXLF
```

2.3 Bezug von Access Tokens

Das Access Token ist das Token, welches für den effektiven Zugriff auf die Applikation verwendet wird.

2.3.1 Request Access Token

Mit dem erhaltenen Auth Code kann das Access Token bezogen werden. Der Bezug erfolgt mittels eines POST auf oauth2.hin.ch/REST/v1/OAuth/GetAccessToken, dabei werden die Parameter als form-data übermittelt (`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
```

Die Redirect_URI muss übereinstimmen mit derjenigen, die beim Bezug des Auth Codes verwendet wurde. Bitte beachten Sie, dass der Parameter «client_secret» je nach Konfiguration nicht notwendig ist.

Bezug eines Access Tokens mit Curl

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

Bitte beachten Sie, dass der Body URL-Encoded sein muss. Allfällige Sonderzeichen im client_secret müssen beachtet werden. Weiter war in älteren Dokument-Versionen die URL für den Bezug der Access Token "https://oauth2.hin.ch/REST/v1/getoAuthToken". Diese URL wird langfristig abgelöst. Systeme, welche die alte URL verwenden, sollten umgestellt werden.

Element	Wert	Beschreibung
grant_type	authorization_code	Der Type ist fix "authorization_code"
code	auth_code	Auth Code, welcher durch den Anwender kopiert wurde
redirect_uri	<"leer">	Endpunkt, an welchen die Response geliefert wird. Muss mit dem Wert übereinstimmen, welcher beim Bezug des Auth Codes definiert wurde
client_id	<Client_ID>	oAuth-Client_ID: HIN vergebene ID für die Drittapplikation. Es handelt sich hierbei nicht um eine HIN Identität im herkömmlichen Sinne.

Element	Wert	Beschreibung
client_secret	<Password>	oAuth-Client_secret: Ein durch HIN definiertes Passwort

2.3.2 Response Access Token

Das Access Token wird in Form eines JSON zurückgeliefert:

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

"Expires_in" definiert in Sekunden, wann das Token ausläuft. Bitte beachten Sie, dass ein Token jederzeit durch den Identitätsbesitzer gelöscht werden kann. Somit kann ein Token auch vor Ablauf der "expires_in" Frist ungültig werden.

Wird ein Access Token mit einem ungültigen Auth Code angefragt, wird ein Fehler zurückgegeben:

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

Hat die Drittapplikation das Token erhalten, ist der Zugriff auf den Ressource Server möglich. Dazu wird das Token als Basic Auth Header mitgegeben. Der Basic Auth Header wird gemäss oAuth Standard "Bearer" angefügt.

Status-Codes

Status-Code	Beschreibung
400	Fehlerhafter Request bspw. fehlende Parameter
403	client_secret oder TokenGruppe ungültig.
404	Client_ID ist nicht für entsprechende TokenGruppe berechtigt. Berechtigung wird durch den HIN Support erteilt. Gewählte TokenGruppe existiert nicht

3. CLIENT CREDENTIALS FLOW

3.1 Ablauf

1. Applikationsanbieter, die den Client Credentials Flow nutzen möchten, melden sich beim HIN Support (support@hin.ch)
2. Der HIN Support erstellt eine HIN ID vom Typ «Device» für den Applikationsanbieter
3. Der Applikationsanbieter erhält die Credentials zur HIN ID und nimmt diese in Betrieb (https://servicecenter.hin.ch/id-activation).
4. Um die client_id und das client_secret zu erhalten, meldet sich der Applikationsanbieter mit der HIN ID bei https://apps.hin.ch/#app=ClientCredentials an. Im gleichen Zug muss eine Notification-E-Mail-Adresse für den Ablauf der Credentials definiert werden.
5. Anschliessend kann der Applikationsanbieter über die Calls, die im Kapitel Calls definiert sind, ein AccessToken beziehen

3.2 Bezug Client Credentials

Der Applikationsanbieter meldet sich mit seiner HIN ID (siehe Ablauf Schritt 2) bei apps.hin.ch an: https://apps.hin.ch/#app=ClientCredentials

Funktionen der Webapplikation:

Funktion	Beschreibung
Definieren einer Notification-E-Mail	An diese wird vor dem Ablauf des client_secrets eine Notification gesendet. Wird keine Adresse hinterlegt, geht die Notification an die verknüpfte HIN ID.
Generieren eines neuen Client_Secrets.	Über das «Schlüssel»-Icon kann ein neues client_secret generiert werden. Das angezeigte client_secret ist für 365 Tage gültig. Durch die erstmalige Benutzung wird dieses aktiv. Um einen unterbruchfreien Übergang von einem Secret zum nächsten zu ermöglichen, können parallel zwei bestehen. Durch den zweiten Klick auf das «Schlüssel»-Icon wird ein zweites Secret generiert. Das initiale Secret bleibt für die restliche Laufzeit der 365 Tage gültig. Wird das neue Secret aktiviert (durch die erstmalige Verwendung), wird das initiale Secret ungültig.
Löschen eines Client_Secrets	Über das «Papierkorb»-Icon kann ein Client_Secret gelöscht werden. Dies ist beispielsweise notwendig, wenn das Secret kompromittiert wurde. Bitte beachten Sie, dass der Eintrag in der Tabelle bestehen bleibt.

