

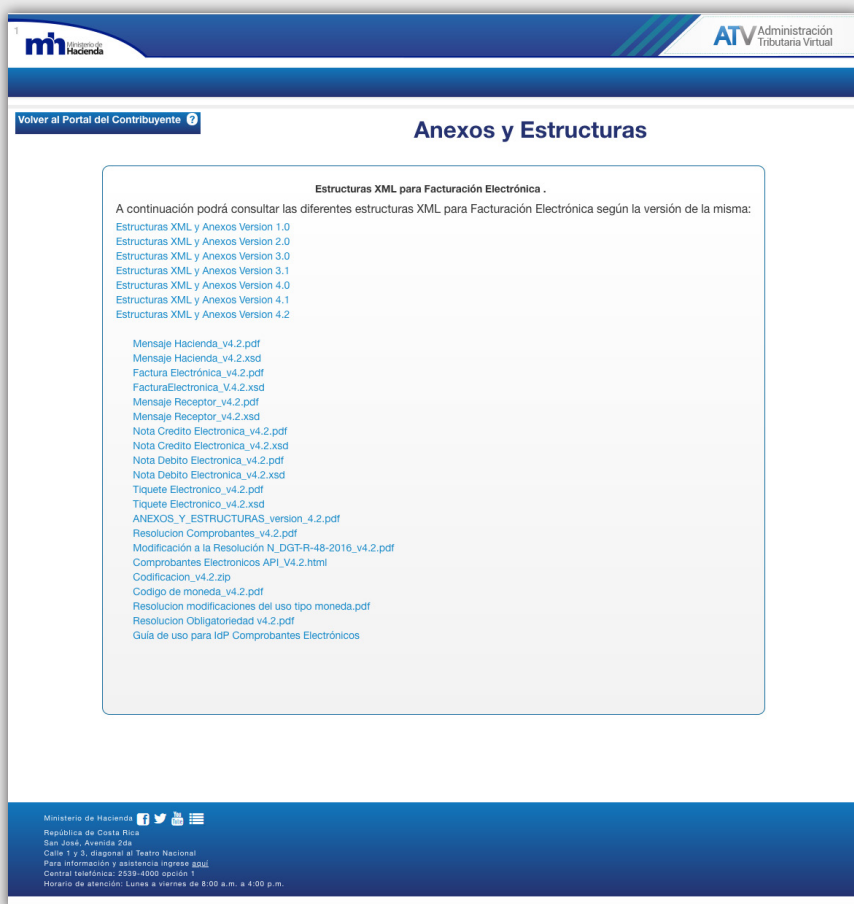
Guía de uso para IdP Comprobantes Electrónicos

Este documento intenta apoyar al lector en el proceso de interactuar con el Identity Provider (IdP) de la plataforma de Recepción de Comprobantes Electrónicos del Ministerio de Hacienda. Le guiará por un paso a paso con información como ¿dónde se encuentra el IdP?, ¿cómo consumirlo?, ¿cuáles estándares se utilizan? Le explicará con imágenes y ejemplos que le permitirán tener una mayor claridad del funcionamiento.

Se le recomienda al lector leer primero la documentación de la plataforma antes de continuar con esta guía. La documentación oficial se encuentra en el URL:

<https://tribunet.hacienda.go.cr/FormatosYEstructurasXML.jsp>

La última versión al momento de escribir este documento es la 4.2, dar principal atención al Anexo #3 del documento de Anexos.



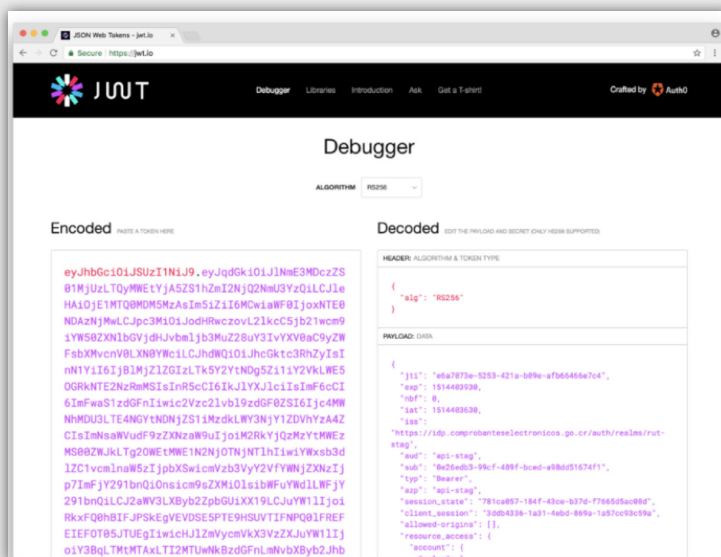
¿Qué es el IdP de Comprobantes Electrónicos?

Un IdP en inglés significa Identity Provider, este se encarga de los procesos de autenticación en la plataforma. Hay varios tipos de IdP y para varias funciones, por ejemplo el IdP se podría utilizar para proteger un sitio web, en este caso, al tratar de acceder al sitio web este redirigiría al IdP para que le solicite los credenciales y luego el IdP lo envía de regreso al sitio web ya con una sesión creada.

