Guía de uso para IdP Comprobantes Electrónicos

Marvin Monge

Version 1.0, 2017-12-16

Índice

Introducción 1
¿Qué es el IdP de Comprobantes Electrónicos? 3
OAuth 2.0 3
Resource Owner Password Credential Grant 3
JSON Web Token 4
Interacción con el IdP 6
¿Cómo obtener un token? 7
¿Cómo refrescar un token? 9
¿Cómo cerrar sesión? (Eliminar el token) 11
Comunicación con la plataforma de Comprobantes Electrónicos 14
Ciclo de vida de la sesión 14

Introducción

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.

IMPORTANTE: Esta no es una guía oficial del *Ministerio de Hacienda* y de ninguna manera se relaciona directamente con ellos. Los puntos de vista y recomendaciones son nuestros únicamente, basados en la documentación y estándares indicados en la documentación de la plataforma de *Recepción de Comprobantes Electrónicos*.

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.

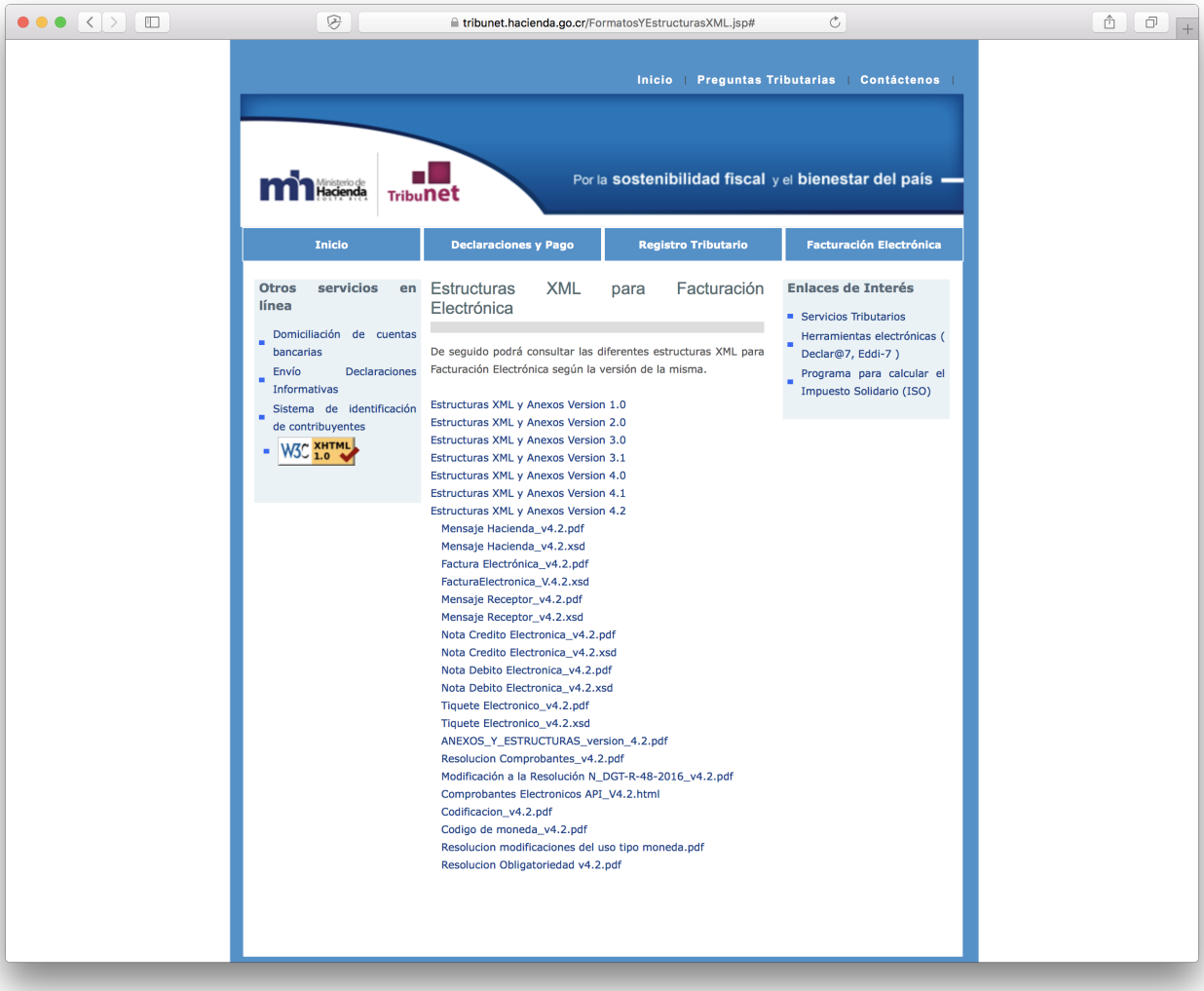


Figura 1. Documentación oficial MH

¿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 accesar el sitio web este redireccionaría 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)* (http://openid.net/connect/), este es una capa de identidad arriba del estándar *RFC6749 The OAuth 2.0 Authorization Framework* (https://tools.ietf.org/html/rfc6749). El *OIDC* utiliza el estándar *RFC7519 JSON Web Token (JWT)* (https://tools.ietf.org/html/rfc7519), 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 si, ¿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

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 unos credenciales, estos son los credenciales que se generan desde el sistema *ATV* del *Ministerio de Hacienda*, serían usuario y contraseña. Hay dos ambientes, producción y sandbox, los 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 esta 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 como 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*.

