UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

Actividades prácticas: Criptografía

MATERIAL DIDÁCTICO

Que para obtener el título de

Ingeniera en Computación

P R E S E N T A

Patricia Nallely Gómez Flores

ASESORA DE MATERIAL DIDÁCTICO

M.C. María Jaquelina López Barrientos

Ciudad Universitaria, Cd. Mx., 2025

Tabla de Contenido

Índice de Figuras

Índice de Tablas

Capítulo I

Introducción

La criptografía se define como «*la ciencia encargada de transformar la información de forma tal que se vuelva incomprensible para todo aquel que no tenga la autorización correspondiente para acceder a ella*» (López Barrientos, M.J., 2016). Por esta razón, es fundamental comprender la importancia que la criptografía tiene para la computación y la seguridad que proporciona.

La criptografía tiene cuatro objetivos principales:

- Confidencialidad: asegurar que la información esté disponible solo para los usuarios autorizados.
- Integridad: garantizar que la información no haya sido alterada.
- Autenticación: verificar la autenticidad de la información o la identidad de un usuario.
- No repudio: evitar que un usuario niegue compromisos o acciones previas.

La criptografía utiliza varios algoritmos criptográficos de bajo nivel para alcanzar uno o más de dichos objetivos de seguridad de la información (Amazon Web Services, s.f.).

Cuando los usuarios navegan por internet, pocos saben que pueden exponerse a muchos peligros. Hemos creado una gran dependencia de la tecnología, cuya evolución es constante, lo que da lugar a la generación continua de nuevas formas de ataque. El uso de internet conlleva varios riesgos, entre los cuales se incluyen: robo de identidad, filtraciones de datos, malware, virus, phishing, correos y sitios web falsos, e incluso estafas (Kaspersky, s.f.).

Por otro lado, la seguridad informática se define como «la práctica de proteger los activos informáticos de una organización, incluyendo sistemas informáticos, redes, dispositivos digitales y datos, frente a accesos no autorizados, filtración de datos, ciberataques y otras actividades maliciosas» (IBM, s.f.).

Por lo tanto, el uso de internet no debe tomarse a la ligera, ya que tener a nuestro alcance esta gran herramienta puede generar consecuencias de diversa índole. Es por eso que se resalta la necesidad de la seguridad, la cual debe ser considerada una prioridad, especialmente para quienes se están formando profesionalmente en el campo de la Ingeniería en Computación, dado que, en la actualidad, surgen problemas relacionados con la seguridad de manera constante.

Por ello, es indispensable que los estudiantes de la Facultad de Ingeniería cuenten con material adicional que complemente sus clases y les permita conocer, mediante la práctica, los distintos algoritmos de cifrado, así como algunas de sus aplicaciones. Esto les permitirá no solo conocerlos y estudiarlos, sino también aplicarlos en situaciones reales y, en el futuro, en su ámbito laboral. Por lo tanto, es necesario contar con recursos prácticos que ayuden a los estudiantes a comprender y aplicar los temas tratados en la asignatura teórica, los cuales puedan ser incorporados en sus labores como ingenieros en computación, contribuyendo así a mitigar los riesgos de seguridad que evolucionan día a día.

Objetivo general

Diseñar y desarrollar una serie de prácticas como complemento a la asignatura de Criptografía de la carrera de Ingeniería en Computación

Antecedentes didácticos de la materia de Criptografía

En la Facultad de Ingeniería, ya se cuenta con un material didáctico enfocado en la materia de Criptografía, titulado *Plataforma educativa en línea para la asignatura de Criptografía*, desarrollado por Ricardo Sáenz Barragán para obtener su grado en Ingeniería.

Asimismo, existe un material complementario titulado *Material de apoyo al proceso de enseñanza y aprendizaje en línea para la asignatura de Criptografía*, desarrollado por Aarón Enrique Mejía Ortiz, también para obtener su grado en Ingeniería.

Ambos trabajos pueden ser consultados en el repositorio de la Facultad de Ingeniería.

Del mismo modo, la M.C. María Jaquelina López Barrientos desarrolló un libro titulado *Criptografía*, el cual se puede consultar en la Biblioteca de la Facultad.

El presente trabajo pretende complementar el material ya existente, con el fin de que el estudiantado disponga de diversos recursos que favorezcan su aprendizaje en la asignatura.

Descripción del problema

La carrera de Ingeniería en Computación, impartida en la Facultad de Ingeniería, cuenta con un plan de estudios amplio que ofrece diversas áreas de profundización, permitiendo a los estudiantes elegir las materias que cursarán para ampliar sus conocimientos según su área de interés.

La materia de criptografía se imparte en el noveno o décimo semestre y es optativa para aquellos alumnos que elijan la especialización en Ingeniería de Software. Esta materia es teórica, por lo que no cuenta con un laboratorio en el que se realicen prácticas para reforzar y ampliar el conocimiento adquirido en la parte teórica.

Además, desde la pandemia por COVID-19, la necesidad de contar con material en línea que apoye el crecimiento académico de los estudiantes ha aumentado, y es de gran ayuda que los alumnos puedan acceder a este tipo de recursos.

Por esta razón, contar con prácticas que los alumnos puedan realizar desde sus propios equipos les proporcionaría material accesible, lo que fortalecería su aprendizaje. Además, sería útil para aquellos que ya tengan conocimientos o deseen aprender sobre criptografía.

Propuesta de solución

Como solución, se plantea crear una serie de prácticas que los estudiantes puedan realizar de manera extracurricular a la materia teórica.

Para la realización de las prácticas, se tomaron como base los contenidos de la asignatura de Criptografía de la carrera de Ingeniería en Computación, según el plan más reciente, disponible al momento de la elaboración de este material.

Se pretende que el conjunto de prácticas se desarrolle a lo largo de un semestre lectivo, ya sea por elección propia de cada estudiante para extender su conocimiento, o bien que el profesorado las considere como parte de las actividades evaluadas.

La elección de los temas que incorpora el manual se realizó con la asesoría de la M.C. Ma. Jaquelina López, quien, al momento de planificar este trabajo, era profesora de dicha asignatura.

Se considera que sean once prácticas las que integren el manual, comenzando a realizarse en la segunda semana de clases. Además, habrá una práctica final y un compendio de dos proyectos finales para ser desarrollados en las tres últimas semanas del semestre.

Cabe señalar que la primera práctica está enfocada en el aprendizaje del uso del sistema operativo Linux, ya que las prácticas estarán diseñadas para hacer uso de las herramientas que este sistema operativo ofrece. En caso de que el alumno no tenga conocimientos previos de Linux, esta práctica servirá como base para comenzar con las demás actividades.

Las prácticas abarcan la aplicación de temas de criptografía, y en algunos casos, el desarrollo de scripts creados en Python (versión 3, la más actual al momento de la realización de este material). Por lo tanto, se sugiere el uso de este lenguaje, aunque la decisión final quedará a consideración del profesor. Además, se incluirá la búsqueda y ejecución de archivos o scripts que mostrarán, dependiendo de la práctica, las actividades a realizar.

Las prácticas se desarrollarán a manera de formulario en PDF, para que el alumno pueda colocar su respuesta o capturas, según se requiera.

Además, se incluirá una versión con respuestas, dirigida al profesorado, para conocer cuál es la respuesta esperada de los alumnos. Esta versión funcionará como una guía de apoyo.

La planificación de cada una de las prácticas, así como el tiempo estimado para su desarrollo, se encuentra en el Anexo A.

En los anexos también se encontrará material complementario que será de ayuda para el alumno, así como los archivos con el material desarrollado, los cuales se encuentran en la máquina virtual para que puedan ser consultados directamente por el profesorado, si es necesario.

Pruebas

El material desarrollado será sometido a pruebas constantes antes de dar por terminada una práctica. Se adopta el rol de estudiante, se utiliza la máquina virtual y se realiza la práctica desde cero. De este modo, también se verifica que no haya errores en la información proporcionada como guía para los profesores, ni que falte algún archivo o documento necesario para la realización de la práctica. Asimismo, se asegura que no se proporcione ninguno de los scripts solicitados a los alumnos.

Alcance

El proyecto tiene como objetivo crear una serie de prácticas que complementen la materia teórica, con el fin de ser de ayuda principalmente para el estudiantado y profesorado involucrado en la asignatura de Criptografía, proporcionando un material adicional que enriquezca la enseñanza.

Cabe destacar que el material desarrollado en este proyecto no tiene relación con los materiales creados previamente.

Capitulo II

Desarrollo de material didáctico

La estructura del trabajo consiste en presentar primero la instalación de la máquina virtual.

El uso de una máquina virtual se realiza considerando que, de esta manera, el alumnado que requiera el material puede tener acceso a él desde su equipo.

La máquina virtual es fundamental, ya que en ella se desarrolla la mayor parte de las actividades que cada una de las prácticas requiere.

Se sugiere que hagan uso de VirtualBox ya que en algunas actividades se pretende el uso de dos o más máquinas virtuales, este software, permite la realización de este ambiente de trabajo, además que es de software libre (Oracle, s.f.).

La máquina virtual Linux es una distribución Debian, debido a la estabilidad y seguridad que ofrece (Debian, s.f.). A mi consideración, es uno de los entornos más amigables para trabajar.

Posteriormente, se presentan las prácticas en el orden en el que se sugiere realizarlas.

La primera práctica se centra en el conocimiento de los elementos básicos de Linux, con el fin de que los estudiantes que no cuenten con antecedentes sobre dicho sistema operativo adquieran las bases mínimas indispensables, de manera que no tengan inconvenientes al realizar las siguientes prácticas.

La práctica sobre el *Concepto de las funciones hash* se presenta antes de lo que se aborda en el curso regular. Esto se hace con el objetivo de que los estudiantes comprendan cómo se utilizan estas funciones, ya que serán fundamentales en las prácticas posteriores. Se diseñó de esta manera para que el estudiante pueda entender las distintas formas en las que se pueden emplear dichas funciones.

Las prácticas se basan en el desarrollo de algoritmos criptográficos mediante la programación en Python. Además de la ejecución y desarrollo de los programas, se incluyen ejercicios relacionados con la seguridad, e incluso algunas prácticas tienen actividades en las que los estudiantes observan la importancia de la seguridad en la criptografía.

En cada práctica, y dependiendo del contenido de cada una, el estudiantado deberá contestar preguntas, encontrar archivos haciendo uso de los comandos de Linux, por ejemplo, y en general, ir resolviendo pistas para completar exitosamente cada actividad.

Las últimas prácticas estarán enfocadas en actividades que hacen uso de herramientas proporcionadas por Linux, y una de ellas utilizará una herramienta de Windows. La importancia radica en que son actividades que se emplean con regularidad en el ámbito laboral.

Por último, los proyectos finales se centran en aspectos relevantes de la criptografía. En ellos, el estudiante, según los conocimientos adquiridos, llevará a cabo la simulación de un Ransomware y, en otro caso, revisará archivos para eliminar aquellos que sean idénticos

A continuación, se presentan cada una de las prácticas que fueron desarrolladas para complementar la asignatura de Criptografía.

Capitulo III

Presentación del material didáctico

Montaje de máquina virtual.

Se muestra cómo realizar el montaje de la máquina virtual en VirtualBox en el sistema operativo Windows.

Puede que en algunas de las Figuras y procedimientos mostrados a continuación no se realicen exactamente igual, pero sí de manera muy similar.

Se recomienda tener una máquina con al menos 500 GB de disco duro, 8 GB de RAM, y procesador Core Intel i5 o AMD Ryzen 5.

Debe descargar las máquinas virtuales que se encuentra en el apartado *Máquinas Virtuales, una* con el nombre *Criptografía* que es una máquina Linux Debian 12 y la máquina que lleva por nombre *Cryptography* que es una máquina Windows 11.

Para realizar el montaje de las máquinas, puede utilizar VirtualBox ya que es software libre y permitirá el uso de 2 o más máquinas virtuales a le vez.

Las especificaciones del software se dan a continuación.

a. VirtualBox:

VirtualBox es software de virtualización de código abierto propiedad de Oracle, por lo cual aquí se tiene más libertad que en VMware ya que aquí se pueden hacer uso de más de una máquina virtual y al ser de código abierto, es gratuito. Puede instalar el software dando clic en la Figura N° MV.1 que lo llevará al sitio de descarga,

Figura N° MV. 1: Enlace a descargar de VirtualBox.

En el sitio, dé clic en *Windows hosts* o en la conveniente para su sistema operativo, de ese modo obtendrá el ejecutable. Realice la instalación siguiendo los pasos por defecto del ejecutable. La pantalla de inicio de VirtualBox se muestra en la Figura N° MV.2.

Figura N° MV. 2: Inicio de VirtualBox.

Para montar la máquina virtual, de clic en *Archivo* → *Importar servicio virtualizado*, o bien, en la flecha ↶ que indica *Importar*, posteriormente verá una pantalla como la mostrada en la parte izquierda de la Figura N° MV.3, seleccione el archivo de la máquina a montar, dé clic en *Siguiente*.

Figura N° MV. 3: Importar máquina.

Aparecerá una ventana como la mostrada en la Figura N° MV.4, puede cambiar el nombre y en caso de que no reconozca el tipo de S.O, elija Linux → Debian (64 bits), en el apartado *Política de direcciones MAC* se recomienda seleccionar la opción mostrada en la Figura, de este modo se evita inconvenientes con la red entre las máquinas.

Figura N° MV. 4: Configuración previa a realizar la importación.

b. Configuración para VirtualBox.

1. Ajuste a la máquina virtual.
   Puede revisar los ajustes de la máquina virtual, para esto, seleccione la máquina y puede dar clic en el icono de *Configuración* que se muestra en la parte superior o bien, dando clic derecho sobre la máquina se muestra una lista de opciones, entre ella *Configuración*, cualquiera de las dos opciones se puede visualizar en el lado izquierdo de la Figura N° MV.7. Del lado derecho de dicha Figura verá la pantalla de ajustes para la máquina virtual.

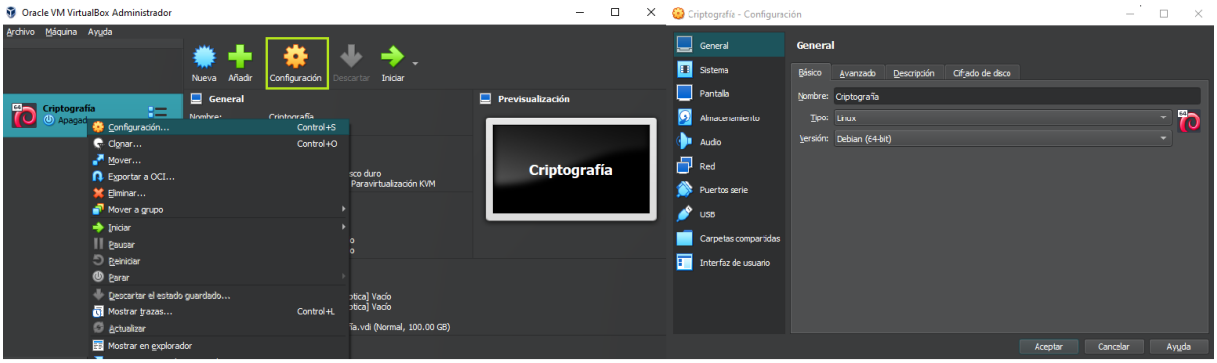


Figura N° MV. 5: Ajuste a la máquina virtual.

1. Ajustes de red.

   En VirtualBox, si se desea usar una red NAT se debe crear, por lo cual debe dar clic en *Archivo → Preferencias → Herramientas → Administrador de red.*

   Se le mostrará una ventana como la mostrada en la parte derecha de la Figura N° Mv.5, en donde en la lista solo verá la red con el nombre *NatNetwork, d*é clic en *Redes NAT* y posteriormente en *Crear*, en la lista ahora verá una nueva red.

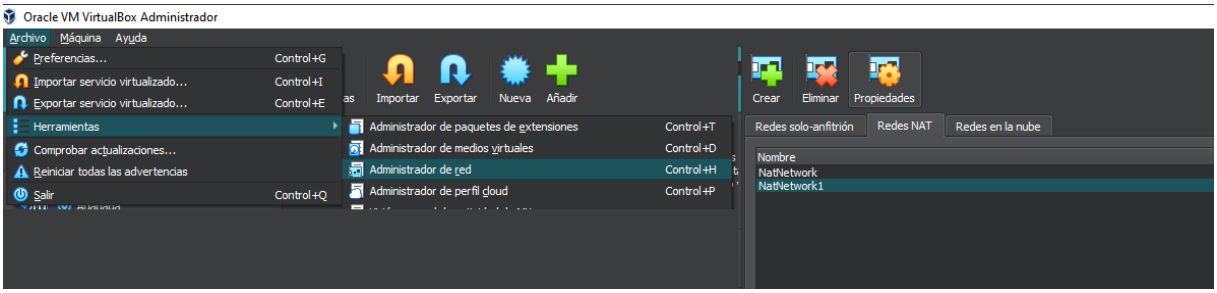


Figura N° MV. 6: Creación de red NAT.

Se creará una nueva red NAT, puede seleccionarla y en la parte inferior tal como se muestra en la Figura N° MV.6, se puede editar tanto el nombre de la red como su prefijo. Para el ejemplo solo se cambia el nombre *NatNetwork1* por *Ciberseguridad*, al final se deben aplicar los cambios.

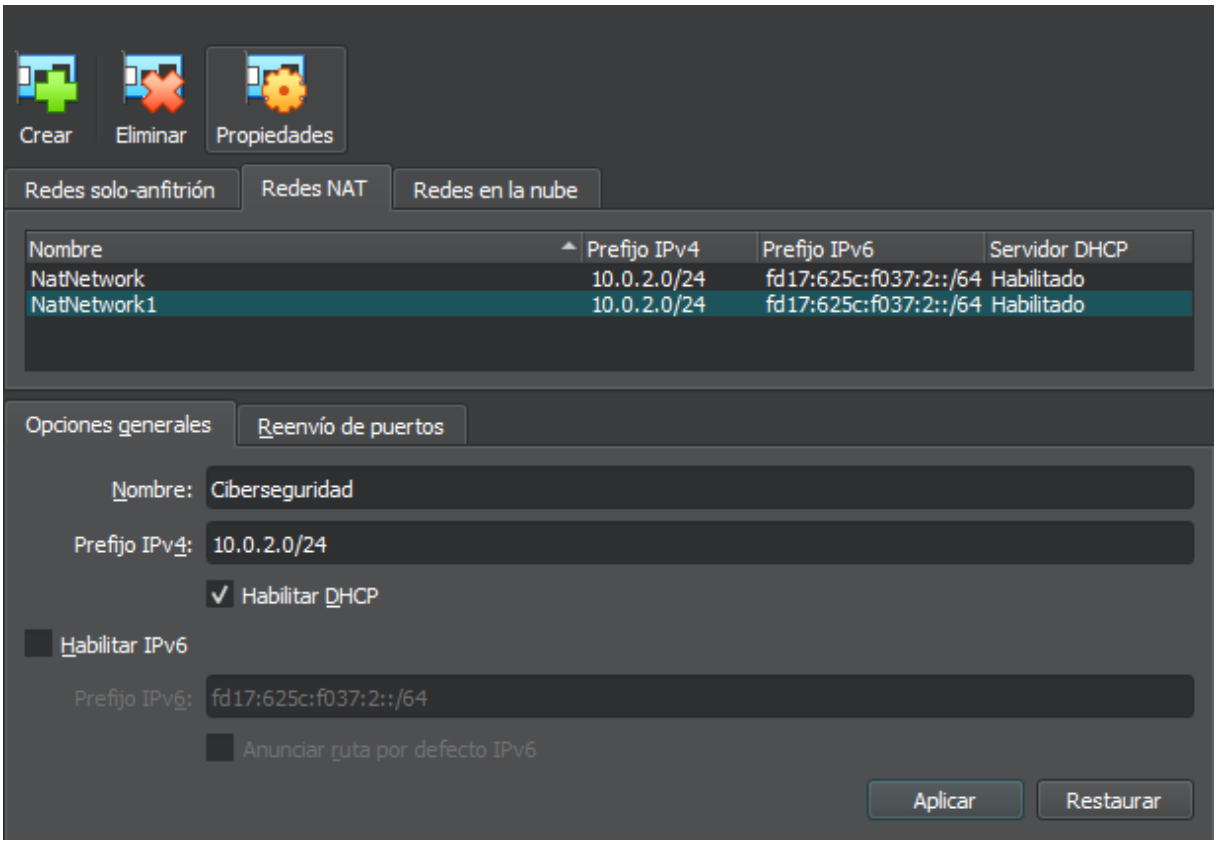


Figura N° MV. 7: Ajustes red NAT.

2. Configuración de adaptador de red para uso de dos o más máquinas.
   Debe tener cuidado con la configuración del adaptador de red, dado que, en caso de utilizar dos o más máquinas virtuales a la vez, es importante que se encuentren en la misma red, para poder realizar la conexión.
   Para esto, ingrese a los ajustes de la máquina virtual, tal como se menciona en el apartado 2.Ajuste a la máquina virtual y dé clic en el apartado *Red,* seguidamente en la sección *Conectado a*, seleccione la opción que dice Red NAT, tal como se muestra en el lado izquierdo de la Figura N° MV.8, por defecto esta seleccionada *NAT* sin embargo, esta red solo tiene una IP misma que le dará a todas sus máquinas por tanto no se podrán ver entre ellas.
   Posteriormente, En la sección *Nombre* seleccione la red que creo en el apartado 1.Ajustes de red, y como paso adicional en la sección *dirección MAC* dé clic en el icono de *Reload*

, esto para que se le asigne una nueva dirección MAC y evite tener problemas con la asignación de IP, esto se muestra en el lado derecho de la Figura MV. 8.

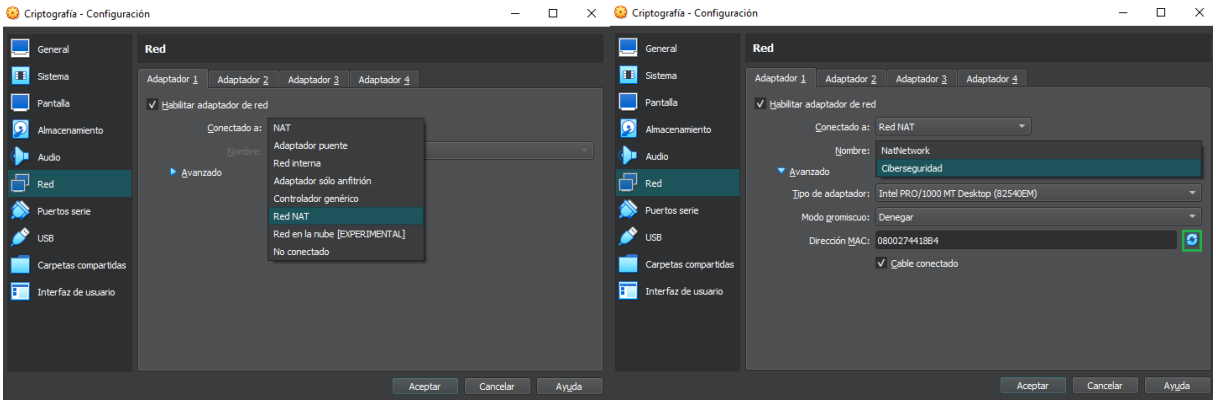


Figura N° MV. 8: Configuración de adaptador de red.

Verifique las IP de las máquinas virtuales, y que exista conexión entre ellas.

3. Configuración de pantalla.
   Adicionalmente, debe revisar en *Configuración,* la sección *Pantalla* en donde el *controlador gráfico* que debe tener seleccionado es *VBoxVGA* tal como se muestra en la Figura N° MV.9.

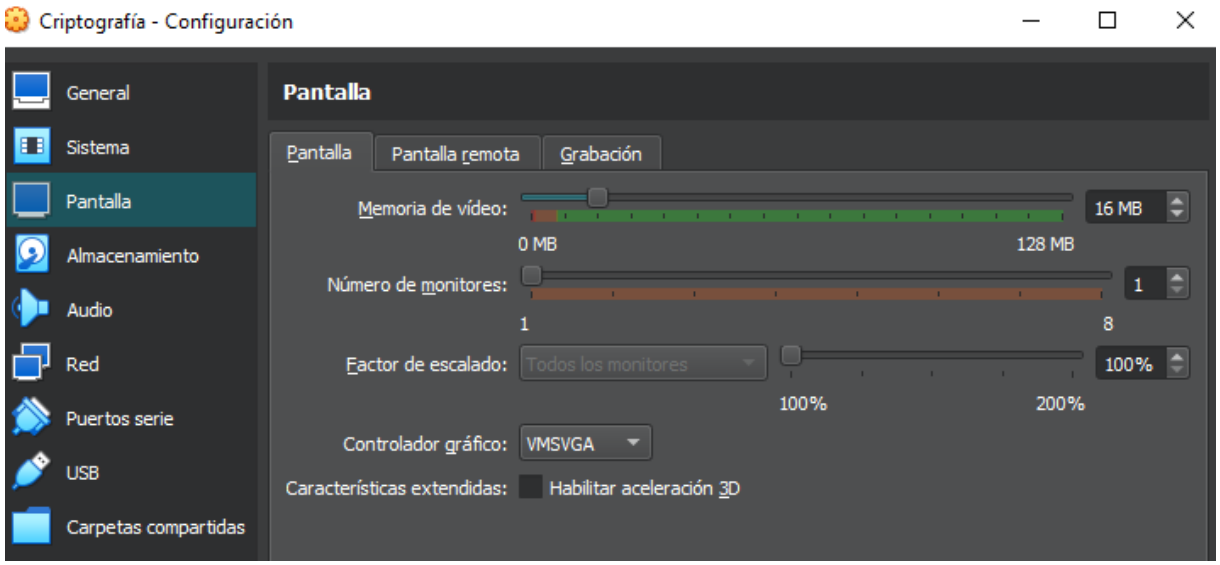


Figura N° MV. 9. Configuración de pantalla.

Para iniciar las máquinas virtuales, dé clic en *Iniciar* . Iniciada la máquina, para conocer las claves a usar consulte el apartado c. Claves para iniciar sesión en las máquinas virtuales de este documento.

c. Claves para iniciar sesión en las máquinas virtuales.

Las claves por usar para las máquinas virtuales se basan en las contraseñas más usadas mundialmente, estás fueron tomadas de la lista publicada referente al año 2023, por Nordpass, con esto se pretende que el alumno(a) tome conciencia en la utilización de contraseñas seguras.

1. Para la máquina Linux

   - Usuario crypto: 111111
   - Usuario root: password

2. Para la máquina Windows

   - Usuario Itzé: 123456

Actividades prácticas

Sección 1: Aspectos Generales

UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

ACTIVIDADES PRÁCTICAS: CRIPTOGRAFÍA

1. CONCEPTOS BÁSICOS DE LINUX

DOCENTE

CONCEPTOS BÁSICOS DE LINUX

Objetivo

Aprender o reforzar, según sea el caso del estudiante, los conceptos básicos del sistema operativo Linux.

Nota: Esta práctica se puede tomar de manera opcional. Si el estudiante considera que cuenta con los conceptos básicos para el uso del sistema Linux, puede omitirla. La decisión queda a consideración tanto del profesor como del estudiante.

Recuerde que para las prácticas se usa la máquina virtual Linux, y por eso es necesario conocer los comandos básicos.

Justificación

Linux es un sistema operativo multiplataforma, multiusuario y multitarea que cuenta con distintas distribuciones entre las más destacadas, Debian, Slackware y Red Hat (ConceptoDefinición, 2021). Muchos servicios son montados en máquinas que cuentan con este sistema operativo, por esta razón es importante conocer la manera en que trabaja, cómo realizar la creación de usuarios, otorgar permisos, entre otros. Los conceptos aprendidos en esta práctica serán utilizados en las prácticas posteriores.

Introducción

Linux es uno de los sistemas operativos de código abierto más utilizado del mundo. Al ser open source lo hace sumamente utilizado en diferentes ámbitos, como lo son servidores web, servidores de correo, monitoreo, firewalls, por mencionar algunos (Tecnología Nolly, s.f.). Sin embargo, se debe tener conocimiento sobre su correcta utilización, pues un simple rm -rf / podría terminar con todo el sistema operativo y junto con él, todo el contenido de este. Es por eso que en esta práctica se realizarán actividades básicas del sistema operativo Linux.

Equipo y material necesario

Máquina Virtual Debian, para montarla en su equipo puede seguir el «Montaje de máquina virtual», el cual es un manual de Instalación MV, en el que se encuentran las instrucciones para VirtualBox. Así mismo puede consultar el Anexo D que contiene ejemplos de cómo debe colocar sus respuestas en la práctica.

Desarrollo

Abrir una terminal en la máquina virtual, se puede realizar de la siguiente manera: Seleccione *Activities* y escriba en la barra de búsqueda *terminal* y dar clic en ella, como se observa en la Figura N° 1.1:

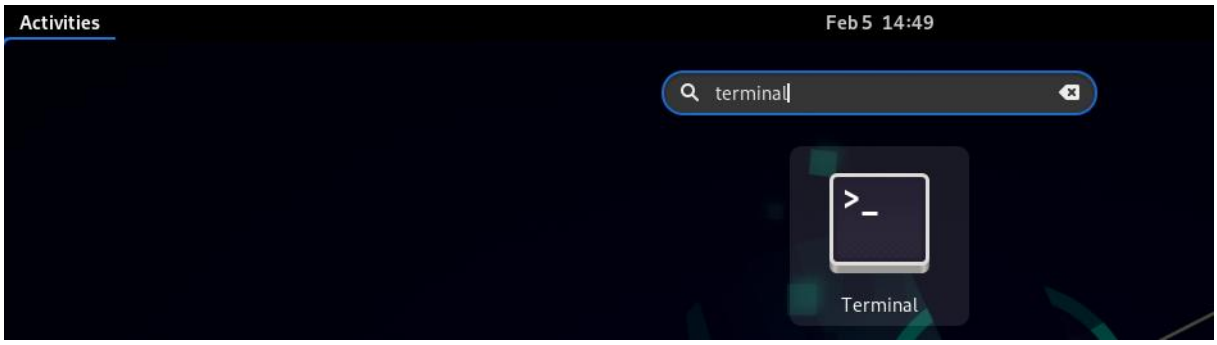


Figura N° 1. 1: Búsqueda de la terminal en Linux.

Entonces se abrirá una ventana como la que se muestra en la Figura N° 1.2:

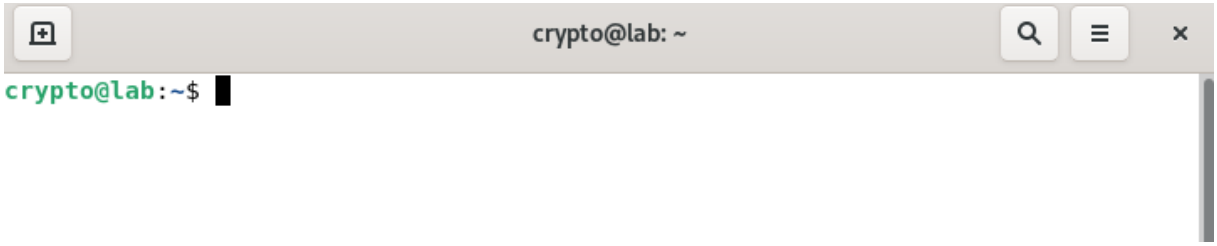


Figura N° 1. 2: Terminal en Linux.

A continuación, en la terminal se deben escribir los siguientes comandos:

pwd: Indica el directorio en donde se encuentra el usuario actualmente. (Véase Figura N° 1.3)

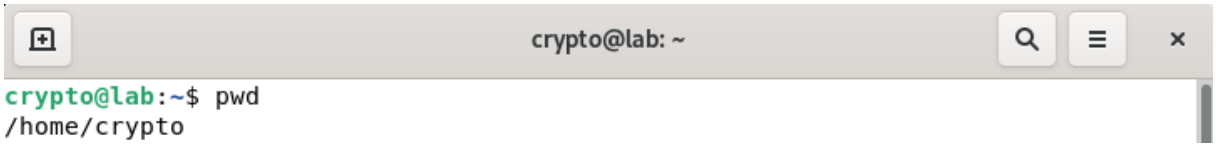


Figura N° 1. 3: Uso del comando pwd.

whoami: Indica el usuario que se encuentra actualmente en el sistema. (Véase Figura N° 1.4)

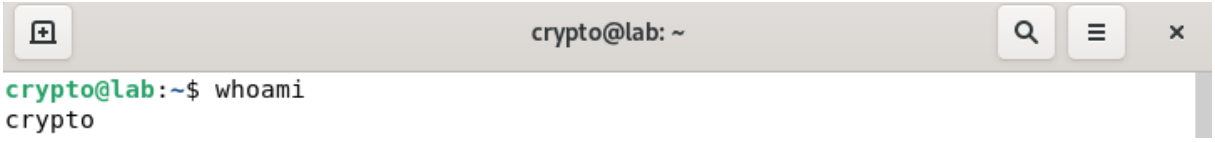


Figura N° 1. 4: Uso del comando whoami.

id: Muestra el ID del usuario y grupos a los que pertenece. (Véase Figura N° 1.5)

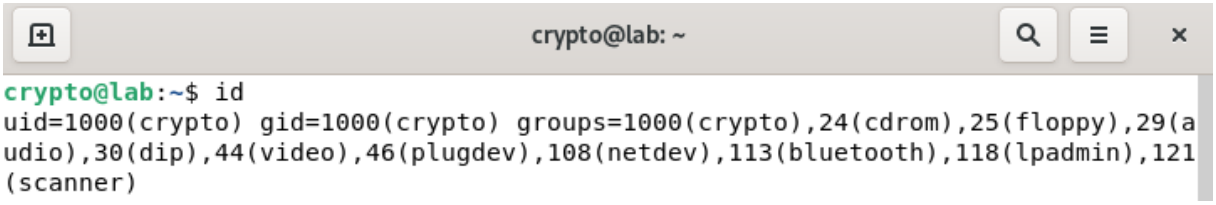


Figura N° 1. 5: Uso del comando id.

hostname: Muestra el nombre del equipo. (Véase Figura N° 1.6)

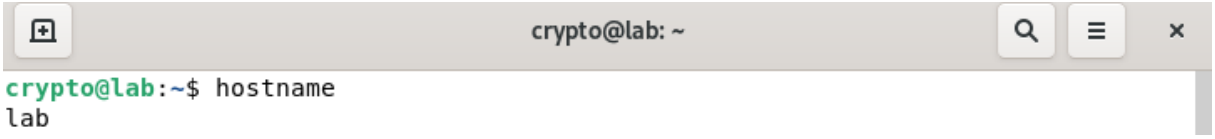


Figura N° 1. 6: Uso del comando hostname.

uname: Muestra información del sistema. El comando por sí solo muestra únicamente el nombre del kernel. (Véase Figura N° 1.7)

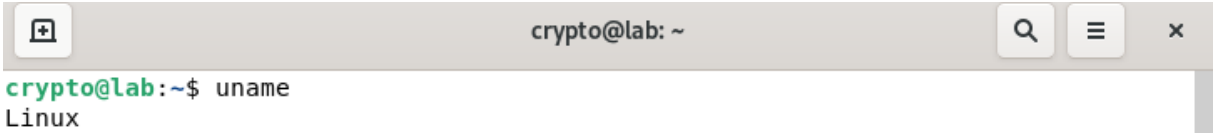


Figura N° 1. 7: Uso del comando uname.

