IAM Y COGNITO

En esta práctica vamos a ver tres ejemplos de cómo se puede utilizar el servicio de autenticación de Cognito de AWS.

En el primer ejemplo utilizaremos los 'User Pools' que permiten gestionar usuarios, su registro y el login. Como ejemplo se utilizara una sencilla aplicación cliente en Javascript (que podría emular una aplicación de móvil o desktop). Esta aplicación permite registrar y realizar login desde un aplicación cliente que podrá utilizar a posteriori el Token obtenido.

La segunda aplicación utilizará además un identity pool , que permite relacionar roles de AWS a los usuarios , permitiendo así acceder a recursos AWS desde una aplicación cliente;-)

El tercer ejemplo permite utilizar credenciales OAuth de otros proveedores , en este caso Google, y obtener credenciales de AWS via un identity pool. De forma que la aplicación cliente puede acceder a servicio de AWS.

Ejemplo User Pool desde aplicación cliente

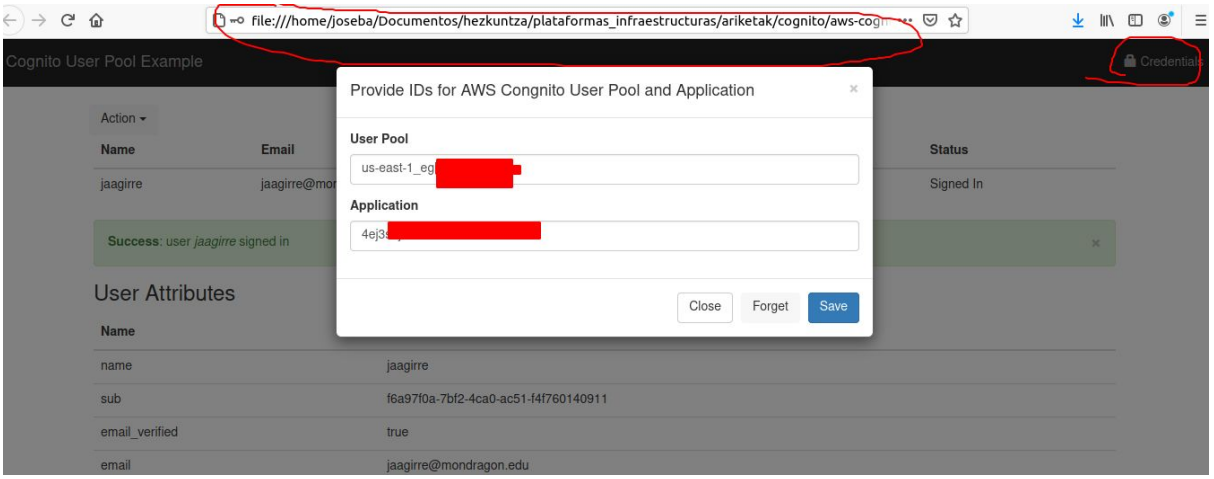
En este ejemplo utilizareis el siguiente código que se ubica en mudle ***aws-cognito-js-example-master.zip***. Este código pertenece al siguiente repositorio Github https://github.com/RomanKosobrodov/aws-cognito-js-example. Este ejemplo es una sencilla aplicación Javascript que permite crear/registrar, realizar login, logout y des registrarnos en un pool de usuarios de cognito.

1- Lo primero que haremos será ir al servicio Cognito y crear un pool de usuarios. El pool de usuario se creará con las propiedades que se indican en los siguientes puntos .Para ello no selecciones la opción de valores por defecto , elegir recorrer propiedades.

