

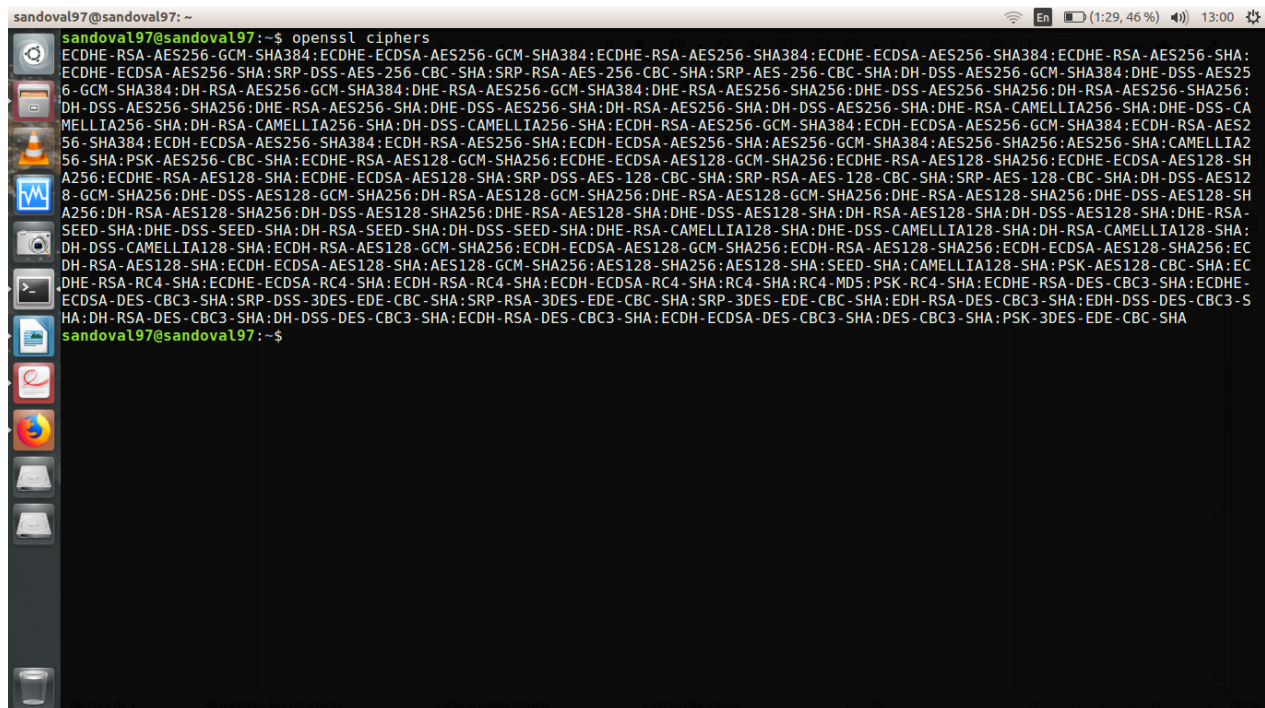
## 1. Instalar OpenSSL

- Obtener OpenSSL para Windows o Linux. En caso de hacerlo para Windows, hay
- que definir el path agregando el nuevo elemento a la variable PATH:
  1. Path c:\openssl\bin.

## 2. Utilización de algoritmos simétricos

### 2.1. Comando ciphers

Para comprobar la lista de algoritmos simétricos soportados ejecutar: **openssl ciphers**