Ejemplo de un JWT

```
eyJhbGciOiJSUzI1NiJ9.eyJqdGkiOiJlNmE3MDczZS01MjUzLTQyMWEtYjA5ZS1hZmI2NjQ2NmU3YzQiLCJle
HAiOjE1MTQ0MDM5MzAsIm5iZiI6MCwiaWF0IjoxNTE0NDAzNjMwLCJpc3MiOiJodHRwczovL2lkcC5jb21wcm9
iYW50ZXNlbGVjdHJvbmljb3MuZ28uY3IvYXV0aC9yZWFsbXMvcnV0LXN0YWciLCJhdWQiOiJhcGktc3RhZyIsI
nN1YiI6IjBlMjZlZGIzLTk5Y2YtNDg5Zi1iY2VkLWE5OGRkNTE2NzRmMSIsInR5cCI6IkJlYXJlciIsImF6cCI
6ImFwaS1zdGFnIiwic2Vzc2lvbl9zdGF0ZSI6Ijc4MWNhMDU3LTE4NGYtNDNjZS1iMzdkLWY3NjY1ZDVhYzA4Z
CIsImNsaWVudF9zZXNzaW9uIjoiM2RkYjQzMzYtMWEzMS00ZWJkLTg2OWEtMWE1N2NjOTNjNTlhIiwiYWxsb3d
lZC1vcmlnaW5zIjpbXSwicmVzb3VyY2VfYWNjZXNzIjp7ImFjY291bnQiOnsicm9sZXMiOlsibWFuYWdlLWFjY
291bnQiLCJ2aWV3LXByb2ZpbGUiXX19LCJuYW1lIjoiRkxFQ0hBIFJPSkEgVEVDSE5PTE9HSUVTIFNPQ0lFREF
EIEFOT05JTUEgIiwicHJlZmVycmVkX3VzZXJuYW1lIjoiY3BqLTMtMTAxLTI2MTUwNkBzdGFnLmNvbXByb2Jhb
nRlc2VsZWN0cm9uaWNvcy5nby5jciIsImdpdmVuX25hbWUiOiJGTEVDSEEgUk9KQSBURUNITk9MT0dJRVMgU09
DSUVEQUQgQU5PTklNQSIsInBvbGljeS1pZCI6IjU4YTYyMDMzNzZlYWUxNDA4Y2U1ZTdkZCIsImVtYWlsIjoiY
3BqLTMtMTAxLTI2MTUwNkBzdGFnLmNvbXByb2JhbnRlc2VsZWN0cm9uaWNvcy5nby5jciJ9.GTZ7WXY5bCsBAU
-E1OsFglkA38XwmZ82AkAAS4RrMM8VKWJxZG4HvMVm44MMzBmciwBJI6Uu94lp
-PRYMGYtoARK3V43SKdx_QLqPEWXa0wkrkNRRhHtimTIX6pvY_e7v6b6UOrPhxBPFyNLiNitW2IsR
-o4bQMtEB4d7gJ_Gengw5Ii_y-Y78eBL2fFGXEQJN-6UFMllmY8oNdTZbq6p
-bKhfoqOJ5OAEwbqK5cOXnHUK91POz5axWHyvxGl879wSFaN98CXG_6XdbpiYLNAd_NsDH7Aza1Z40F_Zx1nim
If9Na_1KAKhRW_NbRbwoTKeRx1Z3Y3YuKhHFpfvjMAA
```

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

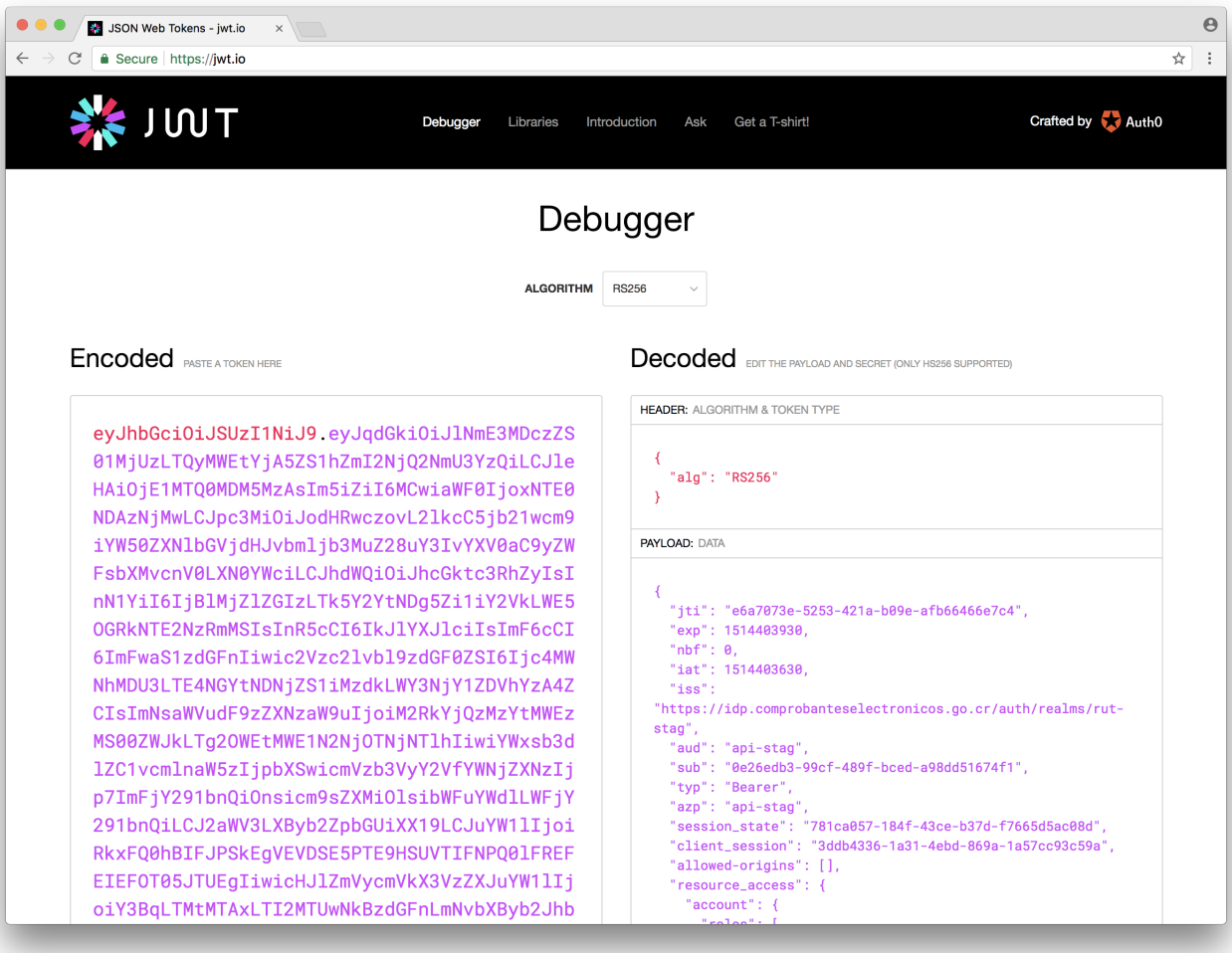


Figura 2. JWT Decoder

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-prod/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-prod/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*

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 como 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 los credenciales adecuados. 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.