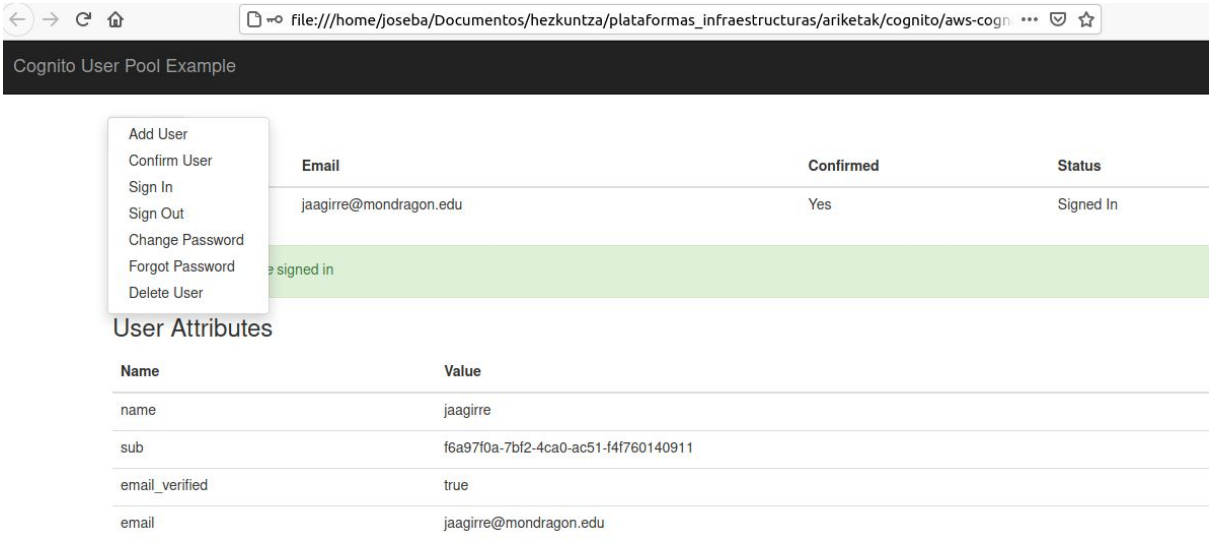
2- Habilitar el loging mediante user name.
   Enable signing in with a user name ("Users can use a username and optionally multiple alternatives to sign up and sign in").
3- Seleccionar la casilla ***"Also allow sign in with verified email address"***
4- Elegir email y preffered_username como atributos requeridos
5- Seleccionar "Allow users to sign themselves up" en la página de politicas
6- Cuando os pregunten "Do you want to require verification of emails or phone numbers?" seleccionar email
7- No selecciones características por defecto
8- Deshabilitar características avanzadas de seguridad
9- Elegir código com metod de verificación
10- Por ultimo un a vez creado el pool de usuarios crear uan aplicación y deshabilitar seceretos. Ya que en la librerai de ASWS de Javascript no se utilizan.

11- Con esto ya estamos preparados para lanzar la aplicación y utilizarla. Para ello copiar el Id del pool y el id de la aplicación.

12- Ahora solo queda ejecutar en local la aplicación. Una vez lanzada la aplicación clickar en credenciales donde configuraréis los ID del user pool.

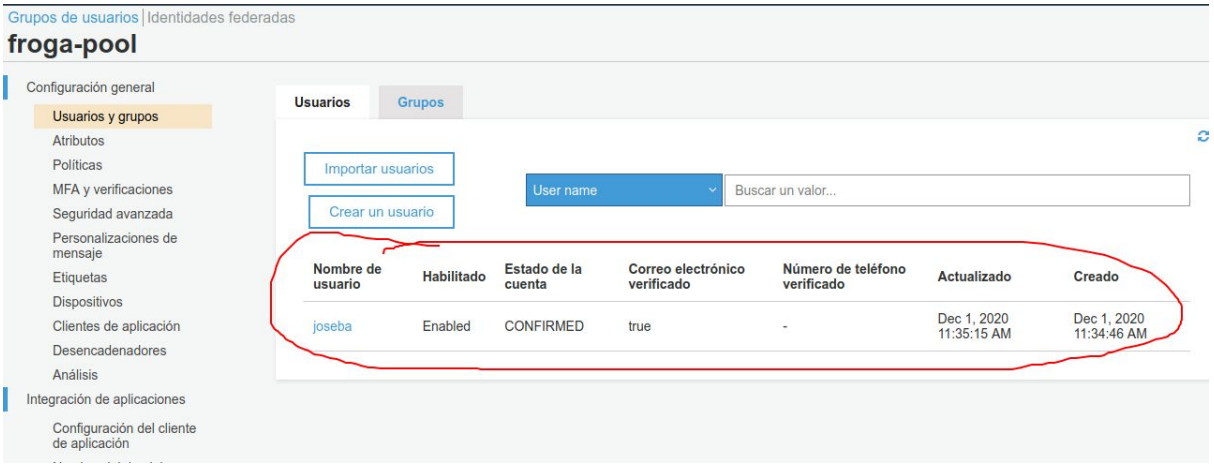


13- Y ahora ya podéis realizar las diferentes acciones.