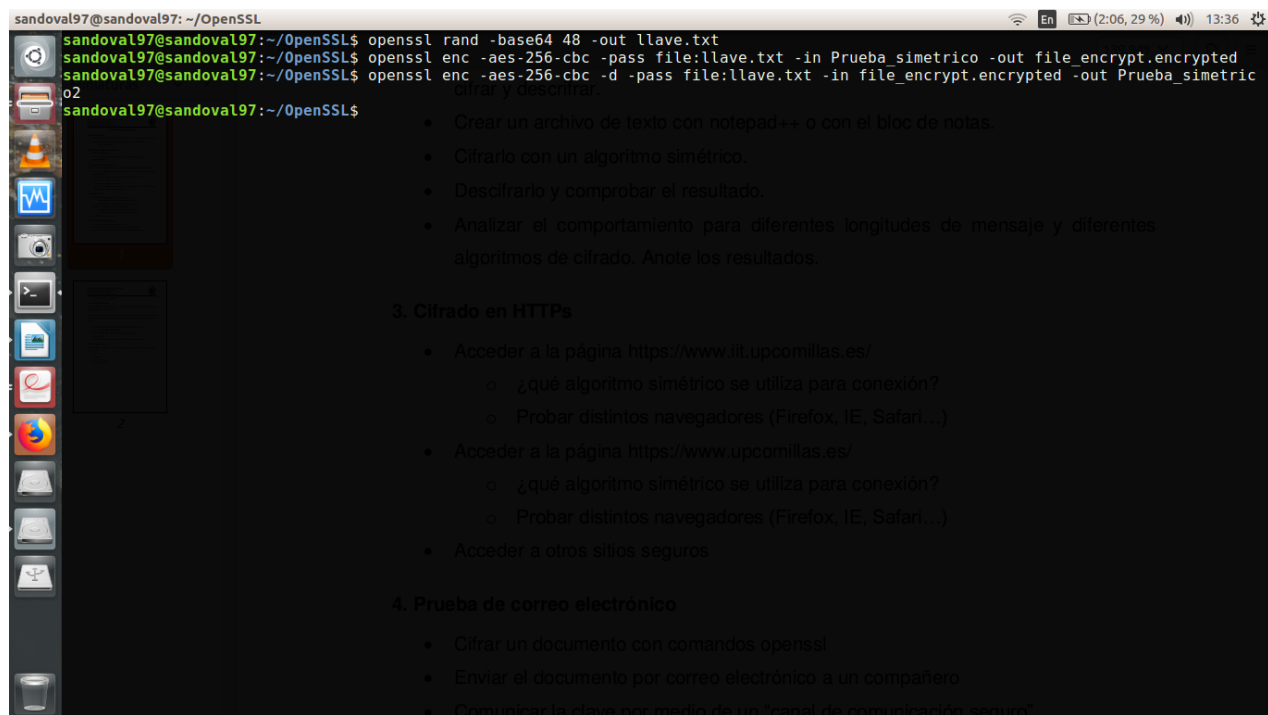


```
sandoval97@sandoval97: ~$ openssl ciphers
ECDHE-RSA-AES256-GCM-SHA384:ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-SHA:
ECDHE-ECDSA-AES256-SHA:SRP-DSS-AES-256-CBC-SHA:SRP-RSA-AES-256-CBC-SHA:SRP-AES-256-CBC-SHA:DH-DSS-AES256-GCM-SHA384:DHE-DSS-AES25
6-GCM-SHA384:DH-RSA-AES256-GCM-SHA384:DHE-RSA-AES256-GCM-SHA384:DHE-RSA-AES256-SHA256:DHE-DSS-AES256-SHA256:DH-RSA-AES256-SHA256:
DH-DSS-AES256-SHA256:DHE-RSA-AES256-SHA:DHE-DSS-AES256-SHA:DH-RSA-AES256-SHA:DH-DSS-AES256-SHA:DHE-RSA-CAMELLIA256-SHA:DHE-DSS-CA
MELLIA256-SHA:DH-RSA-CAMELLIA256-SHA:DH-DSS-CAMELLIA256-SHA:ECDH-RSA-AES256-GCM-SHA384:ECDH-ECDSA-AES256-GCM-SHA384:ECDH-RSA-AES25
6-SHA384:ECDH-ECDSA-AES256-SHA384:ECDH-RSA-AES256-SHA:ECDH-ECDSA-AES256-SHA:AES256-GCM-SHA384:AES256-SHA256:AES256-SHA:CAMELLIA2
56-SHA:PSK-AES256-CBC-SHA:ECDH-RSA-AES128-GCM-SHA256:ECDH-ECDSA-AES128-GCM-SHA256:ECDH-RSA-AES128-SHA256:ECDH-ECDSA-AES128-SH
A256:ECDH-RSA-AES128-SHA:ECDH-ECDSA-AES128-SHA:SRP-DSS-AES-128-CBC-SHA:SRP-RSA-AES-128-CBC-SHA:SRP-AES-128-CBC-SHA:DH-DSS-AES12
8-GCM-SHA256:DHE-DSS-AES128-GCM-SHA256:DH-RSA-AES128-GCM-SHA256:DHE-RSA-AES128-GCM-SHA256:DHE-RSA-AES128-SHA256:DHE-DSS-AES128-SH
A256:DH-RSA-AES128-SHA256:DH-DSS-AES128-SHA256:DHE-RSA-AES128-SHA:DHE-DSS-AES128-SHA:DH-RSA-AES128-SHA:DH-DSS-AES128-SHA:DHE-RSA-
SEED-SHA:DHE-DSS-SEED-SHA:DH-RSA-SEED-SHA:DH-DSS-SEED-SHA:DHE-RSA-CAMELLIA128-SHA:DHE-DSS-CAMELLIA128-SHA:DH-RSA-CAMELLIA128-SHA:
DH-DSS-CAMELLIA128-SHA:ECDH-RSA-AES128-GCM-SHA256:ECDH-ECDSA-AES128-GCM-SHA256:ECDH-RSA-AES128-SHA256:ECDH-ECDSA-AES128-SHA256:EC
DH-RSA-AES128-SHA:ECDH-ECDSA-AES128-SHA:AES128-GCM-SHA256:AES128-SHA256:AES128-SHA:SEED-SHA:CAMELLIA128-SHA:PSK-AES128-CBC-SHA:EC
DHE-RSA-RC4-SHA:ECDH-ECDSA-RC4-SHA:ECDH-RSA-RC4-SHA:ECDH-ECDSA-RC4-SHA:RC4-SHA:RC4-MD5:PSK-RC4-SHA:ECDH-RSA-DES-CBC3-SHA:ECDH-
ECDSA-DES-CBC3-SHA:SRP-DSS-3DES-EDE-CBC-SHA:SRP-RSA-3DES-EDE-CBC-SHA:SRP-3DES-EDE-CBC-SHA:EDH-RSA-DES-CBC3-SHA:EDH-DSS-DES-CBC3-S
HA:DH-RSA-DES-CBC3-SHA:DH-DSS-DES-CBC3-SHA:ECDH-RSA-DES-CBC3-SHA:ECDH-ECDSA-DES-CBC3-SHA:DES-CBC3-SHA:PSK-3DES-EDE-CBC-SHA
```

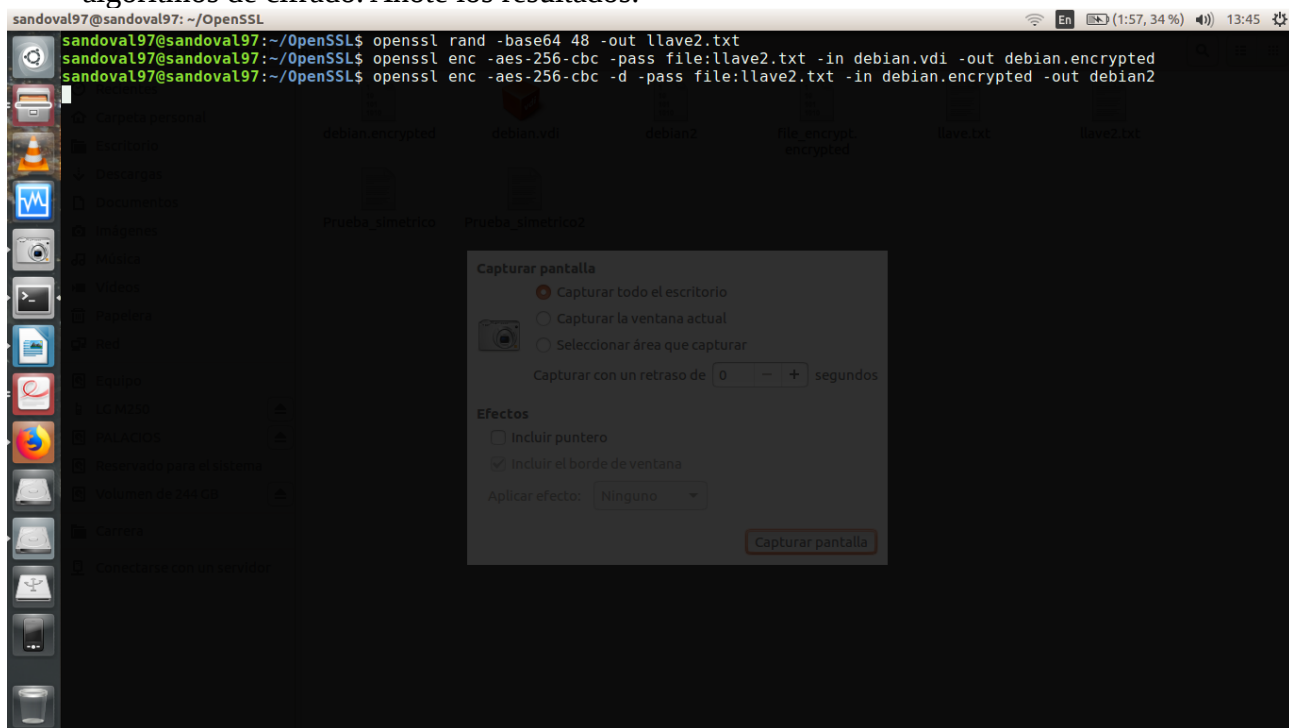
### 2.2. Cifrar un mensaje de prueba

Seguir los siguientes pasos para crear un documento de texto y cifrarlo con openssl.