Si al comando se le agrega el argumento **-a** se obtiene información más detallada, tal como se muestra en la Figura N° 1.8:

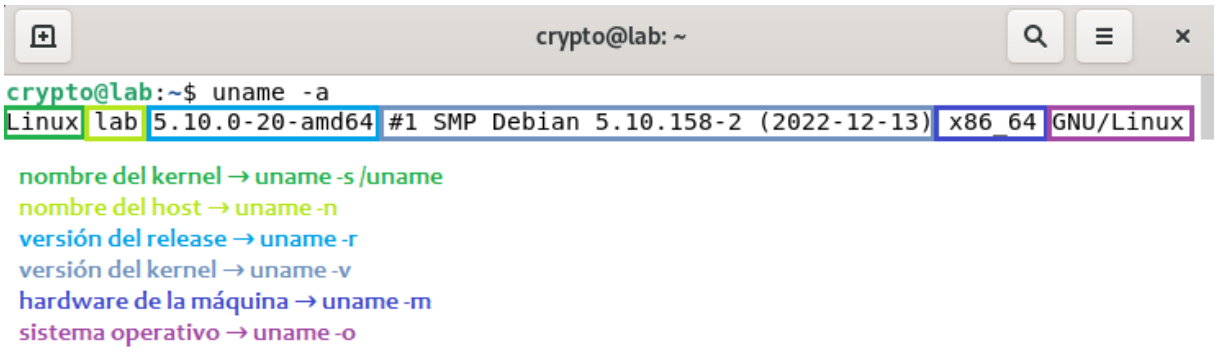


Figura N° 1. 8: Uso del comando uname -a.

ls: Lista los archivos, directorios. (Véase Figura N° 1.9)

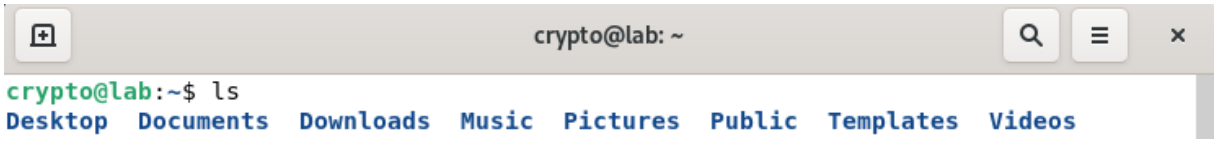


Figura N° 1. 9: Uso del comando ls.

Si se hace uso de **ls -l** se verán los directorios y archivos con más información entre los que destacan las ligas[1], como se muestra en la Figura N° 1.10:

[1] Existen dos tipos de ligas: suaves y duras, y todos los archivos tienen al menos una.

1. *Ligas duras:* Hacen referencia a la ubicación física de los datos, es decir al contenido del archivo. Estas pueden ser creadas con **ln «original» «liga»**. Si se agrega contenido a la liga creada, el contenido también será visto en el archivo original. Aunque se borre el archivo original aún se tendrá en contenido en la liga, por lo cual, para eliminarlo por completo, se tendría(n) que eliminar la(s) liga(s) creada(s).

2. *Ligas suaves:* Hacen referencia a la ubicación del archivo, es como un acceso directo al archivo original. Estas pueden ser creadas **con ln -s «original» «liga»**. A diferencia de las ligas duras, si se elimina la original, la liga suave queda inservible. (De Anda, N., 2018)

Figura N° 1. 10: Uso del comando ls -l.

Para visualizar los elementos ocultos, se hace uso del argumento **-a,** quedando el comando de la siguiente manera **ls -la.** (Véase Figura N° 1.11)

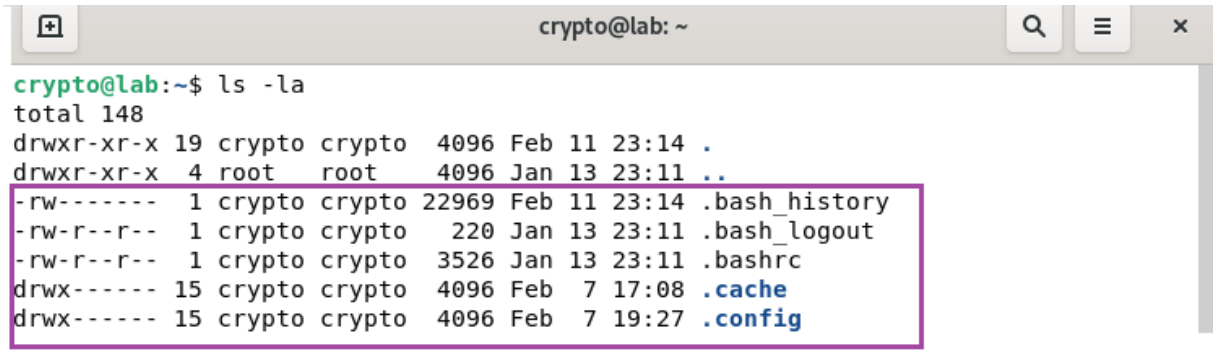


Figura N° 1. 11: Uso del comando ls -la.

mkdir: Crea un directorio. Para hacer uso de este comando se coloca *mkdir «nombre del directorio»*. (Véase Figura N° 1.12)

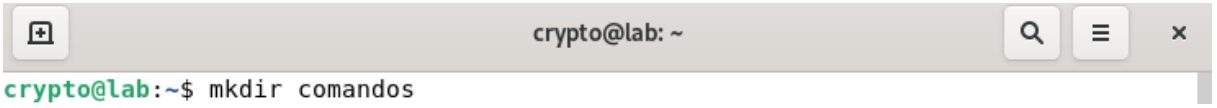


Figura N° 1. 12: Uso del comando mkdir.

Para verificar que el directorio se creó, hacemos uso del comando **ls**.

En caso de que al ejecutar los comandos obtenga el mensaje de «Permiso denegado» haga uso del comando sudo antepuesto al comando principal a utilizar (Ej. sudo mkdir «Directorio»)

ACTIVIDAD 1.1: En el recuadro siguiente, colocar una captura de pantalla indicando la creación del directorio.

Actividad 1. 1: Verificar la creación de directorio.

touch: La función principal de este comando es cambiar las marcas de tiempo[2], sin embargo, también sirve para crear archivos. Se coloca en la terminal: *touch «nombre del archivo».* (Véase Figura N° 1.13)

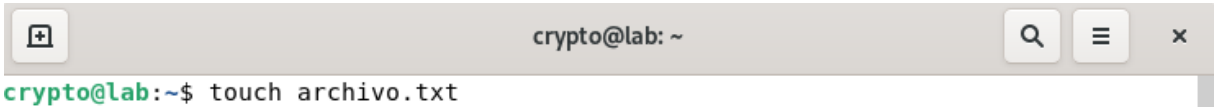


Figura N° 1. 13: Uso del comando touch para crear archivo.

Del mismo modo, se verifica que se ha creado el archivo.

ACTIVIDAD 1.2: En el recuadro siguiente, colocar una captura de pantalla indicando la creación del archivo.

Actividad 1. 2: Verificar la creación de archivo.

[2] Cada archivo, directorio maneja tres tiempos distintos.

1. ctime (*change time*): Indica el tiempo de la última modificación ya sea en cambio de propietario, grupo o permisos.
2. atime (*access time*): Indica el último tiempo en que se accedió al archivo para mostrar el contenido, es decir se realizó una visualización a este.
3. mtime (*modify time*): Indica el tiempo en que se realizó una modificación al archivo, es decir se agregó o eliminó contenido de este.

Para visualizar los tiempos, se hace usando: **stat «*archivo*»** (González, S., s.f.)

echo: Del mismo modo, crea un archivo, pero, desde la línea de comandos se puede realizar escritura en él, donde el contenido se coloca entre comillas. (Véase Figura N° 1.14)

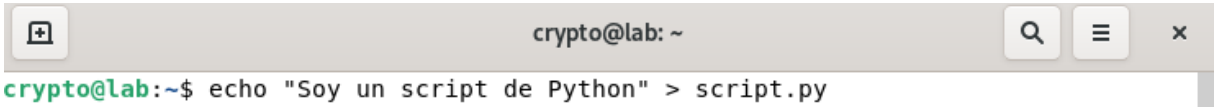


Figura N° 1. 14: Uso de comando echo para crear archivo.

Se comprueba la existencia del archivo, tal como en la Figura N° 1.15

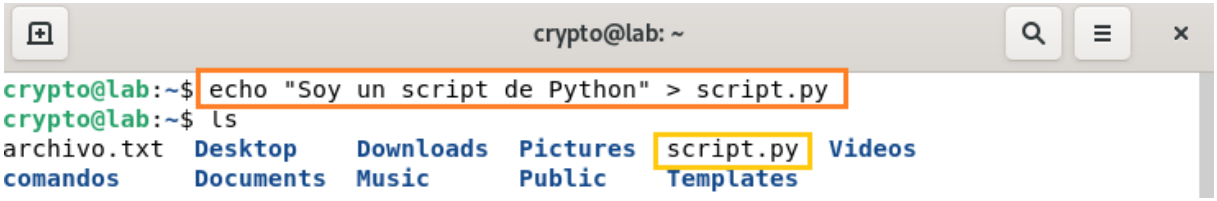


Figura N° 1. 15: Verificar la creación de archivo.

cat: Este comando sirve para visualizar el contenido de un archivo de la siguiente forma: *cat «nombre del archivo»*

ACTIVIDAD 1.3: Utilizar el comando cat para mostrar el contenido del archivo ***script.py*** que acaba de crear, utilice «*cat script.py*». Coloque una captura de pantalla en el siguiente recuadro:

Actividad 1. 3: Uso del comando cat para visualizar el contenido de un archivo.

Además **cat** puede concatenar varios archivos de la siguiente manera: *cat «archivo(1)» «archivo(2)» … «archivo(n)»* en donde la concatenación es mostrada en pantalla.

ACTIVIDAD 1.4: Crear un nuevo archivo llamado **bash.sh**, colocando en el contenido «*Soy un script de bash*», realiza la concatenación de los archivos *bash.sh y script.py*. En el recuadro siguiente, coloque una captura de pantalla.

Actividad 1.4: Uso del comando cat para concatenar archivos mostrados en la pantalla.

Redirección de entrada estándar, salida estándar y salida de error: Cada que se crea un proceso, a su vez se crean 3 archivos (Véase Tabla N° 1.1):

Tabla 1. 1: Entrada, salida estándar y salida de error.

Nombre	Descriptor de archivo	Destino
Entrada estándar (stdin)	0	Teclado
Salida estándar (stdout)	1	Pantalla
Salida de error (stderr)	2	Pantalla

Y estas salidas se pueden redireccionar a archivos haciendo uso del picoparéntesis (>) como se observa en la Figura N° 1.14, para que la frase «*Soy un script de Python*» que es la entrada, se redireccione al archivo *script.py*.

Si solo se hubiera colocado *echo "Soy un script de Python"*, el resultado sería el que se muestra en la pantalla, tal como puede observar en la Figura N° 1.16:

Figura N° 1. 16: Uso de echo sin redireccionamiento.

Por otro lado, al hacer uso del comando **cat** para concatenar los archivos, estos se muestran en la pantalla, pero, también se pueden redireccionar a un archivo como se muestra en la Figura N° 1.17:

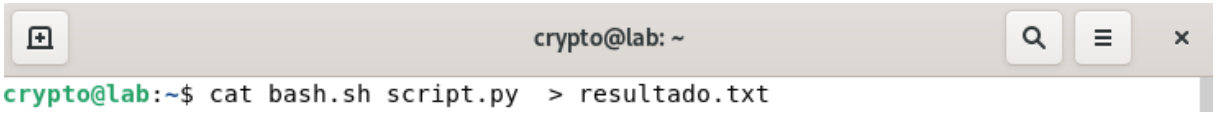


Figura N° 1. 17: Redireccionamiento del contenido de dos archivos, a uno nuevo.

ACTIVIDAD 1.5: Coloque en el siguiente recuadro la comprobación del contenido del archivo **resultado.txt**, haciendo uso de **cat**:

Actividad 1.5: Comprobación del redireccionamiento del contenido de dos archivos, a uno nuevo.

Se puede hacer uso de un doble picoparéntesis (>>) para añadir nuevo contenido a un archivo, si se quiere sobrescribir, se hace uso de un solo picoparéntesis.

En la Figura N° 1.18 se añade contenido nuevo al archivo resultado.txt haciendo uso del comando echo y del comando cat para comprobar:

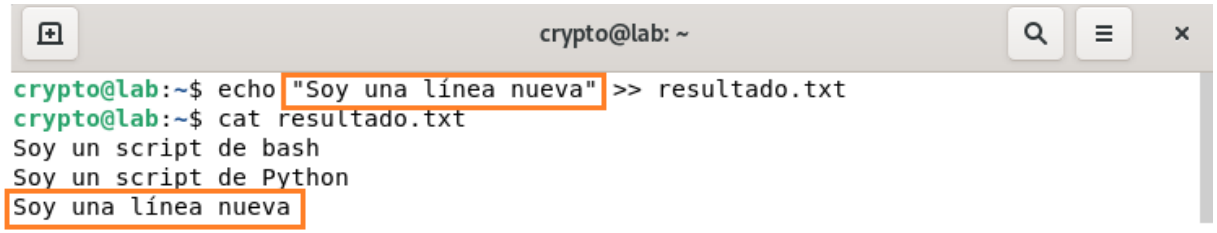


Figura N° 1. 18: Añadir una nueva línea a un archivo.

En la Figura N° 1.19 se sobrescribe el contenido del archivo resultado.txt

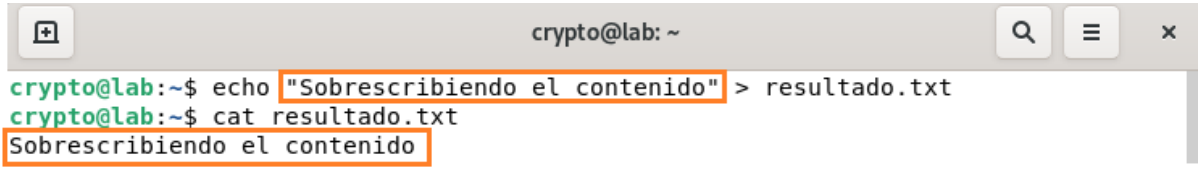


Figura N° 1. 19: Sobrescritura de archivo.

Un ejemplo de la salida de errores se muestra en la Figura N° 1.20, en donde se intenta hacer un **cat** al archivo **no_existe.txt** y regresa un error:

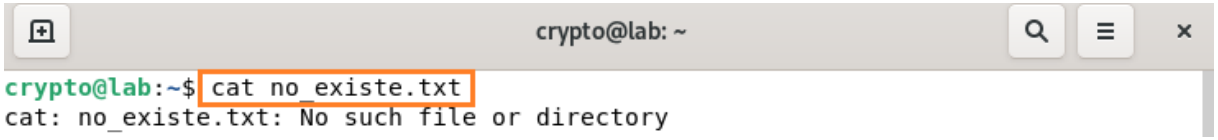


Figura N° 1. 20: Error en la salida estándar.

Para que los errores sean enviados a un archivo se hará uso de un **2** que antecede a los picoparéntesis. En la Figura N°1.21 se realiza un **cat** a tres archivos, los dos primeros existen, el tercero no. Debe quedar claro que la salida NO se quiere ver en pantalla, por lo que es necesario agregar al final la indicación de que el error será enviado a un archivo llamado **errores.log**, lo que se observa en la Figura N° 1.21:

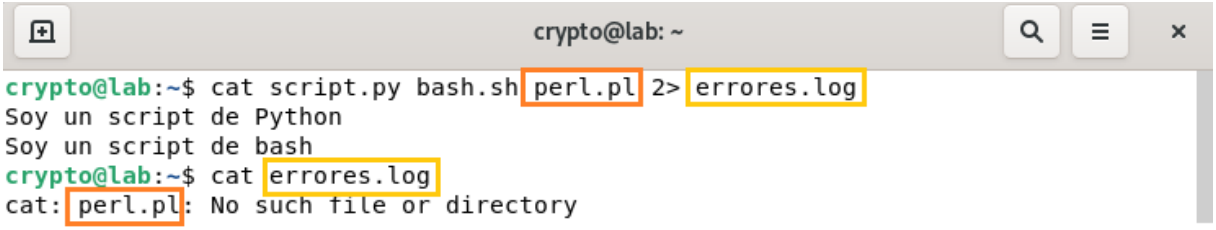
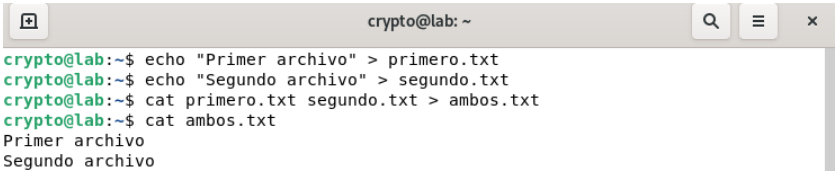


Figura N° 1. 21: Error redireccionado en archivo.

ACTIVIDAD 1.6: Con base en las indicaciones dadas en el lado izquierdo de la siguiente tabla (Véase Tabla N° 1.2) tome la captura de pantalla y colóquela del lado derecho, en dicha captura se debe observar el procedimiento seguido y el resultado obtenido, así mismo, escriba una breve explicación de lo realizado en el apartado de abajo, tal como el primer ejemplo.

Ejemplo:

Tabla 1. 2: Ejemplo para la realización de la Actividad 1.6.

Instrucciones	Captura de pantalla y explicación
Cree 2 archivos llamados primero.txt y 1segundo.txt, los dos deben tener contenido, concatenarlos y agregue su contenido a un nuevo archivo llamado ambos.txt	 Se crea cada uno de los archivos, haciendo uso de echo para colocarles contenido indicando que se escriba en un archivo. Después se hace uso de cat para concatenar los archivos y redireccionar el contenido a un archivo. Finalmente se realiza la visualización del contenido con cat.

Instrucciones	Captura de pantalla y explicación
Agregar una nueva línea al archivo ambos.txt	
Sobrescribir el contenido del archivo segundo.txt	
Escribir en un archivo llamado nuevo.txt el contenido de los archivos primero.txt, segundo.txt y tercero.txt. Este último es un archivo no existente, por lo cual el error debe ser mandando a un archivo llamado errores.log. Todo se debe realizar en una sola línea. Aparte, debe mostrar el contenido de los archivos.	

Actividad 1.6: Uso de redireccionamiento.

more: Este comando permite visualizar el contenido de un archivo, a diferencia de **cat**, en caso de ser un archivo muy largo permite ir visualizando poco a poco su contenido. Para salir de modo lectura oprima la tecla **q**. (Véase Figura N° 1.22)

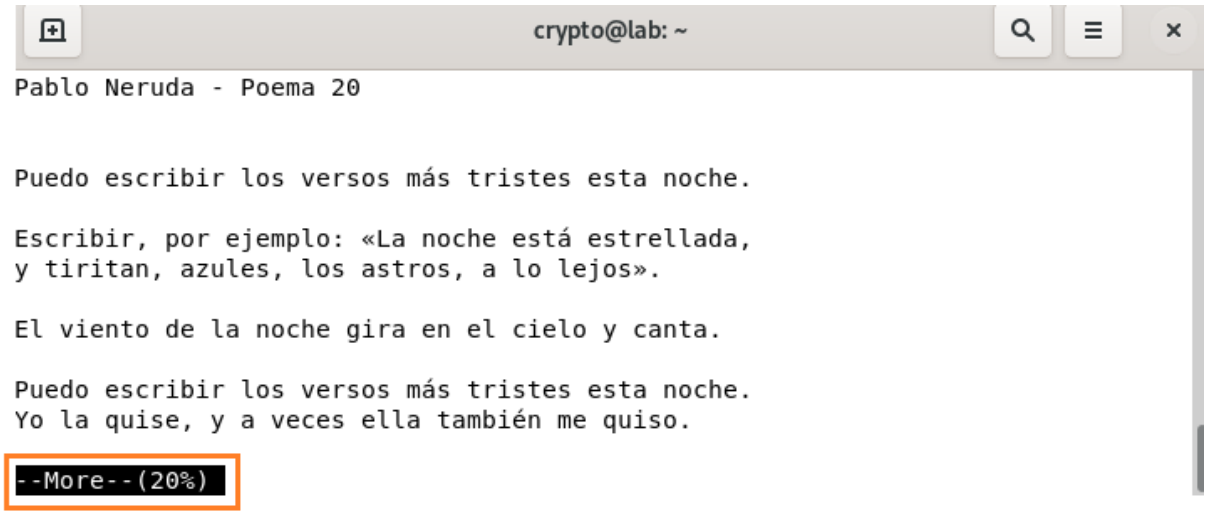


Figura N° 1. 22: Uso de more para visualizar un archivo.

less: Este comando también permite visualizar un archivo, solo que en este caso a diferencia de more, si se quiere regresar a las primeras líneas se puede hacer con el uso de las flechas de dirección ↑↓. Del mismo modo para salir de modo lectura oprima la tecla **q**. (Véase Figura N° 1.23)

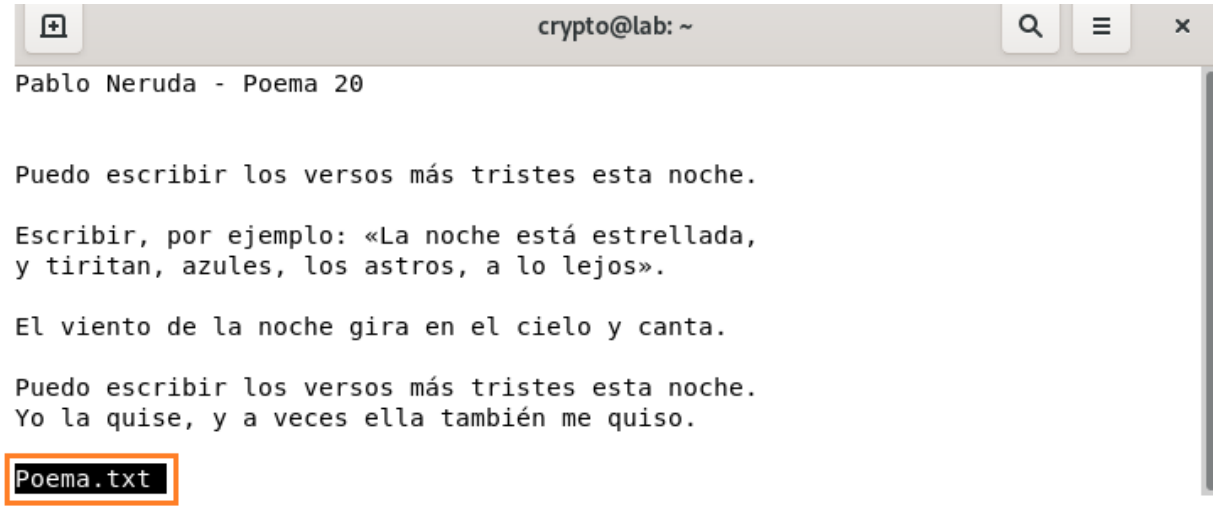


Figura N° 1. 23: Uso de less para visualizar un archivo.

Como se observa en la figura N° 1.24 el texto aparece completo y no permite visualizar poco a poco el archivo, esto pasa, al utilizar el comando **cat**. Es importante ejecutar los comandos para lograr distinguir las diferencias entre las Figuras N° 1.22, 1.23 y 1.24.

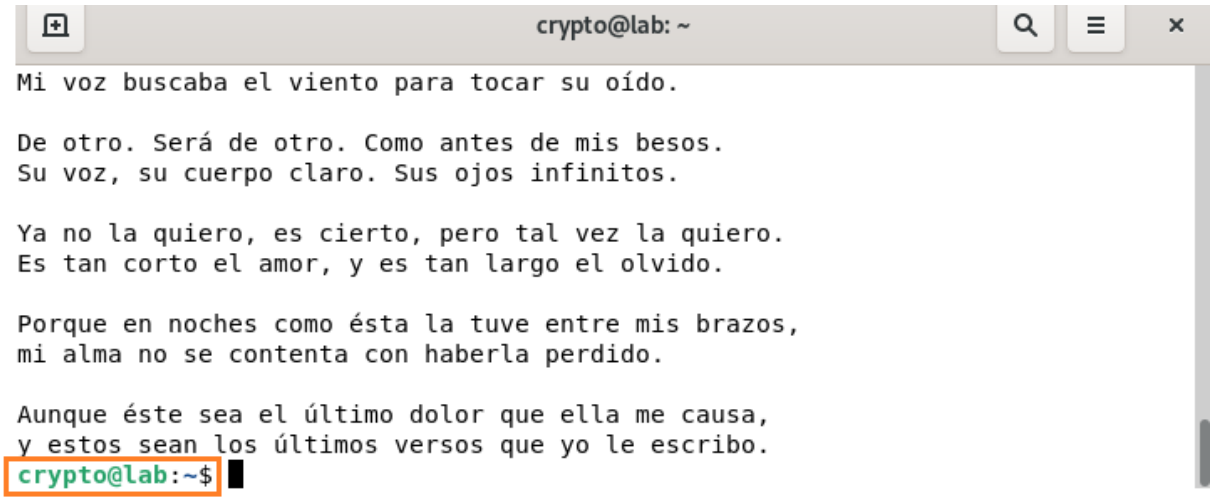


Figura N° 1. 24: Uso de cat para visualizar un archivo.

ACTIVIDAD 1.7: Ejecute los comandos **less, more** y **cat** para visualizar el archivo **Ingeniería.txt** localizado en *Documents/Lecturas* y explique con sus propias palabras en el siguiente recuadro, en qué ocasiones es recomendable utilizar cada uno de los comandos.

Para ingresar al directorio *Documents/Lecturas* se hace uso del comando **cd**[3] de la siguiente manera: *cd «nombre del directorio/ruta del directorio»* (Véase Figura N° 1.25)

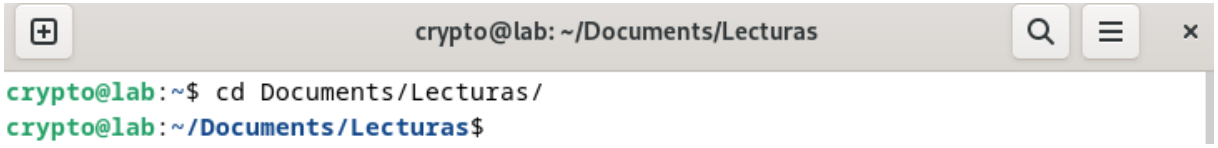


Figura N° 1. 25: Uso del comando cd.

Actividad 1.7: Diferencias entre less, more y cat.

[3] cd: Comando utilizado para ingresar de un directorio a otro. (Medina, L.A., 2013)

file: El comando file indica el tipo de archivo. Se usa de la siguiente manera: *file «nombre del archivo»*

Es importante destacar, que, para Linux, todo es un archivo y las extensiones aquí a diferencia de Windows, no son tomadas en cuenta para definir el tipo de archivo. Sin embargo, para identificar el archivo, las extensiones siguen siendo colocadas.

Cree un nuevo archivo llamado **python.py**, el lenguaje correspondiente a *python*, el archivo queda de la siguiente manera (Véase Figura N° 1.26):

Figura N° 1. 26: Creación de un script de python.

Puede revisar el Anexo N° 2 para saber más sobre los editores de texto nano y vim.

ACTIVIDAD 1.8: Con el uso del comando **file**, indique el tipo de archivo es **python.py** coloque una captura de pantalla del resultado. El archivo **python.py** se encuentra en *Documents/Lecturas.*

Actividad 1.8: Uso del comando file.

Como se observa en el resultado de la Actividad 1.8, el comando nos dice que es un archivo de texto ASCII. En Linux para que este archivo sea considerado un script, se debe colocar en la primera línea la dirección del intérprete[4] de Python. En general para Linux se debe realizar esto con todos los scripts. Se hará uso de Python 3, por lo cual el intérprete es **/usr/bin/python3**, así, el contenido del script queda de la siguiente manera (Véase Figura N° 1.27):

[4] Para obtener la ruta del intérprete se debe usar el comando **which «lenguaje»** en ese caso: *which python3.*

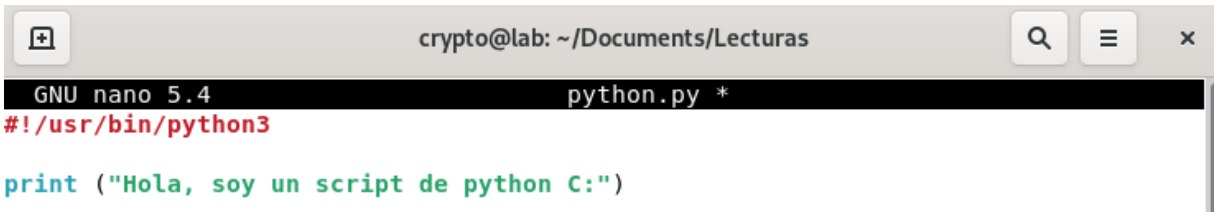


Figura N° 1. 27: Creación de un script de Python.

ACTIVIDAD 1.9: Modifique el archivo **python.py** colocando el contenido mostrado en la Figura N° 1.27, utilice el comando **file** para ese archivo y coloque una captura de pantalla en el siguiente recuadro:

Actividad 1.9: Uso del comando file en script de python.

Rutas: Existen dos distintos tipos de rutas: relativas y absolutas (Carmona, M., 2023).

Antes de indicar las diferencias, en todos los directorios al listar su contenido (**ls -la**) existen dos directorios indicados con **.** y **..** (Véase Figura N° 1.28)

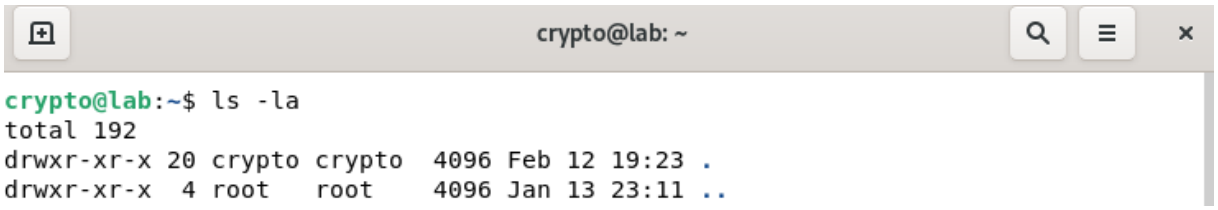


Figura N° 1. 28: Directorios (.) y (..).

El directorio (.) indica el directorio actual, el directorio (..) indica el directorio padre. Es decir, si se encuentra en el directorio «*Documentos*», este sería el directorio padre, los directorios dentro de él, como lo es el directorio «*Lecturas*», es el directorio hijo. Si se ingresa al directorio «*Lecturas*» este se convierte en el directorio padre, y si este tuviera directorios en su contenido, serían los directorios hijos.

Rutas relativas: En términos generales hace referencia a la ruta para ingresar a un directorio desde el directorio actual.

Rutas absolutas: Hace referencia a la ruta para ingresar a un directorio desde el directorio raíz.

Para ejemplificar, cree un directorio llamado **Rutas** en */home/crypto*, dentro del directorio que acaba de crear, cree los directorios **A,B,C,D,E**.

