Visión general

En lo correspondiente a la seguridad de las APIs debe considerarse algunos aspectos de forma general que se deben a la arquitectura de **//ABANCA**. A grandes rasgos mencionaremos los puntos más salientes que luego se profundizarán en secciones específicas:

- **Federación de identidades:** para que los clientes de la entidad utilicen aplicaciones de terceros con sus credenciales de **//ABANCA** se implementa el flujo de autenticación con authorization code de OAuth autenticación con authorization code de OAuth utilizando el esquema de redirecciones estándar. Esto implica que la aplicación deberá redirigir al usuario a un portal de **//ABANCA** que validará las credenciales y luego redirigirá nuevamente al usuario a la aplicación con la información necesaria para continuar con la operación y acceder a las APIs utilizando la identidad correspondiente
- **Autenticación de aplicaciones:** cada aplicación de terceros deberá estar registrada en el portal de desarrolladores de **//ABANCA** y contará con un identificador así como un API Key que deberá utilizar en las llamadas que haga a las distintas APIs y servicios de **//ABANCA.** Estos datos también podrán ser solicitados, sin necesidad de registro, a través de la dirección de correo electrónico openbanking@abanca.com
- **Identidad de los usuarios por access token:** la aplicación de terceros podrá obtener tokens de acceso que representan al usuario que las está utilizando. Estos tokens son de tipo oscuro y deberán ser incluidos en las llamadas correspondientes a las APIs para la correcta identificación de los usuarios
- **Seguridad reforzada:** se soportan escenarios donde se requiere un refuerzo de identidad del usuario a través de un desafío

Autenticación

Para permitir a los usuarios el uso de las APIs de **//ABANCA** se implementa **OAuth** como mecanismo de autenticación. En particular se utiliza el flujo de authorization code con redirección. Es decir, el usuario debe ser redirigido de la aplicación al sistema de autenticación de **//ABANCA**, que se encargará de validar sus credenciales y, si fuera necesario, solicitar su consentimiento. Una vez finalizado esto, se redirigirá nuevamente al usuario a la aplicación original para que pueda operar con sus datos a través del API de **//ABANCA**.

La URL a donde debe redirigirse al usuario para iniciar el proceso de autenticación se compone de la siguiente forma:

```
GET /oauth/{id cliente}/Abanca?response_type=code&redirect_uri={uri
redirección}&scope={scopes}&state={state}

HTTP/1.1

Host: {servidor api}
```

Donde:

- **Id Cliente:** es el identificador que tiene la aplicación que está utilizando el usuario tal como es indicado en el portal de desarrolladores de **//ABANCA**
- **URI Redirección:** es la URL de la aplicación a la que deberá redirigirse al usuario una vez finalizada su autenticación, deberá coincidir con alguna de las registradas para la aplicación (del punto anterior) en el portal de desarrolladores de **//ABANCA.** Alternativamente, podrán darse de alta a través del correo electrónico openbanking@abanca.com
- **Scopes:** es la lista de scopes **OAuth** que se quiere acceder del usuario (opcional)
- **State:** es un valor arbitrario definido por la aplicación llamante para validaciones de seguridad adicionales al recibir la respuesta del proceso de autenticación (opcional)
- **Servidor API:** es la dirección del servidor donde se sirven las **APIs** de **//ABANCA**

Cuando el proceso de autenticación y consentimientos termine exitosamente, el usuario será redirigido la URL indicada, a la cual se le concatenarán algunos parámetros, es decir se recibirá una respuesta como la siguiente:

```
HTTP/1.1 302 Found


Location: {uri redirección}?code={authorization code}&scope={scopes}&state{state}
```

Donde:

- **URI Redirección:** es la URL de la aplicación que se indicó al iniciar el flujo de autenticación **OAuth**
- **Authorization Code:** es el código de autorización que representa al usuario
- **Scopes:** es la lista de scopes se indicó al iniciar el flujo de autenticación **OAuth** (opcional). Los posibles valores son *Accounts*, *Transactions* (AISP), *Transfers* (PISP) y *Funds (*CBII)
- **State:** es un valor arbitrario definido por la aplicación se indicó al iniciar el flujo de autenticación **OAuth** (opcional)

A partir de este punto la aplicación puede obtener el **access token** (para hacer las llamadas a las **APIs**) y **refresh_token** (para refrescar el **access token** sin necesidad de ejecutar todo el flujo de autenticación).