- Mirar en la documentación de OpenSSL cómo se utiliza el comando enc para
- cifrar y descifrar.
- Crear un archivo de texto con notepad++ o con el bloc de notas.
- Cifrarlo con un algoritmo simétrico.
- Descifrarlo y comprobar el resultado.



- Analizar el comportamiento para diferentes longitudes de mensaje y diferentes algoritmos de cifrado. Anote los resultados.

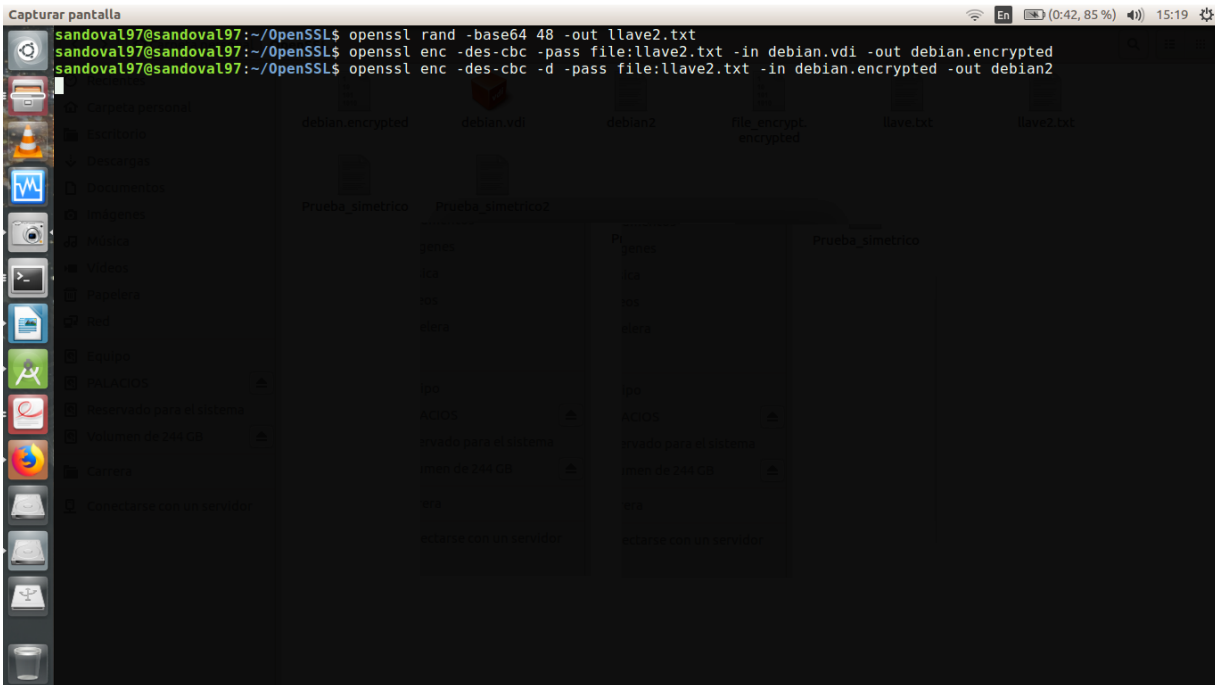


## Analisis:

Con el comando enc de la herramienta open ssl podemos cifrar y descifrar archivos por medio de una clave que tanto el emisor como el receptor deben de conocer, para descifrar es casi el mismo comando nada mas que especificamos -d para indicarle al comando enc que lo que haremos sera descifrar el mensaje o archivo, con el parametro -in indicamos el archivo de entrada a cifrar o descifrar y con el

parametro -out indicamos el nombre que le daremos al archivo cifrado o descifrado (para un archivo descifrado su extension debe de ser .encrypted).

## DES



### Analisis:

Con este algoritmo podemos observar que tanto el descifrado como el cifrado se demora mas en hacer su funcion.

## 3. Cifrado en HTTPs

1. Acceder a la página <https://www.iit.upcomillas.es/>  
¿qué algoritmo simétrico se utiliza para conexión?  
**R:** utiliza AES\_128\_GCM\_SHA256  
Probar distintos navegadores (Firefox, IE, Safari...)
2. Acceder a la página <https://www.upcomillas.es/>  
¿qué algoritmo simétrico se utiliza para conexión?  
**R:** utiliza AES\_256\_GCM\_SHA384  
Probar distintos navegadores (Firefox, IE, Safari...)
3. Acceder a otros sitios seguros

## GITLAB

Pagina de repositorios, informacion y gestiones para desarrolladores, esta pagina utiliza AES\_128\_GCM\_SHA256

## WIKIPEDIA

Página de información y gestiones para estudiantes o apasionados por la lectura informativa, esta página utiliza CHACHA20\_POLY1305\_SHA256

## MEDIUM

Página de información y gestiones para desarrolladores, esta página utiliza TLS\_AES\_128\_GCM\_SHA256