14- Cuando creeis los usuarios deberías de recibir el codigo via email.

15- También podéis comprobar que los usuarios se crean en el pool. En el pool se puede deshabilitar la posibilidad de registro, de forma que solo el admin pueda crear los usuarios.

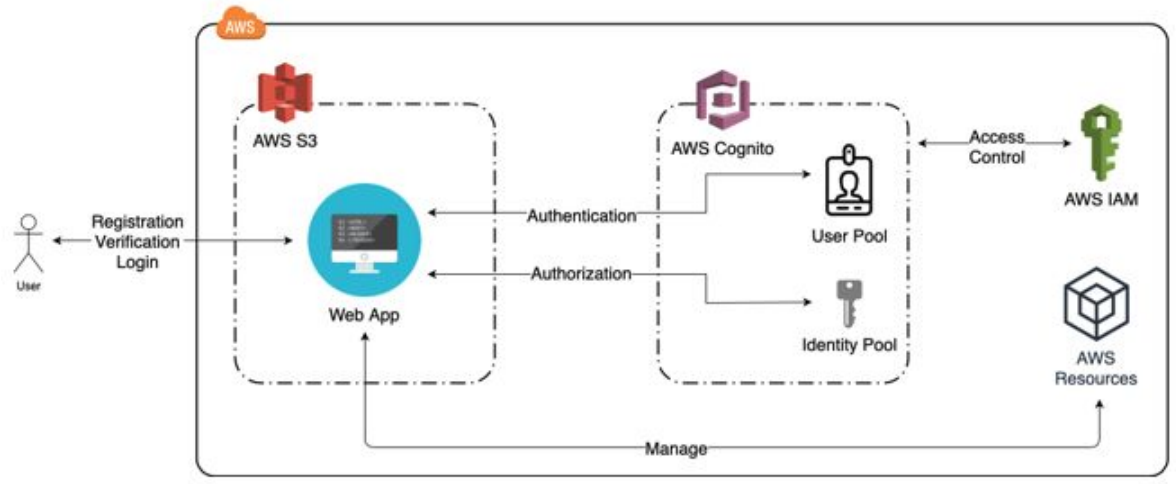


Ejemplo User Pool más Identity Pool

En el ejemplo anterior solo se utilizaba el pool de usuarios para no tener que crear una infraestructura de login desde cero. Pero con la anterior configuración el login no permite al usuario de la aplicación cliente acceder a recursos AWS , ya que no tiene credenciales. Para poder asignar a usuario credenciales utilizaremos en Cognito el identity Pool. El identity pool se asocia a un user pool y permite asignar roles de IAM a los usuarios logeados ;-) En este ejemplo creamos una página web que permite a los usuarios obtener credenciales de AWS y listar cualquier bucket de S3 de nuestra cuenta..

16- En este ejemplo utiliziareis el código "***AWSCognitoUserPools-master.zip***".Este codigo es una pequeña aplicación Javascript que permitirá listar el contenido de los buckets a los clientes, sin dar permisos/role a un servidor y sin crear cuentas ;-)

17- la imagen resume el ejemplo



17- En este ejemplo podríamos usar el pool anterior o crear uno con las propiedades por defecto. Lo que prefiráis.

18- Una vez disponemos de un pool de usuarios con una aplicación y los secretos deshabilitados pasamos a crear un pool de identidades.

19- Al crear el pool de identidades tenemos que tener en cuenta lo siguiente:

- Le damos un nombre
- Y en la sección “Authentication providers” seleccionamos Cognito y le indicamos el ID del User Pool y el Id de la aplicación que utilizaremos.

20- Cuando creeis el pool de identidades fijaros en el Role quu crea y le asocia. Despues en el menú de IAM configuraréis las políticas de este role. Y asi le daréis los permisos sobre los recurso de AWS que decidais.