El funcionamiento con el servicio REST API de Recepción de Comprobantes Electrónicos sería similar, para poder consumir el servicio se necesita previamente tener una sesión en el IdP. Esta sesión no sería con una pantalla de login, sino que es para interacción con APIs.

La plataforma trabaja utilizando un modelo de seguridad con OpenID Connect (OIDC), este es una capa de identidad arriba del estándar RFC6749 The OAuth 2.0 Authorization Framework. El OIDC utiliza el estándar RFC7519 JSON Web Token (JWT), para almacenar la información de la sesión dentro del token del OAuth 2.0. El Grant Type utilizado por el IdP para autenticar usuarios es el Resource Owner Password Credential.

Ahora sí, ¿qué quiere decir el párrafo anterior? Básicamente lo que nos interesa entender es que el modelo de seguridad utilizado en la plataforma es una extensión del OAuth 2.0, y que para la mayoría de los casos se comporta como un OAuth 2.0. El hecho de que sea OpenID Connect agrega funcionalidad interesante pero lo único que nos importa de este es usar la función de logout y utilizar los JWT dentro de los token del OAuth 2.0.



OAuth 2.0

2-12

El OAuth 2.0 tiene varios tipos de Grants y funcionalidades, como lo indica la documentación del Anexo #3, necesitamos utilizar el Resource Owner Password Credential Grant del Auth 2.0. En este es que debemos concentrarnos, sin embargo, si les recomiendo por cultura general leer el documento del estándar para que tengan una idea más completa, y porque toda la documentación no va a estar solamente en la sección de este grant, hay flujos y términos que se encuentran en otras secciones y se van complementando conforme se lee el estándar.

Resource Owner Password Credential Grant

El OAuth 2.0 tiene varios tipos de Grants, el que se debe utilizar es el Resource Owner Password Credential Grant.

La documentación del OAuth 2.0 indica que para el Resource Owner Password Credential Grant se deben utilizar unas credenciales, estos son las credenciales que se generan desde el sistema ATV del Ministerio de Hacienda, serían usuario y contraseña. Hay dos ambientes, producción y sandbox, las credenciales son diferentes para cada ambiente y se deben generar para cada uno.

Utilizando la documentación del OAuth 2.0 y el Anexo #3 de la documentación de la plataforma, se obtiene que para obtener un token se debe enviar un payload de tipo application/x-www-form-urlencoded que contenga los campos:

- grant_type
- client_id
- username
- password

Este debe responder con un payload en application/json;charset=UTF-8 donde se indicará al menos los atributos:

- access_token
- refresh_token

Estos dos token son realmente un JSON Web Token, esto es parte de lo que se extiende al utilizar el OpenID Connect.

En una sección más adelante ya se mostrará con ejemplos reales como quedaría este payload y la respuesta que se obtiene.

JSON Web Token

JSON Web Token es un estándar abierto (RFC 7519) que define una forma segura de transmitir información como un JSON Object. La información puede ser validada y confiable porque está firmada digitalmente, el IdP se encarga de firmarlos. Una ventaja muy importante de utilizar JWT es que este contiene toda la información requerida del usuario, evitando que sea necesario consultar un repositorio de datos cada vez.

En este caso para la plataforma, el JWT se utiliza principalmente de forma interna, pero es importante entender cómo funciona en caso de querer obtener alguna información del JWT como el tiempo de expiración del token, que igualmente se puede obtener al momento de generarlo el token en la respuesta del OAuth 2.0.