3.3 Bezug von Access Tokens

Beim Bezug des AccessTokens muss definiert werden, für welche TokenGruppe das Token gültig sein soll. Die zu verwendende Client_ID wurde bei 3 im Ablauf (3.1) definiert. Das Client_Secret wurde in Schritt 4 bezogen. Die bezogenen Access Tokens sind immer für die HIN ID gültig, welche unter 2. im Ablauf (3.1) generiert wurde.

3.3.1 Request Access Token

```
POST https://oauth2.hin.ch/REST/v1/OAuth/GetAc-
cessToken/ACS-Applikation
  grant_type=client_credentials&
  client_id=CLIENT_ID&
  client_secret=CLIENT_SECRET
```

	Wert	Beschreibung
URL	outh2.hin.ch/REST/v1/OAuth/Get-AccessToken/<TokenGruppe>	Nach dem letzten "/" folgt die Token-Gruppe, für welche das AccessToken bezogen werden soll
Grant_type	client_credentials	Fix definierten Wert. client_credentials"
client_id		client_id, welche durch HIN vergeben wurde
Client_secret		Secret, welches über apps.hin.ch generiert wurde

Bezug eines Access Tokens mit 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/O-
Auth/GetAccessToken/ACS-Applikation
```

3.3.2 Response Access Token

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

Status-Codes

Status-Code	Beschreibung
400	Fehlerhafter Request bspw. fehlende Parameter
403	client_sectret oder Tokengruppe ungültig.
404	Client_ID ist nicht für entsprechende Tokengruppe berechtigt. Berechtigung wird durch den HIN Support erteilt. Gewählte Tokengruppe existiert nicht

4. ZUGRIFF AUF HIN GESCHÜTZTE APPLIKATION MIT ACCESS TOKEN

```
AUTHORIZATION: Bearer RsT5OjbzRn430zqMLgV3Ia
```

Die URL für den Zugriff mit oAuth2 Token unterscheidet sich von der üblichen URL, die bspw. für den Zugang mittels HIN Client verwendet wird. Der üblichen URL wird «oauth2» vorangestellt: <application.hin.ch> → <oauth2.application.hin.ch>.

Zugriff mit AccessToken via Curl

```
curl –header ‘Authorization: Bearer <AccessToken>’ https://<oauth2.application.hin.ch>
```

4.1 Status-Codes und Responses

Status-Codes	Beschreibung
200	Request war erfolgreich.
400	Fehlender Basic-Auth-Header
401	Access Token ist ungültig oder abgelaufen. Bezug eines neuen Tokens notwendig.
403	In der Regel: Berechtigungen auf der HIN ID für die Applikation nicht gesetzt.

Status-Codes können auch durch die aufgerufene Applikation gesetzt werden. Ein 403 kann somit auch durch das Backend generiert worden sein.

5. REFRESH TOKEN

HIN kann auf Basis der Client_ID die Verfügbarkeit von "Refresh Token" steuern. Ein Refresh Token wird verwendet, um ohne Benutzer-Interaktion ein neues Access Token zu beziehen. Ist das Refresh Token für eine Client_ID aktiviert, sieht die Antwort auf den initialen Bezug des Access Tokens folgendermassen aus:


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


Nach Ablauf des Access Token kann das Refresh Token noch 7 Tage für die Erneuerung verwendet werden. Anschliessend verfällt es und es muss ein neues Access Token durch den Anwender generiert werden:

```
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
```

Element	Wert	Beschreibung
grant_type	refresh_token	Der Type ist fix "refresh_token".
refresh_token	<refresh_token>	Refresh Token, welches bei Bezug des initialen Access Token ausgestellt wurde (bspw. "dSY-BoT99PISaGvT5jqK3rrzoUtb").
client_id	<Client_ID>	oAuth-Client_ID: HIN vergebene ID für die Drittapplikation. Es handelt sich hierbei nicht um eine HIN Identität im herkömmlichen Sinne.
client_secret	<Password>	oAuth-Client_secret: Ein durch HIN definiertes Passwort

Neben dem neuen Access Token erhält man wiederum ein Refresh Token, welches wieder für die Erneuerung verwendet werden kann.

6. GLOSSAR

Begriff	Beschreibung
Auth Code	Code für den Bezug eines «Access Tokens». Auth Code ist ein «one time password».
Access Token	Token, welches für den Zugriff auf eine geschützte Ressource genutzt werden kann
Client_ID	oAuth-Client_ID: HIN vergebene ID für die Drittapplikation. Es handelt sich hierbei nicht um eine HIN Identität im herkömmlichen Sinne. Wird benötigt, um mit dem Auth Code das Access Token abfragen zu können
client_secret	Durch HIN definiertes Passwort für die Drittapplikation. Muss neben Client_ID beim Bezug des Access Token mitgegeben werden.
Refresh Token	Token um ohne Benutzer-Interaktion ein neues Access Token zu beziehen