**Cifrado asimétrico**

**Generar un par de claves (pública – privada)**

**Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente**

openssl genrsa -out clave.key 4096

copiar la public key a un fichero aparte

openssl rsa -in clave.key -pubout -out public.key

listar curvas

openssl ecparam -list\_curves

Create private key using the P-224 elliptic curve

openssl ecparam -name secp224k1 -genkey -out ecpriv.key

**Uso del cifrado asimétrico para enviar mensajes confidenciales**

hazle llegar a U la clave correcta de K que permita a U enviar mensajes que solo K pueda

descifrar

1-generas par de claves en kali y ubuntu, y sacas las publicas a un fichero aparte

openssl genrsa -out clave.key 4096

copiar la public key a un fichero aparte

openssl rsa -in clave.key -pubout -out public.key

2- le pasas la public key de kali a ubuntu mediante:

cp kali\_public.key /shared/

3- pasas la clave a ubuntu mediante

cp /shared/kali\_public.key .

4- en ubuntu, encriptas un txt usando la clave publica de kali, mediante

openssl rsautl -encrypt -inkey kali\_public.key -pubin -in hola.txt -out hola\_encriptado.txt

5- envias el fichero encriptado a kali mediante

cp hola\_encriptado.txt /shared/

6- en kali,

cp /shared/hola\_encriptado.txt .

7- desencriptas el fichero con la private key de kali (usamos pkeyutl porque rsautl esta deprecated)

openssl pkeyutl -decrypt -inkey kali\_key.key -in hola\_encriptado.txt -out hola\_desencriptado.txt

**Combinando cifrado simétrico y asimétrico**

**Usar un KDF como contraseña para un cifrado simétrico**

openssl enc -pbkdf2 -aes-128-ecb -k <clave en texto inicial> -P > clave.priv (la mete en un fichero)

**encriptar el mensaje**

openssl enc -aes-128-ecb -in mensaje.txt -out mensajeenc.enc -pass file:./clave.priv

**desencriptarlo**

openssl enc -d -aes-128-ecb -in mensajeenc.enc -out mensajedesenc.txt -pass file:./clave.priv

**Entender cómo funciona en la vida real el cifrado asimétrico**

descargas un pdf de mas de un mb

lo cifras con la clave del ej anterior

openssl enc -aes-128-ecb -in PPT.pdf -out pdfenc.enc -pass file:./clave.priv

cifras la clave.priv con la publica de kali (desde ubuntu)

openssl rsautl -encrypt -inkey kali\_public.key -pubin -in clave.priv -out clavepriv.enc

enviar clave encriptada y pdf encriptado a kali

cp clavepriv.enc /shared/

cp pdfenc.enc /shared/

(en kali) coger los ficheros

cp /shared/clavepriv.enc .

cp /shared/pdfenc.enc .

(en kali) desencriptar la clave con la private key de kali

openssl pkeyutl -decrypt -inkey kali\_key.key -in clavepriv.enc -out clavedesenc.priv

(en kali) desencriptar el pdf con la clave

openssl enc -aes-128-ecb -in pdfenc.enc -out pdfdesenc.pdf -pass file:./clavedesenc.priv

**Integridad y firmas digitales**

**Uso de cifrado autenticado (Authenticated Encryption o AE)**

Abrir cyberchef

El algoritmo necesita una clave, usaré C516DB64096C37093C9297802F7A8AC4

También necesita un IV:  openssl rand -hex 16, yo usaré 00b72b925e77bc11c30b32ed55c2205e

En cyberchef arrastras aes al récipe y rellenas los datos

El input mejor introducirlo raw y el output seleccionarlo en hx

Ahora cifro el resultado y lo copio. Me da esto para la cadena Carmen:

61035cf6457d

Tag: 09f255e6194935f69f83ee4d5add6f28

Ahora desciframos con cyberchef

Arrastrar cuadro correspondiente, acordarse de poner la tag donde pone GCM tag

**Uso de firmas digitales para integridad**

Necesitamos un par de claves pública y privada del emisor (las de antes)

Comando para firmar un fichero cualquiera y escribir la firma a un fichero independiente

openssl dgst -sha256 -sign private\_key.pem -out firma.data mensaje.txt

Esto lo hice en Kali, así que el receptor es Ubuntu

Hay que enviar al receptor el mensaje y la firma

O sea, hay que enviar a Ubuntu el fichero original (mensaje), el fichero firma.data y si no lo tiene ya, la clave pública de kali

Para verificar que el mensaje ha llegado sin alterar

openssl dgst -sha256 -verify kalipublickey.key -signature firma.data mensaje.txt

Debería dar la siguiente salida: Verified OK

Ahora, modificamos el fichero y lo volvemos a validar.

Salida: Verification Failure

**Firmar un PDF con un certificado autogenerado**

Crear un certificado .crt

openssl req -newkey rsa:2048 -nodes -keyout kali.key -x509 -days 365 -out cert.crt

Este certificado no nos vale para firmar porque no tiene la calve pública y privada del emisor

Para firmar necesitamos un .pfx

Al crear el certificado antes, se generó un .crt y un fichero con la clave privada, en este caso kali.key

Para crear el .pfx:

openssl pkcs12 -export -out <nombre para el fichero PFX>.pfx -inkey <fichero de la clave privada que generaste al crear el ctr> -in <certificado ctr>

Se pide una contraseña, da cualquiera

Si haces doble clic en el certificado te debería dejar importarlo, a mi no me deja

Ahora vamos a abobe reader

Buscamos la opción utilizar un certificado

Elegimos un pdf y le damos a firmar digitalmente

**Los conceptos de Confidencialidad, Integridad y Autenticación (CIA) en tu día a día**

**SSH sin contraseña**

Comprobar si existen claves ssh ya

ls -l ~/.ssh/id\_\*.pub

Generate a new 4096 bits SSH key pair with your email address

ssh-keygen -t rsa -b 4096 -C "your\_email@domain.com"

Para verificar si se ha creado:

ls ~/.ssh/id\_\*

Copiamos la public key al servidor remoto

ssh-copy-id remote\_username@server\_ip\_address

Para probarlo, tratamos de conectarnos por ssh

ssh remote\_username@server\_ip\_address

**Comprobación manual de la firma de un programa descargado**

**Marcas de agua**

Marcar una imagen:

sudo apt install imagemagick

Texto corto, marca de agua grande:

convert -density 150 -fill "rgba(255,0,0,0.25)" - gravity Center -pointsize 80 -draw "rotate -45 text 0,0 "

Marca de agua de dos líneas:

convert -density 150 -fill "rgba(255,0,0,0.50)" -pointsize 15 -draw "rotate -15 text 0,200 ''" -draw "rotate -15 text -25,260 ''"

**Metadatos**

sudo apt install exiftool

Ver metadatos: exiftool <imagen>

Eliminar metadatos: exiftool -all= <fichero>