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`	Obtenenido de *ATV*
`password`	Obtenenido de *ATV*

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://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 "username=cpj-3-101-261506@stag.comprobanteselectronicos.go.cr"
\
     --data-urlencode "password=W$/JX/AS@K);1]I;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]I;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%3B
u%7C%2B6
```

Y se obtiene la siguiente respuesta:

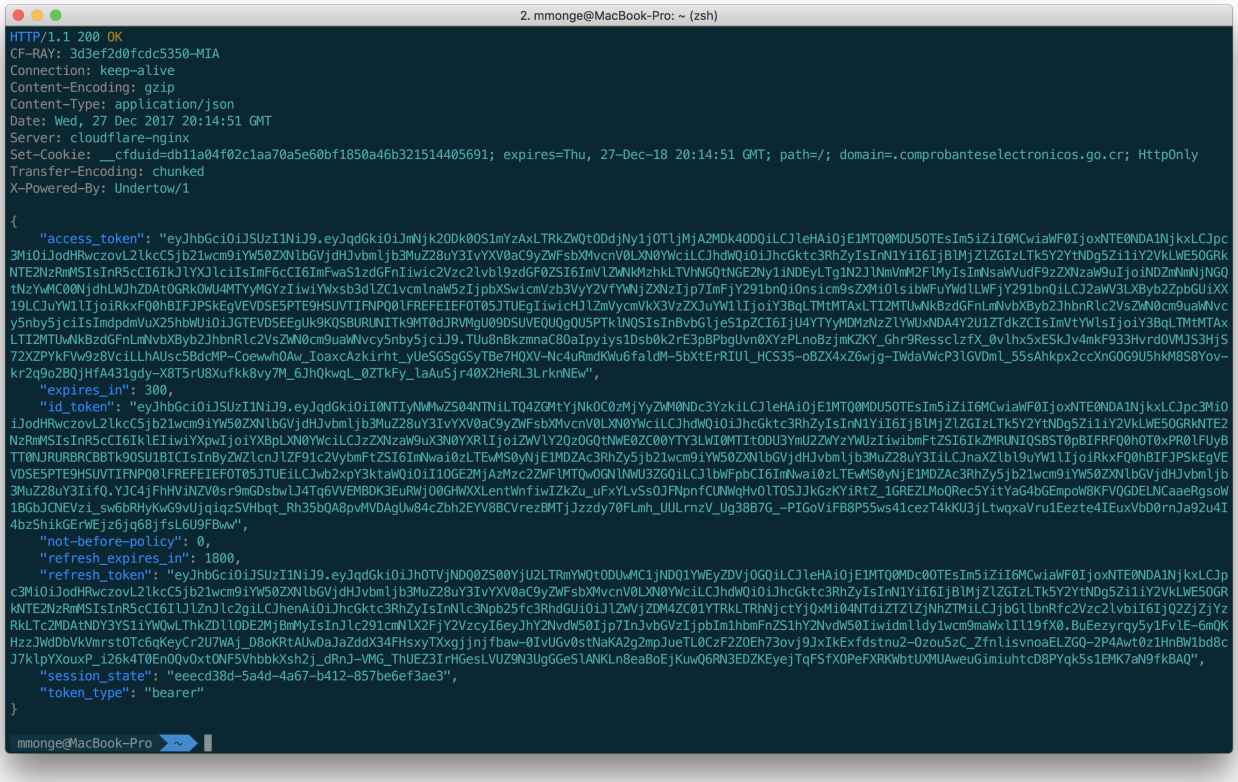


Figura 3. HTTPie, respuesta de obtener token

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 *API de Comprobantes Electrónicos* se debe enviar este.

Form Field	Data
`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=eyJhbGciOiJSUzI1NiJ9.eyJqdGkiOiJjYWU2MGJiMS02OWYxLTQ3MmYtYTE0NC03Y2EzOW
I5MjY3NzIiLCJleHAiOjE1MTQ0MDg5MjQsIm5iZiI6MCwiaWF0IjoxNTE0NDA3MTI0LCJpc3MiOiJodHRwczov
L2lkcC5jb21wcm9iYW50ZXNlbGVjdHJvbmljb3MuZ28uY3IvYXV0aC9yZWFsbXMvcnV0LXN0YWciLCJhdWQiOi
JhcGktc3RhZyIsInN1YiI6IjBlMjZlZGIzLTk5Y2YtNDg5Zi1iY2VkLWE5OGRkNTE2NzRmMSIsInR5cCI6IlJl
ZnJlc2giLCJhenAiOiJhcGktc3RhZyIsInNlc3Npb25fc3RhdGUiOiI0ZGUzZjY5Zi00OTFiLTQ1NWEtYmUxYi
1iMDYyZTQwNmM5NjIiLCJjbGllbnRfc2Vzc2lvbiI6ImM5NDU5YzlmLTU5OTAtNGY3OC05ZjMyLTUzOTU0OWM2
ZmY0OCIsInJlc291cmNlX2FjY2VzcyI6eyJhY2NvdW50Ijp7InJvbGVzIjpbIm1hbmFnZS1hY2NvdW50Iiwidm
lldy1wcm9maWxlIl19fX0.J8ENu7MMNIMlZTon-
wbcGH0CIylfoPlJLjyt6wcvkkbKgdgtPtYQw3vyEBwl23Jos5w0vQKtrJ7CpuUk8HvZbuNUifbf4t8MKystkXd
YgT6YnT3uBkL9zPsOVBeBLWRuKM3MFhspssCZ7nafXTxRAmQ0rxs81jZYhCydzyVu9UIW5VW8bu-
YyBPVKpEx6-
D5s5cp4132Dp7ZMzPaeNvkRdhQr9UL9KdzOEo5RqZZ85TZZrhQYRXheUPH_JOwuJP3WntNmhReXWzBX6RH4zCn
se1yzYnb4DR2LbfikBjPIPU6EH3BzpKBjZdXqGzwQUAhSMJWS_m1rTw0eDDp3Jdi6Q" \
     --data-urlencode "grant_type=refresh_token"
```

