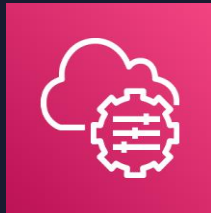


PRÁCTICAS DE LABORATORIO

Guillermina Antonaccio
Felipe Putrelli, Yara Villar,
Guillermo San Martín

Vigésimo quinto
laboratorio (278):

Protección de datos
mediante encriptación



TAREA 1:

En esta tarea había que crear una llave(o clave) en AWS Key Managment Service (AWS KMS) la cual vamos a usar para cifrar y descifrar datos

- 1-El tipo de clave debía ser simétrico
 - 2-Teníamos que ponerle un Alias y una descripción
 - 3-Definir **permisos administrativos clave: elegimos voclabs**
 - 4-Definir **permisos de uso de claves: elegimos voclabs**
 - 5-Una vez creada la llave debíamos copiar el valor ARN de la misma.
-

Aquí podemos ver los datos de la llave

Review

Key configuration

Key type	Key spec	Key usage
Symmetric	SYMMETRIC_DEFAULT	Encrypt and decrypt
Origin	Regionality	
AWS KMS	Single-Region key	

Alias and description

Alias	Description
-------	-------------

Key configuration details:

- Key type: Symmetric
- Key spec: SYMMETRIC_DEFAULT
- Key usage: Encrypt and decrypt
- Origin: AWS KMS
- Regionality: Single-Region key

Warning: You cannot change the key configuration after the key is created.

Steps:

- Step 1: [Configure key](#)
- Step 2: [Add labels](#)
- Step 3: [Define key administrative permissions](#)
- Step 4: [Define key usage permissions](#)
- Step 5: **Review**

Footer: CloudShell Feedback © 2023, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

En esta imagen podemos ver que la llave fue creada con éxito

The screenshot displays the AWS Key Management Service (KMS) console. At the top, a green success banner states: "Success Your AWS KMS key was created with alias **MyKMSKey** and key ID **9f044dc7-f447-42a3-8b86-dc137d74dcd7**." Below this, the "Customer managed keys" section shows a table with one key. The table has columns for selection, aliases, key ID, status, key type, key spec, and key usage. The key "MyKMSKey" is listed with key ID "9f044dc7-f447-42a3-8b86-dc137d74dcd7", status "Enabled", and type "Symmetric".

Key Management Service (KMS)

Success
Your AWS KMS key was created with alias **MyKMSKey** and key ID **9f044dc7-f447-42a3-8b86-dc137d74dcd7**.

[KMS](#) > Customer managed keys

Customer managed keys (1/1)