## 4. Prueba de correo electrónico

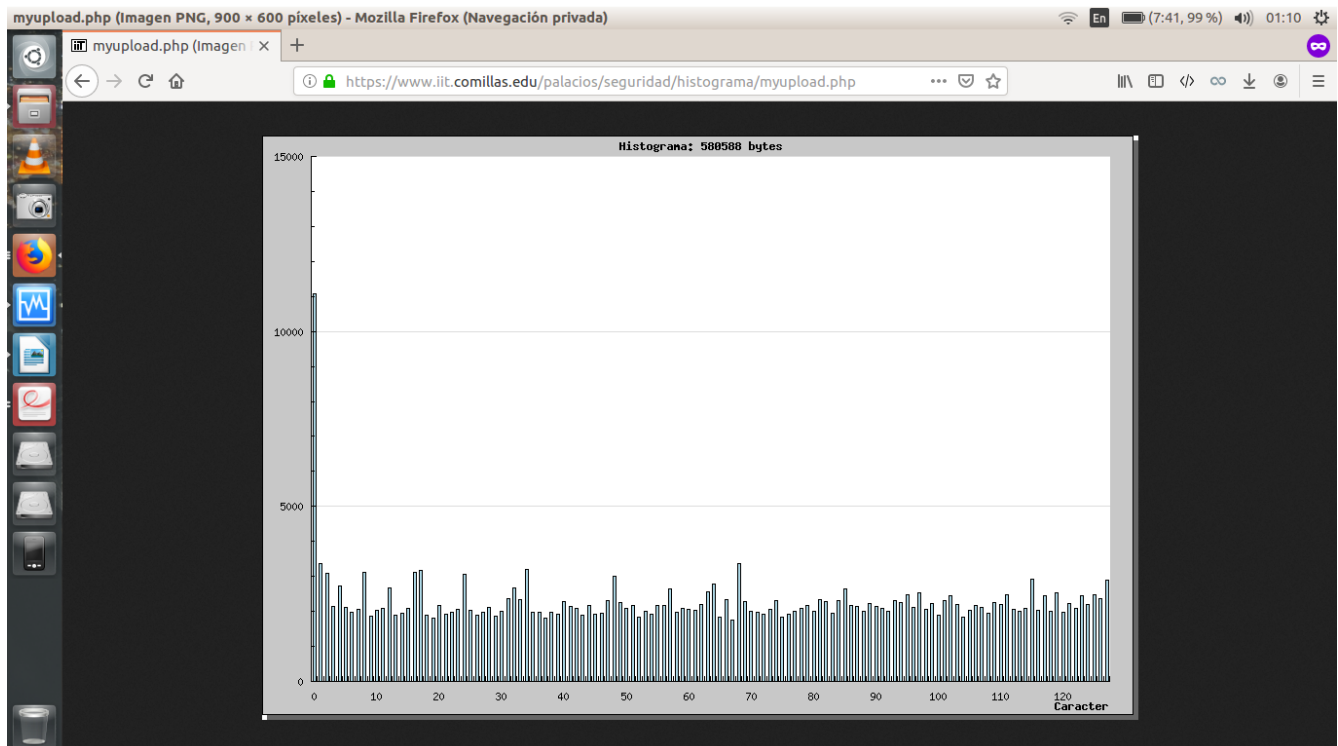
- Cifrar un documento con comandos openssl
- Enviar el documento por correo electrónico a un compañero
- Comunicar la clave por medio de un “canal de comunicación seguro”
- Descifrar el mensaje recibido.

## 5. Análisis por histograma

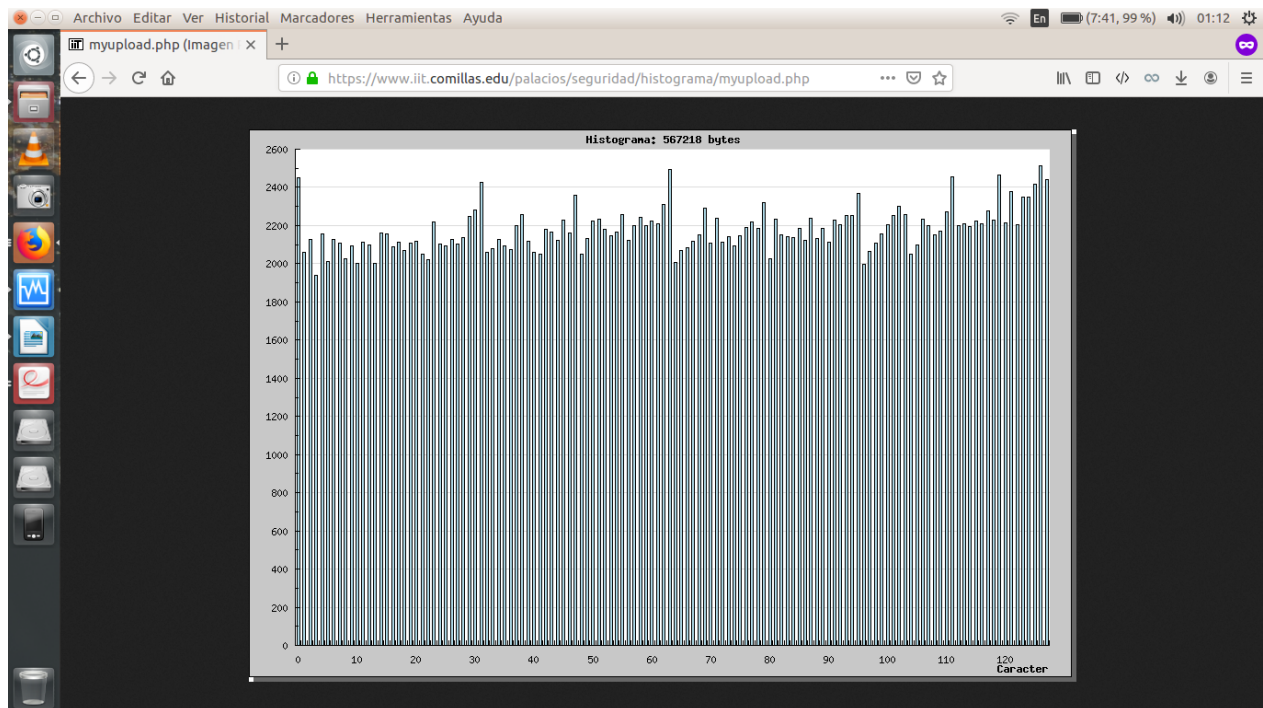
El histograma de un fichero representa, mediante un gráfico de barras, la frecuencia de repetición de cada carácter del fichero.

Utilizar la página <http://www.iit.upcomillas.es/palacios/seguridad/histograma/> para obtener vía web el histograma de cualquier fichero, y hacer pruebas con los siguientes tipos de fichero:

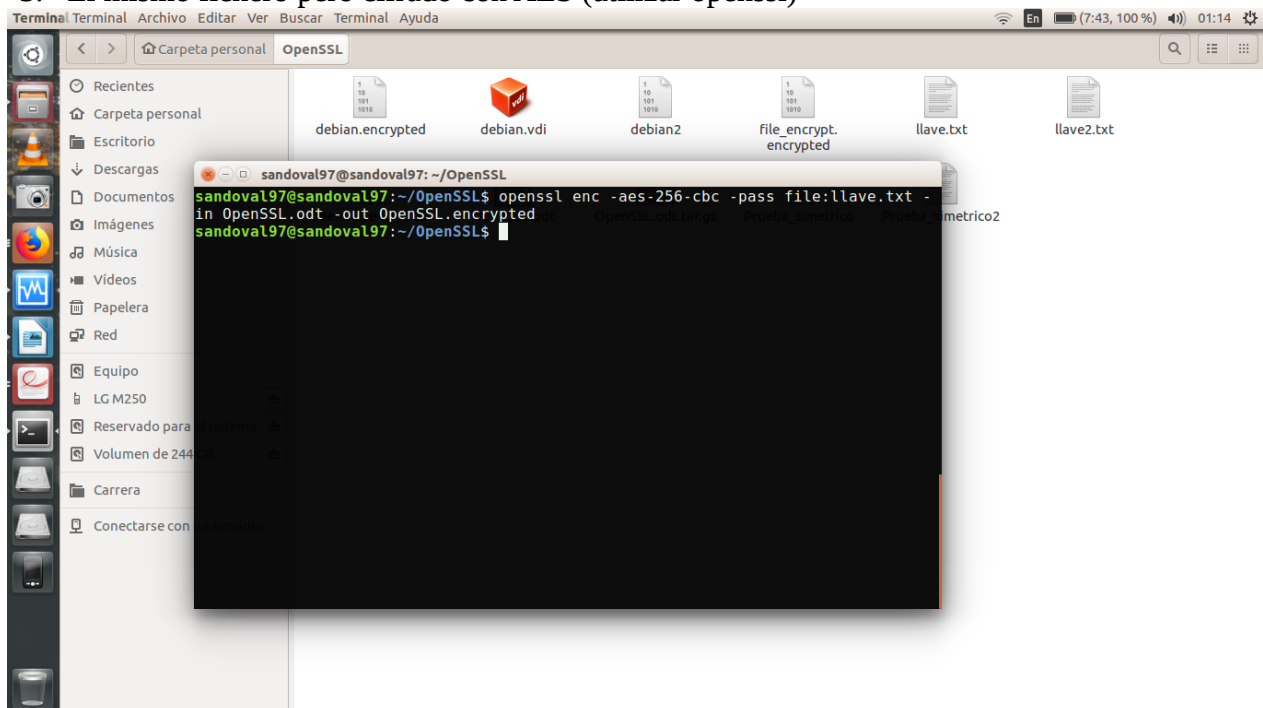
1. Fichero de texto (ejemplo correo electrónico o documento word)