Obtención de access token

Para obtener un **access token** es necesario presentar las **credenciales** del **Usuario** para que las mismas sean validadas por el servicio de Single Sign On de **//ABANCA**. Para ello bastará con realizar las peticiones con el **grant type** adecuado para el caso.

Generación del access token

Para completar el flujo de autenticación descrito en la sección anterior es necesario generar un **access token** con el **authorization code** obtenido. Para esto es necesario realizar una petición como la que se muestra a continuación:

```
POST /oauth2/token HTTP/1.1

Host: {servidor api}

Content-Type: application/x-www-form-urlencoded

AuthKey: {api key}

grant_type=authorization_code&APLICACION={id liente}&code={authorization code}
```

Donde:

- **API Key:** es la clave de identificación de la aplicación en la infraestructura de seguridad, tal como está indicada en el portal de desarrolladores de **//ABANCA**
- **Id cliente:** es el identificador que tiene la aplicación que está utilizando el usuario tal como es indicado en el portal de desarrolladores de **//ABANCA**
- **Authorization Code:** es el código de autorización que representa al usuario y fue obtenido como parte del flujo de autenticación descrito en el apartado anterior
- **Servidor API:** es la dirección del servidor donde se sirven las **APIs** de **//ABANCA**

Es **IMPORTANTE** tener presente que los parámetros deben ser presentados con el formato de un formulario codificado, es decir, el Content-Type de la petición debe ser x-www-form-urlencoded. Esto es algo estándar de OAuth.

En el caso de que el **authorization code** sea válido, el servicio de Single Sign On devolverá el **access token** junto con más información. La respuesta se conoce como **ticket de autenticación** y su contenido puede verse a continuación:

```
HTTP/1.1 200 OK

Content-Type: application/json;charset=UTF-8

Cache-Control: no-storePragma: no-cache

{"access_token": {access token},

"token_type": "bearer",

"expires_in": 599,
```

```
"username": {user name},

"displayName": {display name},

"refresh_token": {refresh token},

".issued": "Sun, 07 Jul 2019 13:11:52 GMT",

".expires": "Sun, 07 Jul 2019 13:21:52 GMT"

}
```

Donde:

- **access token:** es el token OAuth propiamente dicho y su valor es el que deberá ser presentado en las peticiones a las **APIs**
- **username:** identificador del **usuario** que fue autenticado
- **displayname:** nombre visible del **usuario** que fue autenticado si estuviera disponible
- **refresh token:** token de refresco que puede utilizarse para refrescar el **access token** (ver siguiente sección para más detalles)

El resto de los campos se explican a continuación:

- **token_type:** el tipo de esquema que debe utilizarse para presentar este token en las **APIs**. En el caso de **//ABANCA** siempre es Bearer
- **expires_in:** cantidad de segundos durante los cuales el **access token** será válido.
- **issued:** fecha de emisión del **access token**
- **expires:** fecha de expiración del **access token** token (corresponde a la fecha indicada en issued más el tiempo indicado en expires_in)

Refresco del access token

Este mecanismo permite la obtención de un **access token** sin necesidad de completar todo el ciclo de autenticación OAuth. Para ello se utilizará el **refresh token** obtenido con el procedimiento anterior y se solicitará otro **access token** de la siguiente manera:

```
POST /oauth2/token HTTP/1.1

Host: {servidor api}

Content-Type: application/x-www-form-urlencoded
```

```
AuthKey: {api key}


grant_type=refresh_token&APLICACION={id cliente}&refresh_token={refresh token}
```

Donde:

- **API Key:** es la clave de identificación de la aplicación en la infraestructura de seguridad, tal como está indicada en el portal de desarrolladores de **//ABANCA**
- **Id cliente:** es el identificador que tiene la aplicación que está utilizando el usuario tal como es indicado en el portal de desarrolladores de **//ABANCA**
- **Refresh token:** es el token de refresco obtenido cuando se generó originalmente el **access token** con el **authorization code**
- **Servidor API:** es la dirección del servidor donde se sirven las **APIs** de **//ABANCA**