Ahora dentro de cada directorio se deben crear 2 subdirectorios, es decir dentro del directorio **A** debe tener a los directorios **A.1** y **A.2** y el directorio **B** debe contener los directorios **B.1** y **B.2**, y de la misma manera para los directorios **C, D, E,** en donde deben quedar como se describe en la Figura N° 1.29

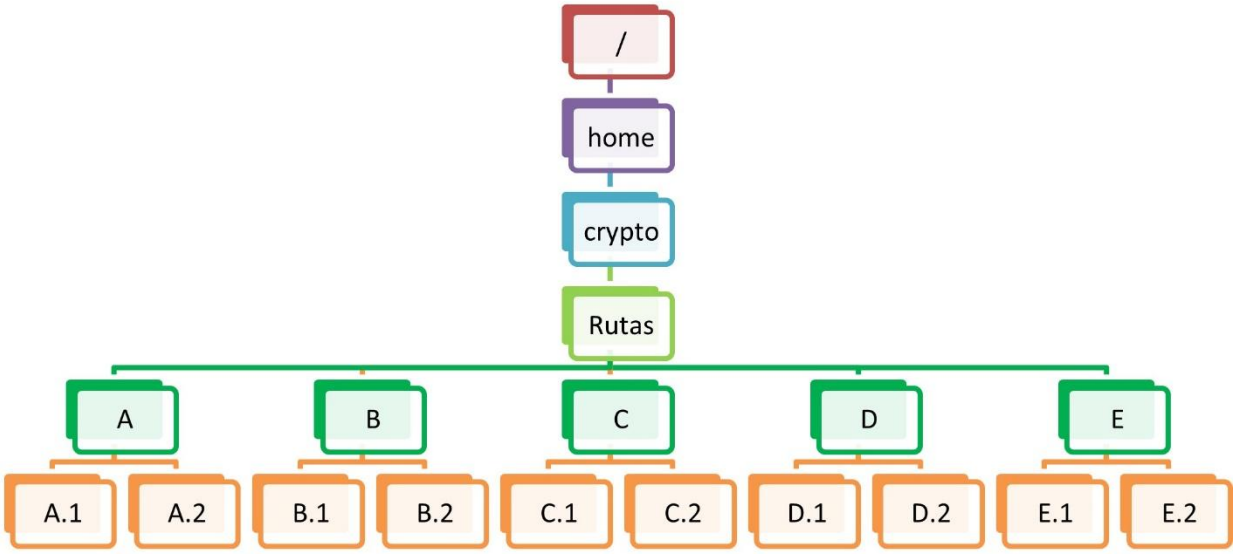


Figura N° 1. 29: Diagrama de directorios.

Siguiendo el diagrama y posicionándose en */home/crypto* que es el directorio home del usuario[5] se quiere llegar a E.1, para esto, debe pasar a Rutas, después a E y posteriormente en E.1.

La ruta relativa del directorio E.1 es **Rutas/E/E.1**

La ruta absoluta del directorio es **/home/crypto/Rutas/E/E.1**

Ahora, desde el directorio E.1, se quiere llegar a A.2 para esto se tiene que pasar por el directorio **E → Rutas → A → A.2**

Esto es posible haciendo uso de **..** que como se mencionó anteriormente es el directorio padre, esto se ejemplifica paso a paso:

1. Primero, se quiere llegar a **E**, para esto colocamos **cd ..** así llegaremos al directorio **E** que es el directorio padre de **E.1** (Véase Figura N° 1.30)

[5] El directorio home del usuario también puede ser representado con **~**

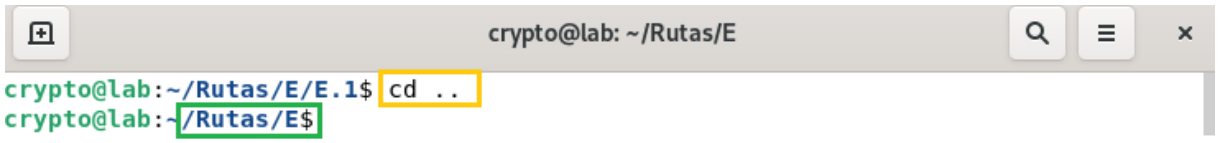


Figura N° 1. 30: Directorio E.

2. Segundo, para llegar a **Rutas**, del mismo modo se realiza con **cd ..**

   En este caso, **Rutas** es el directorio padre de **E**, así como de A,B,C y D. (Véase Figura N° 1.31)

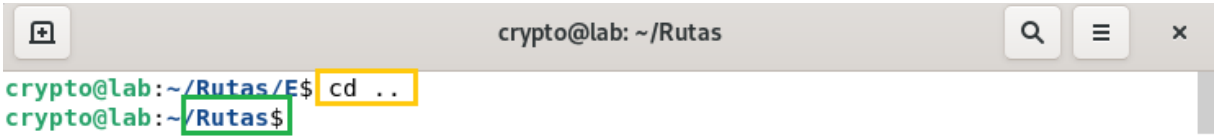


Figura N° 1. 31: Directorio Rutas.

3. Desde el directorio **Rutas** se puede llegar a **A.2** con **cd A/A.2** directamente

   Recuerde, el directorio padre será el que precede al directorio actual, en la Figura N° 1.32, el directorio al que se llegó es **A.2**, por lo cual en este caso su directorio padre es **A.**

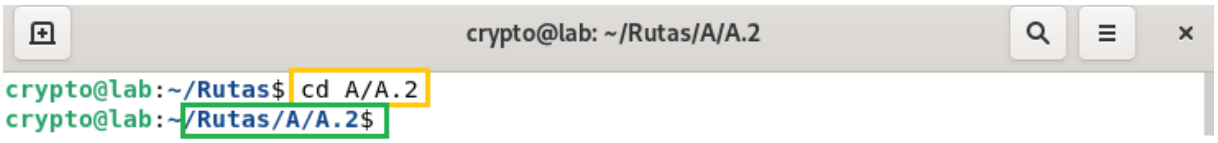


Figura N° 1. 32: Directorio A.2.

   Se puede llegar a dicho directorio en una sola línea (Véase Figura N° 1.33):

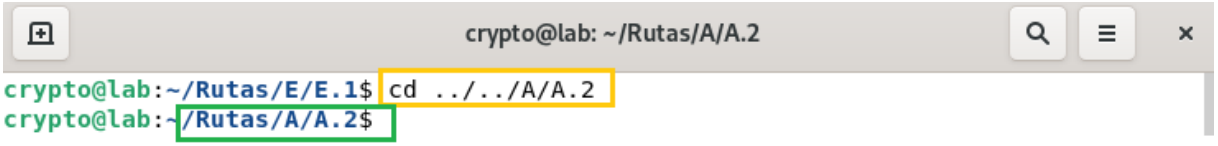


Figura N° 1. 33: Ingresar a un directorio.

Las rutas relativas y absolutas pueden ser usadas de distintas maneras, no solo para ingresar o moverse entre directorios, en los comandos siguientes seguirán siendo utilizadas.

ACTIVIDAD 1.10: Colocarse en */home/crypto* e indique cuál es la ruta relativa y la ruta absoluta para llegar al directorio **D.2**

Ruta relativa: _____

Ruta absoluta: _____

Desde el directorio **D.2** indique la ruta para llegar a **B.1:** _____

cp: El comando **cp** copia archivos de una carpeta a otra, para hacer uso de este comando, se realiza de la siguiente manera: *cp «origen» «destino»*

En la Figura N° 1.34 se realiza la copia del archivo **Poema.txt** usando el comando **cp** y realizando la copia a la carpeta **A.1** para corroborar que se copió, se hace uso del comando **ls** indicando la ruta destino, como se observa el archivo ahora también está en el directorio **A.1**

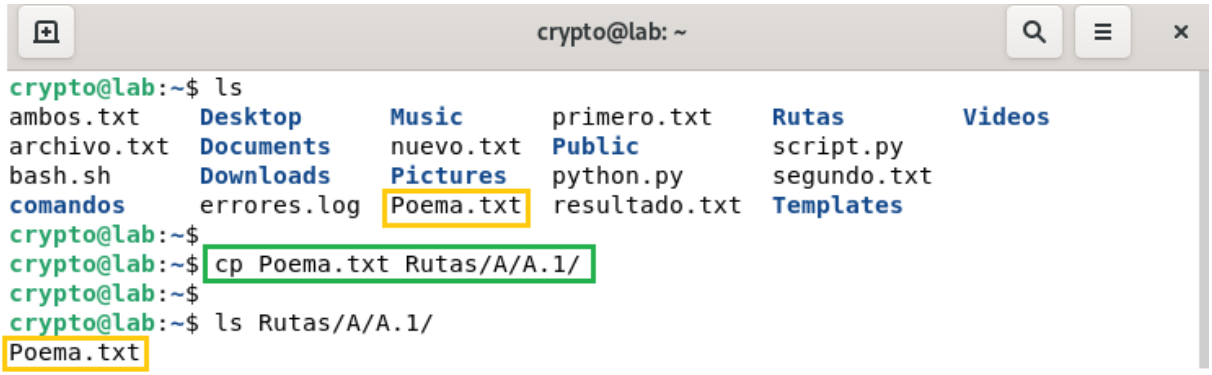


Figura N° 1. 34: Uso del comando cp.

ACTIVIDAD 1.11: De los archivos que ha creado durante la práctica, tome cualquiera de ellos o en su defecto cree un archivo y realice la copia del archivo seleccionado al directorio **C.2**, se debe mostrar que el archivo se encuentra en el nuevo directorio.

Actividad 1.11: Uso del comando cp.

mv: El comando **mv** ayuda a mover los elementos de un lugar a otro. Se debe tener clara la diferencia entre mover y copiar. Al usar el comando **mv** el elemento tal cual pasa del directorio de origen, al directorio de destino. El comando se usa de la siguiente manera: *mv «origen» «destino»*

Como se observa en la Figura N° 1.35, el archivo que se selecciona para mover es **python.py**, este se mueve al directorio **Rutas/B/B.1** y se comprueba que existe haciendo uso del comando **ls** en el destino, finalmente se comprueba que efectivamente el archivo ya no se encuentra en */home/crypto*.

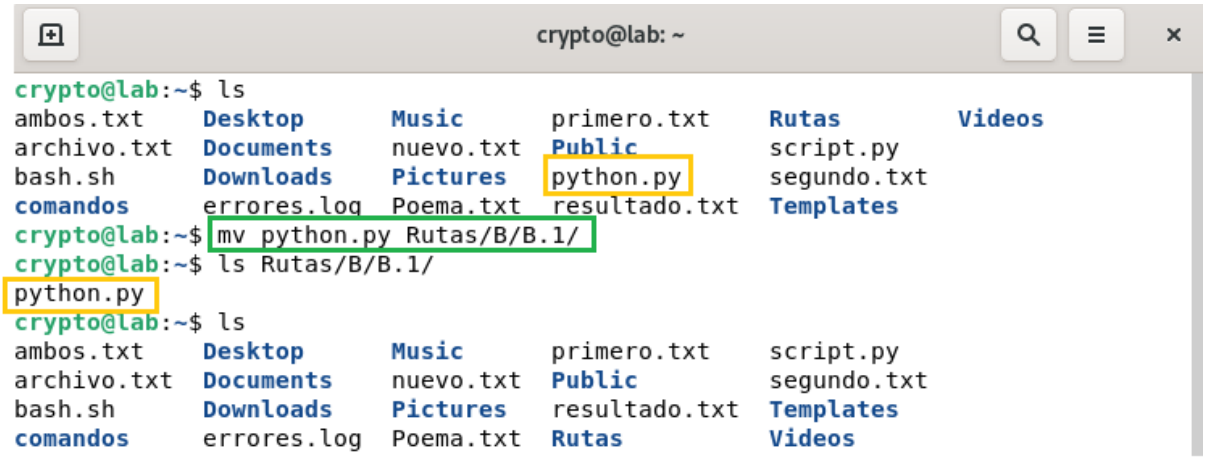


Figura N° 1. 35: Uso del comando mv.

ACTIVIDAD 1.12: Crear un archivo llamado **mover.txt** en */home/crypto*, mover el archivo de */home/crypto* a */home/crypto/Rutas/A/A.2*. Todo debe realizarlo desde el directorio raíz (/). Coloque la captura de pantalla en el recuadro siguiente, en la captura se debe ver la creación del archivo, el archivo dentro del directorio */home/crypto* el uso del comando **mv** y la comprobación de que el archivo **mover.txt** ya no se encuentre en */home/crypto* y sí se encuentre en el directorio **A.2**

Actividad 1.12: Uso del comando mv.

rm: Este comando elimina archivos. El comando se usa de la siguiente manera: *rm «archivo»*. En el caso de los directorios se agrega un **-rf**, la **r** para indicar que es directorio y se eliminen de

manera recursiva, y la ***f*** para forzar la eliminación. El comando queda del siguiente modo: *rm -rf «directorio»*

En la Figura N° 1.36 se muestra cómo se borra el archivo llamado **ambos.txt**.

En la Figura N° 1.37 se muestra cómo se borra el directorio **C.2** haciendo uso de la ruta relativa: al querer listar el contenido del directorio **C** muestra un error dado que el directorio ya no existe.

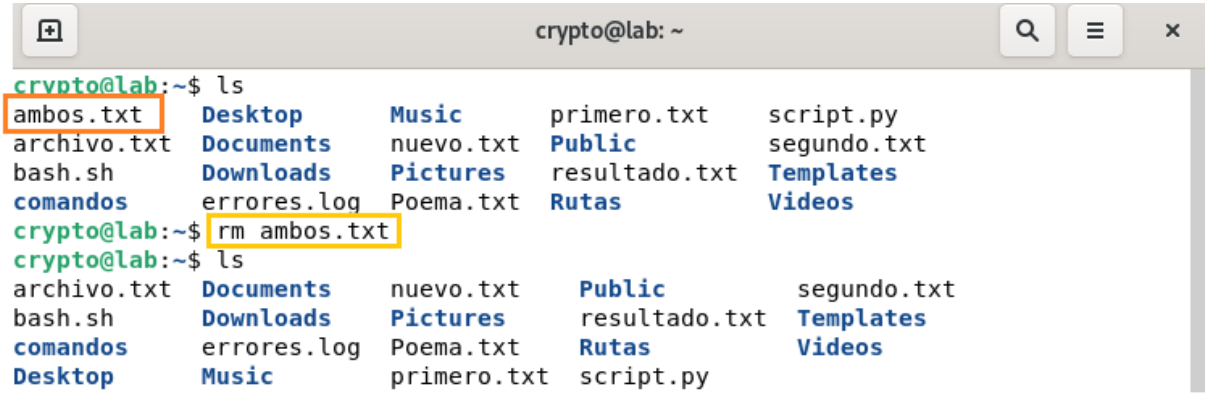


Figura N° 1. 36: Uso del comando rm.

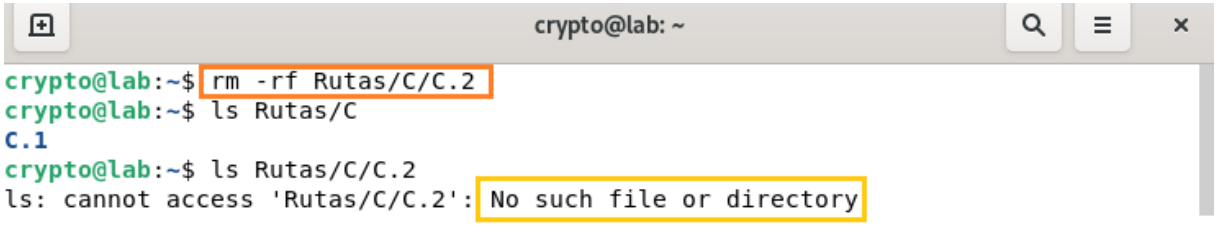


Figura N° 1. 37: Uso del comando rm -rf.

(A partir de esta sección se hace uso del comando sudo, ponga atención en los ejemplos)

Agregar usuario: En este caso se puede encontrar con dos comandos, **adduser** o **useradd**, sin embargo, es recomendable usar **adduser** dado que este comando realiza el agregado de usuario de una manera más completa y correcta, pues la principal diferencia es que **useradd** no agrega contraseña y eso crea conflictos posteriores (Miller,S., 2020).

El comando se usa de la siguiente manera: *adduser «nombre del usuario»* sin embargo es importante destacar que para realizar esta actividad se debe realizar como usuario root[6].

[6] *usuario root:* Este usuario es el que cuenta con los privilegios máximos, tanto de lectura, escritura y ejecución en el sistema operativo, muchos de estos privilegios son limitados para los demás usuarios por razones de seguridad, por esto un usuario root debe ser consciente de lo que va a realizar y los cambios que esto pueda ocasionar.
Para pasar a usuario **root**, se puede usar el comando **su** o anteponer a todo el comando **sudo**, en ambos casos pedirá la contraseña, solo que en el primer caso será permanentemente usuario **root** hasta salir con **exit**; en el segundo caso será solamente usuario **root** al ejecutar ese comando. En caso de que **sudo** no se encuentre instalado, puede realizar su instalación con el comando *apt install sudo*.

En este caso, el usuario ***crypto*** ya está configurado para tener todos los permisos de administrador[7], por lo cual en todo momento se hará uso de sudo para realizar las modificaciones en donde se necesiten los permisos de superusuario.

Como se ve en la Figura N° 1.38 se agrega el usuario llamado **Balam**, los usuarios se agregan con minúsculas, en el apartado «*Full name*» podrá utilizar tanto mayúsculas como minúsculas, este se le agrega una contraseña que es el recuadro amarillo y posteriormente se agrega más información sobre este usuario si así se requiere o bien, los campos pueden quedar en blanco.

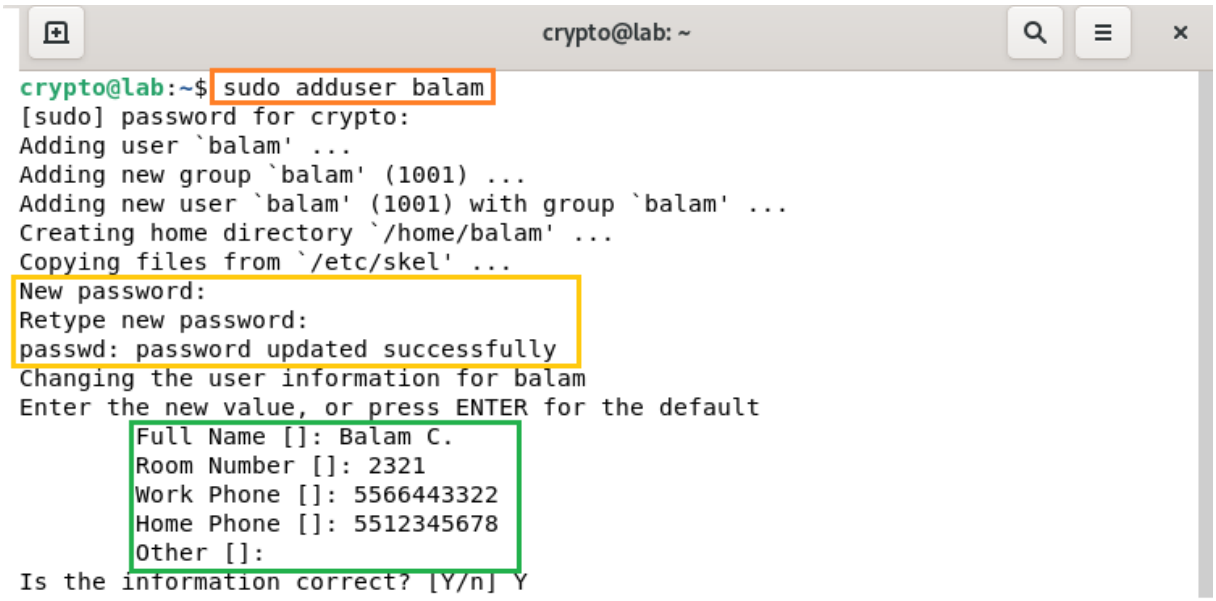


Figura N° 1. 38: Uso del comando adduser.

passwd: Cambia la contraseña de un usuario. Para cambiar la contraseña del usuario actual solo basta con colocar el comando **passwd**, si se quiere cambiar de un usuario específico se indica el usuario *passwd «nombre del usuario»* (Computer Hope, 2021).

En la Figura N° 1.39 se muestra el uso de este comando cambiando la contraseña a otro usuario. Del mismo modo el uso de este comando se debe realizar como usuario root.

[7] *sudoers:* Este archivo está ubicado en **/etc/sudoers,** el archivo contiene la configuración del usuario administrador, lo que se conoce como superusuario o root, y los usuarios que pueden hacer uso de sudo y los privilegios que tienen con este. Si se desea editar el archivo y sus permisos, se realiza con el comando ***sudo visudo***. En caso de no tener ningún usuario configurado con anterioridad, debe realizar con el usuario root.

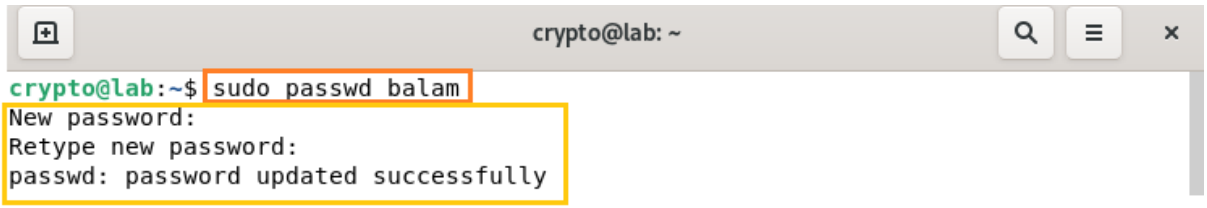


Figura N° 1. 39: Uso del comando passwd.

ACTIVIDAD 1.13: Cree un usuario con su nombre, posteriormente, haga el cambio de contraseña para ese usuario. Coloque una captura de pantalla en donde se muestre lo realizado en el siguiente recuadro:

Actividad 1.13: Uso de los comandos adduser y passwd.

addgroup: Comando utilizado para crear grupos, el comando es simple, solo se coloca: *addgroup «nombre del grupo» (Menjívar, M., 2021).*

En la Figura N° 1.40 se muestra la creación del grupo llamado nuevos.

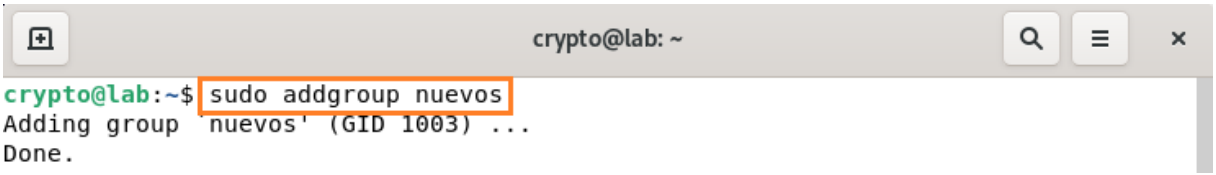


Figura N° 1. 40: Uso del comando addgroup.

usermod: Comando que permite realizar modificación en la cuenta de usuario. Este comando tiene distintas opciones, entre ellas permite agregar un usuario a un grupo, para esto, se usa el siguiente comando: *usermod -a -G «nombre grupo» «nombre usuario»* (Lonston, B., 2023) como se observa en la Figura N° 1.41:

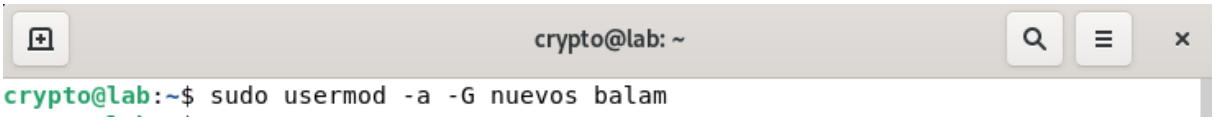


Figura N° 1. 41: Uso del comando usermod.

Se puede revisar que el usuario realmente pertenezca a ese grupo con el comando *getent*[8] *group «nombre grupo»* (Véase Figura N° 1.42)

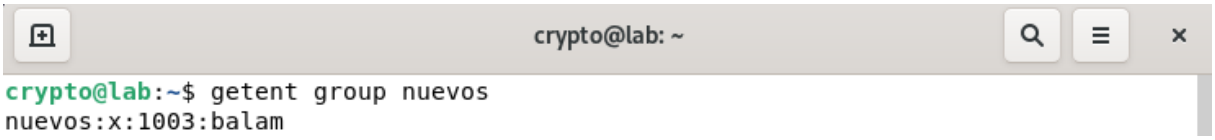


Figura N° 1. 42: Uso del comando getent.

Cabe mencionar que cuando se crea un usuario, por defecto se crea un grupo para este usuario con el mismo nombre (para el caso del ejemplo, balam) tal como se visualiza en la Figura N° 1.38, por lo cual, en este caso balam tiene como grupo principal su propio grupo que lleva el mismo nombre, sin embargo, forma parte ahora también del grupo nuevos, este, vendría a ser su grupo secundario y por tanto Balam pertenece a dos grupos.

En caso de que lo requiera se puede cambiar el grupo secundario como primario, con el comando usermod -g *«nombre grupo» «nombre usuario»*, al realizar esto, de manera automática cambia el nombre del grupo por el definido anteriormente. Un ejemplo de esto se muestra en la Figura N° 1.43:

[8] *getent:* En general este comando obtiene las entradas de la base de datos administrativa, es decir, group, hosts, services, passwd, shadow, protocols o networks (GeeksforGeeks, s.f.)

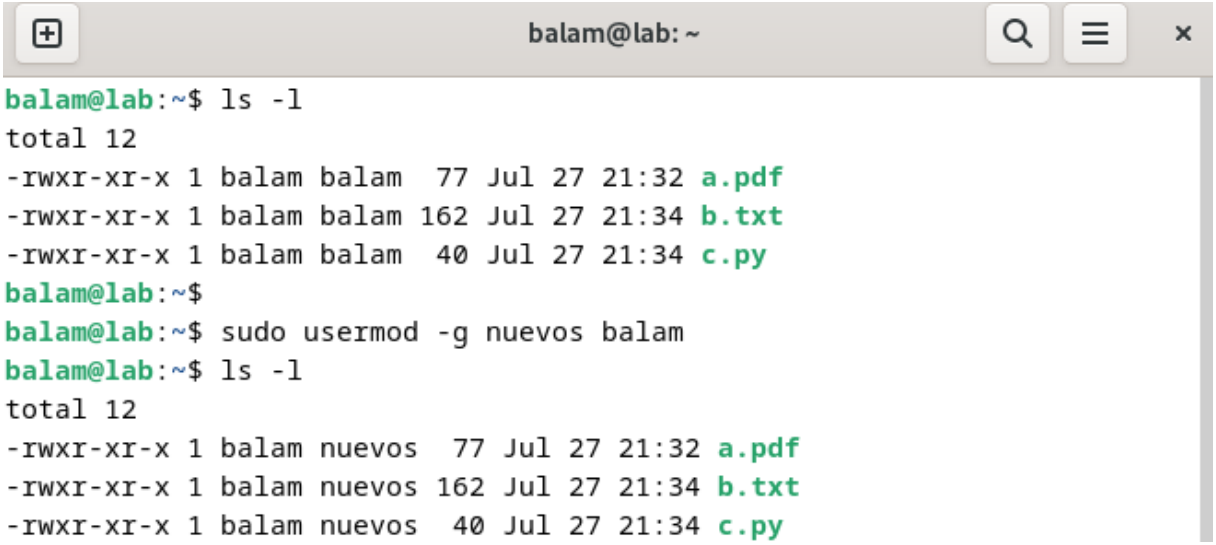


Figura N° 1. 43: Uso del comando groupmod.

groupmod: Comando que permite realizar modificaciones en un grupo. Del mismo modo este comando tiene distintas opciones, en este caso para ejemplificar se cambiará el nombre del grupo, haciendo uso del siguiente comando: *groupmod -n «nombre nuevo» «nombre anterior» (Samdare, B., s.f.).*

En la Figura N° 1.44 se realiza el cambio de nombre del grupo creado anteriormente, de **nuevos** a **personal**, se puede verificar el cambio de nombre con el comando **getent.**

Figura N° 1. 44: Uso del comando groupmod.

ACTIVIDAD 1.14: Cree un grupo llamado **criptografia** y 2 *usuarios más* con el nombre de sus compañeros, agregue a los usuarios que acaba de crear a este grupo, compruebe que los usuarios se encuentran en el grupo creado. Coloque en el siguiente recuadro captura de pantalla de lo realizado, en la captura, *no es necesario visualizar la creación de los otros dos usuarios. El nombre del grupo debe ir sin acento, ya que no es un carácter valido.*

Actividad 1.14: Uso del comando addgroup, agregado y verificación de usuarios en grupo.

finger[9]**:** Comando que muestra información de los usuarios del sistema. Este comando se usa colocando *finger «nombre usuario»* como se muestra en la Figura N° 1.45.

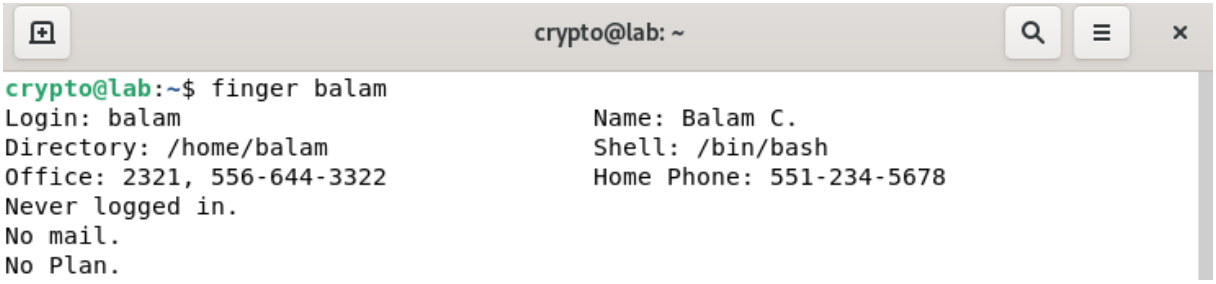


Figura N° 1. 45: Uso del comando finger.

ACTIVIDAD 1.15: De acuerdo con la descripción, coloque en la línea de qué comando se trata:

a. Comando que indica el usuario que se encuentra actualmente en el sistema:

b. Comando que indica el directorio actual en el que se encuentra el usuario:

c. Comando que muestra el id del usuario y grupos a los que pertenece:

Para cambiar de usuario puede realizarlo desde terminar, para eso ejecute: *su - «nombre usuario»*. El ejemplo se muestra en la Figura N° 1.46:

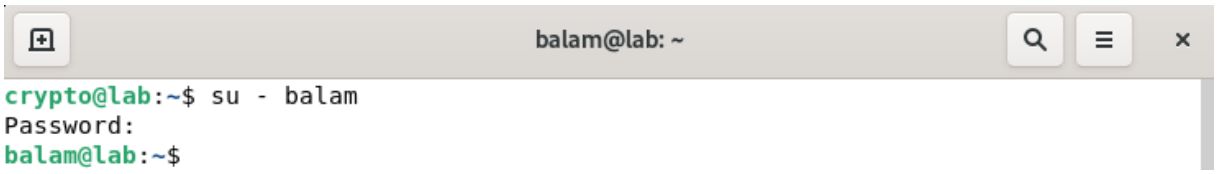


Figura N° 1. 46: Cambio de usuario con el comando su.

Debe tener en cuenta que el cambio de usuario solo permite manipulaciones desde terminal, si quiere ingresar a sus archivos desde el gestor de archivos debe cambiar completamente de usuario e iniciar sesión con el usuario en cuestión, esto para evitar problemas con permisos.

[9] Este comando puede no estar incluido por lo cual debe instalarse, se realiza ejecutando **apt install finger**

Haga uso del comando **finger** para el usuario creado en la *Actividad 1.13*, posteriormente realice el cambio al usuario, esto se realiza como se muestra en la Figura N° 1.45. En el siguiente recuadro, coloque captura de pantalla donde se muestre el uso de los comandos **finger**, además de los descritos en los incisos **a, b y c.**

Actividad 1.5: Cambio de usuario con el comando su.

deluser: Comando para eliminar usuarios, se usa de la siguiente manera: *deluser «nombre usuario»*

En la Figura N° 1.47 se elimina al usuario **Balam,** usando la opción **--remove-home,** para que elimine también su directorio home y en un futuro si se agrega un usuario con el mismo nombre no se creen conflictos.

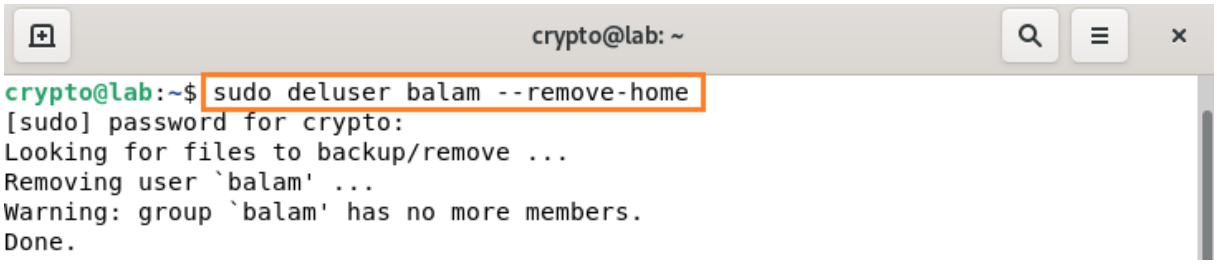


Figura N° 1. 47: Uso del comando deluser.

delgroup: Comando para eliminar un grupo. Para ejemplificar se eliminará el grupo personal, usando el siguiente comando: *delgroup «nombre grupo»* como se muestra en la Figura N° 1.48

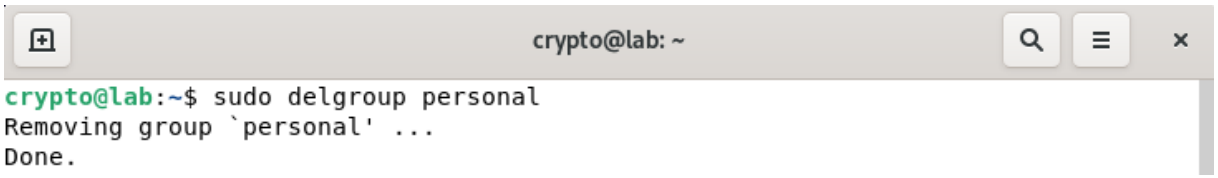


Figura N° 1. 48: Uso del comando delgroup.