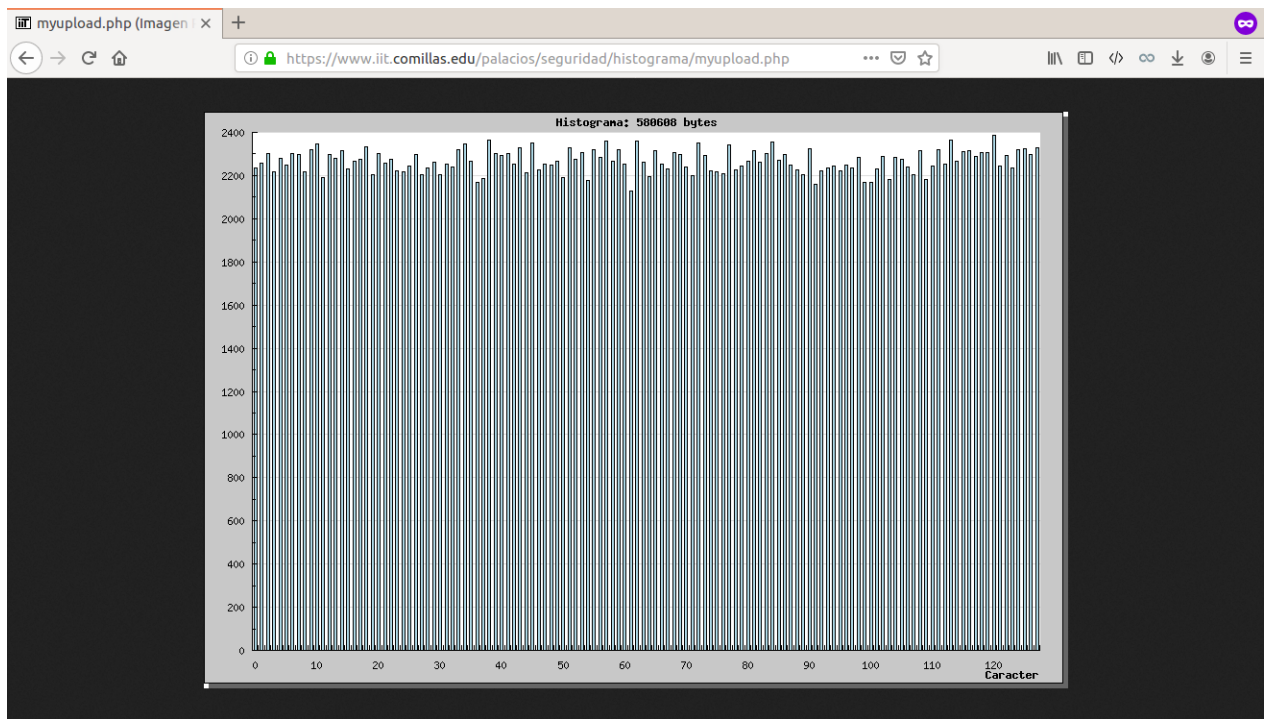


## 2. El mismo fichero pero comprimido

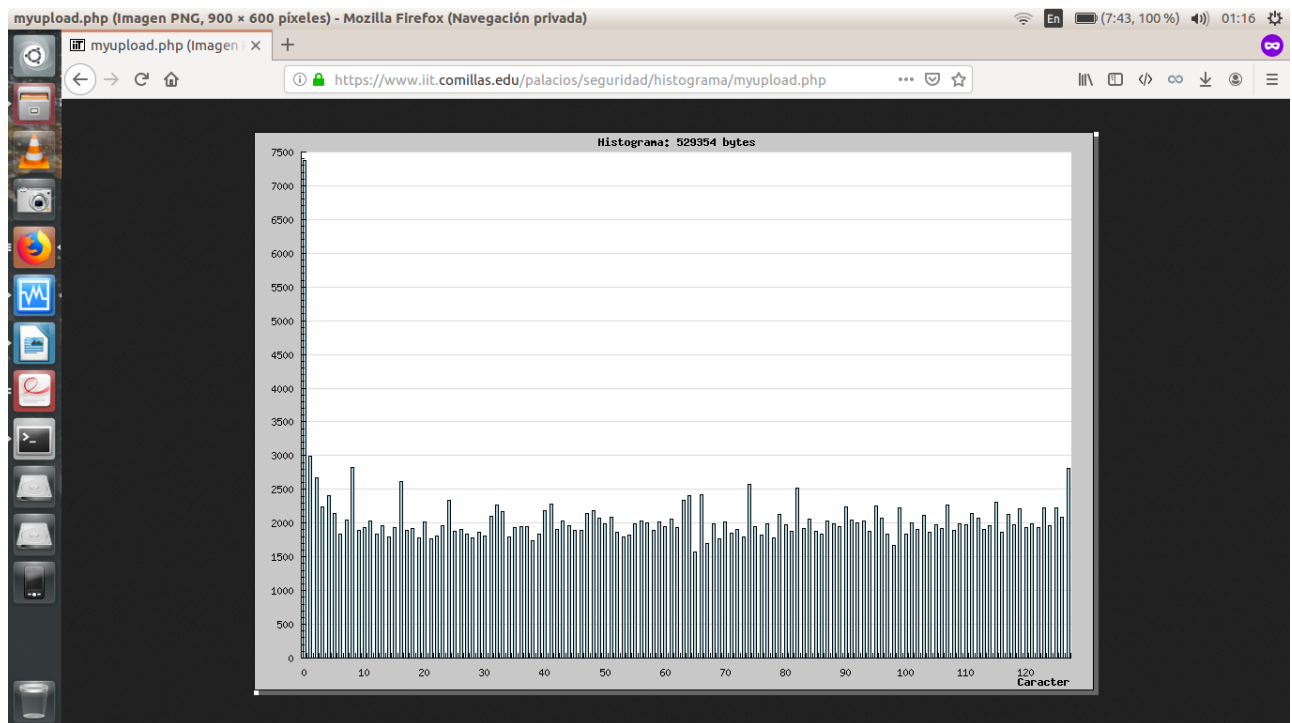


## 3. El mismo fichero pero cifrado con AES (utilizar openssl)

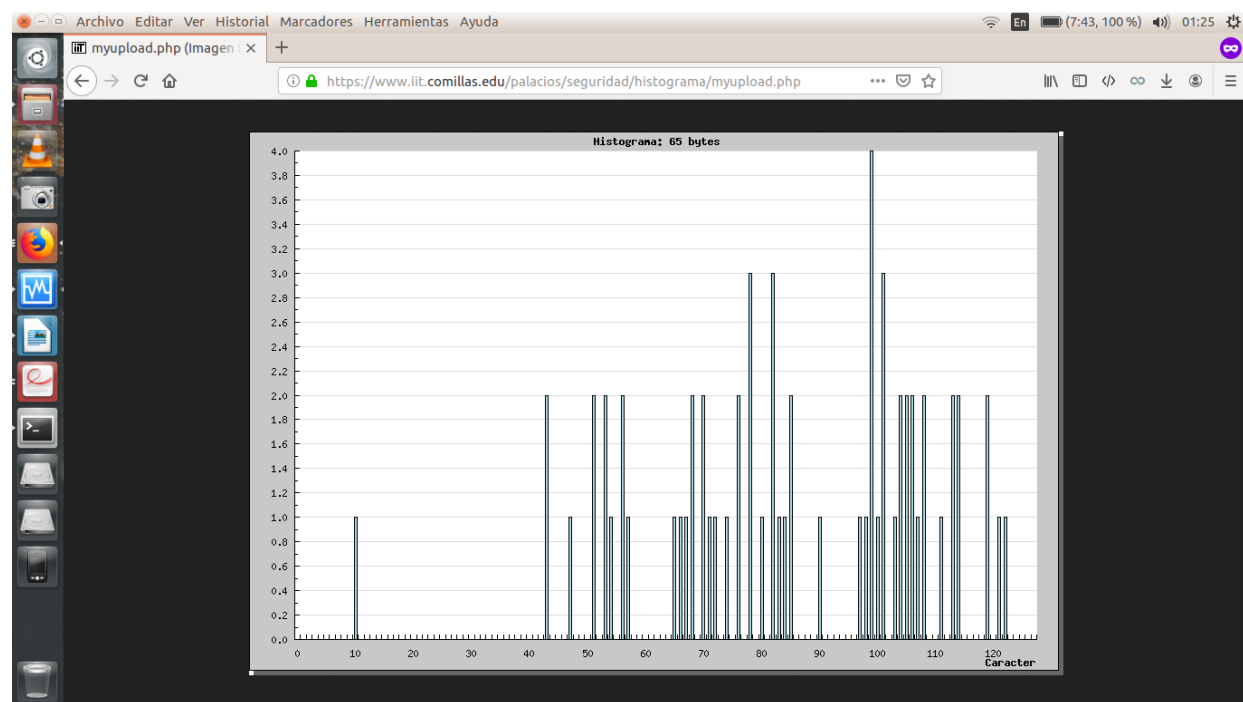
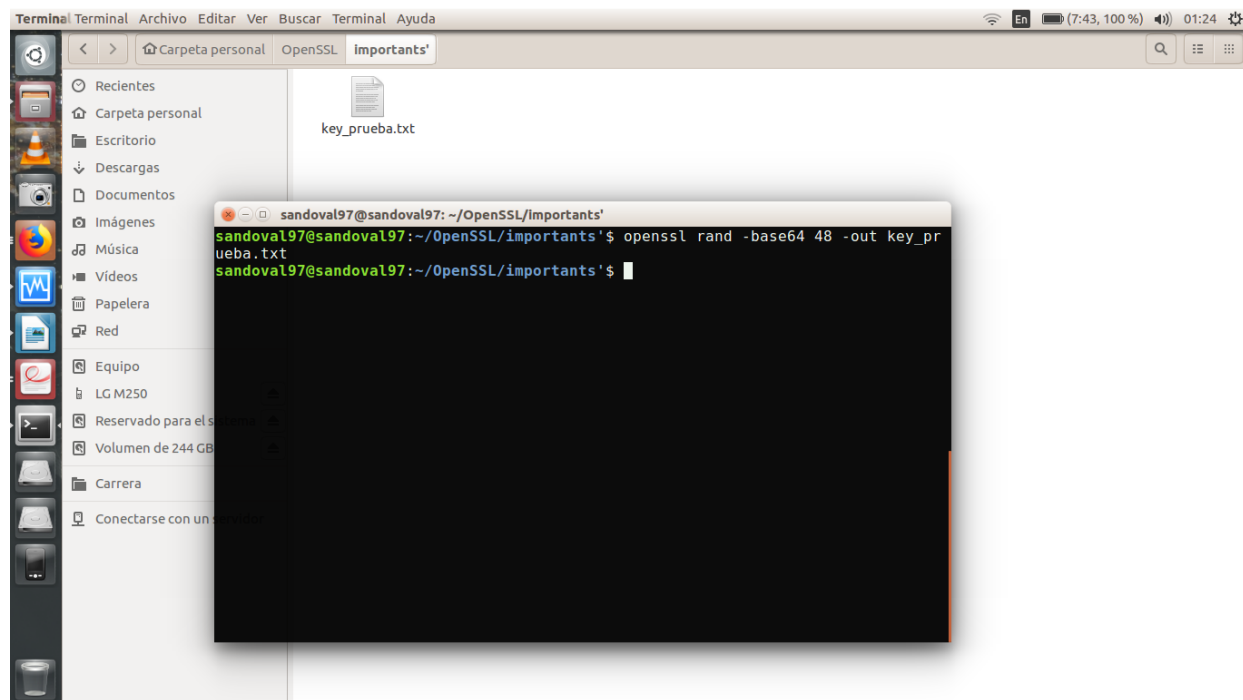