En caso dque el **refresh token** sea válido, el servicio devolverá un ticket de autenticación similar al que se obtiene durante la generación del **access token**, con la diferencia de que no contendrá un nuevo **refresh token**. Esto puede verse a continuación:

```
HTTP/1.1 200 OKContent-Type: application/json;charset=UTF-8Cache-Control:
no-storePragma: no-cache{


"access_token": {access token},


"token_type": "bearer",


"expires_in": 599,


"username": {user name},


"displayName": {display name},


".issued": "Sun, 07 Jul 2019 13:21:52 GMT",


".expires": "Sun, 07 Jul 2019 13:31:52 GMT"


}
```

Errores en la obtención

En caso de que la petición de access token presentara algún error, el servicio devolverá una respuesta como la que se muestra a continuación:

```
HTTP/1.1 400 Bad Request

Content-Type: application/json;charset=UTF-8

Cache-Control: no-store

Pragma: no-cache{

"error": "invalid_client"

}
```

El contenido de la respuesta indicará la causa del error siendo los escenarios más frecuentes:

- Identificador de la aplicación cliente incorrecto
- **Authorization code** incorrecto o expirado
- **Refresh token** incorrecto o expirado

Invocación de APIs

Para invocar cualquier **API** de **//ABANCA** simplemente debe incluirse en la petición la cabecera Authentication con esquema Bearer y el **access token** obtenido previamente así como también la cabecera AuthKey con el **API Key** correspondiente a la aplicación (de la misma forma que se hacía en las peticiones del servicio de Single Sign On). A continuación se muestra un ejemplo de la obtención de un recurso del **API**, aunque esto se aplica para cualquier método de acceso a la misma:

```
GET {ruta recurso} HTTP/1.1

Host: {servidor api}

AuthKey: {api key}

Authorization: Bearer {access token}
```

Donde:

- **Ruta recurso:** es la ruta del recurso del **API** que nos interesa obtener
- **API Key:** es la clave de identificación de la aplicación en la infraestructura de seguridad, tal como está indicada en el portal de desarrolladores de **//ABANCA**
- **Access token:** es el token de acceso obtenido del servicio de Single Sign On
- **Servidor API:** es la dirección del servidor donde se sirven las **APIs** de **//ABANCA**

Posibles escenarios de error

Existen algunos casos muy claros donde pueden encontrarse respuestas de error vinculadas con la seguridad del **API**.

Todos los errores presentan en el contenido de la respuesta una estructura común para facilitar su entendimiento y procesamiento. A continuación detallaremos los más relevantes referentes a la seguridad. Un ejemplo se muestra a continuación.

```
{"errors": [ { "id": "46fcd0c9-74fa-4045-b5f4-cd4605b48fd1", "code":
"API_00001","title": "Unable to determine user's identity", "technicalDescription":
"OAuth token is missing or is not valid (corrupted, expired, etc)" } ] }
```

Donde:

- **id:** identificador único del error
- **code:** código que representa la situación de error encontrada
- **title:** resumen del error
- **technicalDescription:** descripción relativamente detallada de lo que está mal
- **details:** en algunos casos se incluye un objeto con información específica del error obtenido para su correcto tratamiento

También el estado HTTP del error variará de acuerdo a la naturaleza del error.

401 - Unauthorized

Indica que el **access token** presentado no es válido (corrupto o caducado) o ni siquiera fue encontrado en la cabecera Authorization.

El código de error que representa esta situación es API_00001.

403 - Forbidden

En estos casos el **access token** presentado es válido con lo cual se puede validar la **identidad** del **Usuario**, pero por algún motivo se carece de privilegios suficientes para completar la petición. Existen diversos motivos para esto que se corresponden con distintos códigos de error:

- **API_00002:** el token no fue emitido para el uso del **API** que se accede (depende de que permisos tiene el **Cliente**)
- **API_00004:** el usuario no tiene asignados permisos suficientes para acceder al recurso

- **API_00005:** la petición requiere un desafío de seguridad
- **API_00006:** : la respuesta a un desafío de seguridad fue incorrecta
- **API_00008:** el usuario carece de privilegios. Es un escenario bastante genérico
- **API_00016:** el **API** requiere de ciertos **scopes** que el **usuario** usuario no ha autorizado para que el **Cliente** acceda a esa información
- **API_00017:** la petición requiere un desafío de seguridad pero el usuario carece de los mecanismos necesario para responderlo