21- Ir a IAM elegir el role y darle S3FullAccess

22-Ahora ya tenemos todo preparado lo siguiente que haremos sera alojar la palicación cliente en S3.

23- Para ello creamos un bucket. En el bucket deshabilitamos el bloqueo público ya que queremos que se descarguen las páginas. Solo activar el bloqueo de las dos ultimas opciones.

24- Despeues en propiedades le indicamos que el bucket alojara una web estatica. Y le inidcamos la pagina index por defecto y la da error.

25- Por último activamos CORS para se puedan descargar los diferentes archivos de la página. Esto lo haceis en los permisos del Bucket. Aqui en Origins podrias poner el dominio del Bucket ;-)

```
        {
        "AllowedHeaders": [
        "*"
        ],
        "AllowedMethods": [
        "PUT",
        "POST",
        "DELETE",
        "GET"
        ],
        "AllowedOrigins": [
        "*"
        ],
        "ExposeHeaders": []
        }
]
```

26- Y ahora solo queda subir la aplicación al bucket (tanto el index.html como la carpeta scripts). Antes de subir al bucket el codigo teneis que modificar la siguiente información en la aplicación.: el user pool id , el app id y el identity pol id en el index.html

```
<script>

    //=============== AWS IDs ===============
    var userPoolId = 'us-east-1[redacted]
    var clientId = '4ej3s[redacted]';
    var region = 'us-east-1';
    var identityPoolId = 'us-east-1[redacted]';
    //=============== AWS IDs ===============
```

27- Hacer publico la pagina y los scripts.Y probar la aplicación utilizando la url del bucket s3.

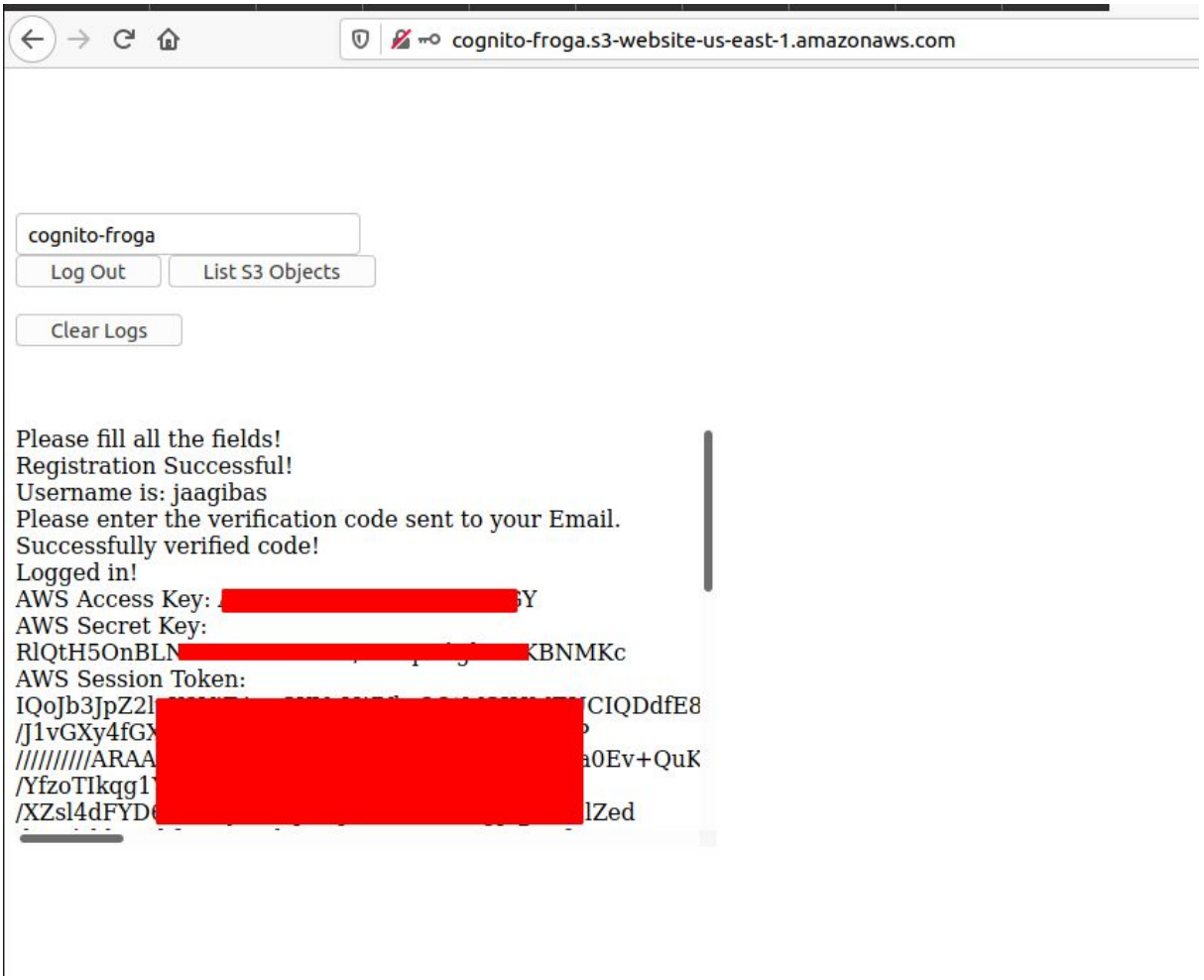
28- Primero registrar un email y validar mediante el código que os llegara