ACTIVIDAD 1.16: Elimine a uno de los dos usuarios creados en la Actividad 1.14. Realice el cambio de nombre del grupo de **criptografia a seguridad**, verifique que, a pesar del cambio de nombre, los usuarios se encuentren en el grupo. En el siguiente recuadro, coloque captura de pantalla de lo realizado.

Actividad 1.16: Eliminación de usuario y cambio de nombre a un grupo.

Si se desea eliminar a un usuario de un grupo, se realiza del mismo modo que al eliminar un usuario del sistema, solo indicando el grupo del cual quiere ser eliminado, es decir se ejecuta de la siguiente manera: *deluser «nombre usuario» «grupo»*

Archivos /etc/passwd /etc/shadow y /etc/group (Sempiterna Serendipia, s.f.).

Linux cuenta con 3 archivos los cuales gestionan los usuarios, sus contraseñas y los grupos.

a. ***/etc/passwd:*** En este archivo se encuentran en forma de lista, todos los usuarios registrados en el sistema, al visualizar el archivo se obtiene algo como lo siguiente (Véase Figura N° 1.49):

```
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/run/ircd:/usr/sbin/nologin
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nologi
n
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
_apt:x:100:65534::/nonexistent:/usr/sbin/nologin
systemd-network:x:101:102:systemd Network Management,,,:/run/systemd:/usr/sbin/n
ologin
systemd-resolve:x:102:103:systemd Resolver,,,:/run/systemd:/usr/sbin/nologin
/etc/passwd
```

Figura N° 1. 49: Archivo /etc/passwd.

Cada usuario cuenta con siete campos separados por dos puntos (:) y cada campo contiene la información mostrada en la Figura N° 1.50:

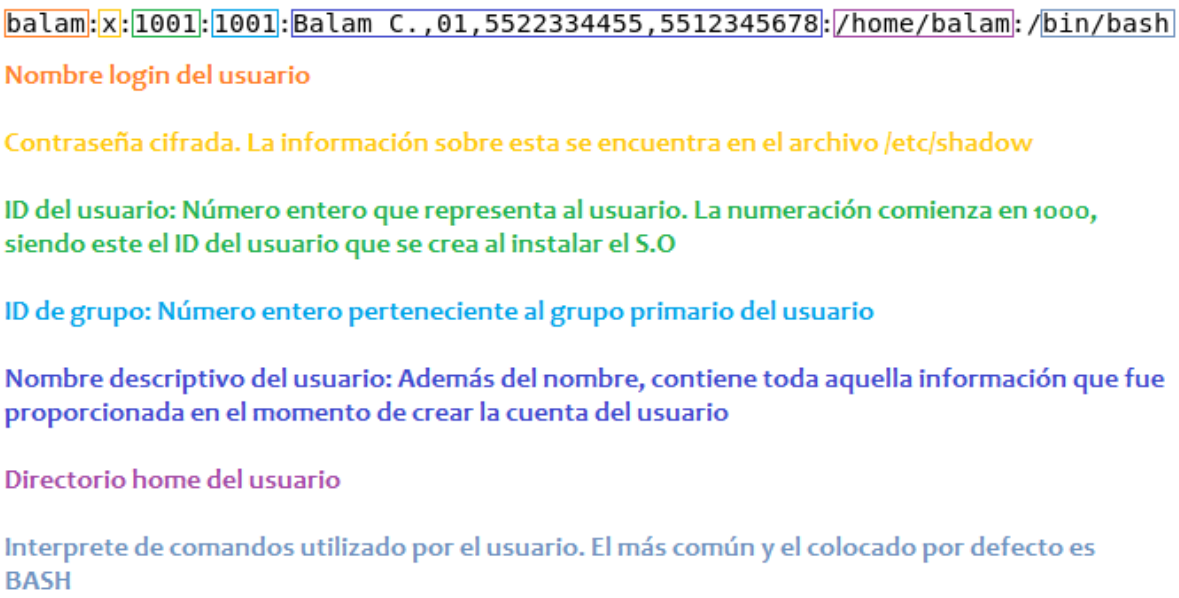


Figura N° 1. 50: Estructura de un usuario en el archivo /etc/passwd.

b. **/etc/shadow:** Archivo que contiene en forma de lista, las contraseñas de cada uno de los usuarios. Para visualizar el archivo se debe tener permisos de superusuario. Al visualizarlo, se obtiene lo siguiente (Véase Figura N° 1.51):

```
root:$y$j9T$w9RxU5qroGkhncIFy.vX7.$9LqeE9mcqiCrb/reqGies25Vx8SiYeOSBSvh8vAboP5:1
9371:0:99999:7:::
daemon:*:19371:0:99999:7:::
bin:*:19371:0:99999:7:::
sys:*:19371:0:99999:7:::
sync:*:19371:0:99999:7:::
games:*:19371:0:99999:7:::
man:*:19371:0:99999:7:::
lp:*:19371:0:99999:7:::
mail:*:19371:0:99999:7:::
news:*:19371:0:99999:7:::
uucp:*:19371:0:99999:7:::
proxy:*:19371:0:99999:7:::
www-data:*:19371:0:99999:7:::
backup:*:19371:0:99999:7:::
list:*:19371:0:99999:7:::
irc:*:19371:0:99999:7:::
gnats:*:19371:0:99999:7:::
nobody:*:19371:0:99999:7:::
_apt:*:19371:0:99999:7:::
systemd-network:*:19371:0:99999:7:::
systemd-resolve:*:19371:0:99999:7:::
tss:*:19371:0:99999:7:::
/etc/shadow
```

Figura N° 1. 51: Archivo /etc/shadow.

La estructura para la contraseña cuenta con nueve campos separados por dos puntos (:) y cada campo contiene la información mostrada en la Figura N° 1.52:

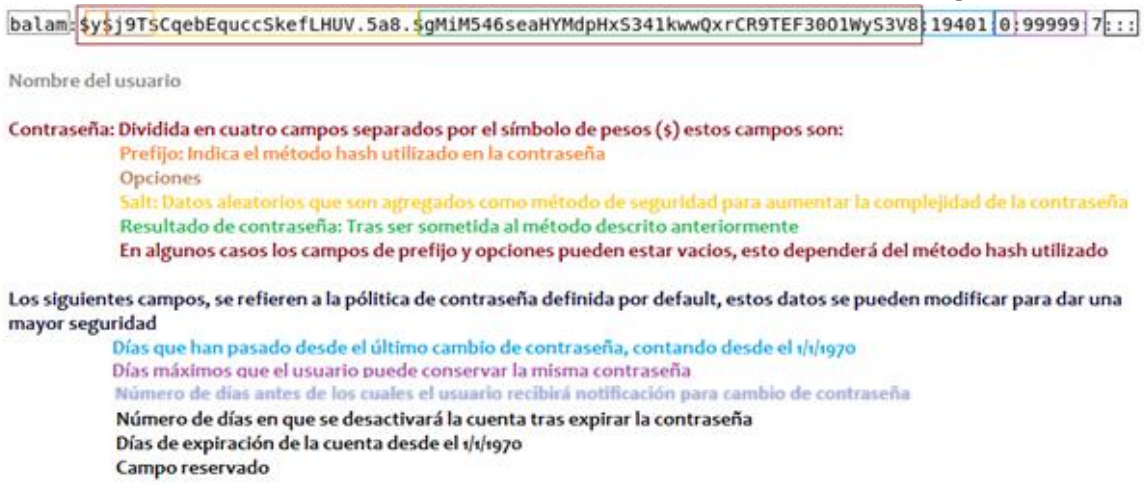


Figura N° 1. 52: Estructura de un usuario en el archivo /etc/shadow.

c. **/etc/group:** Archivo que contiene los grupos y los usuarios pertenecientes a cada uno de ellos. Cuando se crea un usuario se crea un propio grupo para ese usuario, con el mismo nombre que se le asignó al usuario. El archivo se visualiza de la siguiente manera (Véase Figura N° 1.53):

```
root:x:0:
daemon:x:1:
bin:x:2:
sys:x:3:
adm:x:4:
tty:x:5:
disk:x:6:
lp:x:7:
mail:x:8:
news:x:9:
uucp:x:10:
man:x:12:
proxy:x:13:
kmem:x:15:
dialout:x:20:
fax:x:21:
voice:x:22:
cdrom:x:24:crypto
floppy:x:25:crypto
tape:x:26:
sudo:x:27:
audio:x:29:pulse,crypto
dip:x:30:crypto
/etc/group
```

Figura N° 1. 53: Archivo /etc/group.

La estructura para cada uno de los grupos consta de cuatro campos separados por dos puntos (:) y cada campo contiene la información mostrada en la Figura N° 1.54:

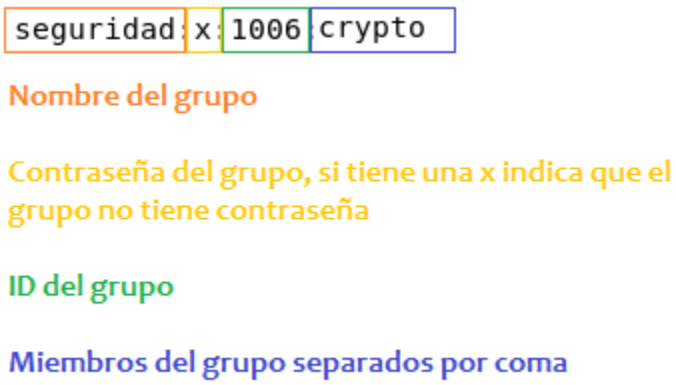


Figura N° 1. 54: Estructura de un grupo en el archivo /etc/group.

Las contraseñas de los grupos se encuentran en otro archivo llamado **gshadow** ubicado en **/etc**, por defecto cuando se crea un grupo, no se coloca contraseña, pero si se quiere agregar contraseña a un grupo, se realiza con **groupmod**.

Permisos: Anteriormente en la Figura N° 1.10, al utilizar el comando **ls -l** se muestra la sección de permisos. Los permisos en Linux se dividen en:

a. **Dueño:** Los permisos correspondientes al usuario dueño del archivo o directorio.

b. **Grupo:** Los permisos correspondientes al grupo al que pertenece el archivo o directorio.

c. **Otros:** Los permisos para los demás usuarios (Perseo, 2019).

Véase Figura N° 1.55:

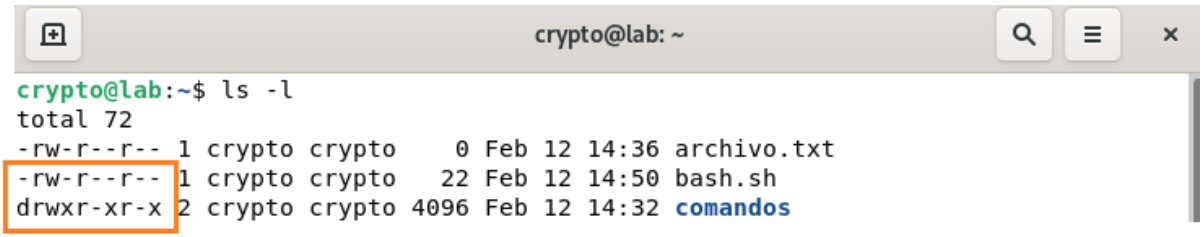


Figura N° 1. 55: Permisos.

En la Figura N° 1.55 dentro del recuadro naranja se encuentran los permisos de un archivo y un directorio, desglosados como se muestra en la Tabla N° 1.3:

Tabla 1. 3: Permisos desglosados.

Directorio	Dueño			Grupo			Otro		
-	r	w	-	r	-	-	r	-	-
d	r	w	x	r	-	x	r	-	x

Como se aprecia en la Tabla N° 1.3 el primer carácter indica si es directorio o archivo, los siguientes tres corresponden a lectura, escritura y ejecución para el dueño, los siguientes tres corresponden a lectura, escritura y ejecución para grupo y los últimos tres corresponden a lectura, escritura y ejecución para otros usuarios.

Como se observa los tres grupos tienen los mismos tres permisos, lectura (r), escritura (w) y ejecución (x), los permisos se pueden otorgar de dos maneras distintas, en modo absoluto, o modo simbólico.

a. **Modo absoluto:** Para la representación en modo absoluto se hace uso del sistema binario y octal, para ejemplificar se toman los datos que se encuentran en la Tabla N° 1.3, el resultado se muestra en la Tabla N° 1.4:

Tabla 1. 4: Permisos en modo absoluto.

Dueño			**Grupo**			**Otro**		
r	w	-	r	-	-	r	-	-
1	1	0	1	0	0	1	0	0
6			4			4		

Como se observa, cada uno de los permisos si se encuentra habilitado pasa a ser un **1 (binario)**, posteriormente se realiza la conversión de cada triada a octal, en el caso del ejemplo en la Tabla N° 1.4, el *dueño* tiene habilitados los permisos de *lectura y escritura*, que en binario para su triada representan **110** y esto traducido a octal corresponde a un **6**; el mismo procedimiento se sigue para la triada del *grupo* y la triada de *otros*.

Otra forma en la cual se puede guiar sin el uso de binarios es tomando los datos de la Tabla N° 1.5:

Tabla 1. 5: Permisos valor en octal.

Permiso	**Valor (octal)**
Lectura	4
Escritura	2
Ejecución	1

Retomando los datos de la Tabla N° 1.4, el archivo tiene los permisos para el *dueño* de *lectura y escritura*, que de acuerdo con la Tabla N° 1.5 sería la suma de **4 y 2** que da un total de **6**; lo mismo se realiza para *grupos y otros*.

El conjunto de estos tres números octales, son los utilizados para otorgar los permisos.

b. **Modo simbólico:** Para el modo simbólico se van a utilizar las siguientes nomenclaturas (Véase Tabla N° 1.6):

Tabla 1. 6: Permisos modo simbólico.

Permisos para	Nomenclatura	Permiso de	Nomenclatura	Asignación	Nomenclatura
Dueño	u	Lectura	r	Agregar	+
Grupo	g	Escritura	w	Eliminar	-
Otros	o	Ejecución	x		
Todos	a				

Asignación de permisos: (González, S., s.f.) Para realizar la asignación de permisos, se realiza mediante el comando **chmod** seguido de los permisos, ya sea en modo absoluto o simbólico, es decir de la siguiente manera: *chmod «permisos» «nombre archivo/directorio» (Véase Figura N° 1.56)*

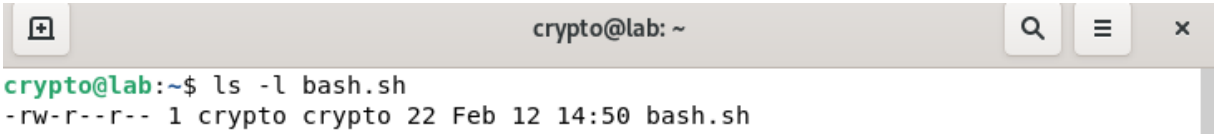


Figura N° 1. 56: Archivo a cambiar permisos en modo absoluto.

Algo que tienen en particular los scripts, es que los usuarios que hagan uso de este deben tener permisos de ejecución.

De forma particular a este archivo se le asignan los *tres permisos al dueño*, *permiso de lectura y ejecución a grupos y permiso de lectura a otros*, con base en la Tabla N° 1.5, los permisos en modo absoluto quedan **754**, usando el comando **chmod**, se realiza el cambio de permisos y se ejecuta el script. *(Véase Figura N° 1.57)*

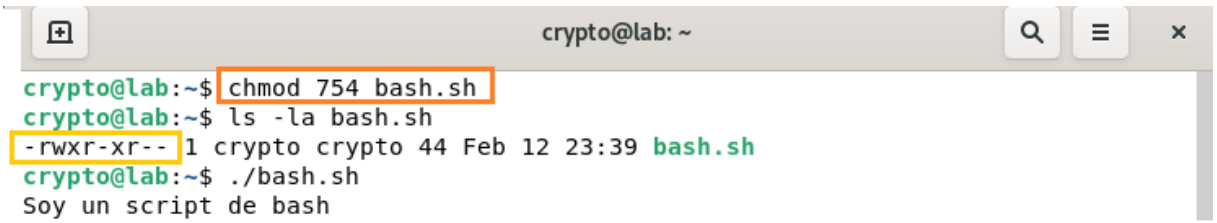


Figura N° 1. 57: Cambio de permisos en modo absoluto.

Como se observa se realiza la asignación de permisos, posteriormente se observa que el cambio de los permisos es reflejado y finalmente se ejecuta el script de bash.

En la Figura N° 1.58 se realiza el cambio de permisos en modo simbólico.

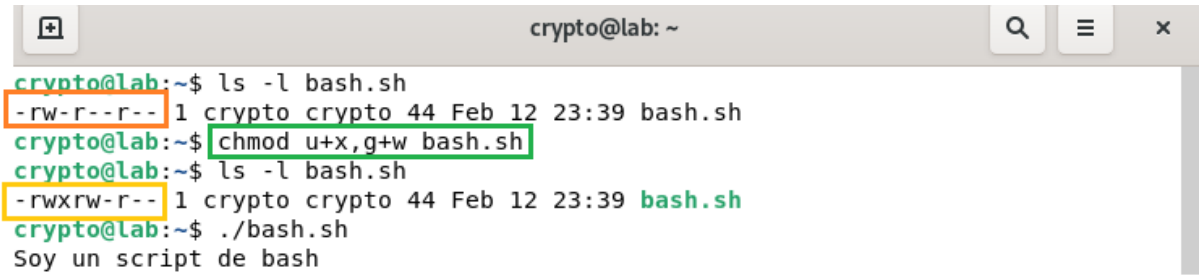


Figura N° 1. 58: Cambio de permisos en modo simbólico.

El cambio de permisos se realiza al mismo archivo, para esto, los permisos son separados por comas. Como se observa, al inicio el archivo para el *dueño solo tiene permisos de lectura y escritura*, y tanto *grupos como otros tienen permisos de lectura*, en este caso, en *grupos se cambia el permiso de ejecución por escritura*, lo demás queda igual a como se estableció anteriormente.

De acuerdo a esto, se observa que al *dueño le faltan permisos de ejecución*, en tanto que a *grupos le faltan permisos de escritura*, para esto se usa **u+x** , de manera que se pueden asignar los *permisos de ejecución al dueño*, y se utiliza **g+w** para asignar *permisos de escritura a grupo*, quedando la línea completa **chmod u+x,g+w bash.sh**, por último se ejecuta el script.

Se pueden aplicar los mismos permisos a todo un directorio haciendo uso **-R** para que estos sean recursivos seguido de los permisos y la ruta en donde se aplicará, en este caso a los directorios y archivos que se encuentren dentro de *Rutas/A* como se indica en la Figura N° 1.59.

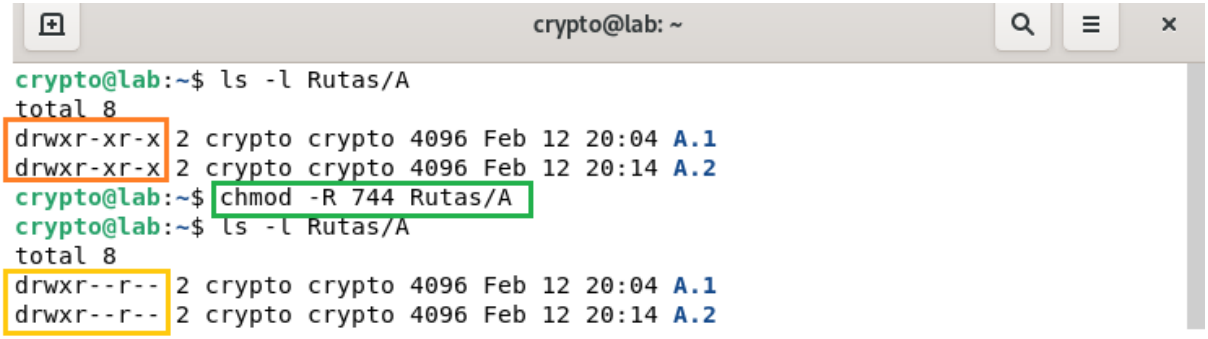


Figura N° 1. 59: Permisos en modo recursivo.

Del mismo modo haciendo uso de ***** que indica lo que sea *«antes de/después de»* según sea el caso, permite aplicar los permisos a muchos archivos que tengan el nombre elementos en común. (Véase Figura N° 1.60)

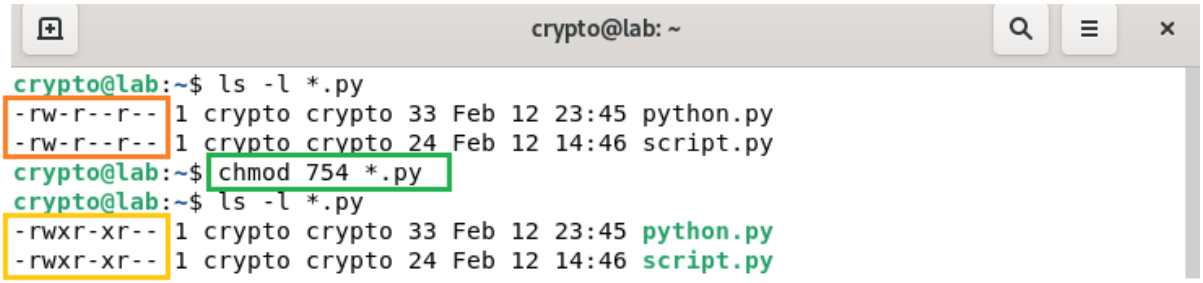


Figura N° 1. 60: Permisos a un conjunto de archivos.

En el ejemplo mostrado en la Figura N° 1.60, se aplican los mismos permisos **754** a todos los archivos que inicien con *lo que sea*, seguido de *.py*.

Cambio de usuario y grupo: Para realizar el cambio de usuario y grupo se hace uso del comando *chown «nuevo usuario»:«nuevo grupo» «nombre archivo/directorio»*

En la Figura N° 1.61 se muestra el cambio de propietario y grupo, de crypto:crypto a crypto:seguridad, por lo cual el usuario crypto es dueño del archivo, solo él puede escribir, leer y ejecutar el script, el grupo de seguridad puede leer y escribir y todos los demás usuarios sólo pueden leer el archivo.

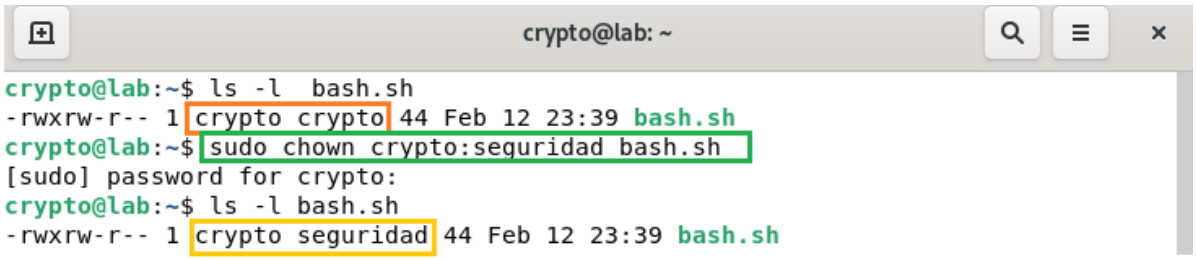


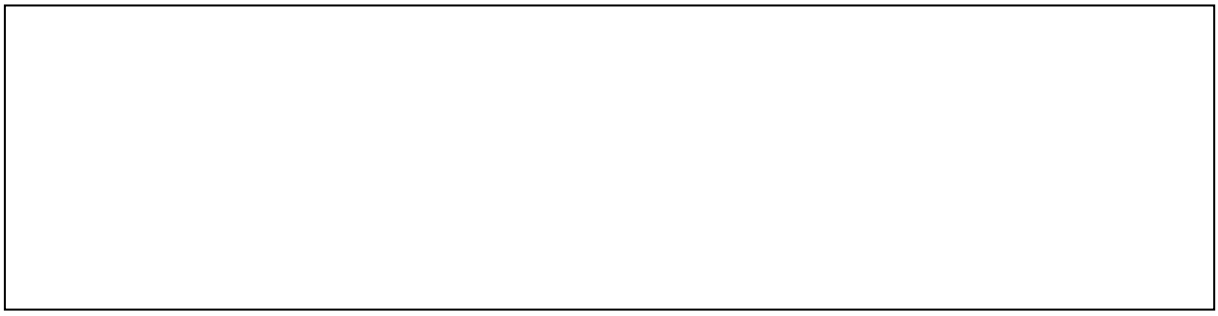
Figura N° 1. 61: Cambio de propietario y grupo.

ACTIVIDAD 1.17: En el directorio */home/crypto/Documents/Permisos*, se encuentran los archivos **primer** y **segundo**, ambos archivos son scripts, por lo cual, a primer debe cambiarle los *permisos en modo absoluto*, y a segundo cambie los *permisos en modo simbólico*, coloque los permisos de modo que ambos scripts deben poder ser ejecutados.

a. Añadir a *crypto* al grupo *seguridad*.

b. Cambie a ambos archivos de grupo en el que se encuentra a *seguridad*, solo cambie al grupo, el dueño sigue siendo crypto.

   En el siguiente recuadro coloque una captura de pantalla en donde se vea lo realizado para los incisos a y b.



Actividad 1.17: Incisos a y b.

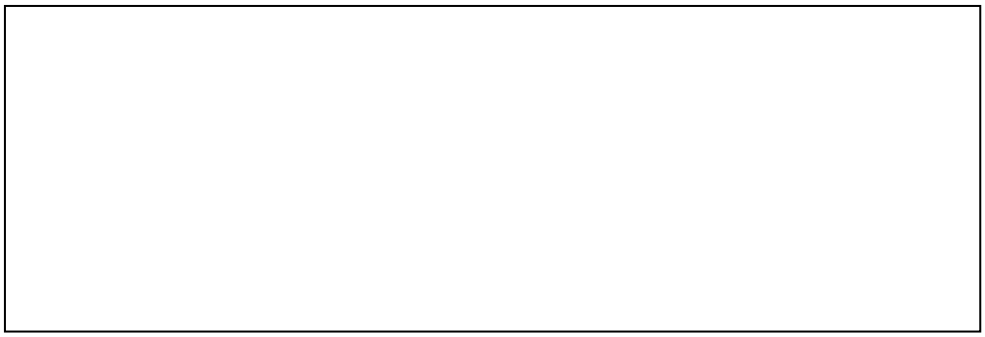
c. Coloque en el siguiente recuadro, según corresponda, una descripción de cómo realizó el cambio de permisos, tomando en cuenta que:

   1. Los scripts lo deben poder ejecutar y visualizar los integrantes del grupo seguridad.
   2. El usuario crypto debe tener permisos de lectura, escritura y ejecución.
   3. Otros usuarios no tienen ningún permiso.

   En el caso de los permisos para *primer*, llene la tabla presentada, como se realizó en la Tabla N° 1.4

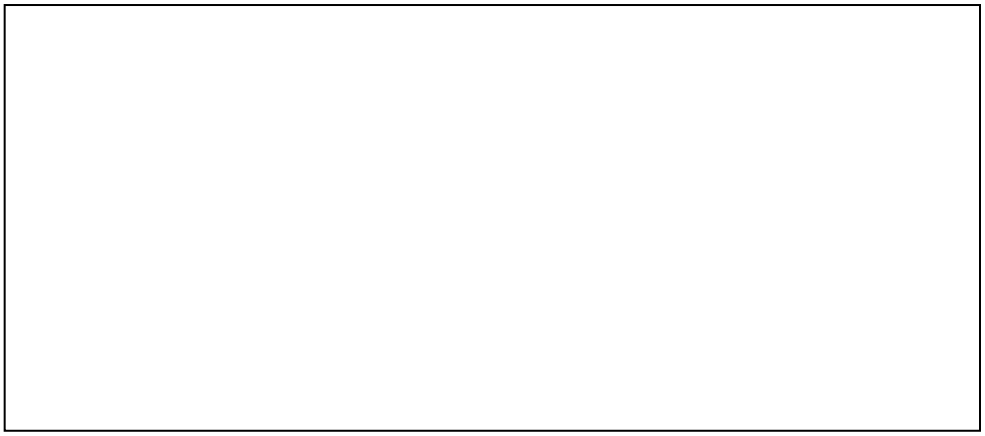
I. Permisos para el script primer.py

Dueño			Grupo			Otro		



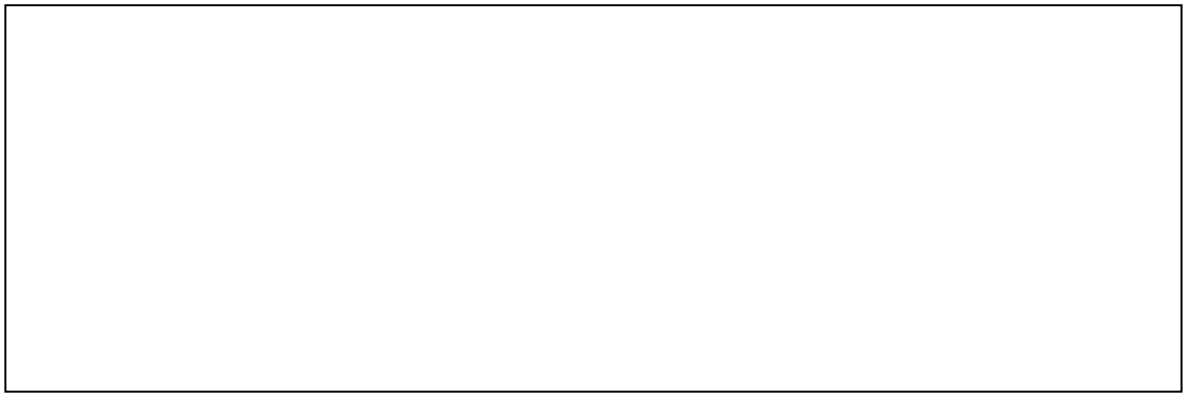
Actividad 1.17: Inciso c.

II. Permisos para el script segundo



Actividad 1.17: Inciso c.

d. Coloque en el siguiente recuadro, una captura de pantalla donde se observe el cambio de permisos de ambos archivos.



Actividad 1.17: Inciso d.

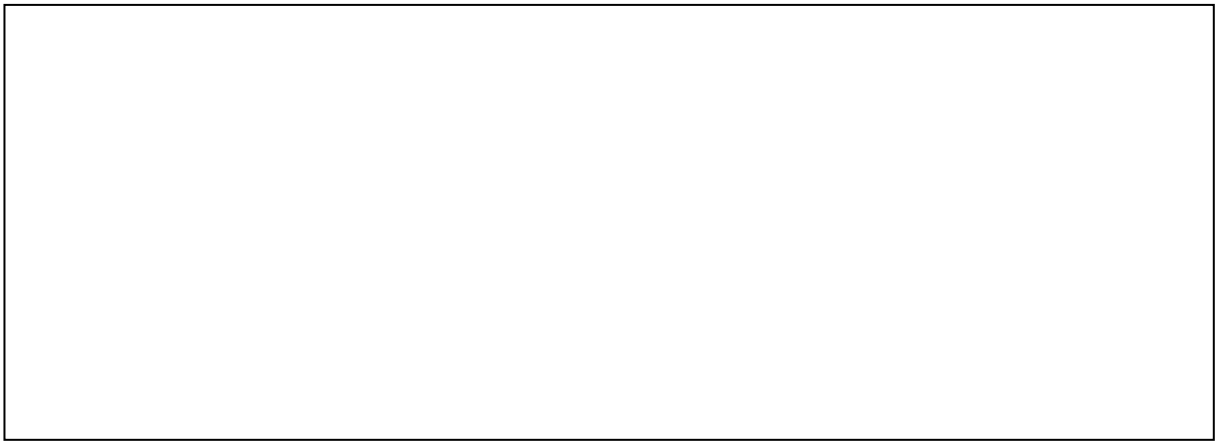
e. Visualice y ejecute cada uno de los scripts.

   Coloque capturas de pantalla de la ejecución de cada uno de los scripts según corresponda.

   Captura para ./primer

Actividad 1.17: Incisos e primer

Captura para ./segundo



Actividad 1.17: Incisos e segundo

Los miembros del grupo seguridad, podrán visualizar y ejecutar los scripts, siempre y cuando tengan acceso también a los directorios en donde se encuentren los scripts. La solución sería copiar los scripts a un directorio nuevo y así no comprometer el directorio home de crypto.

find: Este comando es de gran utilidad cuando se quiere buscar algún archivo o directorio, el cual por alguna razón no se encuentra o no se sabe en dónde está. Su uso es: *find «ruta» «expresión» «nombre archivo/directorio/usuario a buscar» (González, S., s.f.).*

Dentro de las expresiones se puede usar **-name** para buscar por nombre, **-iname** del mismo modo para buscar por nombre, pero sin tomar en cuenta mayúsculas o minúsculas y **-user** para buscar archivos que pertenezcan a un usuario. (Véase Figura N° 1.62)

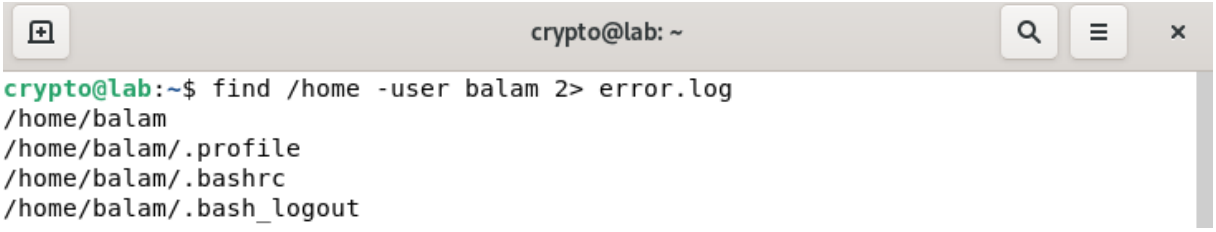


Figura N° 1. 62: Uso del comando find.