eyJhbGciOiJSUzI1NiJ9.eyJqdGkiOiJlNmE3MDczS01MjUzLTQyMWEtYjA5ZS1hZm12NjQ2NmU3YzQlClJleHAiOiJpMTQ0MDM5MzAsIm5iZil6MCwiaWF0IjoxNTE0NDANjMwLClJpc3MiOiJodHRwczovL2lkC5jb21wcm9iYW50ZXNlbiGvJhdHJvbmli3MuZ28uY3lvYXV0aC9yZWZsbXMvbnV0LXN0YWciLCJhdWQiOiJhcGktc3RhZyIsInN1Yil6IjBIMjZlZGlzLk5k5Y2YtNDg5Zi1iY2VKLWE5OGRkNTE2NzRmMSIsInR5cCI6IkllYXJlcilsmF6cCl6ImFwaS1zdGFnIiwic2Vzc2l2b3l9zdGF0ZS16Ijc4MWNhMDU3LTE4NGYtNDNjZS1imzdkLWY3NjY1ZDVhYzA4ZC1mNmNsaWVudF9zZXNzaW9uIjoiM2RkYzQzMzYtMWZlZW500ZWNjLkTg2OWEtMWE1N2NjOTNlNTIhIiwuYXs3d-1ZC1vcmlnaW5Zl3pbiGwXSwicmVzb3VyY2VfYWNjZW50XzNlZj9ImF9Y291bnQ1Oncicm9sZXMiOiNjbnFwYUdlWF-jY291bnQlLCJ2aWV3LXB3b2ZpbiGUXX19LCJuYXW1IljoiriXkFQ0hBIFJPSEgVEVDSE5PTE9HSUVTIFNPQ0IFREFEIEFOT05JTUEgliwicHJlZmVycmVkX3VzZXJuYXW1IljoiriY3BqLTMtMTAxLTl2MTUwNkBzdGFnLmNvbXB3b2JhbnRlc2VsZWN0cm9uaWNvcy5nby5jciIsImdpdmVuX25hbWUiOiJGTEVDSEEGUk9KQSBURUNITk9MT0d-JRVMgU09DSUVEQUQgQU5PTklnQSIsInBvbGljeS1pZC16IjU4YTtyYMDMzNzZlYWUxNDA4Y2U1ZTdkZCIsIm-VtYWlsIjoiY3BqLTMtMTAxLTl2MTUwNkBzdGFnLmNvbXB3b2JhbnRlc2VsZWN0cm9uaWNvcy5nby5jciJ9.GTZ7WY5XbCsBAU-E1OsFglkA38xwmZ82AkAAS4RrMM8VKWJxZUG4HvMvNm44MMzBmcBjBlJ6U94lp-PRYMGyToARK3V43SKdx_QLqPEWXA0wkrkNRRHtImTlIX6pvY_e7v6b16UOrPhBPfYVlNi4tW2lsR-o4wbMTEB4d7gJ_Gengw5li_y-Y78e3L2fFGXEQJN-6UFMllmY8oNdTzbtq6p-bKhfoqJ5OAEwbqK5cOXnHUK91POz5axWHyvxGI879wS-FaN98CXG_6XdbpiYLNAd_NsDH7Aza1Z40F_Zx1nmlf9Na_1KAKhRW_NbRbwoTKerX1Z3Y3YuKhHfPfvjMAA

Para ver la información del JWT se puede utilizar el decoder del sitio web <https://jwt.io/>.

Interacción con el IdP

A continuación, se muestra la información necesaria para interactuar con los dos ambientes (Realms) que expone el IdP de la plataforma, estos dos ambientes son:

- Producción
- Sandbox / Staging

Datos	Producción	Sandbox
TOKEN_URL	https://idp.comprobanteselectronicos.go.cr/auth/realms/rut/protocol/openid-connect/token	https://idp.comprobanteselectronicos.go.cr/auth/realms/rut-stag/protocol/openid-connect/token
LOGOUT_URL	https://idp.comprobanteselectronicos.go.cr/auth/realms/rut/protocol/openid-connect/logout	https://idp.comprobanteselectronicos.go.cr/auth/realms/rut-stag/protocol/openid-connect/logout
client_id	api-prod	api-stag
username	Obtener de ATV	Obtener de ATV
password	Obtener de ATV	Obtener de ATV

TIP: Los body o payload a enviar para interactuar con el IdP son de tipo x-www-form-urlencoded, y el IdP normalmente va a responder con application/json.

En la siguiente sección se explicará cómo utilizar el token que se obtuvo, esta sección se enfoca principalmente en la interacción con el IdP para mayor claridad. Para interactuar con el IdP se van a utilizar principalmente 3 procesos:

- Obtener un token
- Refrescar un token
- Cerrar sesión

Los pasos a continuación se van a explicar utilizando el IdP con el Realm de Sandbox, para que funcione para Producción solo sería necesario cambiar el TOKEN_URL y el client_id, y lógicamente usar las credenciales adecuadas. El password en los ejemplos será sustituido al terminar este documento, por favor no ejecutar los comandos ya que les darán error, estos son principalmente con fines ilustrativos y para que el lector los pueda modificar para sus necesidades.

TIP: Los password que da el **ATV** son cadenas de caracteres de aproximadamente un largo de 20, y pueden tener caracteres especiales. El body o payload que se envía al IdP es de tipo x-www-form-urlencoded, esto quiere decir que se debe revisar que vayan bien codificados los form fields, los caracteres especiales del password si se envían en PLAIN podrían dar errores indicando que los credenciales son inválidos.

Para cada proceso se mostrará la petición HTTP con cURL, HTTPie y el PLAIN HTTP, poner atención a como se ve el password en el PLAIN HTTP para tener claro el tema del x-www-form-urlencoded. Y se mostrará el output recibido únicamente del HTTPie para que sea más visual.

¿Cómo obtener un token?