29- y ahora logearos y probar a listar el bucket. Si queréis listar otros bucket vuestros tenéis que activar CORS en lo bucket , ya que estamos haciendo la petición al bucket desde otro dominio ;-)

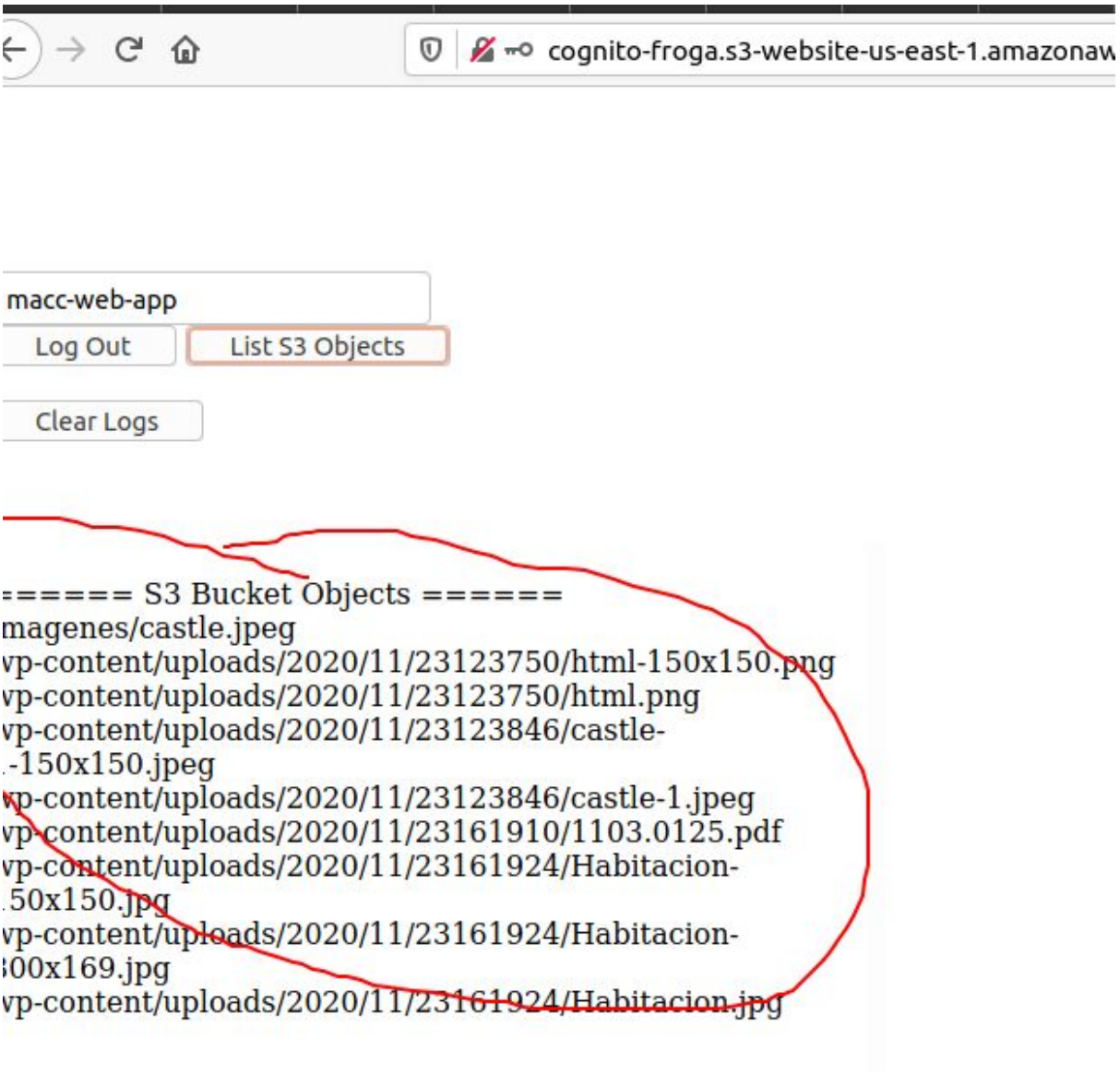
En los otros buckets que queráis utilizar, poniendo en origin la url de vuestro bucket donde se aloja la página web:

```
[
        {
        "AllowedHeaders": [
        "*"
        ],
        "AllowedMethods": [
        "PUT",
        "POST",
        "DELETE",
        "GET"
        ],
```

```
        "AllowedOrigins": [
        "http://cognito-froga.s3-website-us-east-1.amazonaws.com"
        ],
        "ExposeHeaders": []
        }
]
```