Filter keys by properties or tags

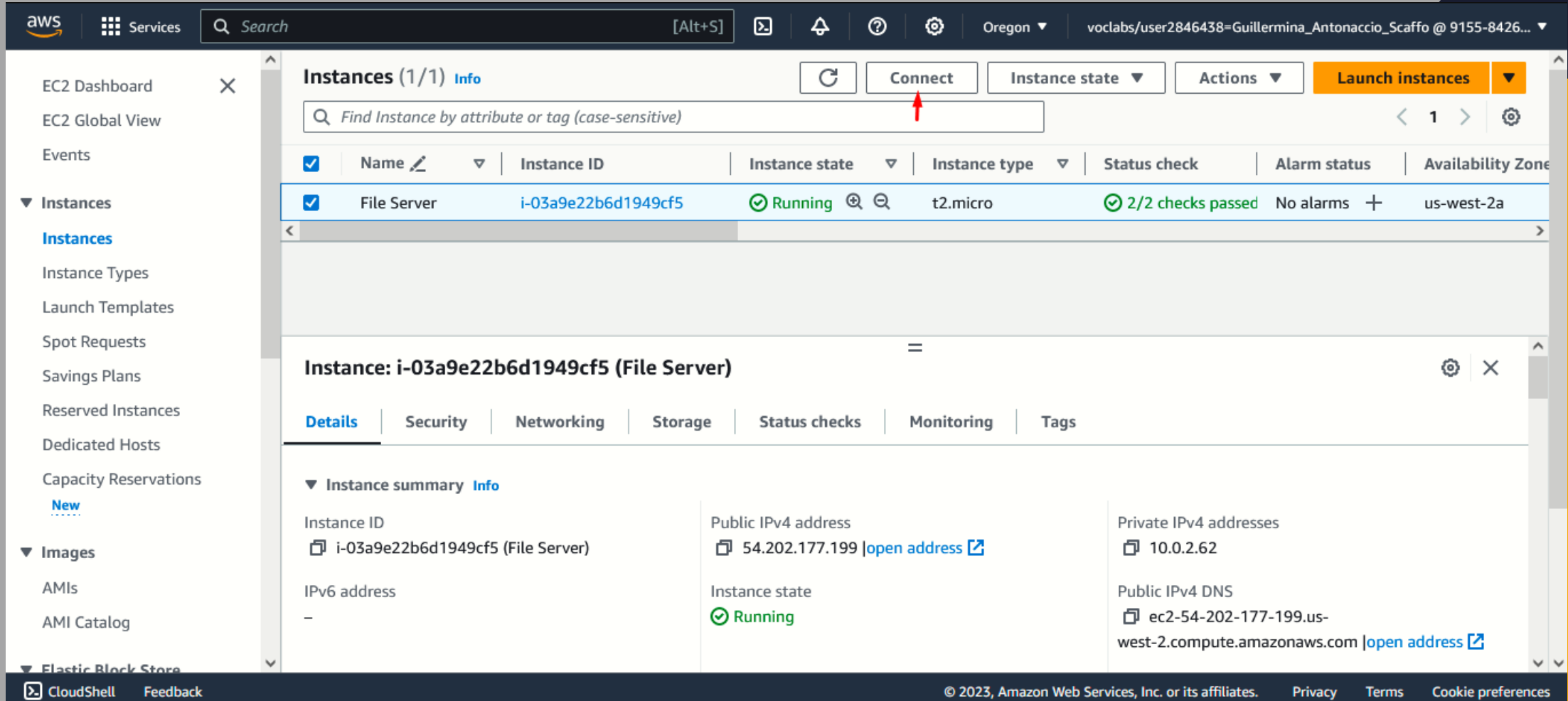
<input checked="" type="checkbox"/>	Aliases	Key ID	Status	Key type	Key spec	Key usage
<input checked="" type="checkbox"/>	MyKMSKey	9f044dc7-f447-42a3-8b86-dc137d74dcd7	Enabled	Symmetric	SYMMETRIC_D...	Encrypt and de...

TAREA 2:

En esta tarea debíamos configurar el file Server instance, de la instancia EC2 e instalar AWS encryption CLI, el cual puedes usar para ejecutar comandos de cifrado y descifrado. Necesitamos hacer esto para poder utilizar la clave que hicimos anteriormente en AWS KMS.

- 1-Fuimos a las instancias EC2 que teníamos, seleccionamos file server y nos conectamos. Elegimos Session Manager para conectarnos. Al hacer esto nos apareció un CLI.
 - 2-Creamos un archivo que se llama AWS credential, y le escribimos ciertos parámetros.
 - 3-En AWS credential, pegamos un bloque de código que estaba en AWS CLI en los detalles de nuestro laboratorio. Guardamos el archivo
 - 4-Por ultimo teníamos que instalar AWS Encryption CLI y exportar el path.
-

Aquí podemos ver la instancia a la que conectamos:



The screenshot displays the AWS Management Console interface. On the left, the navigation menu includes 'EC2 Dashboard', 'EC2 Global View', 'Events', 'Instances', 'Instance Types', 'Launch Templates', 'Spot Requests', 'Savings Plans', 'Reserved Instances', 'Dedicated Hosts', 'Capacity Reservations', 'Images', and 'Elastic Block Store'. The 'Instances' section is expanded, showing a list of instances. The table has columns for Name, Instance ID, Instance state, Instance type, Status check, Alarm status, and Availability Zone. One instance, 'File Server' with ID 'i-03a9e22b6d1949cf5', is shown in a 'Running' state. A red arrow points to the 'Connect' button in the top right of the instance list. Below the table, the details for the selected instance are shown, including the Instance ID, Public IPv4 address (54.202.177.199), Private IPv4 addresses (10.0.2.62), Instance state (Running), and Public IPv4 DNS (ec2-54-202-177-199.us-west-2.compute.amazonaws.com).

Instances (1/1) Info

Find Instance by attribute or tag (case-sensitive)

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone
File Server	i-03a9e22b6d1949cf5	Running	t2.micro	2/2 checks passed	No alarms	us-west-2a

Instance: i-03a9e22b6d1949cf5 (File Server)

Details | Security | Networking | Storage | Status checks | Monitoring | Tags

Instance summary Info

Instance ID i-03a9e22b6d1949cf5 (File Server)	Public IPv4 address 54.202.177.199 open address	Private IPv4 addresses 10.0.2.62
IPv6 address -	Instance state Running	Public IPv4 DNS ec2-54-202-177-199.us-west-2.compute.amazonaws.com open address

© 2023, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

Aquí configuramos por primera vez aws credentials :

Session ID:

Instance ID: i-03a9e22b6d1949cf5

user2846438=Guillermina_Antonaccio_Scaffo-
03c30888848a07c60

```
sh-4.2$ cd
sh-4.2$ cd ~
sh-4.2$ aws configure
AWS Access Key ID [None]: 1
AWS Secret Access Key [None]: 1
Default region name [None]: us-west-2
Default output format [None]:
sh-4.2$
```

Luego volvimos a configurar aws credential, lo cambiamos por este código que aparecía en AWS CLI y lo guardamos:

Session ID:

Instance ID: i-03a9e22b6d1949cf5

Terminate

user2846438=Guillermina_Antonaccio_Scaffo-

03c30888848a07c60

[default]

aws_access_key_id=ASIA5KLITD53YQ6ZYZR7F

aws_secret_access_key=4tWr7NkqK+bhWJixAVzhnq5EB/6VTsxVGRbaPRN3

aws_session_token=FwoGZXIvYXZlEMf////////wEaDLBfmaS7++YvjyruGyLOATu7okNnXlhyslWaiUtiVrqiQDevirx2MDwM5ZrL7+y4xLakToVOL+oaRWIYtZJgEJHyDoa0HJldU0uIYZftsJzdtfzYQf33e1yRplaU43eqWMHTaXZa6P6DxvwZD6D75NItKv5twhIQL8IVEKaWbcLdTgJzrdmgn7NB6fsYqU42mPbRTu+QW/R8ABj1qHorAsAxIlygar21so9CYQtNwtgZAIPDkc5TjTTTyOsoLQLo1fkiiIWgkWvTRH7QY6N3Pt7gU9DtVKUMfwejtEmbKJfquKoGMi1R/814fg2zcactQc1xbo2J/BPe3nbK3uID1ktifyq1q9rXQ5x2G2LiC6D77xc=

4,415

All

Así debería verse credentials si mostramos su contenido con el comando cat y aquí podemos ver que instalamos aws-encryption-sdk-cli

Session ID:

Instance ID: i-03a9e22b6d1949cf5

Terminate

user2846438=Guillermina_Antonaccio_Scaffo-
03c30888848a07c60

```
sh-4.2$ cd
sh-4.2$ cd ~
sh-4.2$ aws configure
AWS Access Key ID [None]: 1
AWS Secret Access Key [None]: 1
Default region name [None]: us-west-2
Default output format [None]:
sh-4.2$ vi ~/.aws/credentials
sh-4.2$ cat ~/.aws/credentials
[default]
aws access key id=ASIA5KLITD53YQ6ZYR7F
aws secret access key=4tWr7NkqK+bhWJixAVzhnq5EB/6VTsxVGRbaPRN3
aws session token=FwoGZXIvYXdzEMf////////wEaDLBfmaS7++YvjyruGyLOATu7okNnXlhyslWaiUtiVrqiQDevirx2MDwM5ZrL7+y4xLakToVOL+oarWIYtZJgEJHyDoa0HJldU0uIYZf
tsJzdtFzYQf33elyRplaU43eqWMHTaXZa6P6Dxvw2D6D75NItKv5twhIQL8IVEKaWbcLdTGjzrdmqn7NB6fsYqU42mPbRTu+QW/R8ABjlqHorAsAxIlyqar21so9CYQtNwtgZAIPDkc5TjTTTyOso
LQLolfkiiIWgkWvTRH7QY6N3Pt7gU9DtVKUMfwejtEmbKJfquKoGMi1R/814fg2zcactQclxbo2J/BPe3nbK3uID1ktifyq1q9rXQ5x2G2LiC6D77xc=
sh-4.2$ pip3 install aws-encryption-sdk-cli
Defaulting to user installation because normal site-packages is not writeable
Collecting aws-encryption-sdk-cli
  Downloading aws_encryption_sdk_cli-4.1.0-py2.py3-none-any.whl (44 kB)
    |████████████████████| 44 kB 3.0 MB/s
Collecting attrs>=17.1.0
  Downloading attrs-23.1.0-py3-none-any.whl (61 kB)
    |████████████████████| 61 kB 10.5 MB/s
Collecting aws-encryption-sdk~=3.1
  Downloading aws_encryption_sdk-3.1.1-py2.py3-none-any.whl (99 kB)
    |████████████████████| 99 kB 14.2 MB/s
Collecting base64io>=1.0.1
  Downloading base64io-1.0.3-py2.py3-none-any.whl (17 kB)
Requirement already satisfied: setuptools in /usr/lib/python3.7/site-packages (from aws-encryption-sdk-cli) (49.1.3)
Collecting importlib-metadata; python_version < "3.8"
  Downloading importlib_metadata-6.7.0-py3-none-any.whl (22 kB)
```

Este es el final de la instalación de aws-encryption-sdk-cli

Session ID:

Instance ID: i-08795077ce081ab98

Terminate

user2846438=Guillermo_Antonaccio_Scaffo-

0f6ee2dfa1a899da8

```
Collecting s3transfer<0.8.0,>=0.7.0
```

```
  Downloading s3transfer-0.7.0-py3-none-any.whl (79 kB)
```

```
| 79 kB 14.5 MB/s
```

```
Collecting cffi>=1.12
```

```
  Downloading cffi-1.15.1-cp37-cp37m-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (427 kB)
```

```
| 427 kB 42.9 MB/s
```

```
Collecting typing-extensions>=3.6.4; python_version < "3.8"
```

```
  Downloading typing_extensions-4.7.1-py3-none-any.whl (33 kB)
```

```
Collecting zipp>=0.5
```

```
  Downloading zipp-3.15.0-py3-none-any.whl (6.8 kB)
```

```
Collecting urllib3<1.27,>=1.25.4; python_version < "3.10"
```

```
  Downloading urllib3-1.26.18-py2.py3-none-any.whl (143 kB)
```

```
| 143 kB 49.5 MB/s
```

```
Collecting python-dateutil<3.0.0,>=2.1
```

```
  Downloading python_dateutil-2.8.2-py2.py3-none-any.whl (247 kB)
```

```
| 247 kB 48.7 MB/s
```

```
Collecting pycparser
```

```
  Downloading pycparser-2.21-py2.py3-none-any.whl (118 kB)
```

```
| 118 kB 50.6 MB/s
```

```
Collecting six>=1.5
```

```
  Downloading six-1.16.0-py2.py3-none-any.whl (11 kB)
```

```
Installing collected packages: base64io, jmespath, urllib3, six, python-dateutil, botocore, s3transfer, boto3, pycparser, cffi, cryptography, wrapt, typing-extensions, zipp, importlib-metadata, attrs, aws-encryption-sdk, aws-encryption-sdk-cli
```

```
  WARNING: The script aws-encryption-cli is installed in '/home/ssm-user/.local/bin' which is not on PATH.
```

```
  Consider adding this directory to PATH or, if you prefer to suppress this warning, use --no-warn-script-location.
```

```
Successfully installed attrs-23.1.0 aws-encryption-sdk-3.1.1 aws-encryption-sdk-cli-4.1.0 base64io-1.0.3 boto3-1.28.83 botocore-1.31.83 cffi-1.15.1 cryptography-41.0.5 importlib-metadata-6.7.0 jmespath-1.0.1 pycparser-2.21 python-dateutil-2.8.2 s3transfer-0.7.0 six-1.16.0 typing-extensions-4.7.1 urllib3-1.26.18 wrapt-1.16.0 zipp-3.15.0
```

```
sh-4.2$
```

```
sh-4.2$ export PATH=$PATH:/home/ssm-user/.local/bin
```

```
sh-4.2$
```