Cifrado de Peticiones

Para el cifrado de las peticiones al API, y de acuerdo a la normativa PSD2, se hará uso de los certificados eIDAS a dos niveles:

- Certificado QWAC para el establecimiento de conexión: se podrá hacer uso de un certificado de cliente cuya cadena de certificación se haya registrado previamente
- Certificado QSEAL para la firma de las peticiones

Para el firmado de las peticiones, deberán de incluirse las siguientes cabeceras HTTP:

- **Date:** fecha en formato UTF-8
- **Digest:** con formato Digest: SHA-256={digest} donde {digest} será un hash codificado en Base64 del body de petición, o de una cadena vacía en caso de GET
- **Request-Target:** que contendrá, en minúsculas, el método y el recurso invocado. Por ejemplo: "get /psd2/me/accounts" o "post /psd2/me/accounts/asd3dsd3servbj5hfswr3vbg2thbn3n4nfe88bld/transfers"
- **X-Request-ID:** identificador de la petición, en formato GUID
- **Signature:** contendrá el algoritmo, las cabeceras utilizadas y la firma generada en formato ="keyId={serial number del certificado}",algorithm={algoritmo de firma},headers="request-target date digest x-request-id",signature={valor de la firma generada}
- **TPP_Signature_Certificate (opcional):** contendrá la clave pública del certificado utilizado para la firma. No será necesario si el certificado ha sido registrado previamente

Seguridad reforzada

Introducción

Existen operatorias sobre el **API** que por su sensibilidad requieren un nivel adicional de validación por parte del usuario. Para ello al momento de recibir una petición el API puede, en vez de procesarla normalmente, devolver un desafío que deberá ser resuelto para poder completar el procesamiento de la petición original.

Aplicabilidad

Cualquier operación sobre un recurso del API es susceptible a requerir una validación adicional. Esto quiere decir que los verbos HTTP que pueden devolver un desafío para completar la operación son:

- GET
- POST
- PUT
- PATCH
- DELETE

Los escenarios donde se aplica el desafío dependerán de varios factores que serán determinados por cada recurso en concreto.

Visión general

Ejecución sin desafíos

Retomemos el caso de acceso a un recurso asumiendo que no hay necesidad de elevar el nivel de seguridad con lo cual el flujo de ejecución es el normal:

Ejecución con desafíos

Pero si asumimos que, por la sensibilidad de la operación, se requiere un refuerzo de seguridad para procesar efectivamente la petición, el flujo es distinto y va a requerir de un paso adicional:

Donde la petición es exactamente la misma en ambos pasos, pero la segunda vez se incluye información adicional con la respuesta al desafío. Entre las dos peticiones el usuario deberá obtener el valor de la solución del desafío, de acuerdo a lo que se le haya indicado como respuesta a la primera petición.

Detalles

A continuación se presentan los detalles técnicos que permiten entender cómo trabajar con los desafíos del **API**. Asumimos que se realiza una petición que corresponde a una operación que requiere un refuerzo de seguridad (por ejemplo: una petición de transferencia a un destinatario no habitual).

La petición podría verse como la siguiente:

```
POST {ruta recurso} HTTP/1.1
```

```
Host: {servidor api}


AuthKey: {api key}


Authorization: Bearer {access token}


Content-Type: application/json


{datos petición}
```

Donde:

- **Ruta recurso:** es la ruta del recurso del **API** que nos interesa obtener
- **API Key:** es la clave de identificación de la aplicación en la infraestructura de seguridad, tal como está indicada en el portal de desarrolladores de **//ABANCA**
- **Access token:** es el token de acceso obtenido del servicio de Single Sign On
- **Datos petición:** es el contenido del cuerpo de la petición
- **Servidor API:** es la dirección del servidor donde se sirven las **APIs** de **//ABANCA**

Recibiendo un desafío

En primer lugar, una vez enviada la petición al **API** la misma nos dará una respuesta indicando que la operación no está permitida (status code 403 - Forbidden) en cuyo detalle se podrá ver que es necesario resolver un desafío.

```
HTTP/1.1 403 Forbidden


Content-Type: application/json;charset=UTF-8


Cache-Control: no-store


Pragma: no-cache
```

```
{ "errors": [ { "id": "e170b6b-6740-4113-8573-e2f14a4bfcdd", "code": "API_00005",
"title": "User should solve a security challenge to access the resource",
"technicalDescription": "The resource has a further security requirement that can
be fulfilled by resolving a security challenge" "details": { "solutionHint":
"**********35", "challengeId":
"58667e9b-4a8f-4706-a71e-5844514a3976","attemptsLeft": 3, type": "OTP_SMS",
"responseMode": "Header", "expiresIn": 599 } } ] }
```