#### 4. Imagen JPEG



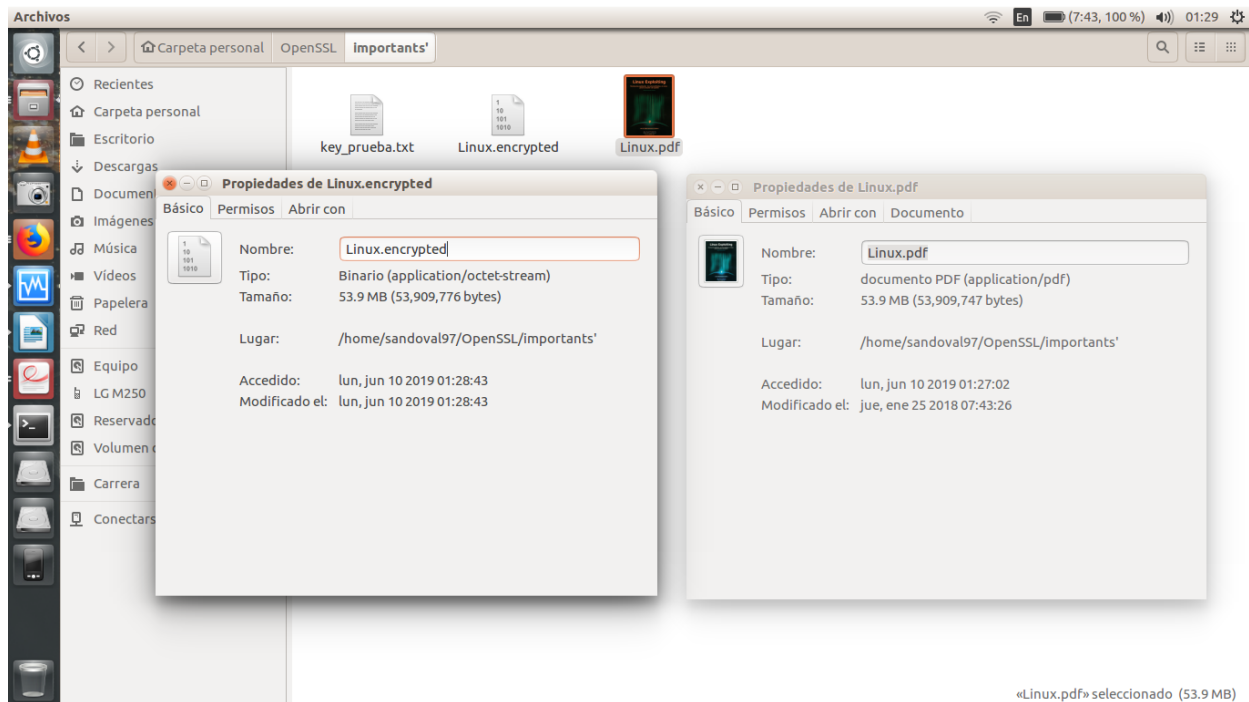
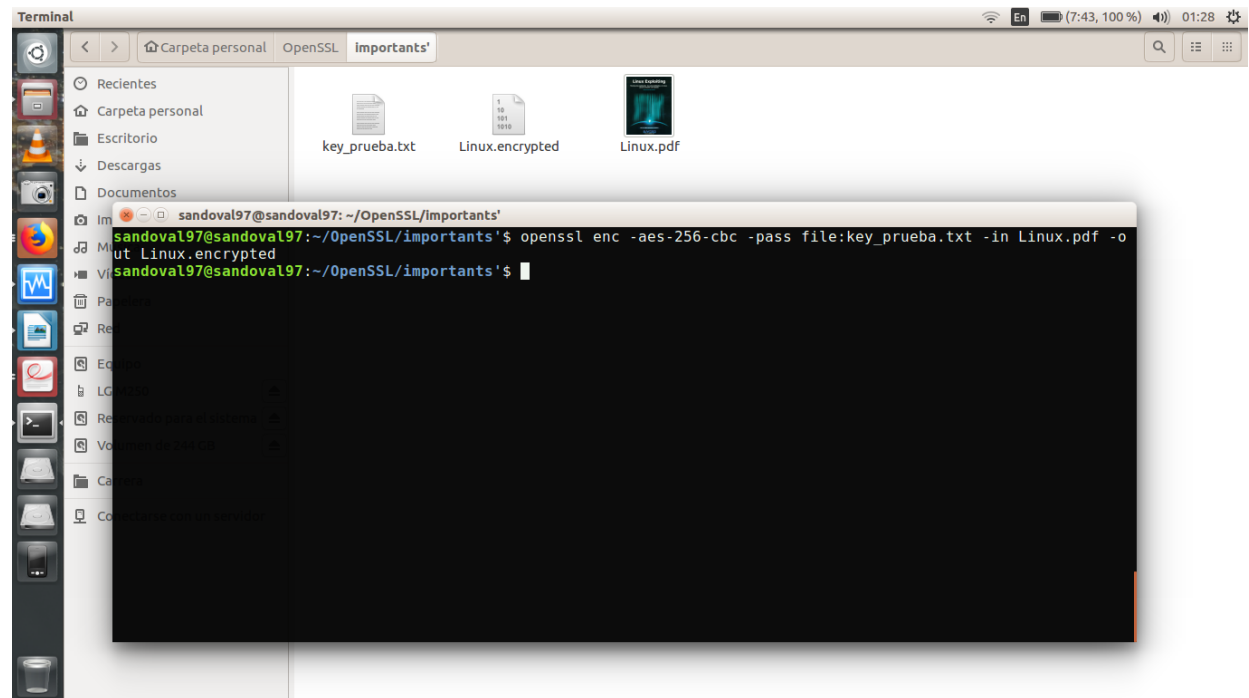
## 5. Fichero aleatorio generado mediante openssl con el comando rand