Para obtener un token se debe enviar una petición HTTP POST al TOKEN_URL con los siguientes form fields:

Form Field	Data
grant_type	password
client_id	api-stag
username	Obtenido de ATV
password	Obtenido de ATV

5-12

Algunas librerías o plataformas podrían solicitar el `scope` o el `client_secret`, en este caso se dejan en blanco, no son necesarias.

cURL, obtener token:

```
curl -X "POST" "https://lap.comprobanteselectronicos.go.cr/auth/realms/rut-stag/protocol/openid-connect/token" \
-H 'Content-Type: application/x-www-form-urlencoded; charset=utf-8' \
--data-urlencode "client_id=api-stag" \
--data-urlencode "username=cpj-3-101-261506@stag.comprobanteselectronicos.go.cr" \
--data-urlencode "password=W$/JX/AS@K);1]l;u|+6" \
--data-urlencode "grant_type=password"
```

HTTPIe, obtener token:

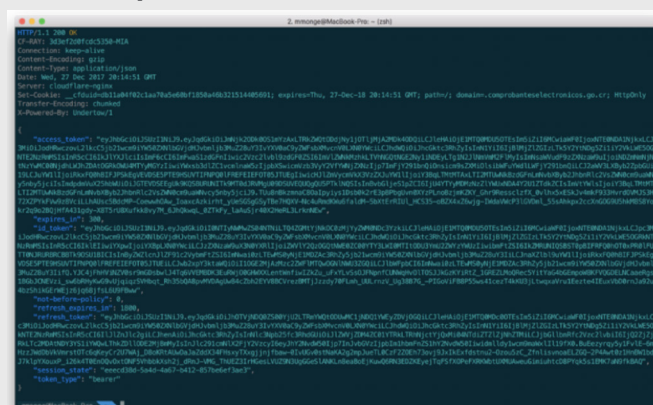
```
http --form POST 'https://idp.comprobanteselectronicos.go.cr/auth/realms/rut-stag/protocol/openid-connect/token' \
  'Content-Type': 'application/x-www-form-urlencoded; charset=utf-8' \
  'client_id'='api-stag' \
  'username'='cpj-3-101-261506@stag.comprobanteselectronicos.go.cr' \
  'password'='W$/JX/AS@K);1]l;u|+6' \
  'grant_type'='password'
```

PLAIN HTTP, obtener token:

```
POST /auth/realms/rut-stag/protocol/openid-connect/token HTTP/1.1
Content-Type: application/x-www-form-urlencoded; charset=utf-8
Host: idp.comprobanteselectronicos.go.cr
Connection: close
User-Agent: Paw/3.1.5 (Macintosh; OS X/10.13.2) GCDHTTPRequest
Content-Length: 152
```

grant_type=password&client_id=api-stag&username=cpj-3-101-261506%40stag.comprobanteselectronicos.go.cr&password=W%24%2FJX%2FAS%40K%29%3B1%5DI%3Bu%7C%2B6

Y se obtiene la siguiente respuesta:



En la respuesta obtenida nos interesa principalmente los siguientes atributos:

Form Field	Data
token_type	Indica el tipo de token que se obtuvo en la petición, normalmente va a ser bearer , este es el que se necesita para trabajar con la plataforma.
access_token	Este es el JWT que se va a utilizar para interactuar con la plataforma, cada vez que se haga una petición al <i>API de Comprobantes Electrónicos</i> se debe enviar este.
expires_in	Tiempo de expiración en segundos del access_token , cuando el access_token expira se debe hacer un proceso de Refresh Token para obtener.
refresh_token	Este es el token utilizado para el proceso de Refresh , este token tiene un tiempo de expiración mucho mayor que el access_token .
refresh_expires_in	Tiempo de expiración en segundos del refresh_token , cuando el refresh_token expira se acabó la sesión y es necesario crear una nueva sesión.

¿Cómo refrescar un token?

Para refrescar un token se debe enviar una petición HTTP POST al TOKEN_URL con los siguientes form fields:

Form Field	Data
grant_type	refresh_token
client_id	api-stag
refresh_token	Data del refresh_token obtenido en el proceso anterior.

cURL, refrescar un token:

```
curl -X "POST" "https://idp.comprobanteselectronicos.go.cr/auth/realms/rut-stag/protocol/openid-connect/token" \
```

```
-H 'Content-Type: application/x-www-form-urlencoded; charset=utf-8' \
```

```
--data-urlencode "client_id=api-stag" \
```

```
--data-urlencode "refresh_token=eyJhbGciOiJIUzI1NiJ9.eyJqdGkiOiJpYWU2MGJiMS02OWYxLTI0NDQ3MTI0LCJpc3MiOiJodHRwczovL2lkC5jb21wcm9iYW50ZXNlbGVjdHJvbmliY3MuZ28uY3IvYXV0aC9yZWZfbXVvcnV0LXN0YWciLCJhdWQiOiJhcGt3RhZyIsbnN1Yil6lGJmZlZGlzLk5k5Y2YtN-Dg5Zi1Y2VkLWE5OGRkNTE2NzRmMSIsbnR5cCI6IjIzLnJlc2giLCJhenAiOiJhcGt3RhZyIsbnNlc3Np-b25fc3RhZGUiOiI0ZGUzZjY5Zi00OTFlTQ1NWETmUxYi1mDYyZTQwNmM5NjliLCJjbGllbnRfc2Vz-c2lvbii6ImM5NDU5YzlmLTU5OTAuNGY3OC05ZjMyLTUzOTU0OWM2ZmY0OCIsbnJlc291cmNIX2F-jY2Vzcy6eyJhY2NvdW50ljp7InJvbGVzljpbIm1hbmFnZS1hY2NvdW50liwidmllYy1wcm9maWxll19fX0.J8ENU7MMNlMIZTon-wbcGH0ClylfoPIJlJyt6wcvkbbKgdgtPtYQw3vyEBw123Jos5w0vQKtrJ7CpuUk8H-vZbuNUifbf4t8MKystkXdYgT6YnT3uBkL9zPsOVBeBLWRuKM3MFhspssCZ7nafXTxRAMQ0rxs81jZYHcY-dzyVu9UIW5VW8bu-YyBPVKpEx6-D5s5cp4132Dp7ZMzPaeNvkRdhQr9UL9KdzOEo5RqZZ85TZZrhQYR-XheUPH_JOWuJP3WntNmhReXWzBX6RH4zCnse1yzYnb4DR2LbfikBjPIPU6EH3BzpKBjZdXqGzwQUAhS-MJWS_m1rTw0eDDp3Jdi6Q" \
```

```
--data-urlencode "grant_type=refresh_token"
```