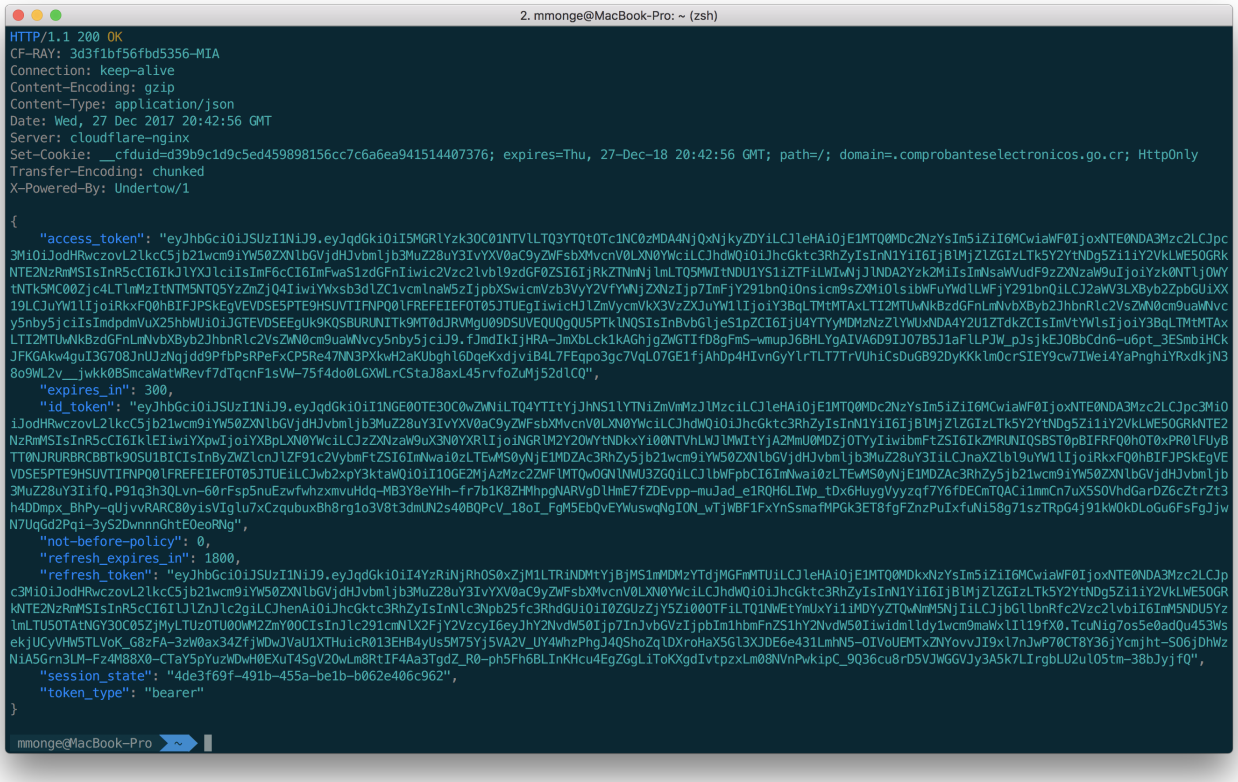
HTTPie, refrescar un token

```
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' \

'refresh_token'='eyJhbGciOiJSUzI1NiJ9.eyJqdGkiOiJjYWU2MGJiMS02OWYxLTQ3MmYtYTE0NC03Y2Ez
OWI5MjY3NzIiLCJleHAiOjE1MTQ0MDg5MjQsIm5iZiI6MCwiaWF0IjoxNTE0NDA3MTI0LCJpc3MiOiJodHRwcz
ovL2lkcC5jb21wcm9iYW50ZXNlbGVjdHJvbmljb3MuZ28uY3IvYXV0aC9yZWFsbXMvcnV0LXN0YWciLCJhdWQi
OiJhcGktc3RhZyIsInN1YiI6IjBlMjZlZGIzLTk5Y2YtNDg5Zi1iY2VkLWE5OGRkNTE2NzRmMSIsInR5cCI6Il
JlZnJlc2giLCJhenAiOiJhcGktc3RhZyIsInNlc3Npb25fc3RhdGUiOiI0ZGUzZjY5Zi00OTFiLTQ1NWEtYmUx
Yi1iMDYyZTQwNmM5NjIiLCJjbGllbnRfc2Vzc2lvbiI6ImM5NDU5YzlmLTU5OTAtNGY3OC05ZjMyLTUzOTU0OW
M2ZmY0OCIsInJlc291cmNlX2FjY2VzcyI6eyJhY2NvdW50Ijp7InJvbGVzIjpbIm1hbmFnZS1hY2NvdW50Iiwi
dmlldy1wcm9maWxlIl19fX0.J8ENu7MMNIMlZTon-
wbcGH0CIylfoPlJLjyt6wcvkkbKgdgtPtYQw3vyEBwl23Jos5w0vQKtrJ7CpuUk8HvZbuNUifbf4t8MKystkXd
YgT6YnT3uBkL9zPsOVBeBLWRuKM3MFhspssCZ7nafXTxRAmQ0rxs81jZYhCydzyVu9UIW5VW8bu-
YyBPVKpEx6-
D5s5cp4132Dp7ZMzPaeNvkRdhQr9UL9KdzOEo5RqZZ85TZZrhQYRXheUPH_JOwuJP3WntNmhReXWzBX6RH4zCn
se1yzYnb4DR2LbfikBjPIPU6EH3BzpKBjZdXqGzwQUAhSMJWS_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.eyJqdGkiOiJjYWU2MGJiMS02OWYxLTQ3MmYtYTE0NC03Y2
EzOWI5MjY3NzIiLCJleHAiOjE1MTQ0MDg5MjQsIm5iZiI6MCwiaWF0IjoxNTE0NDA3MTI0LCJpc3MiOiJodHRw
czovL2lkcC5jb21wcm9iYW50ZXNlbGVjdHJvbmljb3MuZ28uY3IvYXV0aC9yZWFsbXMvcnV0LXN0YWciLCJhdW
QiOiJhcGktc3RhZyIsInN1YiI6IjBlMjZlZGIzLTk5Y2YtNDg5Zi1iY2VkLWE5OGRkNTE2NzRmMSIsInR5cCI6
IlJlZnJlc2giLCJhenAiOiJhcGktc3RhZyIsInNlc3Npb25fc3RhdGUiOiI0ZGUzZjY5Zi00OTFiLTQ1NWEtYm
UxYi1iMDYyZTQwNmM5NjIiLCJjbGllbnRfc2Vzc2lvbiI6ImM5NDU5YzlmLTU5OTAtNGY3OC05ZjMyLTUzOTU0
OWM2ZmY0OCIsInJlc291cmNlX2FjY2VzcyI6eyJhY2NvdW50Ijp7InJvbGVzIjpbIm1hbmFnZS1hY2NvdW50Ii
widmlldy1wcm9maWxlIl19fX0.J8ENu7MMNIMlZTon-
wbcGH0CIylfoPlJLjyt6wcvkkbKgdgtPtYQw3vyEBwl23Jos5w0vQKtrJ7CpuUk8HvZbuNUifbf4t8MKystkXd
YgT6YnT3uBkL9zPsOVBeBLWRuKM3MFhspssCZ7nafXTxRAmQ0rxs81jZYhCydzyVu9UIW5VW8bu-
YyBPVKpEx6-
D5s5cp4132Dp7ZMzPaeNvkRdhQr9UL9KdzOEo5RqZZ85TZZrhQYRXheUPH_JOwuJP3WntNmhReXWzBX6RH4zCn
se1yzYnb4DR2LbfikBjPIPU6EH3BzpKBjZdXqGzwQUAhSMJWS_m1rTw0eDDp3Jdi6Q
```