Aspectos que se deberán de tener en cuenta:

- por tratarse de una situación de error, el formato de la respuesta es el especificado anteriormente
- el código de error (API_00005) que identifica el hecho de que se está solicitando la resolución del desafío para poder continuar
- la propiedad details del error contiene la información del desafío a resolver
- Dentro del detalle del desafío se destaca:
    - **challengeId:** identificador del desafío asociado a la petición que se acaba de realizar: deberá informarse junto con el valor de solución del mismo en la petición siguiente
    - **type:** tipo del desafío a resolver, indica de donde se obtendrá el valor de la solución a informar

Es responsabilidad de la interfaz de usuario presentar la información necesaria para orientar al usuario sobre la situación para que el mismo pueda obtener la solución del desafío, así como también permitirle ingresar la misma para poder confirmar la operación.

Resolviendo el desafío

A esta altura asumimos que el usuario ya obtuvo la solución del desafío, y la indicó a través de la interfaz. A partir de ese punto debe reintentarse la petición incluyendo la solución del desafío, siguiendo el ejemplo y considerando que se indicaba el modo de respuesta Header, corresponde volver a hacer la misma petición agregando dos cabeceras que servirán para indicar la solución como puede verse a continuación:

```
POST {ruta recurso} HTTP/1.1


Host: {servidor api}
```

```
AuthKey: {api key}


Authorization: Bearer {access token}


Content-Type: application/json


x-challenge-id: {challenge id}


x-challenge-response: {challenge response}


{datos petición}
```

Donde:

- **Ruta recurso:** es la ruta del recurso del **API** que nos interesa obtener
- **API Key:** es la clave de identificación de la aplicación en la infraestructura de seguridad, tal como está indicada en el portal de desarrolladores de **//ABANCA**
- **Access token:** es el token de acceso obtenido del servicio de Single Sign On
- **Datos petición:** es el contenido del cuerpo de la petición
- **Challenge id:** es el valor del identificador de desafío obtenido con la petición original
- **Challenge response:** es el valor de solución al desafío planteado a la petición original
- **Servidor API:** es la dirección del servidor donde se sirven las **APIs** de **//ABANCA**

Cabe aclarar que ni los datos de la petición ni la ruta del recurso pueden alterarse respecto a la petición original, de hacerlo las peticiones serían consideradas como diferentes y comenzaría la evaluación de necesidad de refuerzo de seguridad nuevamente.

Solución incorrecta

En el caso de que el valor provisto no fuera válido, se responderá nuevamente con un error (status code 403 - Forbidden) cuyo contenido indicara la situación. La respuesta es muy similar a la recibida cuando se intentó la petición por primera vez:

```
HTTP/1.1 403 Forbidden
```

```
Content-Type: application/json;charset=UTF-8


Cache-Control: no-store


Pragma: no-cache


{"errors": [{"id": "932c432f-32e5-47f6-8efc-2be3fe23f1f7","code":
"API_00006","title": "Challenge response provided is
incorrect","technicalDescription": "The value provided is not a valid response
for the challenge being solved""details": {"solutionHint":
"**********35","challengeId":
"58667e9b-4a8f-4706-a71e-5844514a3976","attemptsLeft": 3,"type":
"OTP_SMS","responseMode": "Header","expiresIn": 599}}]}
```

Lo que es importante tener en cuenta aquí es lo siguiente:

- El identificador del desafío sigue siendo el mismo
- Se redujo la cantidad de intentos disponibles
- Se redujo el tiempo disponible hasta la expiración del desafío

Solución correcta

En caso de que el valor provisto como solución del desafío sea el correcto, el API continuará procesando la petición con normalidad y devolverá el resultado esperado para la operación que se está ejecutando.

Tener en cuenta que haber resuelto correctamente el desafío no implica que el resultado de la petición sea exitoso (es decir, status code2XX); el procesamiento de la petición podría resultar en otras condiciones de error asociadas específicamente con la lógica propia de la operatoria.