HTTPIe, refrescar un token:

```
http --form POST 'https://idp.comprobanteselectronicos.go.cr/auth/realms/rut-stag/protocol/openid-connect/token' \
```

```
'Content-Type': 'application/x-www-form-urlencoded; charset=utf-8' \
```

```
'client_id'='api-stag' \
```

7-12


```
'refresh_token'='eyJhbGciOiJSUzI1NiJ9.eyJqdGkiOiJyYWU2MGJiMS02OWYxLTQ3MmYtYTE0NC03Y2EzOWI5MjY3NzliLCJleHAiOiE1MTQ0MDg5MjQsIm5iZil6MCwiaWF0IjoxNTE0NDA3MTI0LCJpc3MiOiJodHRwczovL2lkccC5jb21wcm9iYW50ZXNlbGVjdHJvbmljb3MuZ28uY3lvYXV0aC9yZWZfbXMvcnV0LXN0YWciLCJhdWQiOiJhcGktc3RhZyIsInN1Yil6IjBIMjZlZGlzLTk5Y2YtNDg5Zi1iY2VkLWE5OGRkNTE2NzRmMSIsInR5cCI6IiJlZnJlc2giLCJhenAiOiJhcGktc3RhZyIsInNlc3Npb25fc3RhZGUuOiI0ZGUzZjY5Zi00OTFILTQ1NWEtYmUxYi1iMDYyZTQwNmM5NjliLCJjbGllbnRfc2Vzc2lvaW50LmM5NDU5YzlmLTU5OTAfNGY3OC05ZjMyLTUzOTU0OWM2ZmY0OCIsInJlc291cmNIX2F-jY2Vzcyl6eyJhY2NvdW50Ijpb7InJvbGVzIjpbIm1hbmFnZS1hY2NvdW50Iiwidmllldy1wcm9maWxlll19fX0.J8ENu7MMNIMIZTon-wbcGH0ClylfoPIJLjyt6wcvkbbKgdgtPtYQw3vyEBwl23Jos5w0vQKtrJ7CpuUk8H-vZbuNUifbf4t8MKystkXdYgT6YnT3uBkL9zPsOVBeBLWRuKM3MFhspssCZ7nafXTxRAMQ0rxs81jZYhCy-dzyVu9UIW5VW8bu-YyBPVKpEx6-D5s5cp4132Dp7ZMzPaeNvkRdhQr9UL9KdzOEo5RqZZ85TZrhQYR-XheUPH_JOwuJP3WntNmhReXWzBX6RH4zCnse1yzYnb4DR2LbfikBjPIPU6EH3BzpKBjZdXqGzwQUAhS-MJWS_m1rTw0eDDp3Jdi6Q' \
```

```
'grant_type'='refresh_token'
```