Y se obtiene la siguiente respuesta:



```
HTTP/1.1 200 OK
CF-RAY: 3d3f1bf56fbd5356-MIA
Connection: keep-alive
Content-Encoding: gzip
Content-Type: application/json
Date: Wed, 27 Dec 2017 20:42:56 GMT
Server: cloudflare-nginx
Set-Cookie: __cfduid=d39b9c1d9c5ed459898156cc7c6a6ea941514407376; expires=Thu, 27-Dec-18 20:42:56 GMT; path=/; domain=.comprobanteselectronicos.go.cr; HttpOnly
Transfer-Encoding: chunked
X-Powered-By: Undertow/1

{
    "access_token": "eyJhbGciOiJSUzI1NiJ9.eyJqdGkiOiI5MGRlYzk3OC01NTVlLTQ3YTQtOTc1NC0zMDA4NjQxNjkyZDYiLCJleHAiOjE1MTQ0MDc2NzYsIm5iZiI6MCwiaWF0IjoxNTE0NDA3Mzc2LCJpc
3MiOiJodHRwczovL2lkcC5jb21wcm9iYW50ZXNlbGVjdHJvbmljb3MuZ28uY3IvYXV0aC9yZWFsbXMvcnV0LXN0YWciLCJhdWQiOiJhcGktc3RhZyIsInN1YiI6IjBlMjZlZGIzLTk5Y2YtNDg5Zi1iY2VkLWE5OGRk
NTE2NzRmMSIsInR5cCI6IkJlYXJlciIsImF6cCI6ImFwaS1zdGFnIiwic2Vzc2lvbl9zdGF0ZSI6IjRkZTNmNjlmLTQ5MWItNDU1YS1iZTFiLWIwNjJlNDA2Yzk2MiIsImNsaWVudF9zZXNzaW9uIjoiYzk0NTljOWY
tNTk5MC00Zjc4LTlmMzItNTM5NTQ5YzZmZjQ4IiwiYWxsb3dlZC1vcmlnaW5zIjpbXSwicmVzb3VyY2VfYWNjZXNzIjp7ImFjY291bnQiOnsicm9sZXMiOlsibWFuYWdlLWFjY291bnQiLCJ2aWV3LXByb2ZpbGUiXX
19LCJuYW1lIjoiRkxFQ0hBIFJPSkEgVEVDSE5PTE9HSUVTIFNPQ0lFREFEIEFOT05JTUEgIiwicHJlZmVycmVkX3VzZXJuYW1lIjoiY3BqLTMtMTAxLTI2MTUwNkBzdGFnLmNvbXByb2JhbnRlc2VsZWN0cm9uaWNvc
y5nby5jciIsImdpdmVuX25hbWUiOiJGTEVDSEEgUk9KQSBURUNITk9MT0dJRVMgU09DSUVEQUQgQU5PTklNQSIsInBvbGljeS1pZCI6IjU4YTYyMDMzNzZlYWUxNDA4Y2U1ZTdkZCIsImVtYWlsIjoiY3BqLTMtMTAx
LTI2MTUwNkBzdGFnLmNvbXByb2JhbnRlc2VsZWN0cm9uaWNvcy5nby5jciJ9.fJmdIkIjHRA-JmXbLck1kAGhjgZWGTIfD8gFmS-wmupJ6BHLYgAIVA6D9IJO7B5J1aFlLPJW_pJsjkEJOBbCdn6-u6pt_3ESmbiHCk
JFKGAkw4guI3G7O8JnUJzNqjdd9PfbPsRPeFxCP5Re47NN3PXkwH2aKUbghl6DqeKxdjviB4L7FEqpo3gc7VqLO7GE1fjAhDp4HIvnGyYlrTLT7TrVUhiCsDuGB92DyKKklmOcrSIEY9cw7IWei4YaPnghiYRxdkjN3
8o9WL2v__jwkk0BSmcaWatWRevf7dTqcnF1sVW-75f4do0LGXWLrCStaJ8axL45rvfoZuMj52dlCQ",
    "expires_in": 300,
    "id_token": "eyJhbGciOiJSUzI1NiJ9.eyJqdGkiOiI1NGE0OTE3OC0wZWNiLTQ4YTItYjJhNS1lYTNiZmVmMzJlMzciLCJleHAiOjE1MTQ0MDc2NzYsIm5iZiI6MCwiaWF0IjoxNTE0NDA3Mzc2LCJpc3MiO