30- y ahora listamos contenido



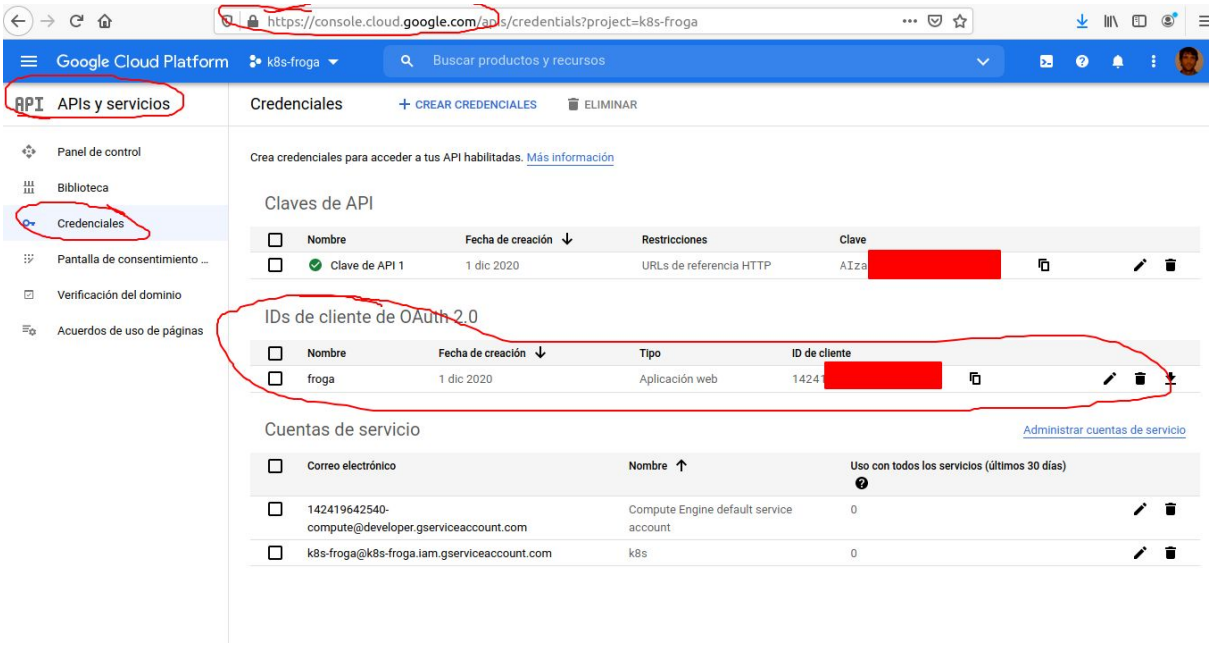
31- Podéis probar como si teneis contenido privado desde la aplicación la podeis ver pero navegando directamente a la URL no.