PLAIN HTTP, refrescar un token:

```
POST /auth/realms/rut-stag/protocol/openid-connect/token HTTP/1.1
Content-Type: application/x-www-form-urlencoded; charset=utf-8
Host: idp.comprobanteselectronicos.go.cr
Connection: close
User-Agent: Paw/3.1.5 (Macintosh; OS X/10.13.2) GCDHTTPRequest
Content-Length: 1009
```

```
grant_type=refresh_token&client_id=api-stag&refresh_token=eyJhbGciOiJSUzI1NiJ9.eyJqdGkiOiJyYWU2MGJiMS02OWYxLTQ3MmYtYTE0NC03Y2EzOWI5MjY3NzliLCJleHAiOiE1MTQ0MDg5MjQsIm5iZil6MCwiaWF0IjoxNTE0NDA3MTI0LCJpc3MiOiJodHRwczovL2lkccC5jb21wcm9iYW50ZXNlbGVjdHJvbmljb3MuZ28uY3lvYXV0aC9yZWZfbXMvcnV0LXN0YWciLCJhdWQiOiJhcGktc3RhZyIsInN1Yil6IjBIMjZlZGlzLTk5Y2YtNDg5Zi1iY2VkLWE5OGRkNTE2NzRmMSIsInR5cCI6IiJlZnJlc2giLCJhenAiOiJhcGktc3RhZyIsInNlc3Npb25fc3RhZGUuOiI0ZGUzZjY5Zi00OTFILTQ1NWEtYmUxYi1iMDYyZTQwNmM5NjliLCJjbGllbnRfc2Vzc2lvaW50LmM5NDU5YzlmLTU5OTAfNGY3OC05ZjMyLTUzOTU0OWM2ZmY0OCIsInJlc291cmNIX2F-jY2Vzcyl6eyJhY2NvdW50Ijpb7InJvbGVzIjpbIm1hbmFnZS1hY2NvdW50Iiwidmllldy1wcm9maWxlll19fX0.J8ENu7MMNIMIZTon-wbcGH0ClylfoPIJLjyt6wcvkbbKgdgtPtYQw3vyEBwl23Jos5w0vQKtrJ7CpuUk8H-vZbuNUifbf4t8MKystkXdYgT6YnT3uBkL9zPsOVBeBLWRuKM3MFhspssCZ7nafXTxRAMQ0rxs81jZYhCy-dzyVu9UIW5VW8bu-YyBPVKpEx6-D5s5cp4132Dp7ZMzPaeNvkRdhQr9UL9KdzOEo5RqZZ85TZrhQYR-XheUPH_JOwuJP3WntNmhReXWzBX6RH4zCnse1yzYnb4DR2LbfikBjPIPU6EH3BzpKBjZdXqGzwQUAhS-MJWS_m1rTw0eDDp3Jdi6Q
```

Y se obtiene la siguiente respuesta:

cURL, cerrar sesión:

```
curl -X "POST" "https://idp.comprobanteselectronicos.go.cr/auth/realms/rut-stag/protocol/openid-connect/logout" \
  -H 'Content-Type: application/x-www-form-urlencoded; charset=utf-8' \
  --data-urlencode "client_id=api-stag" \
  --data-urlencode "refresh_token=eyJhbGciOiJSUzI1NiJ9.eyJqdGkiOiJpYXU2MGJiMS02OWYxLTQ3MmYtYTE0NC03Y2EzOWI5MjY3NzliLCJleHAiOiE1MTQ0MDg5MjQsIm5iZiI6MCwiaWF0IjoxNTE0NDA3MTI0LCJpc3MiOiJodHRwczovL2lkccC5jb21wcm9iYW50ZXNlbGVjdHJvbmli3MuZ28uY3IvYXV0aC9yZWZfbXMvcnV0LXN0YWciLCJhdWQiOiJhcGktc3RhZyIsInN1Yil6IjBIMjZlZGlzLTk5Y2YtNDg5Zi1iY2VklWE5OGRkNTE2NzRmMSIsInR5cCI6IjIzNjIjZnJlc2giLCJhenAiOiJhcGktc3RhZyIsInNlc3Npb25fc3RhZGUiOiI0ZGUzZjY5Zi00OTFILTQ1NWEtYmUxYi1iMDYyZTQwNmM5NjliLCJjbGllbnRfc2Vz-c2l2b2l6ImM5NDU5YzlmLTU5OTAfNGY3OC05ZjMyLTU5OTU0OWM2ZmY0OCIsInJlc291cmNIX2F-jY2Vzcyl6eyJhY2NvdW50Ijp7InJvbGVzIjpbIm1hbmFnZS1hY2NvdW50Iiwidmllldy1wcm9maWxlll19fX0.J8ENu7MMNIMIZTon-wbcGH0ClYlfoPIJlJyt6wcvkbbKgdgtPtYQw3vyEBwL23Jos5w0vQKtrJ7CpuUk8H-vZbuNUifbf4t8MKystkXdyGt6YnT3uBkL9zPsOVBeBLWRuKM3MFhspssCZ7nafXTxRAMQ0rxs81jZYhCy-dzyVu9UIW5VW8bu-YyBPVKpEx6-D5s5cp4132Dp7ZMzPaeNvkRdhQr9UL9KdzOEo5RqZZ85TZZrhQYR-XheUPH_JOwuJP3WntNmhReXWzBX6RH4zCnse1yzYnb4DR2LbfikBjPIPU6EH3BzpKBjZdXqGzwQUAhS-MJWS_m1rTw0eDDp3Jdi6Q"
```

HTTPIe, cerrar sesión:

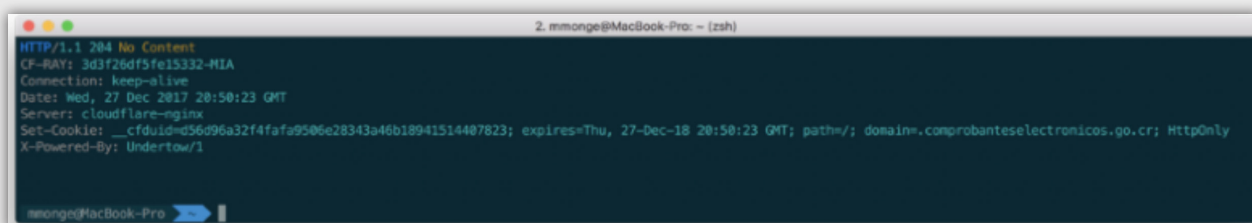
```
http --form POST 'https://idp.comprobanteselectronicos.go.cr/auth/realms/rut-stag/protocol/openid-connect/logout' \
  'Content-Type': 'application/x-www-form-urlencoded; charset=utf-8' \
  'client_id'='api-stag' \
  'refresh_token'='eyJhbGciOiJSUzI1NiJ9.eyJqdGkiOiJpYXU2MGJiMS02OWYxLTQ3MmYtYTE0NC03Y2EzOWI5MjY3NzliLCJleHAiOiE1MTQ0MDg5MjQsIm5iZiI6MCwiaWF0IjoxNTE0NDA3MTI0LCJpc3MiOiJodHRwczovL2lkccC5jb21wcm9iYW50ZXNlbGVjdHJvbmli3MuZ28uY3IvYXV0aC9yZWZfbXMvcnV0LXN0YWciLCJhdWQiOiJhcGktc3RhZyIsInN1Yil6IjBIMjZlZGlzLTk5Y2YtNDg5Zi1iY2VklWE5OGRkNTE2NzRmMSIsInR5cCI6IjIzNjIjZnJlc2giLCJhenAiOiJhcGktc3RhZyIsInNlc3Npb25fc3RhZGUiOiI0ZGUzZjY5Zi00OTFILTQ1NWEtYmUxYi1iMDYyZTQwNmM5NjliLCJjbGllbnRfc2Vz-c2l2b2l6ImM5NDU5YzlmLTU5OTAfNGY3OC05ZjMyLTU5OTU0OWM2ZmY0OCIsInJlc291cmNIX2F-jY2Vzcyl6eyJhY2NvdW50Ijp7InJvbGVzIjpbIm1hbmFnZS1hY2NvdW50Iiwidmllldy1wcm9maWxlll19fX0.J8ENu7MMNIMIZTon-wbcGH0ClYlfoPIJlJyt6wcvkbbKgdgtPtYQw3vyEBwL23Jos5w0vQKtrJ7CpuUk8H-vZbuNUifbf4t8MKystkXdyGt6YnT3uBkL9zPsOVBeBLWRuKM3MFhspssCZ7nafXTxRAMQ0rxs81jZYhCy-dzyVu9UIW5VW8bu-YyBPVKpEx6-D5s5cp4132Dp7ZMzPaeNvkRdhQr9UL9KdzOEo5RqZZ85TZZrhQYR-XheUPH_JOwuJP3WntNmhReXWzBX6RH4zCnse1yzYnb4DR2LbfikBjPIPU6EH3BzpKBjZdXqGzwQUAhS-MJWS_m1rTw0eDDp3Jdi6Q'
```

PLAIN HTTP, cerrar sesión:

```
POST /auth/realms/rut-stag/protocol/openid-connect/logout HTTP/1.1
Content-Type: application/x-www-form-urlencoded; charset=utf-8
Host: idp.comprobanteselectronicos.go.cr
Connection: close
User-Agent: Paw/3.1.5 (Macintosh; OS X/10.13.2) GCDHTTPRequest
Content-Length: 984
```

client_id=api-stag&refresh_token=eyJhbGciOiJSUzI1NiJ9.eyJqdGkiOiJjYWU2MGJiMS02OWYxLT-Q3MmYtYTE0NC03Y2EzOWI5MjY3NzliLCJleHAiOiJlMTQ0MDg5MjQsIm5iZil6MCwiaWF0IjoxNTE0N-DA3MTI0LCJpc3MiOiJodHRwczovL2lkC5jb2Iwcm9iYW50ZXNlbGVjdHJvbmliY3MuZ28uY3lvYX-V0aC9yZWZfbXMvcnV0LXN0YWciLCJhdWQiOiJhcGktc3RhZyIsInN1Yil6IjBIMjZlZGZlLtk5Y2YtND-g5Zi1iY2VkLWE5OGRkNTE2NzRmMSIsInR5cCI6IjZnJlc2giLCJhenAiOiJhcGktc3RhZyIsInNlc3Np-b25fc3RhdGUiOiI0ZGUzZjY5Zi00OTFlLTQ1NWEtYmUxYi1iMDYyZTQwNmM5NjliLCJjbGllbnRfc2Vz-c2lvdil6ImM5NDU5YzlmLTU5OTAfNGY3OC05ZjMyLTUzOTU0OWM2ZmY0OCIsInJlc291cmNIX2F-jY2Vzcyl6eyJhY2NvdW50Ijpb7InJvbGVzIjpbIm1hbmFnZS1hY2NvdW50Iiwidmllldy1wcm9maWxlll19fX0.J8ENu7MMNIMIZTon-wbcGH0ClylfoPIJLjyt6wcvkbbKgdgtPtYQw3vyEBw123Jos5w0vQKtrJ7CpuUk8H-vZbuNUifbf4t8MKystkXdYgT6YnT3uBkL9zPsOVBeBLWRuKM3MFhspssCZ7nafXTxRAMQ0rxs81jZYhCy-dzyVu9UIW5VW8bu-YyBPVKpEx6-D5s5cp4132Dp7ZMzPaeNvkRdhQr9UL9KdzOEo5RqZZ85TZrhQYR-XheUPH_JOwJP3WntNmhReXWzBX6RH4zCnse1yzYnb4DR2LbfikBjPIPU6EH3BzpKBjZdXqGzwQUAhS-MJWS_m1rTw0eDDp3Jdi6Q