iJodHRwczovL2lkcC5jb21wcm9iYW50ZXNlbGVjdHJvbmljb3MuZ28uY3IvYXV0aC9yZWFsbXMvcnV0LXN0YWciLCJhdWQiOiJhcGktc3RhZyIsInN1YiI6IjBlMjZlZGIzLTk5Y2YtNDg5Zi1iY2VkLWE5OGRkNTE2
NzRmMSIsInR5cCI6IklEIiwiYXpwIjoiYXBpLXN0YWciLCJzZXNzaW9uX3N0YXRlIjoiNGRlM2Y2OWYtNDkxYi00NTVhLWJlMWItYjA2MmU0MDZjOTYyIiwibmFtZSI6IkZMRUNIQSBST0pBIFRFQ0hOT0xPR0lFUyB
TT0NJRURBRCBBTk9OSU1BICIsInByZWZlcnJlZF91c2VybmFtZSI6ImNwai0zLTEwMS0yNjE1MDZAc3RhZy5jb21wcm9iYW50ZXNlbGVjdHJvbmljb3MuZ28uY3IiLCJnaXZlbl9uYW1lIjoiRkxFQ0hBIFJPSkEgVE
VDSE5PTE9HSUVTIFNPQ0lFREFEIEFOT05JTUEiLCJwb2xpY3ktaWQiOiI1OGE2MjAzMzc2ZWFlMTQwOGNlNWU3ZGQiLCJlbWFpbCI6ImNwai0zLTEwMS0yNjE1MDZAc3RhZy5jb21wcm9iYW50ZXNlbGVjdHJvbmljb
3MuZ28uY3IifQ.P91q3h3QLvn-60rFsp5nuEzwfwhzxmvuHdq-MB3Y8eYHh-fr7b1K8ZHMhpgNARVgDlHmE7fZDEvpp-muJad_e1RQH6LIWp_tDx6HuygVyyzqf7Y6fDECmTQACi1mmCn7uX5SOVhdGarDZ6cZtrZt3
h4DDmpx_BhPy-qUjvvRARC80yisVIglu7xCzqubuxBh8rg1o3V8t3dmUN2s40BQPcV_18oI_FgM5EbQvEYWuswqNgION_wTjWBF1FxYnSsmafMPGk3ET8fgFZnzPuIxfuNi58g71szTRpG4j91kWOkDLoGu6FsFgJjw
N7UqGd2Pqi-3yS2DwnnnGhtEOeoRNg",
    "not-before-policy": 0,
    "refresh_expires_in": 1800,
    "refresh_token": "eyJhbGciOiJSUzI1NiJ9.eyJqdGkiOiI4YzRiNjRhOS0xZjM1LTRiNDMtYjBjMS1mMDMzYTdjMGFmMTUiLCJleHAiOjE1MTQ0MDkxNzYsIm5iZiI6MCwiaWF0IjoxNTE0NDA3Mzc2LCJp
c3MiOiJodHRwczovL2lkcC5jb21wcm9iYW50ZXNlbGVjdHJvbmljb3MuZ28uY3IvYXV0aC9yZWFsbXMvcnV0LXN0YWciLCJhdWQiOiJhcGktc3RhZyIsInN1YiI6IjBlMjZlZGIzLTk5Y2YtNDg5Zi1iY2VkLWE5OGR
kNTE2NzRmMSIsInR5cCI6IlJlZnJlc2giLCJhenAiOiJhcGktc3RhZyIsInNlc3Npb25fc3RhdGUiOiI0ZGUzZjY5Zi00OTFiLTQ1NWEtYmUxYi1iMDYyZTQwNmM5NjIiLCJjbGllbnRfc2Vzc2lvbiI6ImM5NDU5Yz
lmLTU5OTAtNGY3OC05ZjMyLTUzOTU0OWM2ZmY0OCIsInJlc291cmNlX2FjY2VzcyI6eyJhY2NvdW50Ijp7InJvbGVzIjpbIm1hbmFnZS1hY2NvdW50Iiwidmlldy1wcm9maWxlIl19fX0.TcuNig7os5e0adQu453Ws
ekjUCyVHW5TLVoK_G8zFA-3zW0ax34ZfjWDwJVaU1XTHuicR013EHB4yUs5M75Yj5VA2V_UY4WhzPhgJ4QShoZqlDXroHaX5Gl3XJDE6e431LmhN5-OIVoUEMTxZNYovvJI9xl7nJwP70CT8Y36jYcmjht-SO6jDhWz
NiA5Grn3LM-Fz4M88X0-CTaY5pYuzWDwH0EXuT4SgV2OwLm8RtIF4Aa3TgdZ_R0-ph5Fh6BLInKHcu4EgZGgLiToKXgdIvtpzxLm08NVnPwkipC_9Q36cu8rD5VJWGGVJy3A5k7LIrgbLU2ulO5tm-38bJyjfQ",
    "session_state": "4de3f69f-491b-455a-be1b-b062e406c962",
    "token_type": "bearer"
}

mmonge@MacBook-Pro  ~ 
```