Ejemplo Proveedor externo e Identity Pool

Por último en esta última parte en lugar de logearnos con el user pool utilizaremos el login de GMAIL. Para ello en el identity pool crearemos otro provider en este caso admeas del de cognito una de Google ;-)

32- Para esta práctica utilizareis el código “cognito-idp-tst-master.zip” que consta de una página y un pequeño script. Crear un buclet y como antes activar acceso publico y CORS y subir todo código.

33- Para esta práctica tenéis que daros de alta en google cloud con vuestro usuario de mail y en API crear un cliente ID Oauth para aplicaciones web. Ahi creais el id de cliente que permiteris que haga login en cognito. Para ello en dominio teneis que darle el origen del bucket dondehabeis alojado la aplicación.



34- Ahora copiar el ID de google y en Cognito en el identity pool de la práctica anterior ir a provvedores y en Google copiar el id del servicio qu ehabeis creado. Y ya esta.

35- Ahora solo os queda copiar el id del identity pool de Cognito y el id de cliente de google en el código.

36- El id de google en index.html

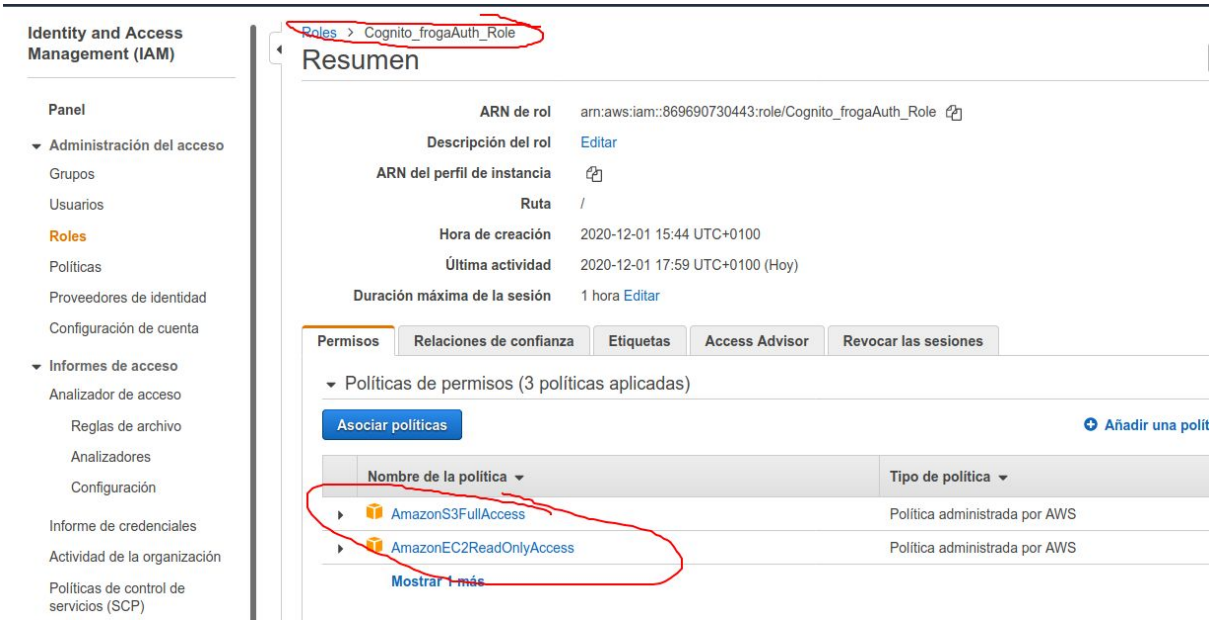
```
<html lang="en">
  <title>| COGNITO-IDP-AUTH |</title>
  <head>
    <meta name="author" content="sebolabs">
    <meta name="google-signin-scope" content="profile email">
    <!-- Google Client ID -->
    <meta name="google-signin-client_id" content="142[redacted]">
    <script src="https://apis.google.com/js/platform.js" async defer></script>
    <script src="https://sdk.amazonaws.com/js/aws-sdk-2.2.19.min.js"></script>
    <script src="scripts.js"></script>
  </head>
  <body>
    <div class="g-signin2" data-onsuccess="onSignIn" data-theme="dark"></div>
    <p />
    <button onclick="onSignOut()">Sign out</button>
    <p />
    <div id="output"></div>
  </body>
</html>
```