### 5.1. Tamaño de ficheros

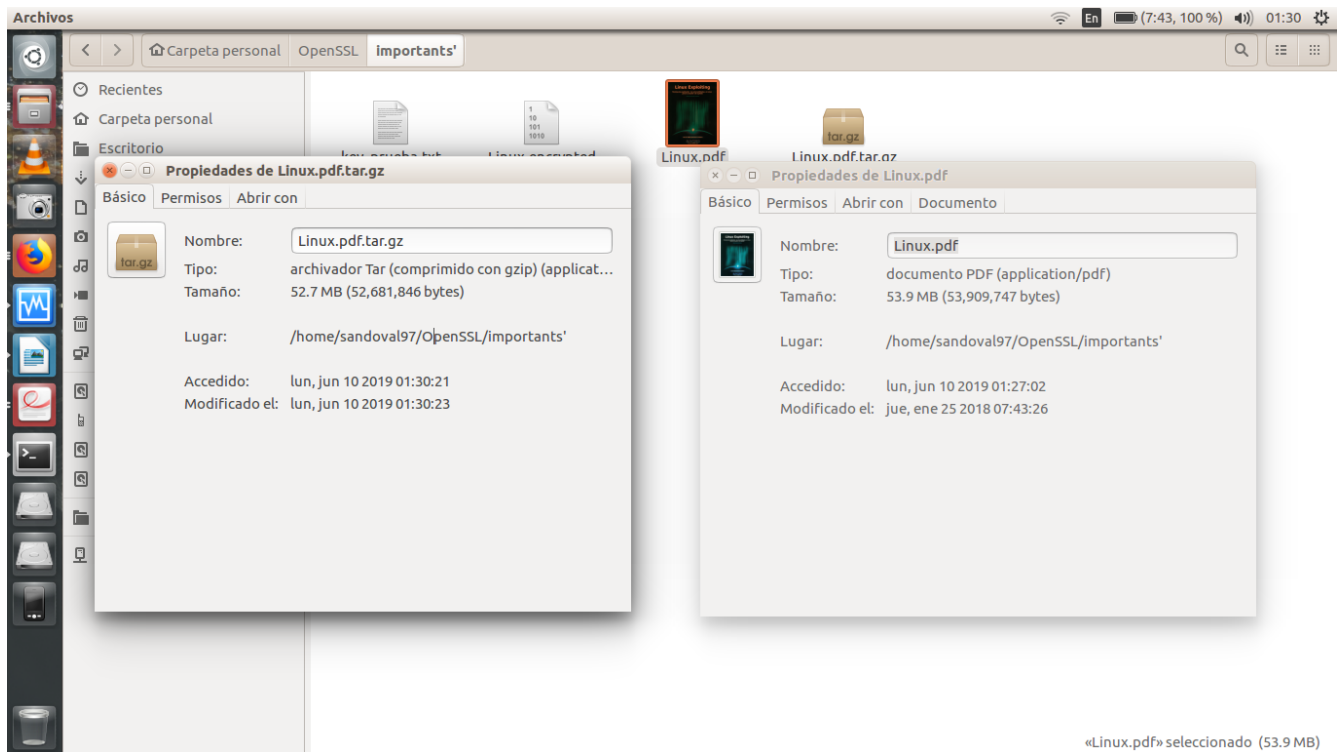
Anotar el tamaño original de un fichero (Ej. Documento de texto ó html largo) y el tamaño resultante de:

## 1. Cifrar

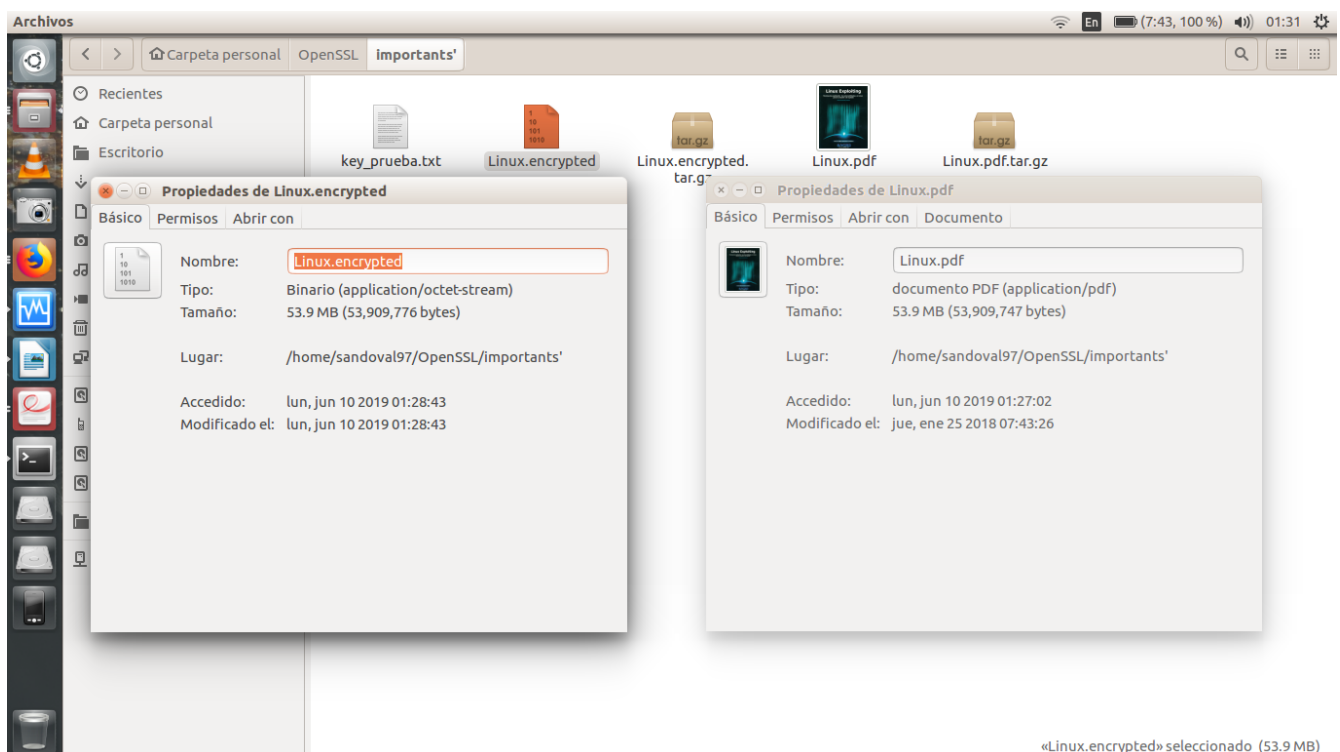




## 2. Comprimir



## 3. Cifrar + comprimir



#### 4. Comprimir + cifrar