Figura 4. HTTPie, respuesta de refrescar token

La respuesta obtenida es prácticamente la misma del proceso anterior de obtener un token, acá lo que se hace es igualmente obtener un token pero por medio de un `refresh_token`. 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 *API de Comprobantes Electrónicos* 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 cerrar sesión? (Eliminar el token)

Para cerrar sesión se debe enviar una petición *HTTP POST* al *LOGOUT_URL* con los siguientes *form fields*:

Form Field	Data
client_id	api-stag
refresh_token	Data del refresh_token.

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.eyJqdGkiOiJjYWU2MGJiMS02OWYxLTQ3MmYtYTE0NC03Y2EzOW
I5MjY3NzIiLCJleHAiOjE1MTQ0MDg5MjQsIm5iZiI6MCwiaWF0IjoxNTE0NDA3MTI0LCJpc3MiOiJodHRwczov
L2lkcC5jb21wcm9iYW50ZXNlbGVjdHJvbmljb3MuZ28uY3IvYXV0aC9yZWFsbXMvcnV0LXN0YWciLCJhdWQiOi
JhcGktc3RhZyIsInN1YiI6IjBlMjZlZGIzLTk5Y2YtNDg5Zi1iY2VkLWE5OGRkNTE2NzRmMSIsInR5cCI6IlJl
ZnJlc2giLCJhenAiOiJhcGktc3RhZyIsInNlc3Npb25fc3RhdGUiOiI0ZGUzZjY5Zi00OTFiLTQ1NWEtYmUxYi
1iMDYyZTQwNmM5NjIiLCJjbGllbnRfc2Vzc2lvbiI6ImM5NDU5YzlmLTU5OTAtNGY3OC05ZjMyLTUzOTU0OWM2
ZmY0OCIsInJlc291cmNlX2FjY2VzcyI6eyJhY2NvdW50Ijp7InJvbGVzIjpbIm1hbmFnZS1hY2NvdW50Iiwidm
lldy1wcm9maWxlIl19fX0.J8ENu7MMNIMlZTon-
wbcGH0CIylfoPlJLjyt6wcvkkbKgdgtPtYQw3vyEBwl23Jos5w0vQKtrJ7CpuUk8HvZbuNUifbf4t8MKystkXd
YgT6YnT3uBkL9zPsOVBeBLWRuKM3MFhspssCZ7nafXTxRAmQ0rxs81jZYhCydzyVu9UIW5VW8bu-
YyBPVKpEx6-
D5s5cp4132Dp7ZMzPaeNvkRdhQr9UL9KdzOEo5RqZZ85TZZrhQYRXheUPH_JOwuJP3WntNmhReXWzBX6RH4zCn
se1yzYnb4DR2LbfikBjPIPU6EH3BzpKBjZdXqGzwQUAhSMJWS_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.eyJqdGkiOiJjYWU2MGJiMS02OWYxLTQ3MmYtYTE0NC03Y2Ez
OWI5MjY3NzIiLCJleHAiOjE1MTQ0MDg5MjQsIm5iZiI6MCwiaWF0IjoxNTE0NDA3MTI0LCJpc3MiOiJodHRwcz
ovL2lkcC5jb21wcm9iYW50ZXNlbGVjdHJvbmljb3MuZ28uY3IvYXV0aC9yZWFsbXMvcnV0LXN0YWciLCJhdWQi
OiJhcGktc3RhZyIsInN1YiI6IjBlMjZlZGIzLTk5Y2YtNDg5Zi1iY2VkLWE5OGRkNTE2NzRmMSIsInR5cCI6Il
JlZnJlc2giLCJhenAiOiJhcGktc3RhZyIsInNlc3Npb25fc3RhdGUiOiI0ZGUzZjY5Zi00OTFiLTQ1NWEtYmUx
Yi1iMDYyZTQwNmM5NjIiLCJjbGllbnRfc2Vzc2lvbiI6ImM5NDU5YzlmLTU5OTAtNGY3OC05ZjMyLTUzOTU0OW
M2ZmY0OCIsInJlc291cmNlX2FjY2VzcyI6eyJhY2NvdW50Ijp7InJvbGVzIjpbIm1hbmFnZS1hY2NvdW50Iiwi
dmlldy1wcm9maWxlIl19fX0.J8ENu7MMNIMlZTon-
wbcGH0CIylfoPlJLjyt6wcvkkbKgdgtPtYQw3vyEBwl23Jos5w0vQKtrJ7CpuUk8HvZbuNUifbf4t8MKystkXd
YgT6YnT3uBkL9zPsOVBeBLWRuKM3MFhspssCZ7nafXTxRAmQ0rxs81jZYhCydzyVu9UIW5VW8bu-
YyBPVKpEx6-
D5s5cp4132Dp7ZMzPaeNvkRdhQr9UL9KdzOEo5RqZZ85TZZrhQYRXheUPH_JOwuJP3WntNmhReXWzBX6RH4zCn
se1yzYnb4DR2LbfikBjPIPU6EH3BzpKBjZdXqGzwQUAhSMJWS_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.eyJqdGkiOiJjYWU2MGJiMS02OWYxLTQ3MmYtYTE0NC03Y2
EzOWI5MjY3NzIiLCJleHAiOjE1MTQ0MDg5MjQsIm5iZiI6MCwiaWF0IjoxNTE0NDA3MTI0LCJpc3MiOiJodHRw
czovL2lkcC5jb21wcm9iYW50ZXNlbGVjdHJvbmljb3MuZ28uY3IvYXV0aC9yZWFsbXMvcnV0LXN0YWciLCJhdW
QiOiJhcGktc3RhZyIsInN1YiI6IjBlMjZlZGIzLTk5Y2YtNDg5Zi1iY2VkLWE5OGRkNTE2NzRmMSIsInR5cCI6