En la Figura N° 1.62 se hace uso del comando *find* para listar archivos/directorios pertenecientes al usuario Balam, y para hacer una búsqueda más limpia se envían los errores a un archivo.

Otros comandos:

- **history:** Este comando ayuda a ver la historia de comandos que se han utilizado últimamente
- **script:** Este comando ayuda a crear un archivo que contendrá todo lo que se ejecute en la terminal. Para iniciarlo, basta con colocar *script «nombre»*

  Todo lo que se esté ejecutando en la terminal y los resultados que se arrojen, serán guardados en este archivo, y para terminar y cerrar el archivo ejecutamos «exit»
- **Autocompletar:** Se pueden autocompletar las rutas u opciones para comandos si se hace uso de la tecla tabulador.
- **Comandos usados:** Para ver los comandos que han sido usados últimamente se puede hacer uso de las flechas de dirección ↑↓, esto es para que sea más fácil hacer uso de algún comando que hace poco fue usado y se quiere volver a utilizar.
- **Uso de | (pipe):** Sirve para tomar la salida de un comando como entrada de otro. (Véase Figura N° 1.63)

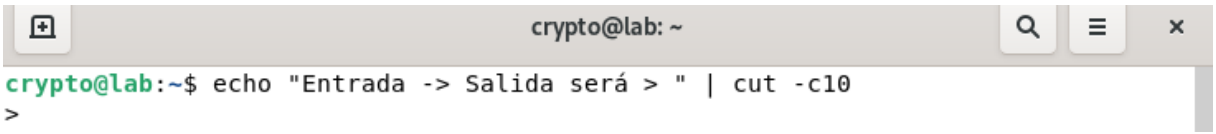
  

  Figura N° 1. 63: Uso de pipe.

  Como entrada se coloca mediante el uso del comando **echo** el mensaje *"Entrada -> Salida será >"* este mensaje pasa al comando **cut**[10] que indica que como salida se mostrará solo el carácter *número 10*, contando los espacios, el carácter que corresponde es el picoparéntesis. El pipe sirve en muchas ocasiones para simplificar las búsquedas.

[10] cut: Comando utilizado para eliminar secciones de líneas en archivos de texto.

apt: El comando apt sirve para instalar, buscar actualizar y eliminar paquetes. Para hacer uso de este comando se necesita ser usuario root.

Consultar dirección IP y MAC: El comando **ip address** muestra las interfaces de red con la que cuente el equipo y de cada uno muestra sus datos de red, tal como la IP, MAC.

man: Este comando funciona como manual de cada uno de los comandos con los que cuenta Linux. Se utiliza *man «comando»* y cuenta con distintas secciones. Inclusive man tiene su propio manual. (archlinux, 2023) Al ejecutar **man man** se obtiene más información del manual, sus secciones y como utilizarlo (Veáse Figura N° 1.64).

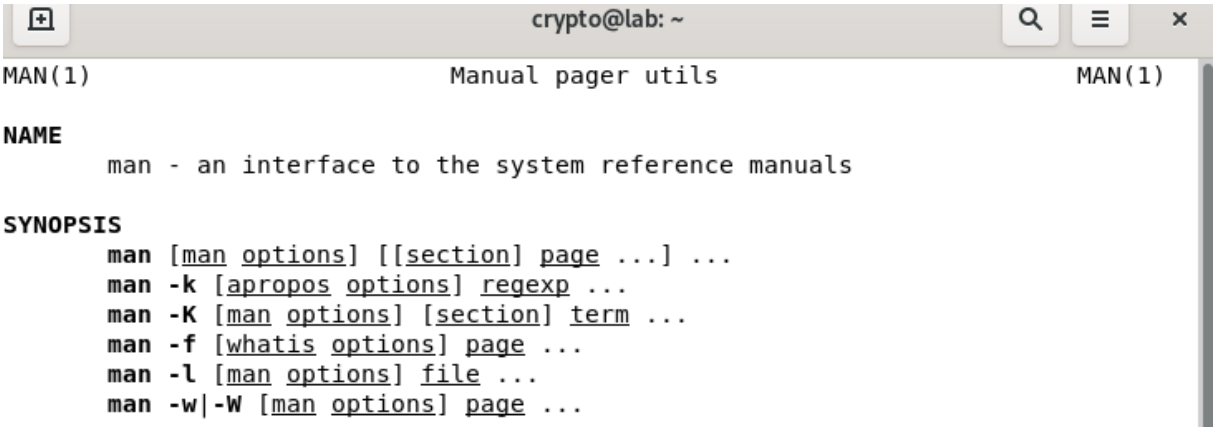


Figura N° 1. 64: Manual de man, secciones con las que cuenta.

Conclusiones:

1. ¿Por qué es importante que sólo quien administra sea usuario root? ¿es conveniente que lo sea en todo momento?

2. ¿Cuál es la diferencia entre las rutas absolutas y las rutas relativas?

3. ¿Es correcto usar chmod 777 «archivo/directorio»? ¿Sí o no y por qué?

4. ¿Cómo se revisa con uso de man los archivos passwd, shadow y group?

5. Comentarios o conclusiones adicionales:

UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

ACTIVIDADES PRÁCTICAS: CRIPTOGRAFÍA

2. CONCEPTO DE FUNCIONES HASH

DOCENTE

CONCEPTO DE FUNCIONES HASH

Objetivo

En esta práctica, se espera que el estudiante conozca los diferentes tipos de funciones hash que se han desarrollado, las identifique, aprenda sobre algunas herramientas para generarlas y comprenda las aplicaciones que estas funciones tienen.

Justificación

La obtención y verificación de funciones hash es de gran utilidad para proteger la confidencialidad de las contraseñas y garantizar la integridad de los datos. Los conceptos aprendidos en esta práctica serán de gran utilidad para comprender la integridad de la información, además de que serán utilizados posteriormente.

Introducción

Es necesario tener claro que «cifrar» y obtener valores «hash» son dos procesos diferentes con objetivos también distintos. Cuando se cifra se hace uso de una clave con la cual se va a realizar tanto el cifrado como el descifrado si se habla de cifrado simétrico, si es cifrado asimétrico será una clave para cifrar y otra para descifrar. El tema del cifrado se abordará en la siguiente práctica.

A diferencia del cifrado que es usado para garantizar la confidencialidad de la información, el proceso correspondiente a la obtención del valor hash es que éste es utilizado para garantizar la integridad de la información, esto es, garantizar que la información no ha sido modificada. A partir del texto a proteger se obtiene un hash, para lo que este no tiene inversa y no utiliza una clave. El hash consta de una longitud fija[1].

Equipo y material necesario

Máquina Virtual Debian, para montarla en su equipo puede seguir el «Montaje de máquina virtual», el cual es un manual de Instalación MV, en el que se encuentran las instrucciones tanto para VMware como para VirtualBox. Así mismo puede consultar el Anexo D que contiene ejemplos de cómo debe colocar sus respuestas en la práctica.

Desarrollo

1 «La triada de la seguridad de la información» consta de confidencialidad, integridad y disponibilidad.

a) Confidencialidad: Solo aquellos autorizados pueden acceder a la información
b) Integridad: La información no se puede modificar sin autorización
c) Disponibilidad: La información estará utilizable siempre que se necesite

Como definición formal, un hash es: «Un algoritmo matemático que trasforma cualquier bloque arbitrario de datos en una nueva serie de caracteres con una longitud fija. Independientemente de la longitud de los datos de entrada, el valor hash de salida siempre tendrá la misma longitud» (López, A., 2023), tomando en cuenta que dependerá del tipo de hash para determinar la cantidad de caracteres que conformen la longitud.

MD5 (RFC 1321): Este algoritmo fue desarrollado por Ronald Rivest y está basado en dos algoritmos anteriores MD2 y MD4. Todos estos protocolos producen un número de **128 bits**, que corresponde a **32 caracteres**.

Sin embargo, en 2005 se encontraron vulnerabilidades en su desarrollo, ya que fue capaz de crear dos certificados X.509 distintos con igual hash MD5 (UPM, 2017), lo cual rompía con la propiedad de integridad que caracteriza a las funciones hash.

Puede consultar su documentación en https://www.rfc-editor.org/rfc/rfc1321

Se puede obtener la función hash de un documento, de una palabra, o una frase, haciendo uso de *md5sum* en la terminal de Linux (Archivo Geek, 2017).

Como ejemplo, se tiene un archivo, llamado *primero*, ubicado en */home/crypto/Documents/Hash/MD5*, este archivo contiene la palabra *Criptografía*, de manera que, para obtener el **md5** del archivo este se realiza como: *md5sum «nombre del archivo» (Véase Figura Nª 2.1)*

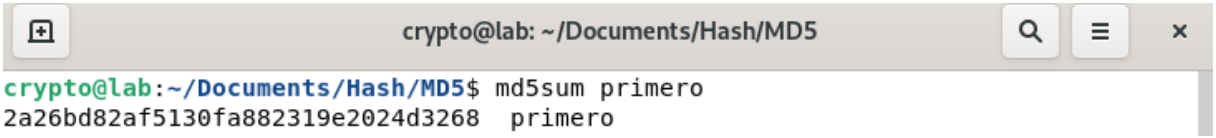


Figura N° 2. 1: MD5 de un archivo.

ACTIVIDAD 2.1: Realice una copia del archivo *primero*, puede poner el nombre primero_v2. Modifique el contenido del archivo primero_v2, que acaba de copiar, de Criptografía a Criptografia (sin acento). Obtenga MD5 de ambos archivos, Coloque captura de pantalla en el siguiente recuadro y conteste: Si ambos archivos tienen el mismo nombre, ¿El MD5 es igual? ¿Por qué? Coloque su respuesta en el recuadro de texto.

Actividad 2.1: Comparar MD5.

SHA: Secure Hash Algorithm, tiene principalmente dos variantes: SHA-1 Y SHA-2 (Landman, N. Williams, C. Ross, E. y Khim, J., s.f.)

1. **SHA-1:** Este algoritmo fue desarrollado en 1993 por la agencia de estándares de Estados Unidos, el Instituto Nacional de Estándares y Tecnología «NIST». Este algoritmo produce una salida de **160 bits** que corresponden a **40 caracteres** sin importar la longitud de la entrada.
   Puede consultar su documentación en RFC 3174: US Secure Hash Algorithm 1 (SHA1)

2. **SHA-2:** En 2001 la Agencia de Seguridad Nacional «NSA» de los Estados Unidos, una familia de algoritmos hash que constituye en cuatro funciones: SHA-224, SHA-256, SHA- 384 y SHA-512. De los cuales los reconocidos son SHA-256 que produce una salida de **64 caracteres** y SHA-512 que produce una salida de **128 caracteres**.
   Puede consultar su documentación en https://www.rfc-editor.org/rfc/rfc6668

SHA-3 (UPM, 2015): Fue publicado por el NIST (National Institute of Standards and Technology) en 2015, después de un concurso lanzado en 2007 debido a las vulnerabilidades de las funciones hash anteriores.

El algoritmo ganador fue Keccak. SHA-3 funciona bajo el método de esponja, donde ingresa (absorbe) cualquier cantidad de datos y genera (exprime) la cantidad de datos que se desee.

Puede consultar su documentación en NIST Releases SHA-3 Cryptographic Hash Standard | NIST

ACTIVIDAD 2.2: Realice la copia del archivo *Ingenieria.txt* ubicado en el directorio */home/crypto/Documents/Lecturas* a un nuevo directorio llamado *SHA* que se debe ubicarlo en el directorio */home/crypto/Documents/Hash*. Obtenga el hash para SHA1, SHA-256 y SHA-512 de dicho archivo usando las funciones de Linux como se indica en la siguiente tabla.

Coloque la captura de pantalla en el recuadro correspondiente.

Hash	Captura del hash
SHA1 *sha1sum «nombre del archivo»*	
SHA-256 *sha256sum «nombre del archivo»*	
SHA-512 *sha512sum «nombre del archivo»*	

Actividad 2.2: Obtener hash de distintas versiones de SHA.

Como notó en la actividad anterior, la cantidad de caracteres dependerá del hash utilizado, independientemente de la longitud de entrada. La cantidad de caracteres es una característica que permite identificar el tipo de hash, tal como se observa en la actividad que acaba de realizar.

Ahora, compruebe que el hash que acaba de obtener tenga los caracteres correspondientes e incluya en el recuadro correspondiente la captura de pantalla.

Ayuda: Python tiene una función llamada *len*, que ayuda a contar los caracteres de una cadena. Copie la cadena que dio como resultado el hash y use la función len de Python. Para ejemplificar, se realiza lo solicitado con MD5 para el archivo utilizado en la Figura N° 2.1, el ejemplo se muestra en la Figura N° 2.2:

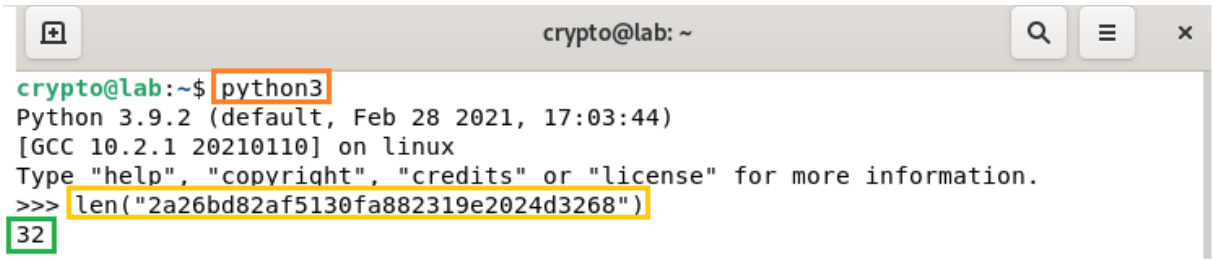


Figura N° 2. 2: Uso de len para comprobar caracteres devueltos por MD5.

Como se observa en la Figura N° 2.2, la función devuelve el número **32** que representa la cantidad de caracteres correspondientes a MD5.

En la captura se solicita enmarcar la parte en donde se hace uso de la función en color naranja, y su resultado en color verde. Verifique que está usando la cadena correcta para evitar errores.

Hash	Captura de comprobación de caracteres
SHA1 40 caracteres	
SHA-256 64 caracteres	

Actividad 2.2: Comprobación de caracteres para las distintas versiones de SHA.

Hash	Captura de comprobación de caracteres
SHA-512 128 caracteres	

Actividad 2.2: Comprobación de caracteres para las distintas versiones de SHA.

RIPEMD-160 (RFC 2857): Cuyo acrónimo es Race Integrity Primituves Evaluation Message Digest es una función hash que como su nombre lo dice, consta de **160 bits**, es decir, **40 caracteres**. Propuesto por Hans Dobbertin, Antoon Bosselaers, y Bart Preneel. Fue realizado para reemplazar a las funciones hash de **128 bits** como MD5 y RIPEMD en dicha versión (Wikipedia, 2024).

Puede consultar su documentación en ietf.org/rfc/rfc2857.txt

Existen otras herramientas que ayudan a obtener hashes, una de ellas es OpenSSL.

OpenSSL es una herramienta de código abierto de propósito general para la criptografía y la comunicación segura (OpenSSL Project, s.f.).

OpenSSL proporciona distintas herramientas, algunas de las cuales serán utilizadas en el presente manual.

ACTIVIDAD 2.3: Revise el manual de OpenSSL y encuentre:

1. ¿Con qué parámetro se obtiene una lista de los comandos disponibles para hashes en OpenSSL?

   openssl

Coloque una captura de pantalla del resultado

Actividad 2.3: Pregunta 1.

¿Cómo revisó el manual de OpenSSL?

En la Figura N° 2.3 se hace uso de OpenSSL para obtener el hash del archivo *Ingeniería.txt*, que fue utilizado en la Actividad 2.2, se realiza de la siguiente manera: *openssl «opción hash» «archivo»*

En el apartado opción hash, se hace referencia a los comandos obtenidos al ejecutar lo indicado en la Actividad 2.3 en la primera pregunta.

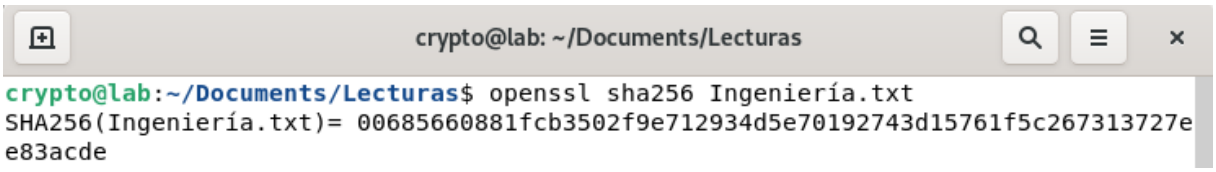


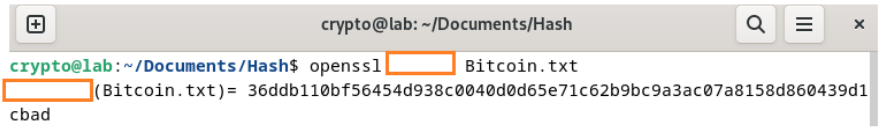
Figura N° 2. 3: Uso de OpenSSL con SHA-256.

ACTIVIDAD 2.4: Use OpenSSL para obtener el Hash del archivo *Bitcoin.txt* ubicado en */home/crypto/Documents/Lecturas*, copie dicho archivo al directorio Hash que creó anteriormente en la Actividad 2.1

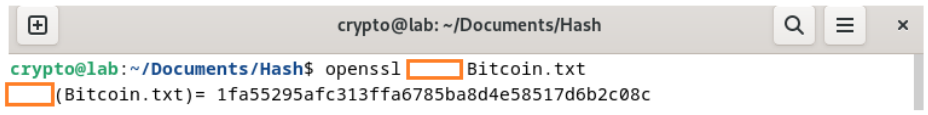
a) ¿Cuáles son los permisos que tiene asignados el archivo *Bitcoin.txt*? Además, explique cuáles son los cambios necesarios para que pueda hacer uso del archivo.

Actividad 2.4: Permisos asignados a Bitcoin.txt.

b) Complete la siguiente tabla según corresponda a la captura del hash o escribir el tipo de hash:
   a. Añadir la captura con la obtención de la función hash.
   b. Indicar en el recuadro de qué función hash se trata.

Hash	Captura del hash
RIPEMD-160	
	
MD5	

Actividad 2.4: Reconocer función hash.

Hash	Captura del hash
	
SHA-512	

Actividad 2.4: Reconocer función hash.

ACTIVIDAD 2.5: De acuerdo con la siguiente información dada, investigue de qué función hash se trata:

> Función hash que se caracteriza por ser más rápida que MD5, SHA-1, SHA-2, SHA-3 y BLAKE2, además de ser más segura que MD5 o SHA-1. La longitud por defecto es de 256 bits (Darkcrizt, 2023). Este algoritmo está diseñado para verificar la autenticación de archivos, mensajes y creación de firmas digitales. Algunos de sus desarrolladores son Jack O'Connor y Samuel Neves. Fue publicado en el 2020 y tiene su portafolio de desarrollo en GitHub.

Utilice la función hash descrita en el ejercicio anterior y úsela para obtener el hash del archivo *Bitcoin.txt*. En el repositorio de GitHub del hash, se indica cuál es el comando que se utiliza.

La actividad debe realizarla desde el usuario crypto, en él no es necesario instalar nada, solo hacer uso del comando que termina al igual que los utilizados anteriormente con **sum.** *En caso de querer hacer uso del comando desde otro usuario, debe realizar su instalación, la cual está indicada en el Github.*

Pista: Son dos caracteres los que completan el comando **_ _ sum**, y se usa del mismo modo que en la actividad 2.2, es decir: _ _*sum «nombre del archivo»*

Coloque una captura de pantalla del uso de la función hash en el siguiente recuadro:

Actividad 2.5: Uso de la función descrita para obtener el hash de un archivo.

ACTIVIDAD 2.6: Cree un programa con el cual se pueda hacer uso de funciones hash para archivos o cadenas de texto. Las funciones hash a usar serán SHA1, SHA-256, SHA-512 y RIPEMD160. Si desea utilizar Python, puede hacer uso del módulo *hashlib*, o bien, realizarlo en el lenguaje de su preferencia o la que su docente indique.

Adicionalmente agregue una opción para que pueda comparar si dos archivos tienen el mismo hash.

En la salida, el programa debe arrojar:

a. Si se está obteniendo el hash de un archivo o de una frase, en el caso del archivo se debe comprobar si este existe o no.
b. La función hash utilizada.
c. En el caso de comparación de archivos, se debe mostrar el nombre de los archivos, el resultado del hash y una leyenda que indique si son iguales o no.

En la Figura N° 2.4 se muestran ejemplos de la obtención de hash para un archivo, una palabra y una frase mediante el uso de banderas (con el uso del módulo *argparse*):

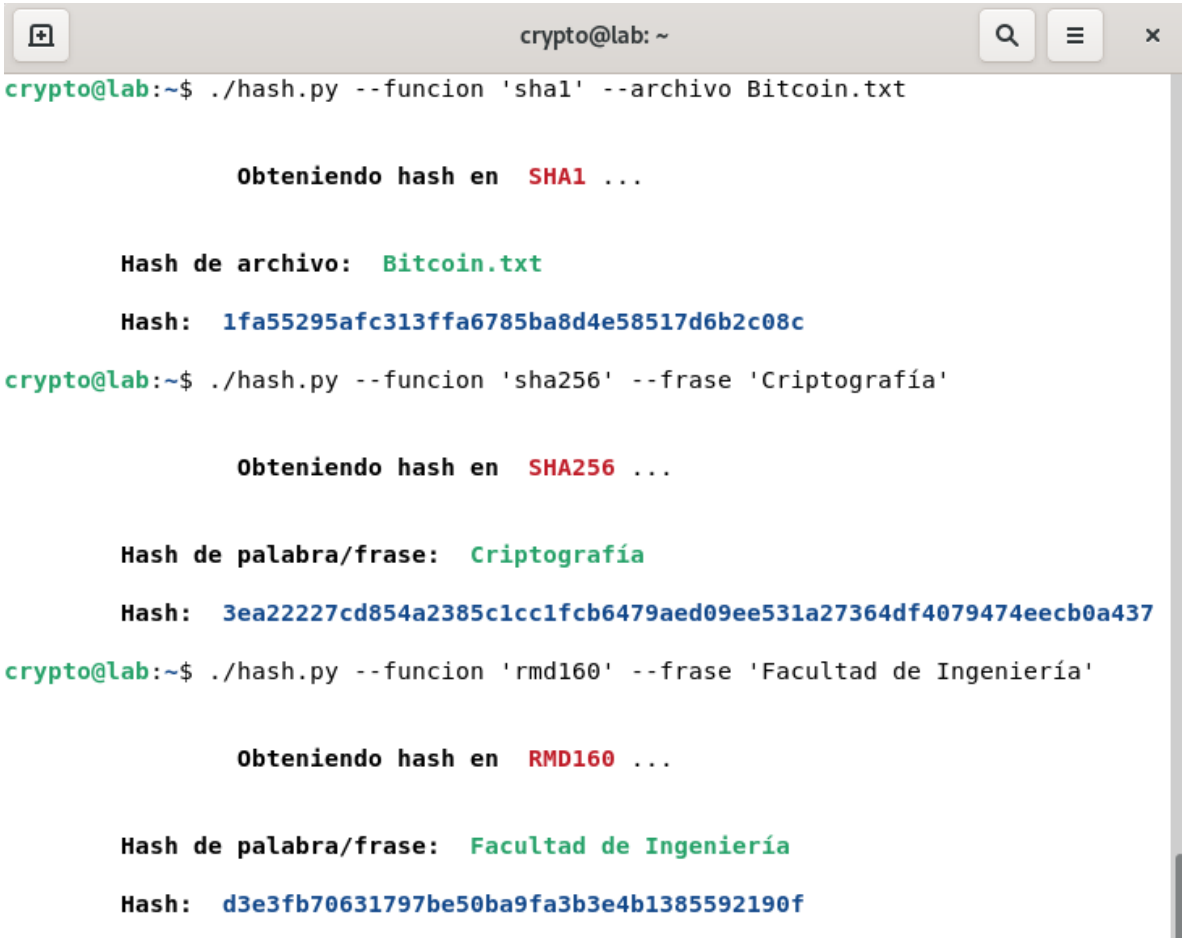


Figura N° 2. 4: Ejemplos ejecución script.

En la Figura N° 2.5 se muestran ejemplos de salida para la comparación de archivos para *Bitcoin.txt* e *Ingeniería.txt*, archivos de contenido diferente, por lo cual el resultado es que tanto el hash como los archivos son diferentes.

Para mostrar el caso de archivos iguales, se copia el archivo *Bitcoin.txt* con un nuevo nombre a *Criptomoneda.txt*, al compararlo, se muestra que tanto la función hash como los archivos, son iguales, ya que recuerde que lo que el hash compara es el contenido, no el nombre.

Su script también debe realizar un manejo de errores y excepciones, por lo cual debe evitar que el programa arroje dichos mensajes y colocar los propios, tal como se muestra en la Figura N° 2.6.

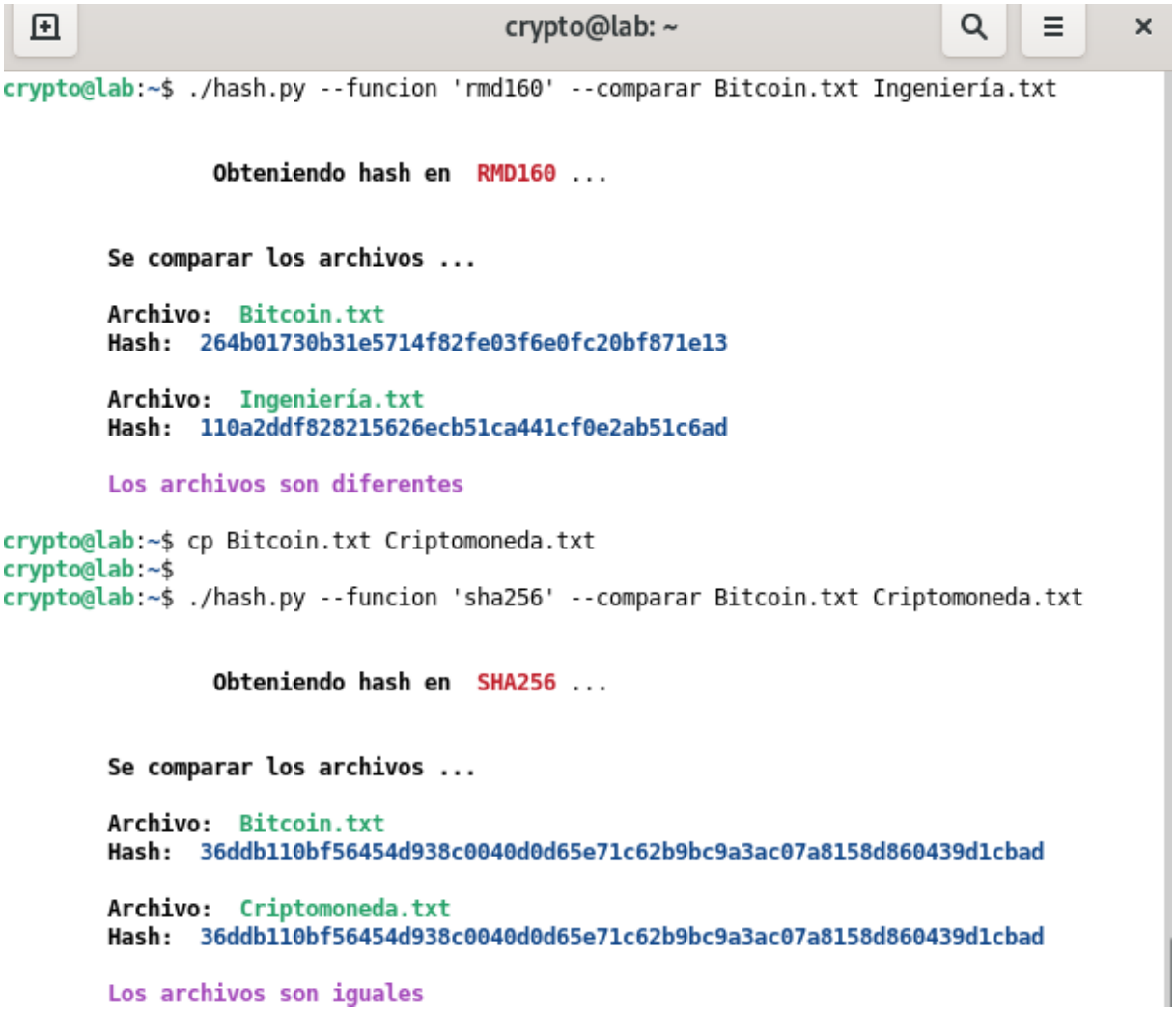


Figura N° 2. 5: Ejemplo comparación de archivos.

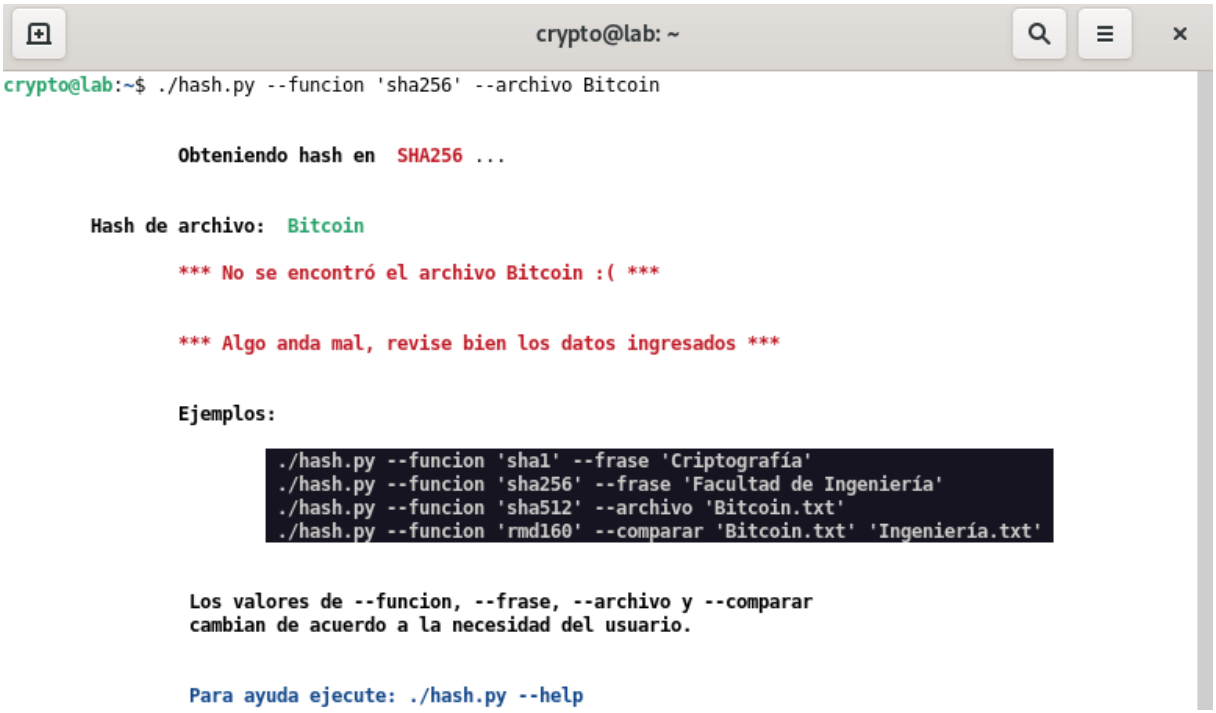


Figura N° 2. 6: Ejemplo de manejo de errores en el script.

Coloque en el siguiente recuadro una captura de pantalla de la ejecución de su script donde se muestre la obtención de hash para un archivo, una palabra o frase y la comparación de archivos:

Actividad 2.6: Ejecución de script creado por el alumno.

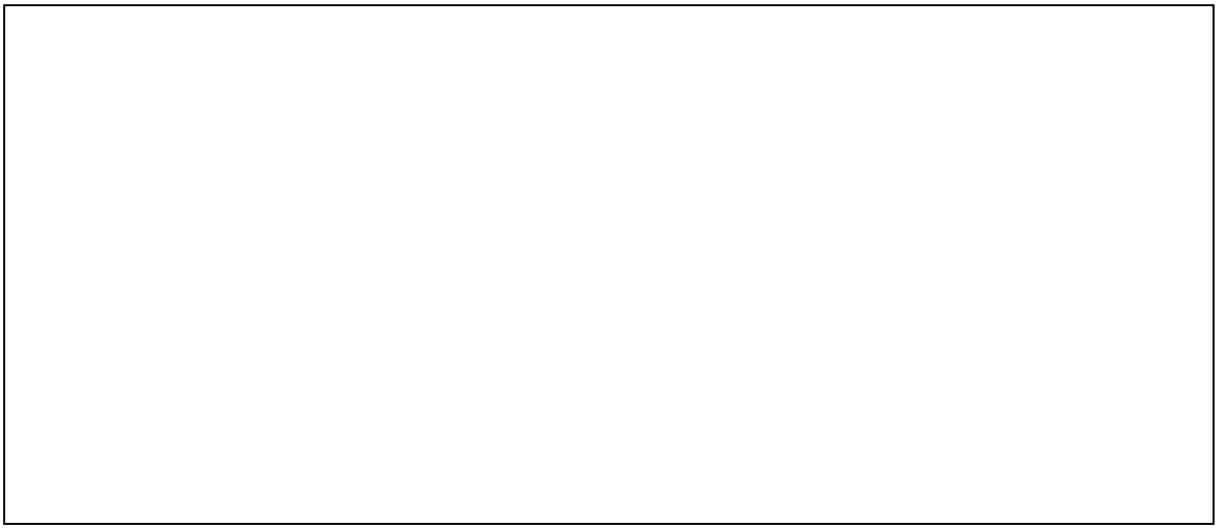
El script debe estar debidamente comentado, en las primeras líneas debe tener su nombre completo. El método de entrega dependerá de la indicación de su docente.