TAREA 3:

En esta tarea debíamos cifrar y descifrar datos, creando un archivo de texto con datos confidenciales simulados. Luego utilizamos el cifrado para proteger el contenido del archivo y luego desciframos los datos para ver el contenido del mismo.

- 1-Creamos 3 archivos de texto y en el primero escribimos ";;;TOP SECRET 1!!!"
 - 2-Creamos un directorio llamado output
 - 3-Copiamos y pegamos el KMS ARN que copiamos al principio en una variable llamada keyArn
 - 4-Encriptamos el primer archivo secret1.txt y el archivo se volvió secret1.txt.encrypted, si ejecutábamos `cat secret1` nos aparecía encriptado e ilegible.
 - 5-Luego desciframos el archivo y usando el comando `cat`, comprobamos que podíamos leerlo bien y había sido descifrado con éxito.
-

Aquí podemos ver que creamos 3 archivos de texto y en uno de ellos escribimos TOP SECRET, luego creamos un directorio llamado output, guardamos el ARN que habíamos copiado anteriormente en keyArn y encriptamos el archivo secret1.txt

Session ID:

Instance ID: i-08795077ce081ab98

Terminate

user2846438=Guillermina_Antonaccio_Scaffo-
0f6ee2dfa1a899da8

```
sh-4.2$ ls
config  credentials  secret1.txt  secret2.txt  secret3.txt
sh-4.2$ pwd
/home/ssm-user/.aws
sh-4.2$ cd ..
sh-4.2$ pwd
/home/ssm-user
sh-4.2$ ls
sh-4.2$ touch secret1.txt secret2.txt secret3.txt
sh-4.2$
sh-4.2$ echo 'TOP SECRET 1!!!!' > secret1.txt
sh-4.2$ ls
secret1.txt  secret2.txt  secret3.txt
sh-4.2$ pwd
/home/ssm-user
sh-4.2$ ls
secret1.txt  secret2.txt  secret3.txt
sh-4.2$ mkdir output
sh-4.2$ cd
sh-4.2$ ls
output  secret1.txt  secret2.txt  secret3.txt
sh-4.2$ pwd
/home/ssm-user
sh-4.2$ keyArn=arn:aws:kms:us-west-2:915584262007:key/ce8bd69e-c1f1-4096-8054-34ed2d643ecd
sh-4.2$ aws-encryption-cli --encrypt \
>
usage: aws-encryption-cli [-h] [--version] [-e] [-d] [--decrypt-unsigned] [-S]
                        [--metadata-output METADATA_OUTPUT]
                        [--overwrite-metadata] -w WRAPPING_KEYS
                        [WRAPPING_KEYS ...]
                        [--commitment-policy {forbid-encrypt-allow-decrypt,require-encrypt-allow-decrypt,require-encrypt-require-decrypt}]
```