IlJlZnJlc2giLCJhenAiOiJhcGktc3RhZyIsInNlc3Npb25fc3RhdGUiOiI0ZGUzZjY5Zi00OTFiLTQ1NWEtYm
UxYi1iMDYyZTQwNmM5NjIiLCJjbGllbnRfc2Vzc2lvbiI6ImM5NDU5YzlmLTU5OTAtNGY3OC05ZjMyLTUzOTU0
OWM2ZmY0OCIsInJlc291cmNlX2FjY2VzcyI6eyJhY2NvdW50Ijp7InJvbGVzIjpbIm1hbmFnZS1hY2NvdW50Ii
widmlldy1wcm9maWxlIl19fX0.J8ENu7MMNIMlZTon-
wbcGH0CIylfoPlJLjyt6wcvkkbKgdgtPtYQw3vyEBwl23Jos5w0vQKtrJ7CpuUk8HvZbuNUifbf4t8MKystkXd
YgT6YnT3uBkL9zPsOVBeBLWRuKM3MFhspssCZ7nafXTxRAmQ0rxs81jZYhCydzyVu9UIW5VW8bu-
YyBPVKpEx6-
D5s5cp4132Dp7ZMzPaeNvkRdhQr9UL9KdzOEo5RqZZ85TZZrhQYRXheUPH_JOwuJP3WntNmhReXWzBX6RH4zCn
se1yzYnb4DR2LbfikBjPIPU6EH3BzpKBjZdXqGzwQUAhSMJWS_m1rTw0eDDp3Jdi6Q
```

Y se obtiene la siguiente respuesta:

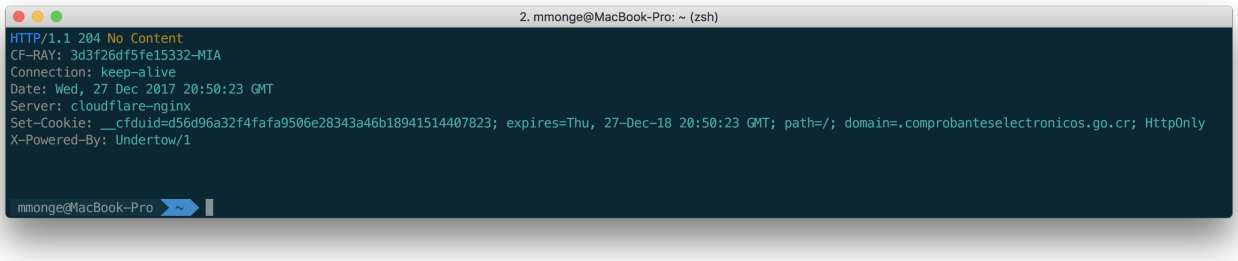


Figura 5. HTTPie, respuesta de cerrar sesión

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 pretente 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.eyJqdGkiOiJkNmVlYmE1OC1jNGNiLTQ4NmItYTZhMy04ZTk5M2ZjY2NlMjgiLCJle
HAiOjE1MTQ0MDc0MjQsIm5iZiI6MCwiaWF0IjoxNTE0NDA3MTI0LCJpc3MiOiJodHRwczovL2lkcC5jb21wcm9
iYW50ZXNlbGVjdHJvbmljb3MuZ28uY3IvYXV0aC9yZWFsbXMvcnV0LXN0YWciLCJhdWQiOiJhcGktc3RhZyIsI
nN1YiI6IjBlMjZlZGIzLTk5Y2YtNDg5Zi1iY2VkLWE5OGRkNTE2NzRmMSIsInR5cCI6IkJlYXJlciIsImF6cCI
6ImFwaS1zdGFnIiwic2Vzc2lvbl9zdGF0ZSI6IjRkZTNmNjlmLTQ5MWItNDU1YS1iZTFiLWIwNjJlNDA2Yzk2M
iIsImNsaWVudF9zZXNzaW9uIjoiYzk0NTljOWYtNTk5MC00Zjc4LTlmMzItNTM5NTQ5YzZmZjQ4IiwiYWxsb3d
lZC1vcmlnaW5zIjpbXSwicmVzb3VyY2VfYWNjZXNzIjp7ImFjY291bnQiOnsicm9sZXMiOlsibWFuYWdlLWFjY
291bnQiLCJ2aWV3LXByb2ZpbGUiXX19LCJuYW1lIjoiRkxFQ0hBIFJPSkEgVEVDSE5PTE9HSUVTIFNPQ0lFREF
EIEFOT05JTUEgIiwicHJlZmVycmVkX3VzZXJuYW1lIjoiY3BqLTMtMTAxLTI2MTUwNkBzdGFnLmNvbXByb2Jhb
nRlc2VsZWN0cm9uaWNvcy5nby5jciIsImdpdmVuX25hbWUiOiJGTEVDSEEgUk9KQSBURUNITk9MT0dJRVMgU09
DSUVEQUQgQU5PTklNQSIsInBvbGljeS1pZCI6IjU4YTYyMDMzNzZlYWUxNDA4Y2U1ZTdkZCIsImVtYWlsIjoiY
3BqLTMtMTAxLTI2MTUwNkBzdGFnLmNvbXByb2JhbnRlc2VsZWN0cm9uaWNvcy5nby5jciJ9.IhVlTY0be_Pjcq
R9e0Aekd9STZZlcGz2NyawImaHzpHPX7SiZS_eNhXg8wNLoX0lk36APCyxmdtLEd077ZdVkAUnc5vMiEBSVnFD
2v02DXzhCvn3VmT3aTomLxqqpR5n4IHUPQDlT41PTSrvOBIMemocfm5VIqzNbeBPAS28qhJMYInSkVRyyQ6YLy
S0fmGd9Rtqu7bnYx431QbFn5Z2rYXGD2JEZVkGnFptarKXFFUdBzC_XjJQZZnA_I2aaCwS5xpr4ZjBSedPoJAf
gkzfuDXCCN49Igt7KFSprN75TU6r3lr_TIS9LvzP-oDlucnRPZRECs-_Qtx-tbbAyW2yJg
```

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.

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. Esto se podría hacer de varias maneras, ejemplo se podría tener un proceso que cuando va a enviar a la plataforma revisa si tiene un `access_token` que no haya expirado, sino lo tiene entonces revisa que tenga un `refresh_token` no expirado (el `refresh_expires_in` al momento de hacer el documento son 10 horas), para poder refrescar el token, y si tampoco lo tiene entonces crea una sesión.

El *IdP* de la plataforma por el momento no tiene un límite de sesiones que un mismo usuario puede

tener abiertas, pero 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.