ACTIVIDAD 2.7: MD5 es un algoritmo que fue declarado como vulnerable, gracias al trabajo de Xiaoyun Wang[2], él y su equipo comprobaron que al obtener el MD5 de dos archivos, pueden existir colisiones, esto quiere decir que, aunque los archivos sean diferentes se obtiene el mismo Hash.

En el directorio */home/crypto/Documents/Colisión* hay un directorio llamado *evilize-0.2.tar.gz* con el desarrollo de un software, se trata de una demostración de colisión desarrollada por Patrick Stach, basándose en el trabajo desarrollado por Wang. Esta versión fue modificada y distribuida por Peter Selinger[3].

En los casos en donde se encuentre un recuadro colocar captura de pantalla de lo realizado

1. Descomprima el archivo: *sudo tar -xvf evilize-02.tar.gz*
2. Ingrese al directorio que acaba de descomprimir
3. Inspeccione los archivos que tiene el directorio.
4. Estando en el directorio, ejecute el comando **make**, como lo indica el README, al ejecutar este comando, creará 3 nuevos archivos «*evilize*», «*md5coll*» y el archivo objeto «*goodevil.o*». En caso de que tenga problemas al hacer uso de make, asigne permisos de ejecución.



Actividad 2.7 quinto paso: Ejecución de make.

2 **Wang Xiaoyun.** How to Break MD5 and Other Hash Functions. Recuperado de: http://merlot.usc.edu/csac-f06/papers/Wang05a.pdf

3 **Selinger Peter.** MD5 Collision Demo. Recuperado de: https://www.mathstat.dal.ca/~selinger/md5collision/

6. Compile el programa hello-erase.c vinculado a goodevil.o ejecutando lo siguiente:

«gcc hello-erase.c goodevil.o -o hello-erase»

7. Ejecutar el siguiente comando para la creación del vector:

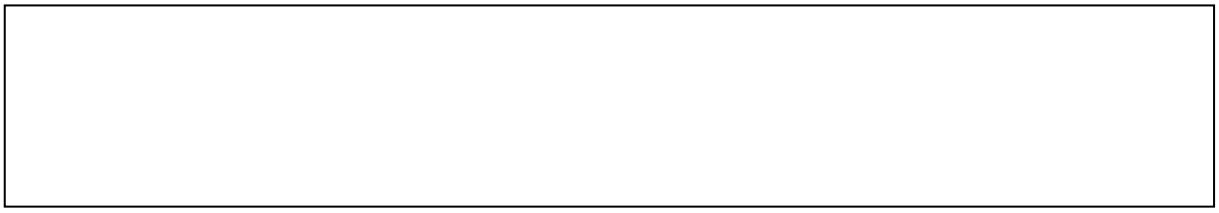
«./evilize hello-erase –i»

Actividad 2.7 séptimo paso: Obtención del vector.

8. Con el vector obtenido, ejecute lo siguiente, donde debe cambiar *vector* por el **vector** obtenido en el paso anterior:

*«./md5coll **vector** > init.txt»*

Este proceso genera la colisión MD5. Este paso puede *tardar algunas horas*, todo dependerá de los recursos de su equipo. Se dará cuenta que el proceso finalizó en el momento en el que en la terminal se visualice *(done)*.



Actividad 2.7 octavo paso: Generador de colisión MD5.

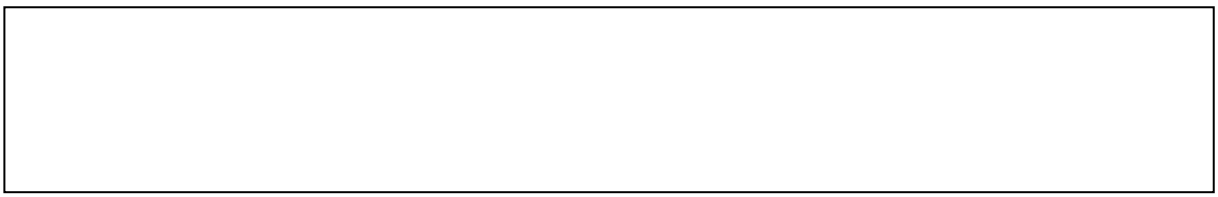
Explique, ¿Qué es lo que realiza > init.txt?

Actividad 2.7 octavo paso: Generador de colisión MD5.

9. Se crean dos programas llamados «good» y «evil», ejecutando lo siguiente:

« ./evilize hello-erase –c init.txt –g good –e evil»

Quien realiza la colisión es el script *evilize*, puede revisar el mismo script, o bien, el README que se encuentra en el directorio que se descomprimió, para saber más acerca de la colisión.



Actividad 2.7 noveno paso: Generador de programas a utilizar para la colisión.

10. Al obtener el MD5 de los programas generados, se observa que el valor del hash es el mismo, como se muestra en la Figura N° 2.7:

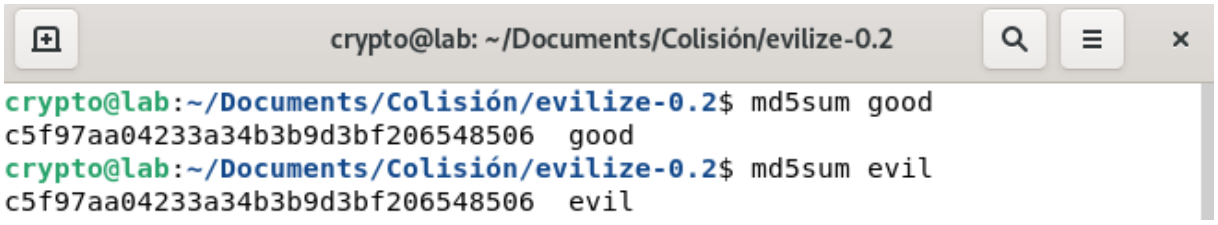
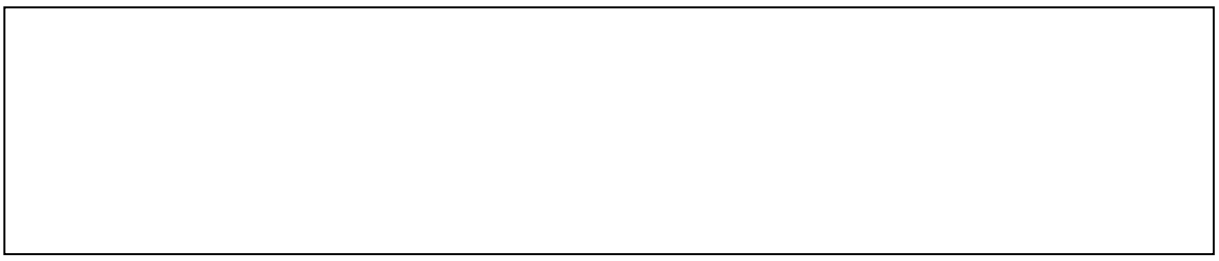


Figura N° 2. 7: Generador de programas a utilizar para demostración de colisión.



Actividad 2.7: Comprobación de colisión en MD5 de dos archivos.

Si se ejecutan los scripts *good* y *evil*, se dará cuanta que son programas distintos.

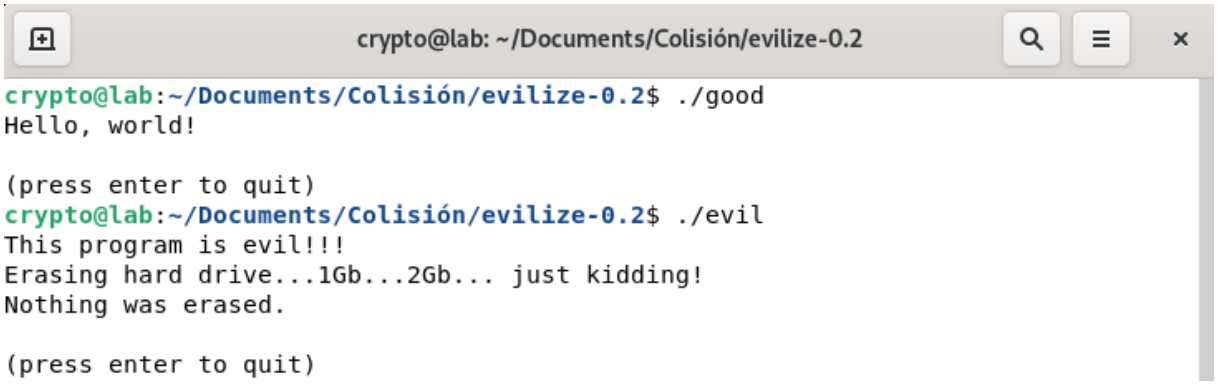


Figura N° 2. 8: Comprobación de que los archivos son distintos.

ACTIVIDAD 2.8: Las funciones de derivación de clave son utilizadas para las contraseñas que son almacenadas en una base de datos, regularmente esto se ve en un formulario de registro, ya que por seguridad es mejor hacer uso de estas funciones que guardar las claves en claro.

Las funciones de derivación de clave se consideran como no vulnerables a ataques de diccionario o fuerza bruta, son lentas, con el propósito de dificultar los ataques.

La robustez de estas funciones consiste en que además de hacer uso de **salt**[4], la mayoría hace uso de funciones hash, que es aplicada a la combinación de la contraseña con salt, el resultado de esto se repite N veces, comúnmente arriba de las cinco mil iteraciones, de manera que así se garantice su seguridad (KeepCoding, 2022).

Una de estas funciones de derivación es Argon2 . Diseñado en 2015 y que resultó ser el ganador de un concurso de descifrado de contraseñas. Tiene distintas versiones, Argon2i, cuyo propósito es mantener la resistencia a ataques de canal lateral; por otro lado, Argon2d es más resistente a ataques de crackeo de GPU. Argon2 hace uso de Blake2 como hash. *(Wikipedia, 2023)*

Descrito lo anterior, su actividad consiste en crear un programa en donde se haga uso de Argon2 para la verificación de contraseñas. Un ejemplo de esto es un formulario de registro, en donde verifica que la contraseñas sean iguales, si estas lo son, hace la simulación de que sus datos han sido registrados, si no lo son las rechaza, limpia los campos de contraseña y pide al usuario que vuelva a ingresar los datos. Si son iguales, le indica al usuario que se almacenaron sus datos, limpia el formulario y da la opción de realizar un nuevo registro. Puede hacer uso de Python para el desarrollo de su script, instalando la librería de *Argon2*, y para la interfaz puede hacer uso de *Tkinter (Amos, D., s.f.)*, o bien, realizarlo en el lenguaje de su preferencia o la que su docente indique.

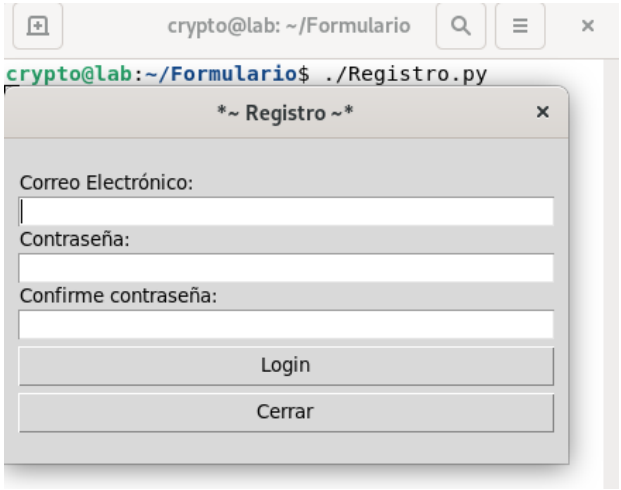
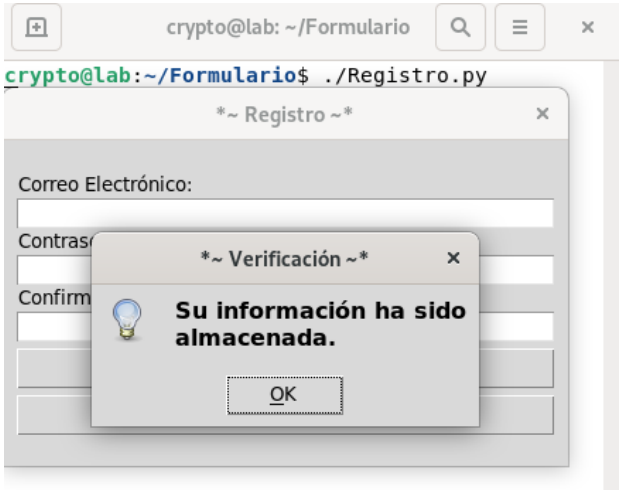
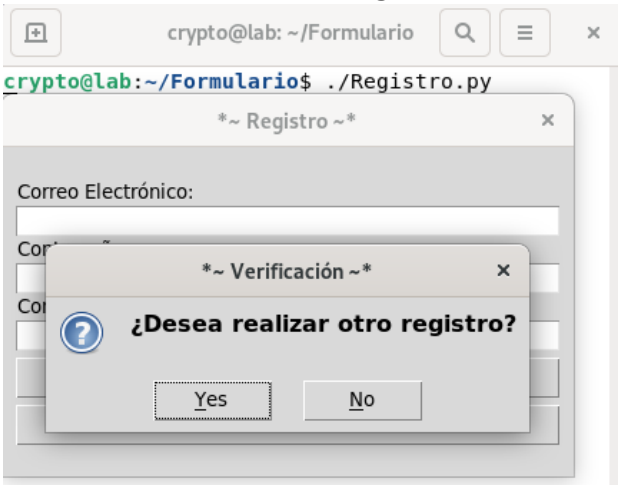
NOTA: Al ser simulación solo basta con que se indique si son iguales o no, no es necesario guardar la información en algún archivo, el propósito principal de esta actividad es hacer uso de Argon2.

El script debe estar comentado en su totalidad. En caso de usar otro lenguaje dentro del script mencione porqué el uso de ese lenguaje y haga saber a su docente qué se debe instalar para tener éxito al usar su programa. La entrega del script dependerá de lo indicado por su docente.

En la siguiente tabla se muestran ejemplos que pueden servirle para realizar su programa:

4 **Salt:** Caracteres aleatorios que son agregados ya sea al inicio a la final de una contraseña para garantizar la seguridad y así evitar la vulnerabilidad.

Tabla 2. 1: Ejemplo de formulario con el uso de Argon2.

<table>
<tr><th colspan="2">Formulario de Registro.</th></tr>
<tr><td>inicio de formulario.
 
</td><td>Llenado con contraseñas distintas.
 
</td></tr>
<tr><td>Llenadcd o de manera correcta.
 
</td><td>Opción para realizar otro registro.
 
</td></tr>
<tr><td colspan="2">Opciones de alerta, ningún campo se permite en blanco.
 
</td></tr>
</table>

En la siguiente tabla coloque las capturas de su script tal como en la Tabla 2.1:

Formulario de Registro.	
inicio de formulario.	Llenado con contraseñas distintas.
Llenado de manera correcta.	Opción para realizar otro registro.
Opciones de alerta, ningún campo se permite en blanco.	

Actividad 2.8: Implementación de formulario con el uso de Argon2.

Conclusiones:

1. Escriba, ¿Qué es un hash?

2. Escriba, ¿En dónde es usado SHA-256?

3. Explique, ¿Cuál es la diferencia entre derivación de clave y hash?

4. ¿Qué fue lo más destacado de la realización de los scripts?

5. Comentarios o conclusiones adicionales

Actividades prácticas

Sección 2: Criptografía Simétrica

UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

ACTIVIDADES PRÁCTICAS: CRIPTOGRAFÍA

3. DATA ENCRYPTION STANDARD

DOCENTE

DATA ENCRYPTION STANDARD

Objetivo

Conocer y entender el modo de cifrado y descifrado del algoritmo Data Encryption Standard (DES) a través de su programación.

Justificación

Los algoritmos simétricos se caracterizan por utilizar la misma clave para cifrar y descifrar. El algoritmo Data Encryption Standard (DES) es uno de los que cumple con esta característica. A través de la programación, usted aprenderá cómo funciona.

Introducción

En 1973 el NBS (National Bureau of Standards) lanzó la convocatoria para desarrollar el Data Encryption Standard (DES), que es un algoritmo de cifrado simétrico por bloques. IBM (International Business Machines), presenta un algoritmo basado en uno llamado «Lucifer» de Horst Feistel, miembro del equipo de IBM(International Business Machines), el cual fue modificado por equipo de investigadores de la empresa y un equipo externo con miembros de la NSA (Agencia Nacional de Seguridad) (UPM, 2015).

Aplicados los cambios, dicho algoritmo fue adoptado como estándar para las comunicaciones seguras y el almacenamiento de información no clasificada por el Gobierno de los EE. UU. En un principio la idea de la NSA era implementar este algoritmo en hardware, por lo que el algoritmo debería mantenerse en secreto. Finalmente, el algoritmo se hizo público en 1977 y publicado en FIPS PUB[1] 46-1 y su implementación en software se hizo posible (González Velarde, E.J., s.f.)

Puede consultar su documentación en https://doi.org/10.6028/NIST.FIPS.46-1

Si bien el algoritmo DES es actualmente considerado inseguro, es importante conocer su historia y funcionamiento para comprender su papel en la evolución de la Criptografía.

Equipo y material necesario

Máquina Virtual Debian, para montarla en su equipo puede seguir el «Montaje de máquina virtual», el cual es un manual de Instalación MV, en el que se encuentran las instrucciones tanto para VMware como para VirtualBox. Así mismo puede consultar el Anexo D que contiene ejemplos de cómo debe colocar sus respuestas en la práctica.

Desarrollo

[1] Federal Information Processing Standards Publication

Si se busca en Internet «Algoritmos de cifrado desarrollados en Python» se obtendrán un sinfín de resultados de los cuales, algunos serán más entendibles que otros, sin embargo, lo importante es entender lo que se realiza en el código para tener la oportunidad de llevar otros desarrollos a la práctica.

En esta actividad se realizará un script del algoritmo Data Encryption Standard usando Python 3. Se tomó como base el siguiente código[2], por su claridad y apego al desarrollo del algoritmo. Como podrá notar, el código no está comentado, eso adicional a la adecuación del programa según se solicite, será parte de las actividades a realizar.

ACTIVIDAD 3.1: Conteste lo siguiente:

Acrónimo del algoritmo «Data Encryption Standard»: [ ]

Busque un directorio cuyo nombre es la respuesta a la pregunta anterior. Describa qué fue lo que realizó para encontrar el directorio. Coloque el(los) comando(s) utilizados.

Pista: El directorio está oculto

Actividad 3.1: Búsqueda del directorio.

En dicho directorio tiene distintos archivos, los cuales conforman los pasos para la realización del cifrado haciendo uso del algoritmo Data Encryption Standard. El contenido de directorio se puede observar en la Figura N° 3.1.

Compruebe comparando su resultado con el de la Figura N° 3.1

2 SuQinghang. Github CourseCode. Recuperado de https://github.com/SuQinghang/CoursesCode/tree/master/Cryptology_Exp/code/DES
En caso de no poder ingresar al enlace, una copia del repositorio se encuentra en el directorio encontrado en la Actividad 3.1

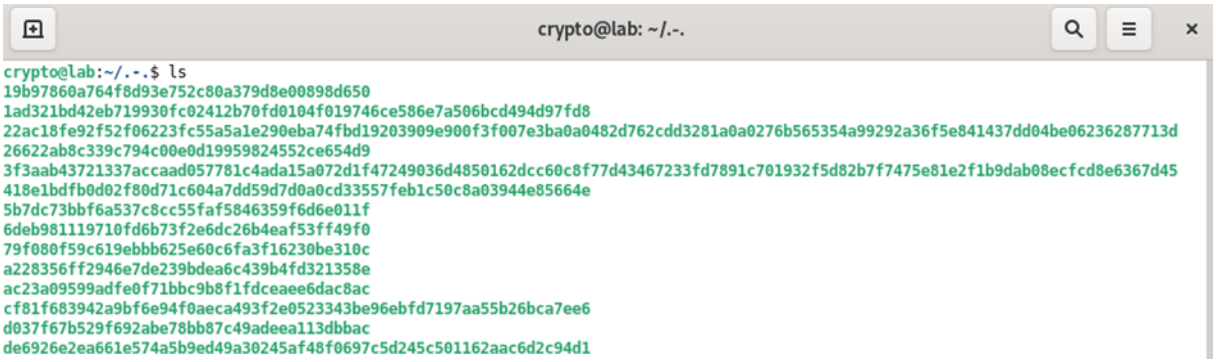


Figura N° 3. 1: Contenido del directorio.

ACTIVIDAD 3.2: Ahora, se le mostrará una serie de preguntas de las cuales la respuesta a dicha pregunta es el nombre de uno de los archivos ubicado en el directorio encontrado anteriormente.

Para cada respuesta que usted obtenga, obtendrá su hash, esto haciendo uso del programa que desarrolló en la práctica pasada. El tipo de hash a utilizar depende del indicado en cada una de las preguntas.

Ejemplo: Conteste la siguiente pregunta

1. ¿Qué tipo de cifrado simétrico es Data Encryption Standard? Obtenga **RMD160** de su respuesta. **por bloques**

   Por lo cual, como la respuesta es «*por bloques*» se obtiene el *RMD160*, como se muestra en la Figura N° 3.2

Figura N° 3. 2: Ejemplo obtención de hash.

   En la carpeta que encontró en la *Actividad 3.1*, se encuentra el script cuyo nombre es el hash mostrado en la Figura N° 3.2. Ejecute dicho script, en él encontrará la respuesta a la pregunta en claro y un título breve con lo que realiza el bloque de código de dicho archivo. Esto es mostrado en la Figura N° 3.3

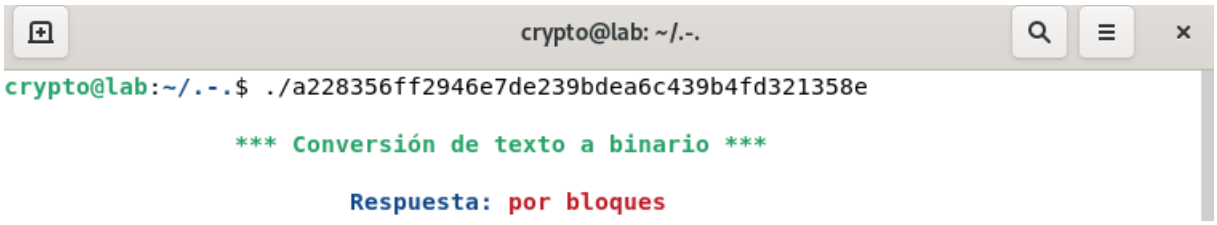


Figura N° 3. 3: Conversión de texto a binario.

Todas las respuestas son con letras *minúsculas*, excepto, cuando la respuesta sea un *número* se coloca el carácter *numérico*, **no** con *letra* y si se trata de algún instituto, o asociación se colocan las *siglas en mayúsculas*. Algunas preguntas tienen resaltada en negritas una palabra, la cual forma parte de la respuesta. Tome en cuenta si es singular o si es plural.

Tomando como ejemplo la pregunta N° 2, si la respuesta fuera 8, se obtiene el hash de «8 bits».

En todas las preguntas, coloque dentro del primer recuadro la captura de pantalla donde obtiene el hash de su respuesta como en la Figura N° 3.2. En el siguiente recuadro coloque la captura de pantalla del script ejecutado tal como en la Figura N° 3.3, y en el último recuadro explique con sus propias palabras lo que realiza el bloque de código, tomando en cuenta el uso de bibliotecas y funciones si es el caso. Para no generar errores, algunas partes del código están comentadas con **#** como se muestra en la Figura N° 3.4.

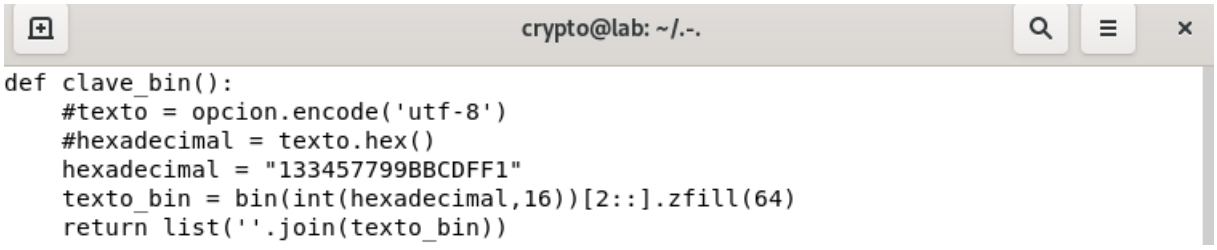
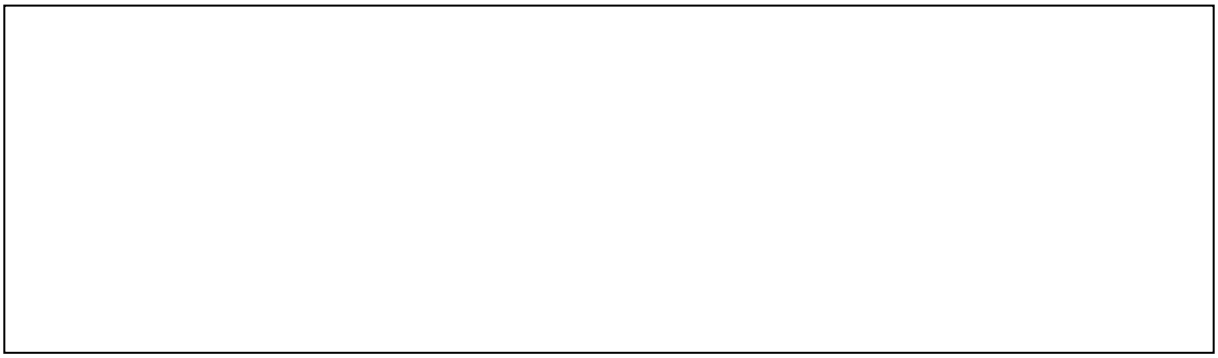


Figura N° 3. 4: Parte del código correspondiente a la conversión de texto a binario.

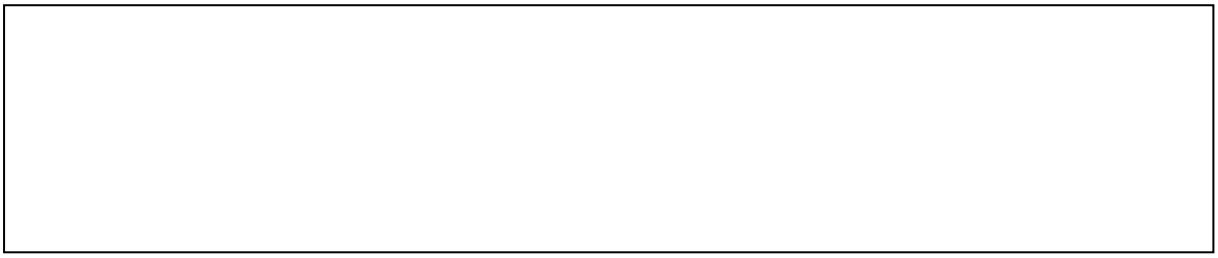
De la pregunta N° 2 a la pregunta N° 5 se trabaja con la clave.

NOTA: Uno de los scripts tiene un enlace que puede ser de ayuda para entender el funcionamiento del algoritmo y por lo tanto su programación. (Grabbe, J.O., s.f.)

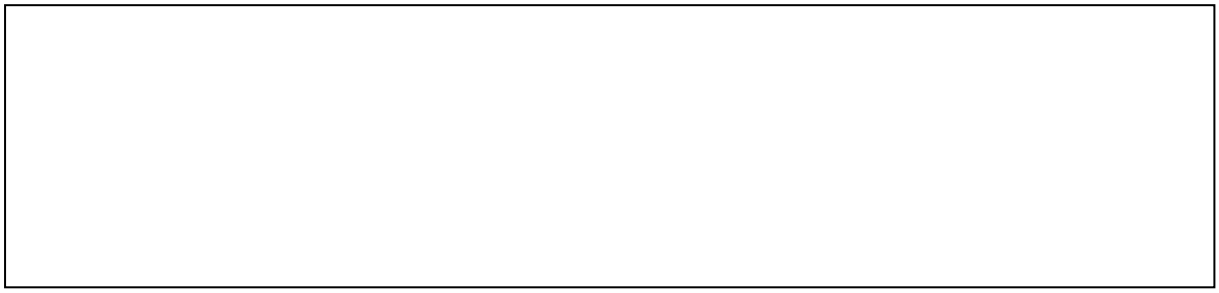
2. ¿De cuántos **bits** está conformada la clave a utilizar para cifrar el mensaje? Obtenga SHA- 1 de su respuesta.



Actividad 3.2: Hash de pregunta N° 2.

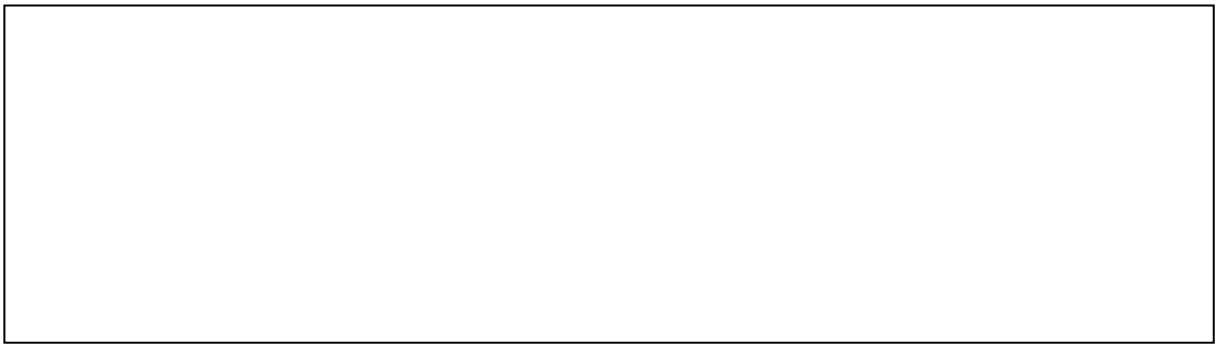


Actividad 3.2: Ejecución del script, pregunta N° 2.



Actividad 3.2: Explicación del script, pregunta N° 2.

3. ¿De cuántos **bits** está conformado el bloque de entrada correspondiente al mensaje en claro? Obtenga RIPEMD-160 de su respuesta.

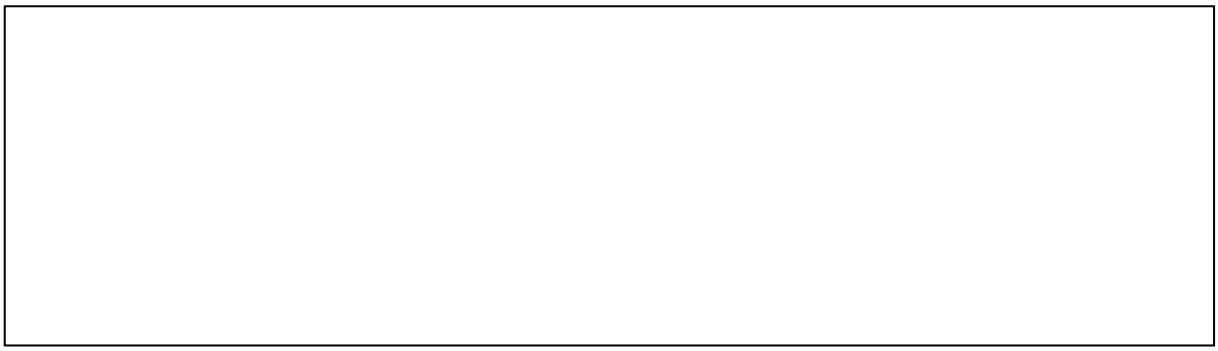


Actividad 3.2: Hash de pregunta N° 3.

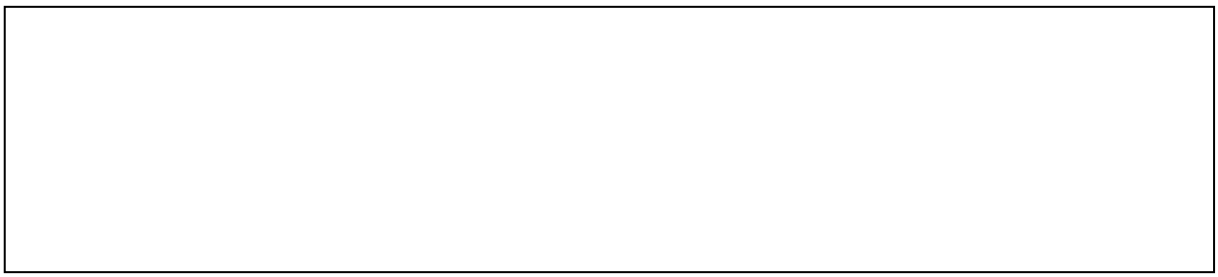
Actividad 3.2: Ejecución del script, pregunta N° 3.

Actividad 3.2: Explicación del script, pregunta N° 3.