Una vez que encriptamos el archivo con éxito se ve así:

```
sh-4.2$ aws-encryption-cli --encrypt \  
> --input secret1.txt \  
> --wrapping-keys key=$keyArn \  
> --metadata-output ~/metadata \  
> --encryption-context purpose=test \  
> --commitment-policy require-encrypt-require-decrypt \  
> --output ~/output/.  
sh-4.2$ echo $?  
0  
sh-4.2$ ls output  
secret1.txt.encrypted  
sh-4.2$ cd output  
sh-4.2$ cat secret1.txt.encrypted  
x000n'000H00]00R00S000u  
naws-crypto-public-keyDAmdduEEh/itJmqTwOIDLRDm30nQiw7T87wFQjen9T8ryrmIXxdi0edNZlGLcxMgs6Q==purposetestaws-kmsKarn:aws:kms  
F00[jest-2:915584262007:key/11fb3c33-7507-4869-8b0e-0755492927880x00w00,{  
0o0m0hL0`0He.00Yz0`j00]0B801UL~0|*0H00  
0nI9r000@0K0;00<Ea00M0W00A00S0C00]0Q0B0040000xcq6600w00K0(0X+!I00(+[<0000(;tl000^!q0000.0ip0700!g\0000000005?'0Zg0}0|8  
x0h00*0R00M0g0e10s0I`2000A000070R0(S0f00nJws`07C0000O$00cf0&kw0n|s0ez  
-0r.{00005K0!Q0}BU0"zh0?r.0L00=Qsh-4.2$
```

Cuando desciframos el archivo de texto, logramos que se vuelva legible con éxito

Session ID:


Instance ID: i-001602de7742cecc4

Terminate

user2846438=Guillermina_Antonaccio_Scaffo-

01da0ba8c48d7e5b2

```
> --metadata-output ~/metadata \
> --encryption-context purpose=test \
> --commitment-policy require-encrypt-require-decrypt \
> --output ~/output/.
sh-4.2$ echo $?
0
sh-4.2$ ls output
secret1.txt.encrypted
sh-4.2$ cd output
sh-4.2$ cat secret1.txt.encrypted
x0000n'0000H00[]000R0S0000u
                                naws-crypto-public-keyDAmdduEEh/itJmqTwOIDLRDm30nQiw7T87wFQjen9T8ryrmIXxdi0edN2lGLcxMgs6Q==purposetestaws-kmsKarn:aws:kms
F00[jest-2:915584262007:key/11fb3c33-7507-4869-8b0e-0755492927880x00w00,{0
0o0m0hL0`0He.000Yz0`j00]0B801UL~0|    *0H00
    0nI9r0000K0;00<Ea    0M0W000A000S0C0    00]0Q0B0040000xcq66000w00K0(0X+!I00(+<0000(;tl000^!q0000.0ip    0700lg\000000n05?'0Zg    0}08
x0h00*0R00M    g0e10s0I`2000A000070R0(S0f00n0ws`07C0000O$00cf0&kw0n|s0ez
-0r.{0005K0!Q0}BU0"zh0?r.0L00=Qsh-4.2$
sh-4.2$ aws-encryption-cli --decrypt \
> --input secret1.txt.encrypted \
> --wrapping-keys key=$keyArn \
> --commitment-policy require-encrypt-require-decrypt \
> --encryption-context purpose=test \
> --metadata-output ~/metadata \
> --max-encrypted-data-keys 1 \
> --buffer \
> --output .
sh-4.2$ ls
secret1.txt.encrypted secret1.txt.encrypted.decrypted
sh-4.2$ cat secret1.txt.encrypted.decrypted
TOP SECRET 1!!!
sh-4.2$ Yara Villar Felipe Putrelli Guillermo San Martin Guillermina Antonaccio
```



Aquí termina el
laboratorio, muchas
gracias