Reenvío del desafío

Podría darse el caso de que un desafío cuya solución se envía al usuario a través de un canal independiente no se hubiera recibido (por ejemplo: una notificación push que se envió mientras el usuario tenía su móvil apagado). En esos casos es posible solicitar el reenvío de la información, para esto simplemente bastará con hacer una petición similar a cuando se

responde el desafío con la diferencia que se indica que se solicita su reenvío. Esto puede verse en el siguiente ejemplo

```
POST {ruta recurso} HTTP/1.1

Host: {servidor api}

AuthKey: {api key}

Authorization: Bearer {access token}

Content-Type: application/json

x-challenge-id: {challenge id}

x-challenge-action: resend{datos petición}
```

La respuesta será equivalente a la de la petición original, con la información del desafío actualizada (el tiempo hasta la expiración).

Resumen

Este apartado presenta una referencia rápida de las cuestiones técnicas asociadas a los desafíos en las APIs.

Respuesta básica

- **Status Code:** 403. Se indica que no se permite la operación tal como fue hecha la petición. Existen otros escenarios donde se utiliza este status code por lo tanto es necesario revisar también el código de error.
- **Códigos de Error:** el código de error recibido (errors[].code) puede indicar distintas situaciones
    - **API_00005:** es necesario resolver el desafío para procesar correctamente la petición
    - **API_00006:** se intentó resolver el desafío con un valor inválido

    - **API_00007:** ocurrió un error inesperado procesando la solución del desafío
- **Información del desafío:**
    - **challengeId:** identificador del desafío que queda asociado a la petición. Deberá ser utilizado para indicar la solución del mismo
    - **attemptsLeft:** cantidad de intentos que quedan para resolver el desafío
    - **type:** tipo de desafío; indica de dónde el usuario debe obtener la información de la solución del desafío. Ver más detalle en el apartado de *Tipos de Desafíos*
    - **responseMode:** modo en que debe ser presentada la solución del desafío; indica a la aplicación cómo debe proceder para informar la solución del desafío con el objeto de poder completar el procesamiento de la petición de forma satisfactoria. Ver más detalle en el apartado de *Modos de Respuesta*
    - **expiresIn:** cantidad de segundos disponibles para resolver el desafío
    - **solutionHint:** este campo es opcional y dependerá del type, pero su objetivo es que si el tipo de desafío requiere algo específico, se lo indique a través del mismo

Tipos de Desafíos

Existen distintos tipos de desafíos, que tienen que ver con configuraciones y preferencias de los usuarios. Cada tipo indica una manera distinta de obtener el valor de solución y se detallan a continuación.

OTP_SMS

En este caso la solución al desafío corresponde con un código que se envía al teléfono móvil del usuario a través de un mensaje de texto. Ese código es el que se debe informar como valor de respuesta al desafío.

Este tipo de desafío indicará en el SolutionHint una cadena con el número de teléfono móvil (enmascarado) al que se envio el mensaje SMS.

Este tipo de desafío acepta la acción de reenvío (Resend).

OTP_Mobile

En este caso la solución al desafío corresponde con un código que se envía al teléfono móvil del usuario a través de una notificación de la aplicación de Banca Móvil. Ese código es el que se debe informar como valor de respuesta al desafío.

Este tipo de desafío indicará en el SolutionHint una cadena que puede ser ingresada de forma manual en la aplicación de Banca Móvil para generar la respuesta correcta al desafío lanzado.

Este tipo de desafío acepta la acción de reenvío (Resend).

OTP_Device

En este caso la solución al desafío corresponde con un valor que se genera en un token físico que se encuentra en posesión del usuario. El usuario simplemente debe obtener un valor del dispositivo e informarlo como valor de respuesta al desafío.

Este tipo de desafío no utiliza SolutionHint.

Modos de Respuesta

De momento sólo se soporta un modo, que se detalla a continuación.

Header

En el caso del modo de respuesta Header, indica que la información de solución del desafío debe ser incluida a través de cabeceras HTTP que deben ser agregadas a la petición original que resultó en el desafío. Las cabeceras a informar en este caso son:

- **x-challenge-id:** identificador del desafío recibido. Es el mismo al obtenido de la respuesta (errors[].details.challengeId)
- **x-challenge-response:** es el valor de la solución al desafío recibido, deberá ser ingresado por el usuario a través de la aplicación
- **x-challenge-action:** identificador de la acción a realizar sobre el desafío. Requiere que se informe x-challenge-id y no se informe x-challenge-response

Acciones sobre desafíos

De acuerdo al escenario se pueden habilitar acciones especiales sobre un desafío que ya fue lanzado. Actualmente la única acción disponible, además de la resolución de los desafíos, es el reenvío de la información de solución asociada.