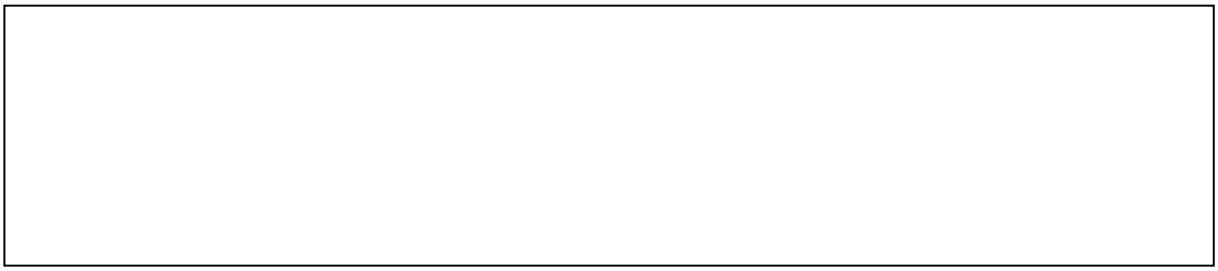
4. ¿Cuántos **bits** son realmente usados para realizar el cifrado? Obtenga SHA-256 de su respuesta.



Actividad 3.2: Hash de pregunta N° 4.

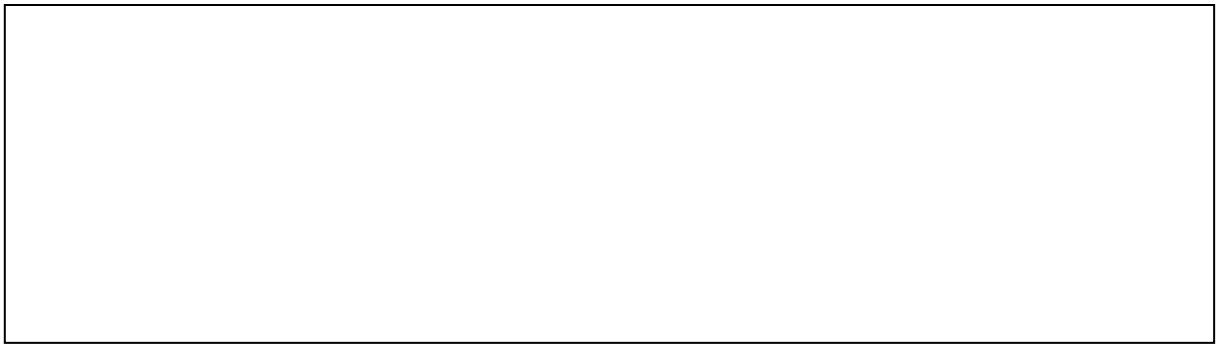


Actividad 3.2: Ejecución del script, pregunta N° 4.



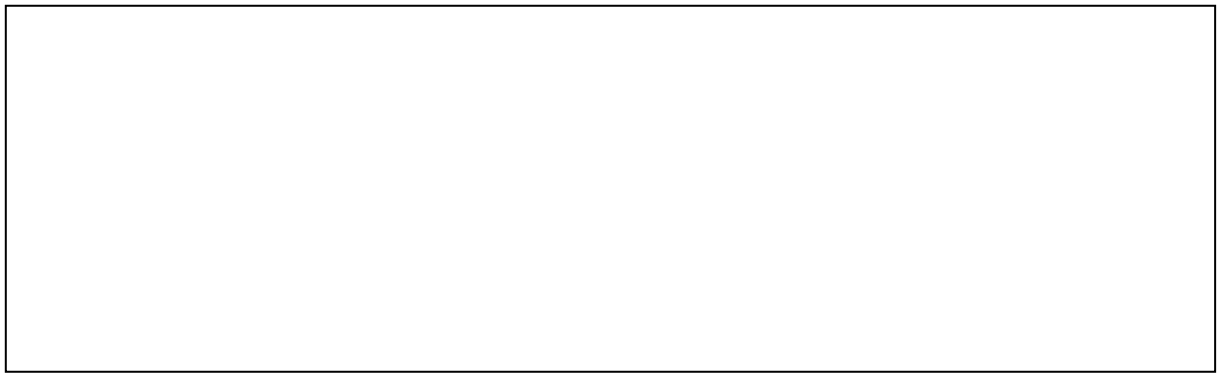
Actividad 3.2: Explicación del script, pregunta N° 4.

5. ¿Cuál fue el primer nombre del algoritmo Data Encryption Standard? Obtenga RMD-160 de su respuesta.



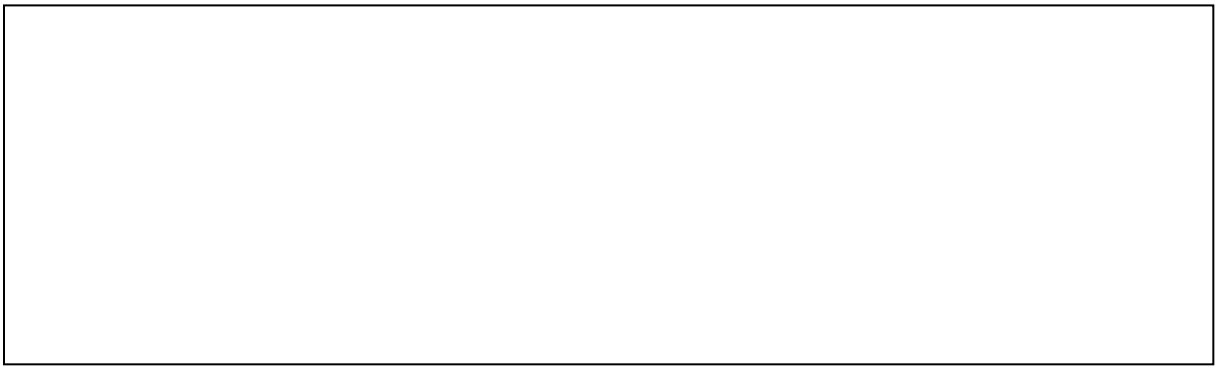
Actividad 3.2: Hash de pregunta N° 5.

Actividad 3.2: Ejecución del script, pregunta N° 5.

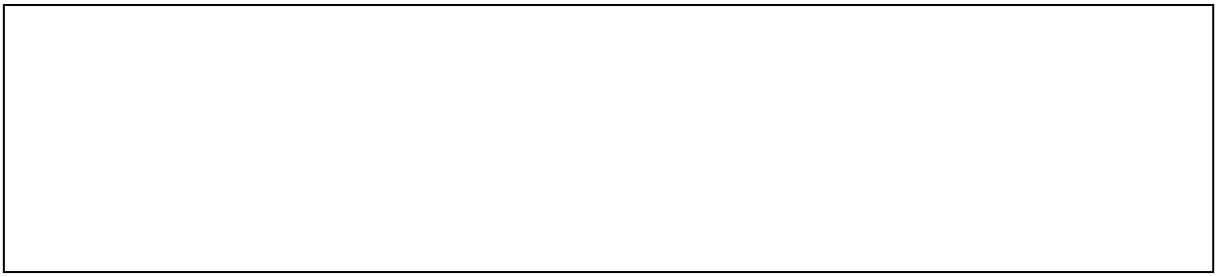


Actividad 3.2: Explicación del script, pregunta N° 5.

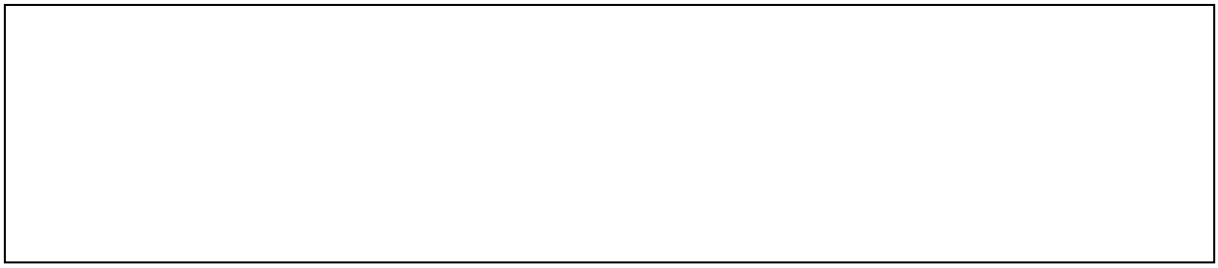
A partir de la pregunta N° 6 hasta la pregunta N° 14 corresponden en su mayoría al mensaje.

6. ¿De cuántas **subclaves** está compuesto el algoritmo Data Encryption Standard? Obtenga SHA-512 de su respuesta.
7. 

Actividad 3.2: Hash de pregunta N° 6.

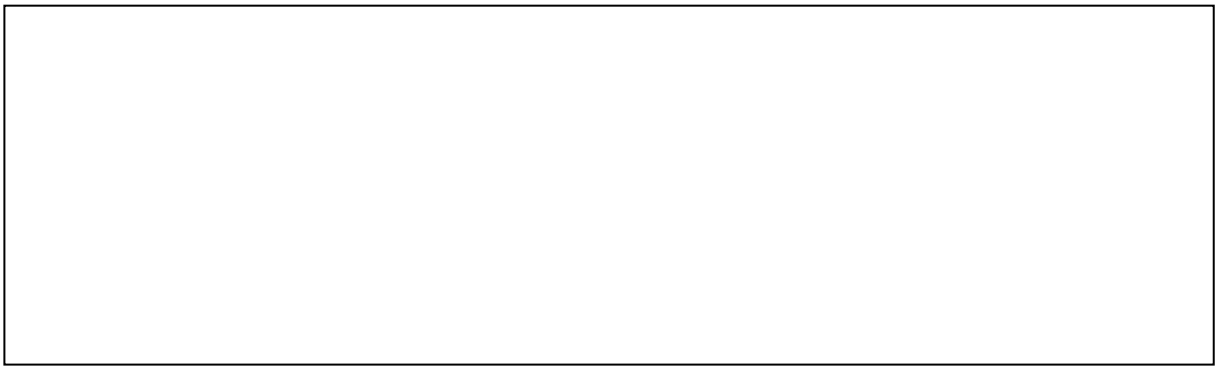


Actividad 3.2: Ejecución del script, pregunta N° 6.

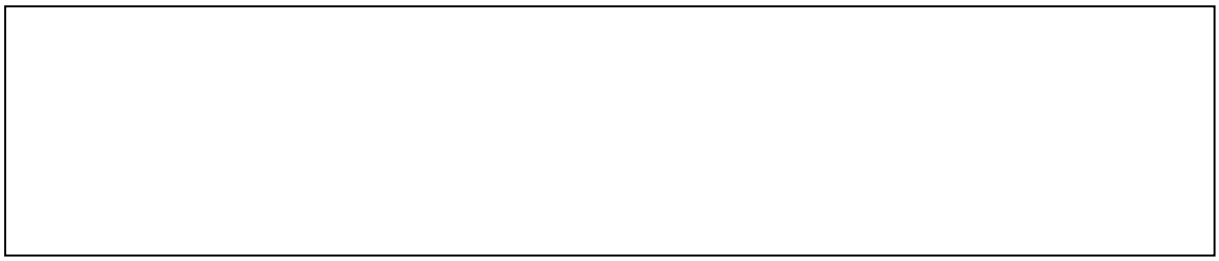


Actividad 3.2: Explicación del script, pregunta N° 6.

8. ¿Cuántos **bytes** son usados como paridad? Obtenga RIPEMD-160 de su respuesta.



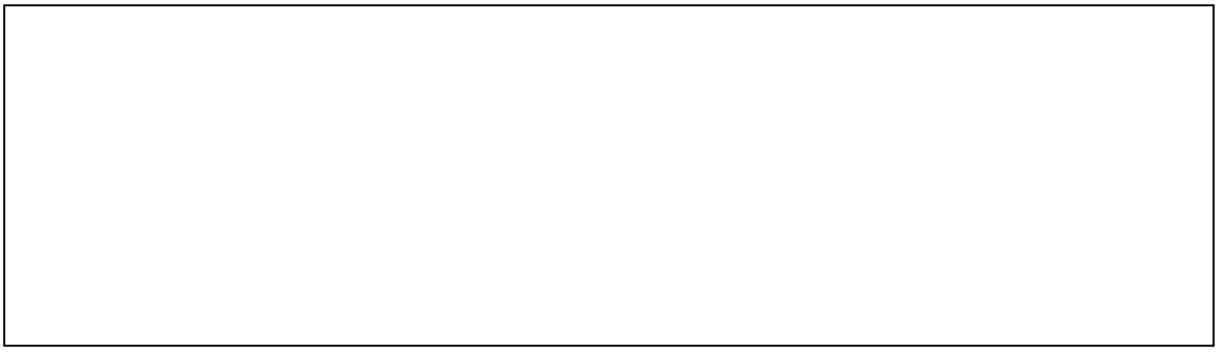
Actividad 3.2: Hash de pregunta N° 7.



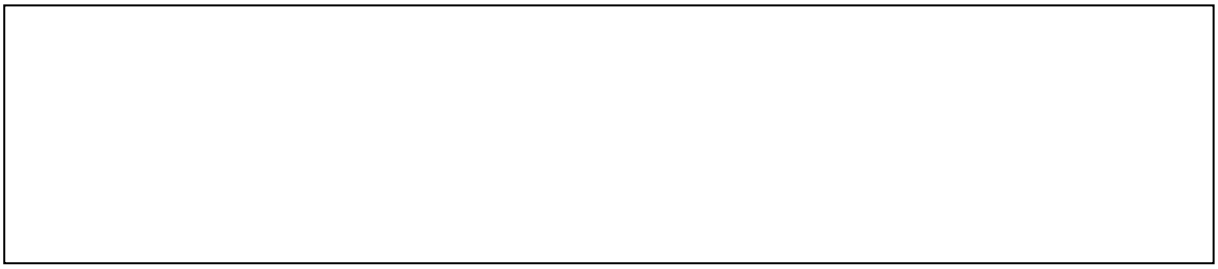
Actividad 3.2: Ejecución del script, pregunta N° 7.

Actividad 3.2: Explicación del script, pregunta N° 7.

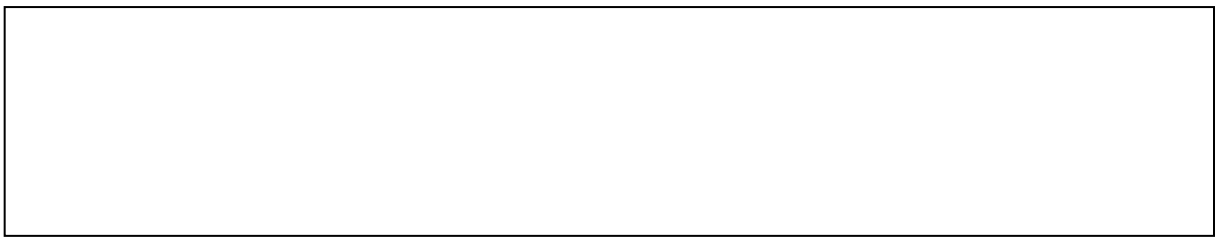
9. ¿Qué operación lógica es utilizada en el algoritmo Data Encryption Standard? Obtener SHA-256 de su respuesta.



Actividad 3.2: Hash de pregunta N° 8.

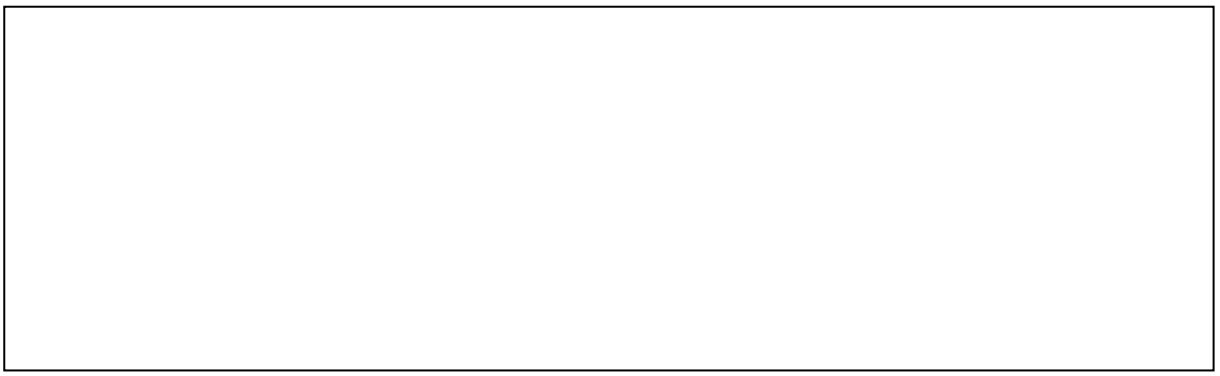


Actividad 3.2: Ejecución del script, pregunta N° 8.

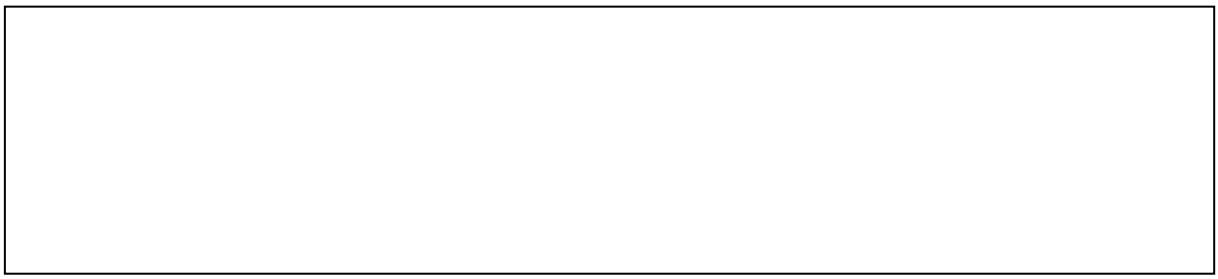


Actividad 3.2: Explicación del script, pregunta N° 8.

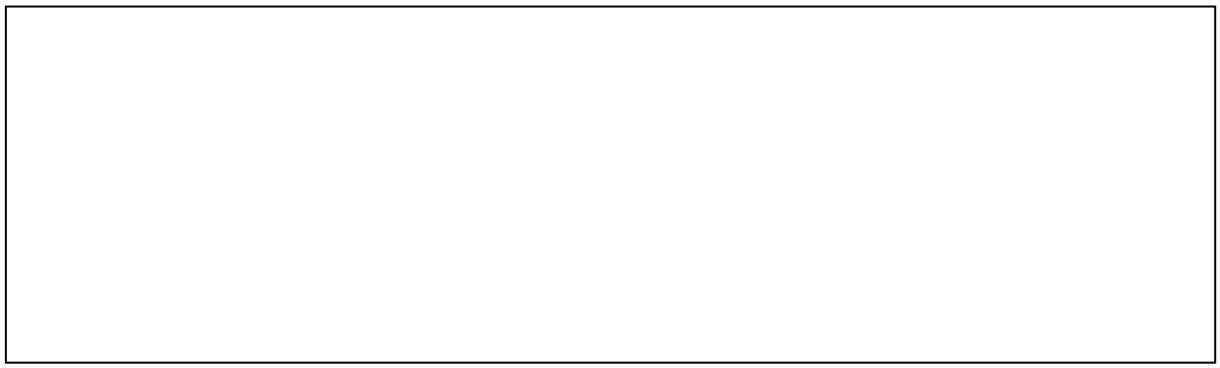
10. ¿En qué año fue desarrollado el algoritmo Data Encryption Standard? Obtenga SHA-512 de su respuesta.



Actividad 3.2: Hash de pregunta N° 9.

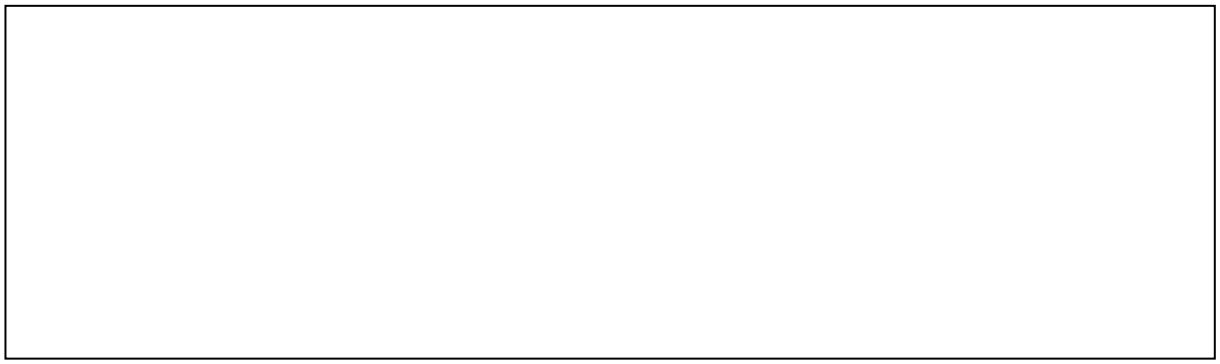


Actividad 3.2: Ejecución del script, pregunta N° 9.

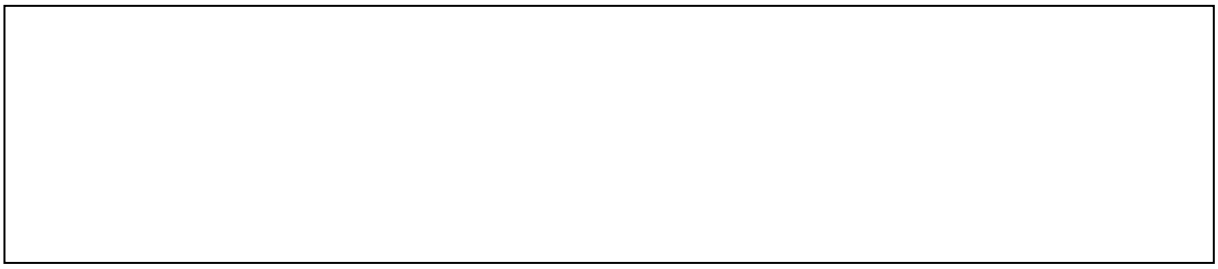


Actividad 3.2: Explicación del script, pregunta N° 9.

11. ¿Qué instituto eligió al algoritmo Data Encryption Standard? Obtenga SHA-256 de su respuesta.



Actividad 3.2: Hash de pregunta N° 10.

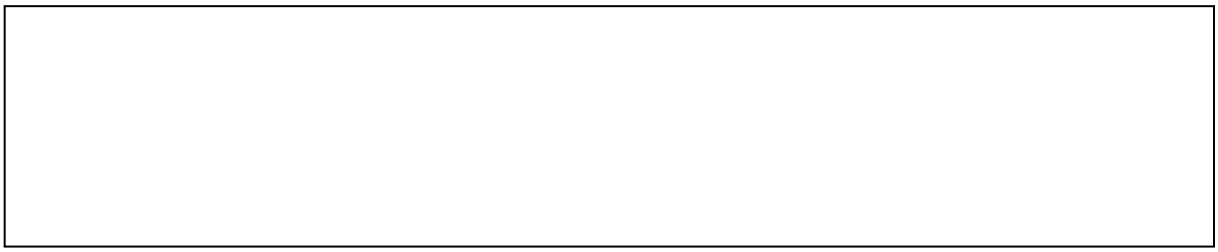


Actividad 3.2: Ejecución del script, pregunta N° 10.

Actividad 3.2: Explicación del script, pregunta N° 10.

12. La clave es dividida en dos partes, ¿Cuántos **bits** conforman cada una de las partes? Obtenga SHA-1 de su respuesta.

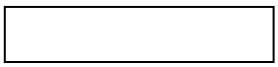
Actividad 3.2: Hash de pregunta N° 11.

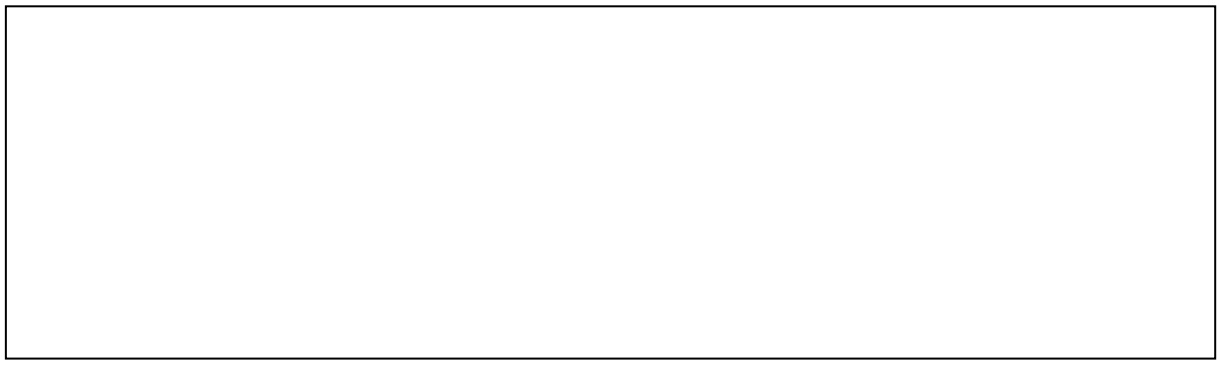


Actividad 3.2: Ejecución del script, pregunta N° 11.

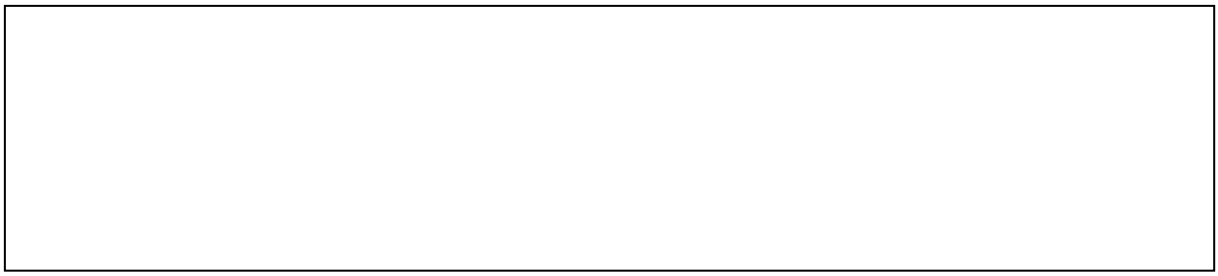
Actividad 3.2: Explicación del script, pregunta N° 11.

13. El mensaje también es dividido en dos partes, izquierda y derecha, ¿Cuántos **bytes** conforman cada una de las partes? Obtenga SHA-256 de su respuesta.

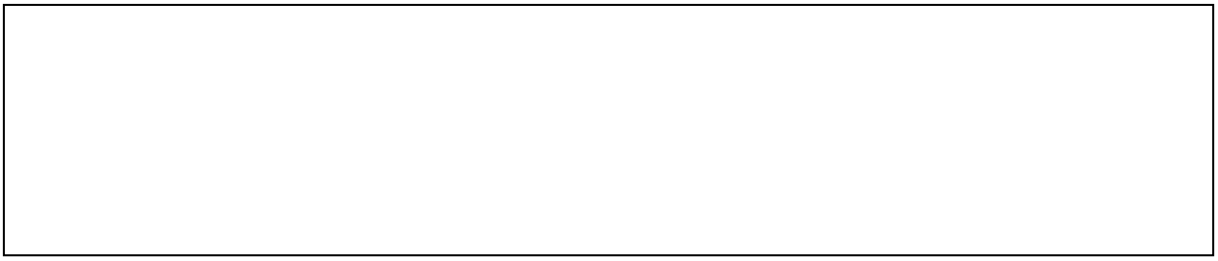




Actividad 3.2: Hash de pregunta N° 12.

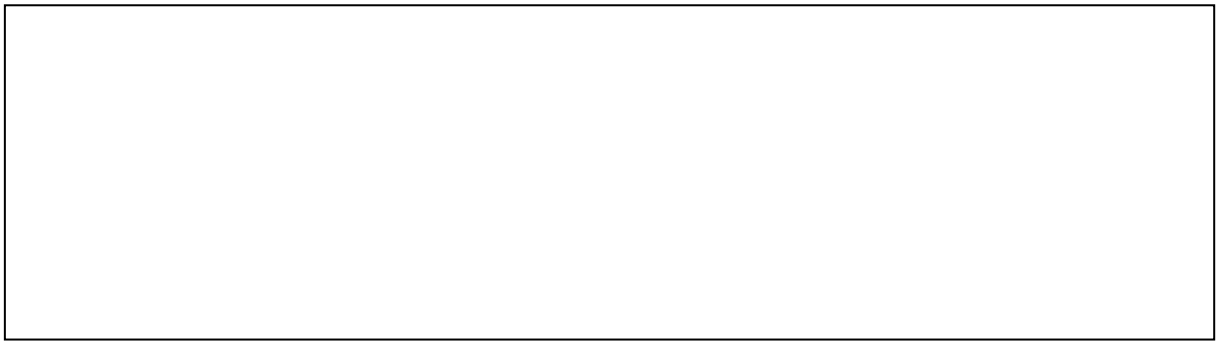


Actividad 3.2: Ejecución del script, pregunta N° 12.



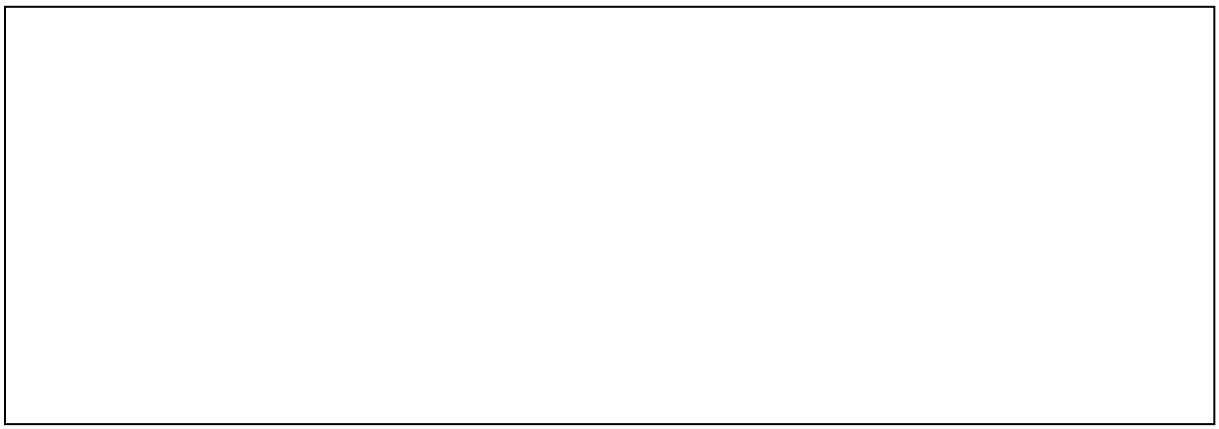
Actividad 3.2: Explicación del script, pregunta N° 12.

14. ¿Qué instituto propuso el algoritmo Data Encryption Standard? Obtenga RIPEMD-160 de su respuesta.



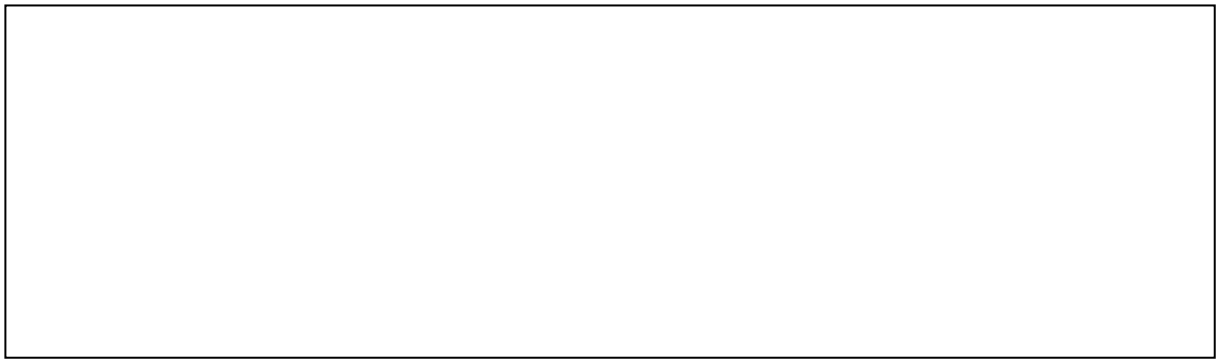
Actividad 3.2: Hash de pregunta N° 13.

Actividad 3.2: Ejecución del script, pregunta N° 13.

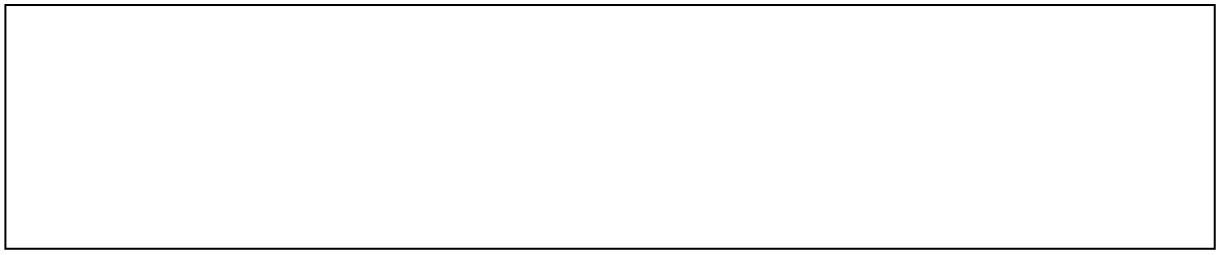


Actividad 3.2: Explicación del script, pregunta N° 13.

15. ¿Quién fue el principal desarrollador del algoritmo Data Encryption Standard? Obtenga SHA-1 de su respuesta.



Actividad 3.2: Hash de pregunta N° 14.



Actividad 3.2: Ejecución del script, pregunta N° 14.

Actividad 3.2: Explicación del script, pregunta N° 14.

Al unir todas las partes, obtiene un script que cifra y descifra usando el siguiente mensaje y clave, los dos, representados en hexadecimal:

Clave: ***133457799BBCDFF1***

Mensaje: ***0123456789ABCDEF***

El resultado de cifrar se observa en la Figura N° 3.5, el resultado se da en hexadecimal dado que al realizar la conversión a caracteres no todos son entendibles o como tal no son caracteres dentro del código ASCII.