Y se obtiene la siguiente respuesta:



```
2. mmongo@MacBook-Pro: ~ (zsh)
HTTP/1.1 204 No Content
CF-RAY: 3d3f26df5fe15332-MIA
Connection: keep-alive
Date: Wed, 27 Dec 2017 20:50:23 GMT
Server: cloudflare-nginx
Set-Cookie: __cfduid=d56d96a32f4fafa9506e28343a46b18941514487823; expires=Thu, 27-Dec-18 20:50:23 GMT; path=/; domain=.comprobanteselectronicos.go.cr; HttpOnly
X-Powered-By: Undertow/1
mmongo@MacBook-Pro
```

Importante notar de esta respuesta que se obtiene un HTTP 204 No Content, esta es la respuesta correcta al cerrar sesión.

Comunicación con la plataforma de Comprobantes Electrónicos

En las secciones anteriores se explicó un poco ¿qué es?, ¿cómo funciona? y ¿cómo interactuar? con el IdP, en esta sección se pretende explicar al lector como poner en práctica todo para comunicarse con la plataforma.

Ya luego de haber entendido las secciones anteriores, esta es muy simple. Para interactuar con el API de Recepción de Comprobantes Electrónicos es necesario enviar un header en cada request, este header es el Authorization header. El contenido del Authorization header se obtiene concatenando el bearer_type + [un espacio] + access_token. Por ejemplo:

Authorization: bearer

eyJhbGciOiJSUzI1NiJ9.eyJqdGkiOiJkNmVIYmE1OC1jNGNiLTQ4NmItYTZhMy04ZTk5M2ZjY2NIMj-gilCJleHAiOiJlMTQ0MDc0MjQsIm5iZil6MCwiaWF0IjoxNTE0NDA3MTI0LCJpc3MiOiJodHRwczov-L2lkC5jb2Iwcm9iYW50ZXNlbGVjdHJvbmliY3MuZ28uY3lvYXV0aC9yZWZfbXMvcnV0LXN0YWciLCJhdWQiOiJhcGktc3RhZyIsInN1Yil6IjBIMjZlZGZlLtk5Y2YtNDg5Zi1iY2VkLWE5OGRkNTE2NzRmMSIsInR5cCI6IjZnJlc2giLCJhenAiOiJhcGktc3RhZyIsInNlc3Np-b25fc3RhdGUiOiI0ZGUzZjY5Zi00OTFlLTQ1NWEtYmUxYi1iMDYyZTQwNmM5NjliLCJjbGllbnRfc2Vz-c2lvdil6ImM5NDU5YzlmLTU5OTAfNGY3OC05ZjMyLTUzOTU0OWM2ZmY0OCIsInJlc291cmNIX2F-jY2Vzcyl6eyJhY2NvdW50Ijpb7InJvbGVzIjpbIm1hbmFnZS1hY2NvdW50Iiwidmllldy1wcm9maWxlll19fX0.J8ENu7MMNIMIZTon-wbcGH0ClylfoPIJLjyt6wcvkbbKgdgtPtYQw3vyEBw123Jos5w0vQKtrJ7CpuUk8H-vZbuNUifbf4t8MKystkXdYgT6YnT3uBkL9zPsOVBeBLWRuKM3MFhspssCZ7nafXTxRAMQ0rxs81jZYhCy-dzyVu9UIW5VW8bu-YyBPVKpEx6-D5s5cp4132Dp7ZMzPaeNvkRdhQr9UL9KdzOEo5RqZZ85TZrhQYR-XheUPH_JOwJP3WntNmhReXWzBX6RH4zCnse1yzYnb4DR2LbfikBjPIPU6EH3BzpKBjZdXqGzwQUAhS-MJWS_m1rTw0eDDp3Jdi6Q

11-12

Lo importante acá que se debe tener en cuenta es siempre enviar un `access_token` que no haya expirado, es por esto que desde la aplicación cliente que va a interactuar con la plataforma se debe manejar el ciclo de vida de la sesión.

La sesión se crea la primera vez que se obtiene el `access_token`, tendrá un tiempo de vida definido por el `expires_in`, al momento de hacer este documento son 5 minutos. Esto quiere decir que se debe controlar este tiempo de vida del `access_token` para saber que antes de enviarlo si ya expiró es necesario refrescarlo.

La mejor práctica es si ya terminó de usar una sesión que la cierre utilizando el proceso de cerrar sesión de la sección anterior. Esto sería similar a cuando se entra al portal bancario en un navegador web, la práctica segura es cerrar la sesión al terminar para evitar alguna actividad maliciosa.