37- Y el id del pool de identidades en el scripts.js

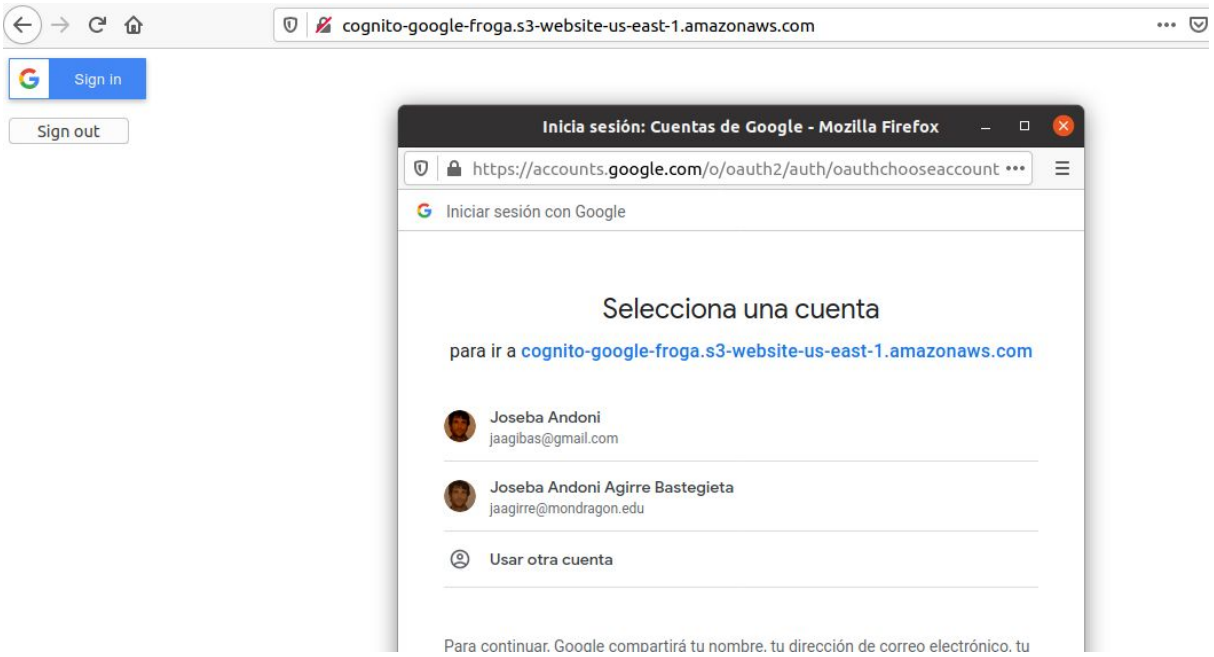
```
function signInCallback(authResult) {
  if (authResult['access_token']) {

    // adding google access token to Cognito credentials login map
    AWS.config.region = 'us-east-1';
    AWS.config.credentials = new AWS.CognitoIdentityCredentials({
      IdentityPoolId: 'us-east-1:e[redacted]
      Logins: {
        'accounts.google.com': authResult['id_token']
      }
    });
```

38- Lo unico antes de probarlo este código mira iinformacić¸on de las instancias Ec2 por lo que tendréis que darle al Role del identity desde IAM permisos para EC2. Darselo solo para probar la práctica luego quitarlo por seguridad. Dar solo permisos de lectura en EC2



39- Y ahora probar la aplicación



40- Y aqui vemos lo que obtenemos ;-)