Resend

Permite que los casos donde la información de solución de un desafío es enviada al cliente por un canal distinto a la misma API, dicha información vuelva a ser enviada sin detrimento de la cantidad de intentos disponibles para la resolución del desafío.

Consideraciones prácticas

- Cualquier cambio en los parámetros de la petición será interpretado como si se tratara de una petición nueva e independiente
- Del punto anterior se infiere que si hacemos la petición con valores a, b y recibimos el desafío con identificador x; si luego intentamos una petición con los valores a, c indicando la resolución del desafío x, recibiremos un nuevo desafío: y
- Una vez expirado el desafío, intentar resolverlo resultará en un nuevo desafío
- No se puede resolver dos veces el mismo desafío
- Reintentar una petición que dio origen a un desafío sin proveer la solución del mismo no se considera un intento inválido, pero volverá a presentar el mismo desafío (aunque con menos segundos de validez correspondientes al paso del tiempo entre ambos momentos)

Acceso a los servicios de //Abanca

URLs de Base

Los distintos servicios provistos se exponen con una URL base que es propia para cada entorno e instancia como se muestra en la siguiente tabla.

Entorno	Autenticación	API V2	API V1
Sandbox	https://api.abanca.com/oauth/{id cliente}/Sandbox	https://api.abanca.com/V2/sandbox	https://api.abanca.com/sandbox
Explotación	https://api.abanca.com/oauth/{id cliente}/Abanca	https://api.abanca.com/V2/psd2	https://api.abanca.com/psd2

Autenticación de la aplicación

Para poder acceder a los servicios ofrecidos, **todas** las peticiones que se envíen deben identificar al consumidor de **APIs** a través de una cabecera específica para tal fin. La cabecera a incluir es AuthKey y su valor será el que se asocia a la aplicación en el portal de desarrolladores, como ya se pudo ver en los ejemplos presentados.

Identificación del usuario

Las peticiones a las **APIs** que requieran de tener la identidad del usuario, deberán incluir la cabecera Authorization con esquema Bearer y utilizar el access token OAuth obtenido.

Acceso a APIs

Para acceder a las APIs simplemente será necesario concatenar la ruta correspondiente a los recursos a la URL Base que corresponda al entorno e instancia con el que se debe trabajar. Cada API en su documentación indica las rutas que expone, que en su mayoría comenzarán con el prefijo /me.

Acceso al Servicio de Autenticación

En el caso de los servicios de autenticación del usuario de aplicaciones externas, simplemente es necesario utilizar la ruta adecuada dentro de la URL base que corresponda al entorno e instancia de interés:

- **/oauth/{id cliente}/Abanca:** ruta a donde redirigir a los usuarios para iniciar el flujo de autenticación OAuth (recordar incluir los parámetros indicados en la sección de **Autenticación**)
- **/oauth2/token:** ruta de donde se debe obtener los **access token** que serán necesarios para identificar a los usuarios en las peticiona las APIs

Acceso a la información de persona jurídica

Para poder acceder a los servicios de información de persona jurídica de **//ABANCA** se debe agregar la cabecera propietaria personalizada **x-clienteContratoId** la cual contendrá un identificador. Dicho identificador se obtendrá del attributo "id" proporcionado en la respuesta del recurso **/psd2/me/contracts.**.

Errores en la obtención

En caso de que la cabecera de la petición no informase la cabecera propietaria, el servicio devolverá una respuesta como la que se muestra a continuación:

```
HTTP/1.1 403 Forbidden


Content-Type: application/json;charset=UTF-8


{"errors": [{"id": "6e39ae13-e73b-4b7f-926e-545edd251f11","code":
"ERR_1011","title": "Operación no autorizada","technicalDescription": "Revise
los parámetros de entrada","detail": "Revise los parámetros de entrada"}]}
```

Usuarios sandbox

Identificador	Pin
1	12345
2	abcde
3	67890
4	fghij