Figura N° 3. 5: Resultado de cifrar con DES.

Al descifrar el mensaje, se debe obtener el mensaje original, es decir *0123456789ABCDEF* como se observa en la Figura N° 3.6

Figura N° 3. 6: Resultado de descifrar con DES.

ACTIVIDAD 3.3: Una todos los bloques de código en un solo script, tome en cuenta, que en los fragmentos de código que se presentaron en la actividad anterior *(Actividad 3.2)*, se hace uso de la librería *colorama* para darle color a los textos mostrados en pantalla, si gusta puede seguir haciendo uso de estos o bien, eliminar las líneas donde se utiliza.

Realice las modificaciones necesarias para que el script funcione para cifrar y descifrar, ya sea *mensajes o archivos de texto plano*. La clave siempre se ingresa en texto, no en archivo, los archivos solo serán manejados para lo que se desee cifrar o descifrar.

El mensaje, la clave, o el contenido del archivo pueden ser de la longitud que sea, por lo cual, usted debe adecuar su script de forma que continúe con la forma establecida del algoritmo DES, es decir, tanto mensaje como clave de 64 bits.

Una forma puede ser, para el mensaje y contenido del archivo, ir fragmentando el texto de manera que quede en listas de 64 bits, y en caso de que alguna sea menor a 64 bits, realizar el relleno con ceros.

Para la clave, puede hacer uso de alguna función hash, y adecuar de manera que solo se haga uso de los números de bits que son requeridos.

En este caso, los ejemplos se le muestran haciendo uso de banderas, con el módulo *argparse*, pero si usted desea hacer uso de algunos otros elementos de Python como *tkinter*, que ayuda a abrir una ventana para buscar directamente los archivos, o pedir el ingreso del nombre al correr el programa.

Cuando se haga uso de archivos, el resultado del cifrado o descifrado, debe guardarse en un archivo también. Para los resultados de cifrado, debe agregar una codificación, es decir, después del cifrado, al resultado lo pasa por algún codificador, en los ejemplos mostrados, se hace uso de base64.

Todo su código debe hacer uso correcto del manejo de errores, y debe estar comentado en su totalidad.

A continuación, se muestran los ejemplos de cifrado y descifrado, tanto para mensajes como para archivos de texto.

1. **Cifrado de mensaje:**

   En este caso, al indicarse las banderas se ingresan el texto que se utiliza como mensaje: 'Data Encryption Standard y la clave: 'cipher' el programa, regresa el resultado del cifrado. (Véase Figura N° 3.7)

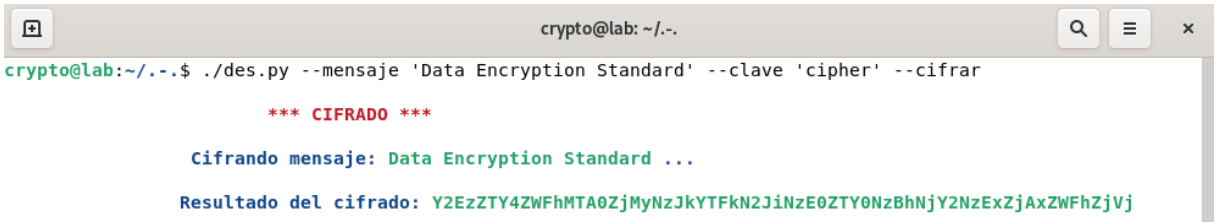


Figura N° 3. 7: Cifrado con mensaje y clave en texto.

2. **Descifrado de mensaje:**

   Ahora, como argumento de la bandera –mensaje, se coloca el resultado del cifrado, es decir *'Y2EzZTY4ZWFhMTA0ZjMyNzJkYTFkN2JiNzE0ZTY0NzBhNjY2NzExZjAxZWFhZjVj'*, la clave es la misma, es decir 'cipher', y la bandera ahora será descifrar. (Véase Figura N° 3.8)

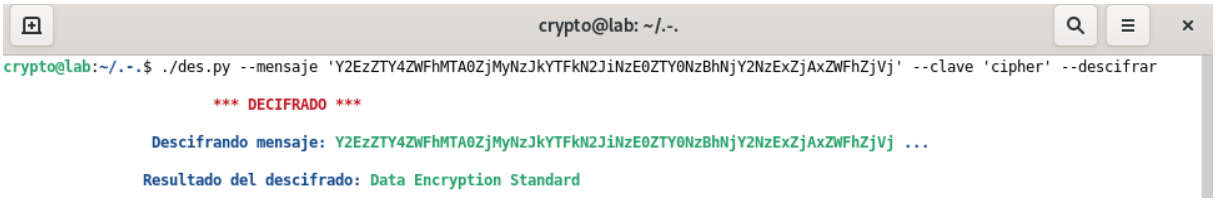


Figura N° 3. 8: Cifrado con mensaje y clave y en archivo.

3. Cifrado de archivo:

Para este caso, se cifra un archivo llamado *DES.txt*, para los archivos, en este caso como es el cifrado, se guarda con el nombre original, seguido de la palabra «cipher» y la fecha y hora en la que se está realizando el cifrado. La clave para este caso será '*crypto*'. (Véase Figura N° 3.9)

Figura N° 3. 9: Cifrado con mensaje en archivo y clave en texto.

4. Descifrado de archivo:

Ahora, se descifra el archivo. En este caso, se coloca en la bandera --archivo, el nombre del archivo cifrado, es decir DES.txt_cipher-SunMar121331462023.txt, la clave es la misma, y la bandera cambia por la de descifrado. En este caso, como es descifrado, el nombre coloca «descipher», junto con la fecha en la que se crea el archivo. (Véase Figura N° 3.10)

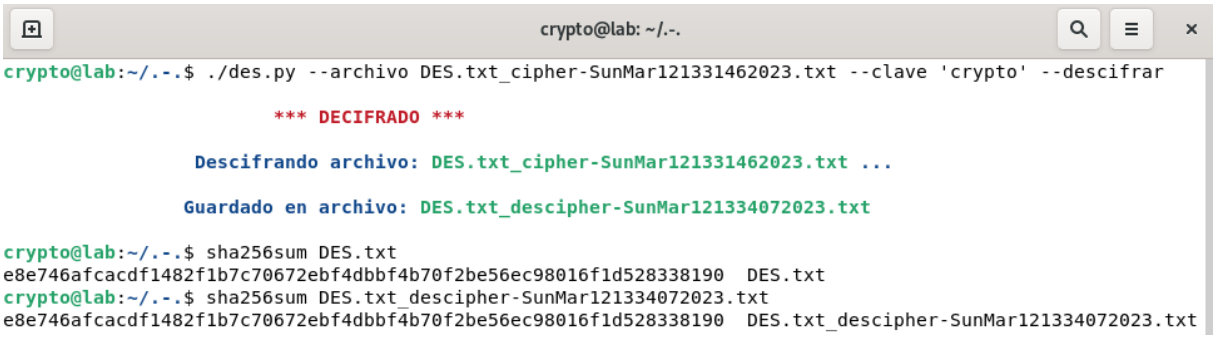


Figura N° 3. 10: Cifrado con mensaje en texto y clave en archivo.

En la Figura N° 3.10, se muestra el resultado, y una comprobación de que el descifrado es igual al archivo original, esto se comprueba con el hash de los archivos.

En caso de que no se coloque la clave de manera correcta, es decir, que no corresponda con la misma a la hora de descifrar, los datos no podrán ser interpretados, y debe indicarlo con un mensaje, como en la Figura N° 3.11:

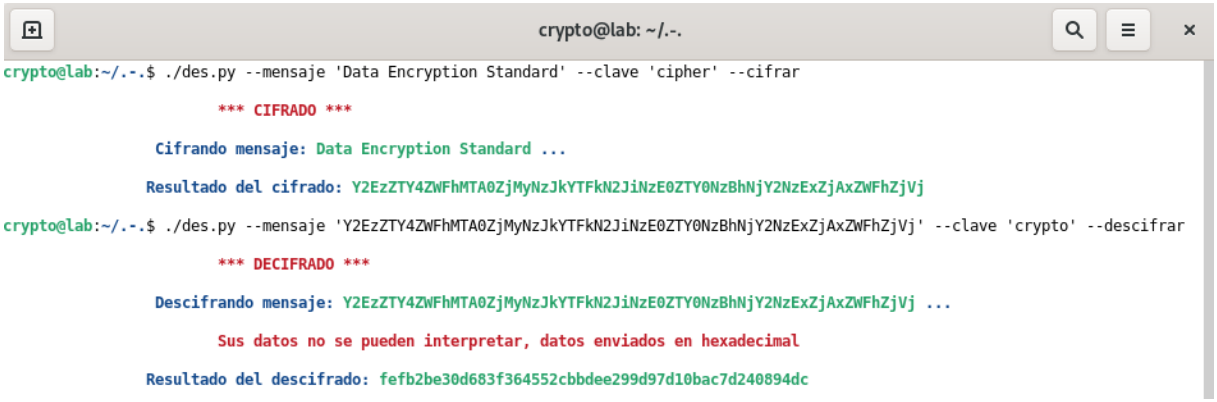


Figura N° 3. 11: No se realiza el descifrado de manera correcta por clave distinta.

En los siguientes recuadros, coloque capturas correspondientes a la ejecución de su script, usando el mensaje y archivo de su preferencia. Tanto el archivo usado como su script deben ser enviados a su profesor en el formato como se le indique.

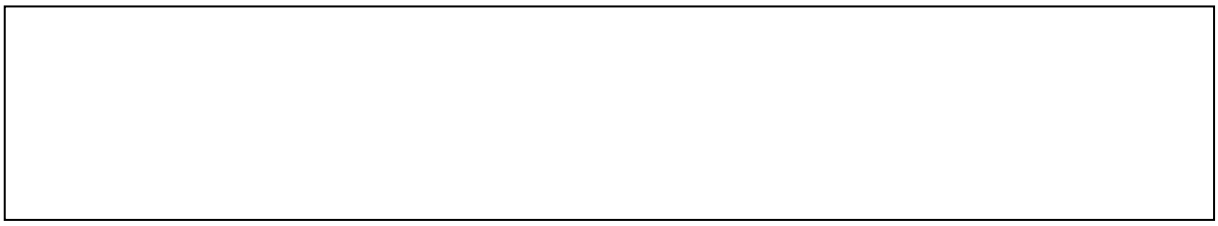
1. **Cifrado de mensaje:**

Actividad 3.3: Uso de script para cifrado/descifrado.

2. **Descifrado de mensaje:**

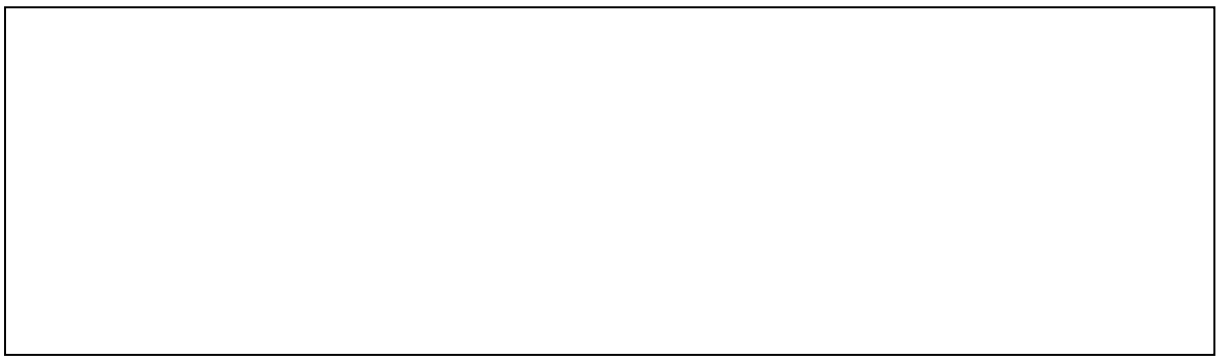
Actividad 3.3: Uso de script para cifrado/descifrado.

3. **Cifrado de archivo:**



Actividad 3.3: Uso de script para cifrado/descifrado.

4. **Descifrado de mensaje:**



Actividad 3.3: Uso de script para cifrado/descifrado.

ACTIVIDAD 3.4: Como lo realizó en la Actividad 2.3 y 2.4 de la «*Práctica N° 2: Funciones Hash*», también se puede hacer uso de *OpenSSL*, para el cifrado de archivos con DES. Cabe destacar que DES tiene una versión llamada 3DES, la cual consiste en realizar 3 veces el algoritmo, usando como entrada de uno la salida del anterior, esta versión fue la sustitución para el algoritmo DES. (ShellHacks, 2016)

De igual forma, se pueden hacer uso de distintos modos de operación, en donde destaca CBC y ECB. En esta ocasión se realizará el cifrado con el algoritmo 3DES.

Para eso, debe ejecutar el comando: *openssl enc -des3 -in «nombre_de_archivo_a_cifrar» -out «nombre_archivo_cifrado» -pbkdf2* tal como se muestra en la Figura N° 3.12 en donde se cifra el archivo *Bitcoin.txt*, se guarda en el nuevo archivo llamado *Bitcoin_3DES.cipher*.

Cuando realice la ejecución del comando, se le solicitará que ingrese una contraseña, misma contraseña que será utilizada cuando descifre el archivo.

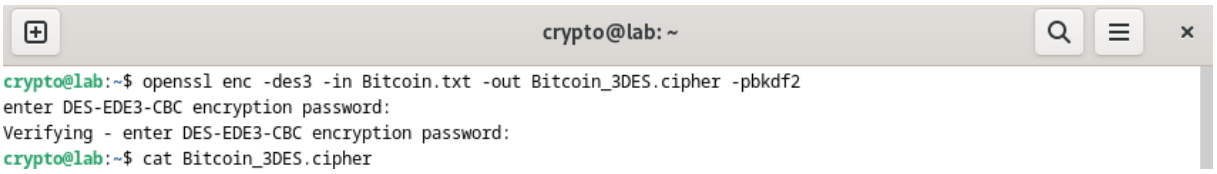


Figura N° 3. 12: Uso de OpenSSL para cifrado de archivo con 3DES.

Para descifrar el archivo, se hace uso del comando: *openssl enc -des3 -d -in «nombre_de_archivo_a_descifrar» -pbkdf2* al final se puede agregar la bandera *-out nombre_archivo* para guardar el contenido del descifrado en un archivo o bien, omitirlo y el resultado se mostrará en pantalla, tal como se muestra en la Figura N° 3.13

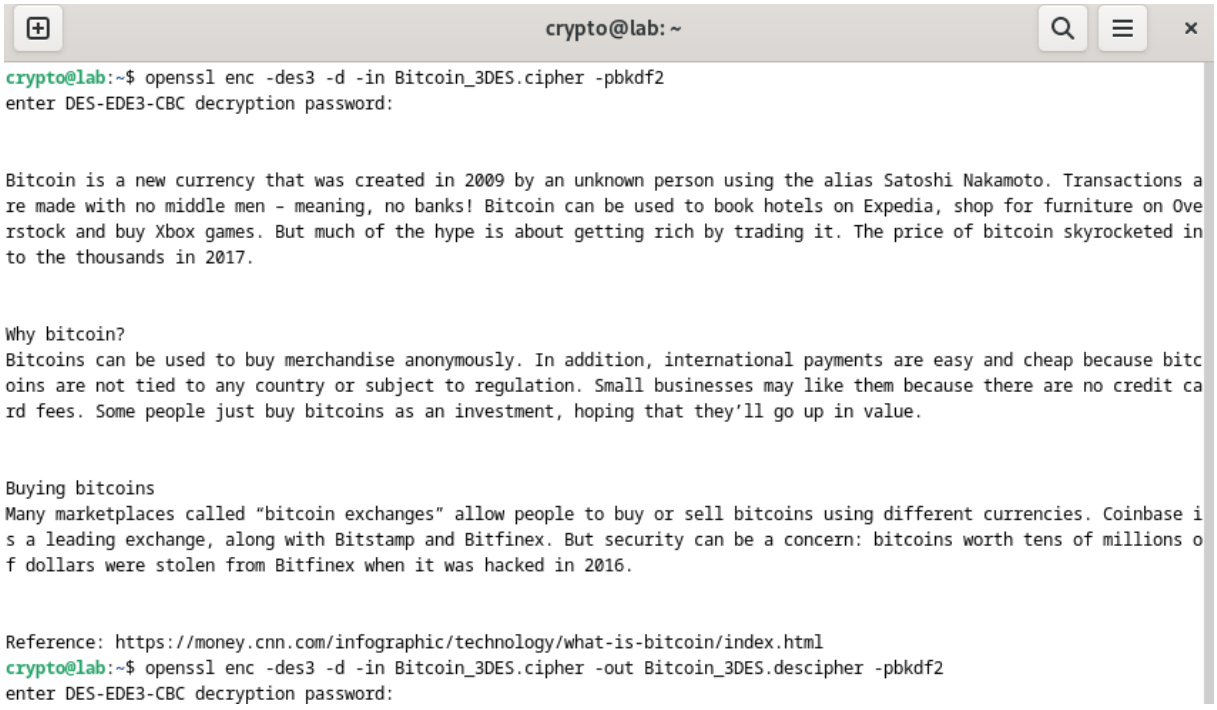
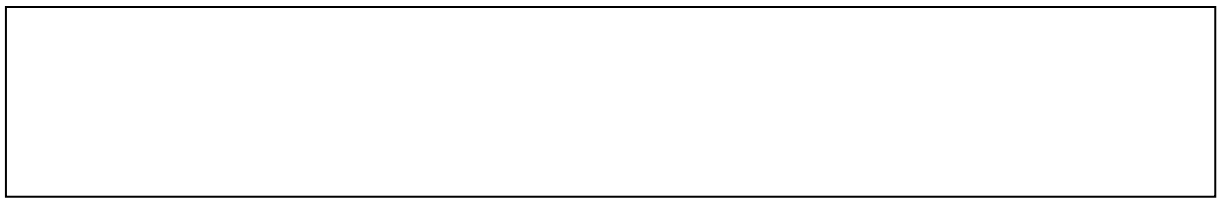


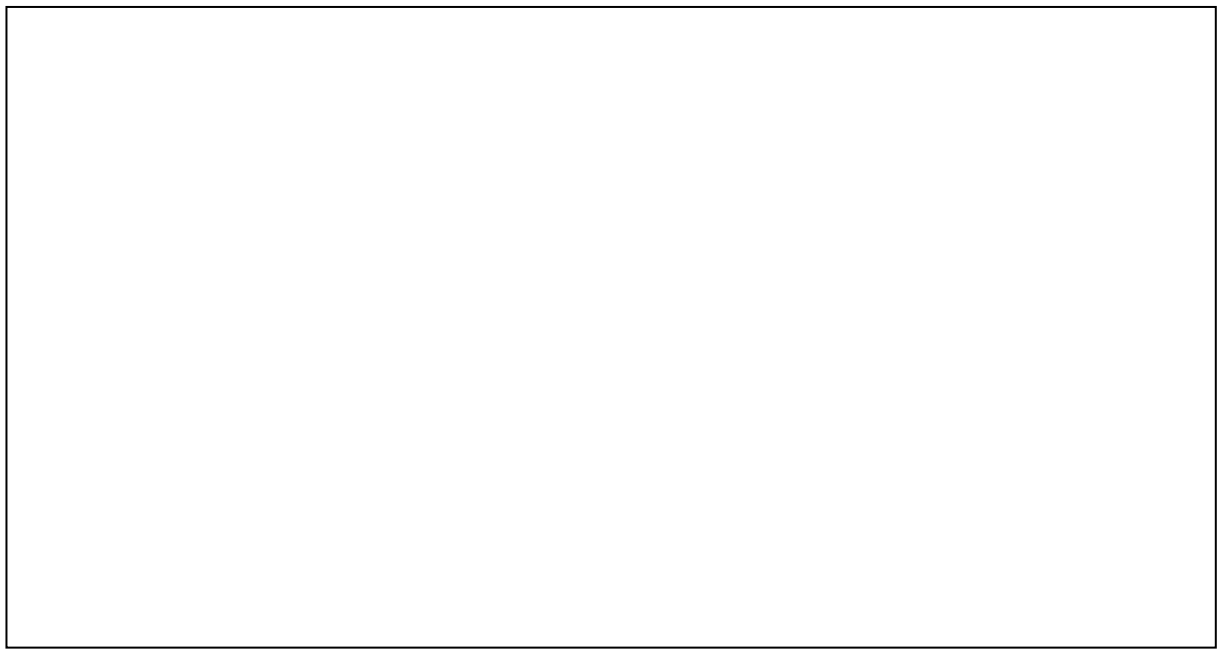
Figura N° 3. 13: Uso de OpenSSL para descifrado de archivo con 3DES.

Si desea conocer los demás métodos de cifrado que realiza openSSL ejecute *openssl enc -ciphers.*

En los siguientes recuadros, coloque la captura de pantalla en donde corresponda, del cifrado y descifrado de un archivo tal como se mostró en las Figuras N° 3.12 y 3.13.



Actividad 3.4: Uso de OpenSSL para cifrado de archivo.



Actividad 3.4: Uso de OpenSSL para cifrado de archivo.

Conclusiones:

1. ¿Con qué fines era usado el algoritmo Data Encryption Standard?

2. Investigue, ¿Cómo se aplica el método de cifrado 3DES?

3. ¿Cómo es que el algoritmo Data Encryption Standard fue vulnerado?

4. ¿Qué considera que fue lo más difícil de la realización de los scripts?

5. Comentarios o conclusiones adicionales

UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

ACTIVIDADES PRÁCTICAS: CRIPTOGRAFÍA

4. ADVANCED ENCRYPTION STANDARD

DOCENTE

ADVANCED ENCRYPTION STANDARD

Objetivo

Conocer el modo de cifrado y descifrado del algoritmo Advanced Encryption Standard y sus variantes.

Justificación

El algoritmo Advanced Encryption Standard (AES) es el algoritmo que sustituyo a Data Encryption Standard (DES). Así como las diferencias, mejoras y beneficios que representa frente a dicho algoritmo. En esta práctica conocerá las mejoras en la criptografía simétrica en cuanto al algoritmo Data Encryption Standard.

Introducción

Anunciado por el NIST, como el sucesor del algoritmo Data Encryption Standard, en el año 1997.

Es un algoritmo de cifrado simétrico por bloques, que fue elegido por el gobierno de los Estados Unidos para proteger la información clasificada aplicada tanto en software como en hardware.

Al igual que con Data Encryption Standard, Advanced Encryption Standard fue elegido entre al menos quince competidores, de los cuales: tenían que ser suficientemente seguros para soportar ataques, tener gran eficiencia computacional y de memoria, que contara con una relativa simplicidad de implementación y capaz de realizar el cifrado de bloques de 128 bits, utilizando claves de 128, 192 y 256 bits.

Finalmente, en 2001, el Advanced Encryption Standard (AES) entró en vigor como estándar del gobierno federal, siendo publicado por el NIST en el FIPS PUB 197. Este algoritmo sigue vigente y es ampliamente utilizado, además de que, hasta la fecha, no presenta vulnerabilidades significativas en su cifrado (López Barrientos, M. J., 2016)

Puede consultar su documentación en https://doi.org/10.6028/NIST.FIPS.197

Equipo y material necesario

Máquina Virtual Debian, para montarla en su equipo puede seguir el «Montaje de máquina virtual», el cual es un manual de Instalación MV, en el que se encuentran las instrucciones tanto para VMware como para VirtualBox. Así mismo puede consultar el Anexo D que contiene ejemplos de cómo debe colocar sus respuestas en la práctica.

Desarrollo

ACTIVIDAD 4.1: En la *Práctica N° 1 «Introducción a Linux», se creó un usuario con su nombre, a partir de esta práctica hará uso de dicho usuario. De cualquier modo, puede revisar la práctica para la creación de este.* Agregue dicho usuario al archivo *sudoers*, para que su usuario tenga los privilegios de super usuario.

***Recuerde**, a partir de esta actividad, y durante las prácticas siguientes, hará uso de este usuario.*

A partir de las actividades siguientes, las capturas se mostrarán con el usuario: **Atziri** para alumnos y con el usuario **Criptografía** para el docente.

ACTIVIDAD 4.2: Cree un directorio llamado **AES** en el home de su usuario. Ahora, haga la búsqueda de un archivo llamado «*Rijndael*» y copie ese archivo a la carpeta recién creada. Coloque una captura de pantalla donde se observe la creación del directorio y la realización de la copia de dicho archivo.

Actividad 4.2: Creación de directorio y copiar archivo.

¿Qué comando utilizó para realizar la búsqueda del archivo?

Actividad 4.2: Búsqueda de archivo.

Lea el archivo, en él encontrará los datos de un libro el cual podrá consultar para contestar las siguientes preguntas:

1. ¿Cuántos bytes son utilizados para mensaje?
2. ¿Cuántos bits son utilizados como clave?
3. ¿En qué modo se hace uso del vector de inicialización cuya documentación se encuentra en el RFC 3602?

4. ¿Cuántos bits son utilizados para vector de inicialización?

5. ¿Cuál es el número de vueltas que realiza al utilizar una clave de 128 bits?

6. ¿Cuál es el número de vueltas que realiza al utilizar una clave de 192 bits?

7. ¿Cuál es el número de vueltas que realiza al utilizar una clave de 256 bits?

8. ¿Cuáles son los principales procesos para el cifrado Advanced Encryption Standard?

9. ¿Quiénes son los desarrolladores de dicho algoritmo?

Coloque una captura de pantalla donde se visualice el contenido del archivo.

Actividad 4.2: Contenido del archivo encontrado.

ACTIVIDAD 4.3: En el mismo directorio donde localizó el archivo de la Actividad 4.2, se encuentra un archivo con el nombre **«des_cipher»**, este archivo contiene un mensaje, cifrado con el algoritmo 3DES, tal como se realizó en la *Actividad 3.4* de la *Práctica N° 3 «Data Encryption Standard»*. Haga uso de OpenSSL para descifrar dicho mensaje, use como clave: «*Rijndael*».

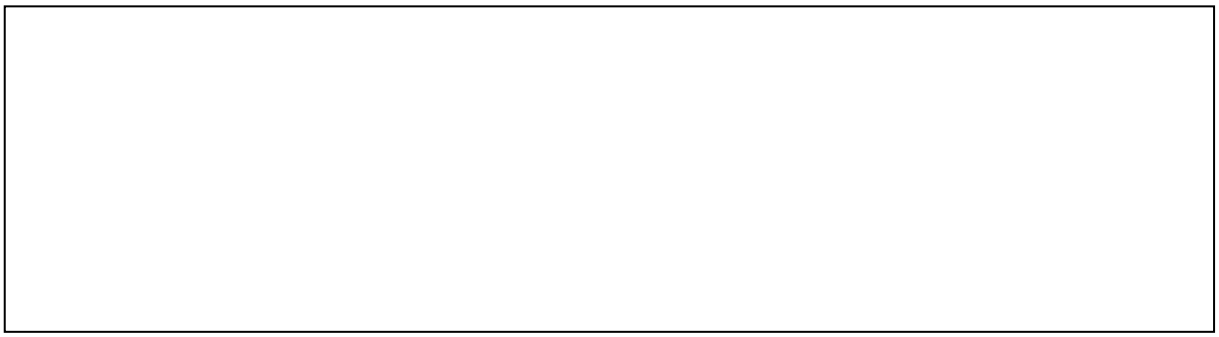
¿Cuál es el mensaje obtenido? Coloque una captura de pantalla en donde se muestre el resultado.

Actividad 4.3: Uso de script DES.

Busque un archivo que en su contenido se encuentre el mensaje obtenido anteriormente. ¿Coloque el o los comandos utilizados para la búsqueda?

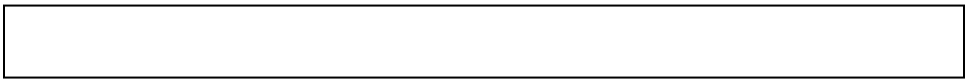
Pista:

- Busque información acerca de grep
- Busque en el directorio home del usuario root
- Haga uso de 2>
- Realice la búsqueda como superusuario.

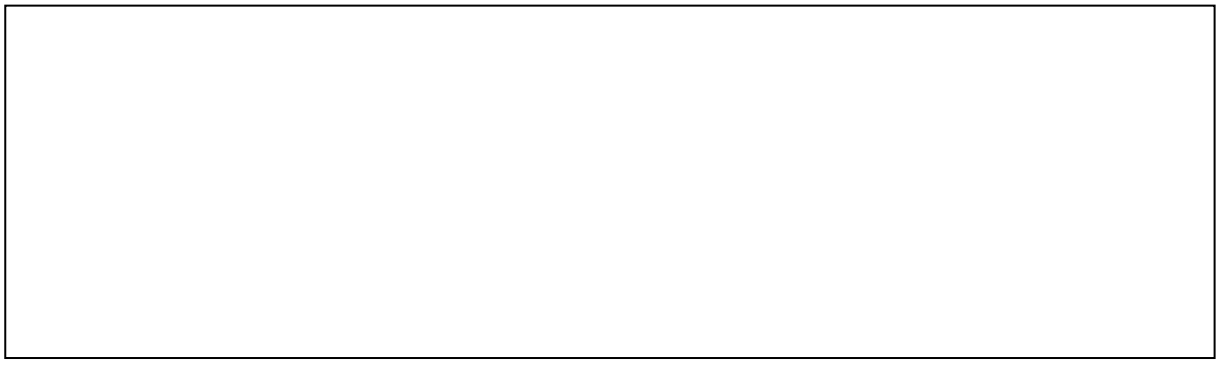


Actividad 4.3: Búsqueda del archivo.

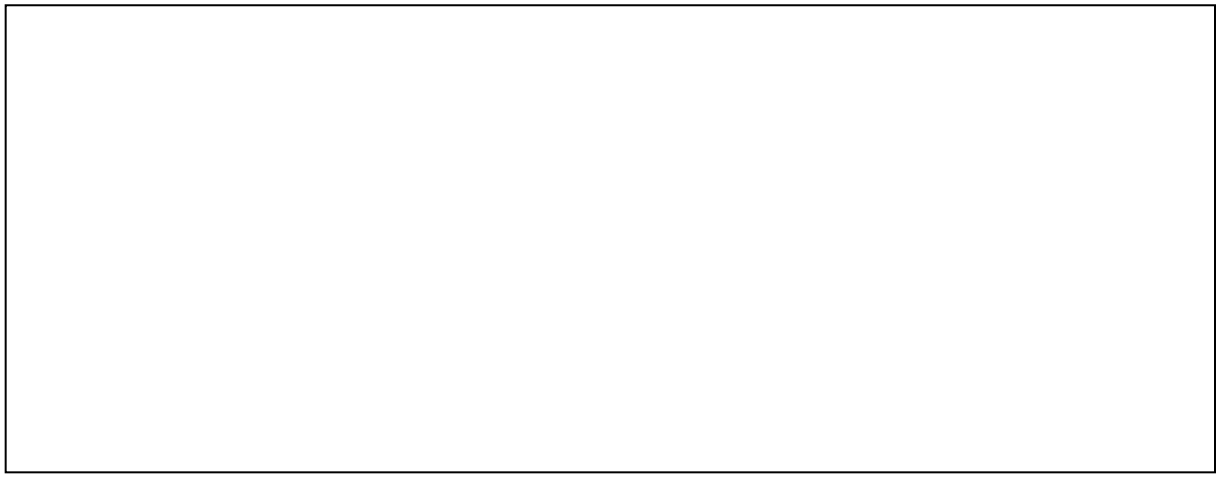
Copie el archivo a su carpeta AES. ¿Cuál es el nombre del archivo?



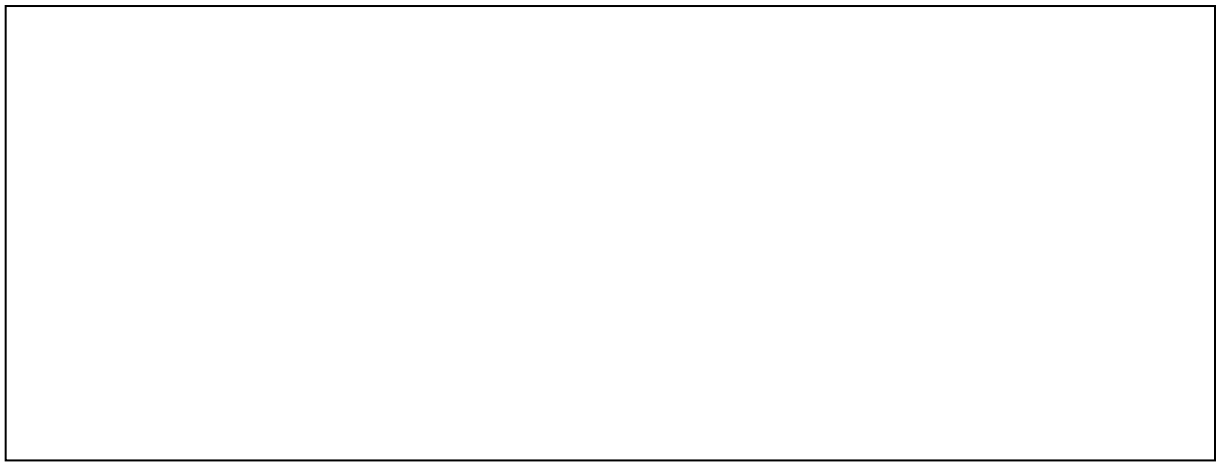
Ahora, visualice el documento indicado en el archivo que acaba de localizar, dicho documento tiene paso a paso el desarrollo del algoritmo AES para valores específicos. Siga las instrucciones que dicho archivo le da, y llene los recuadros y tablas siguientes, el archivo de instrucciones le dice qué debe colocar en cada una de ellas.



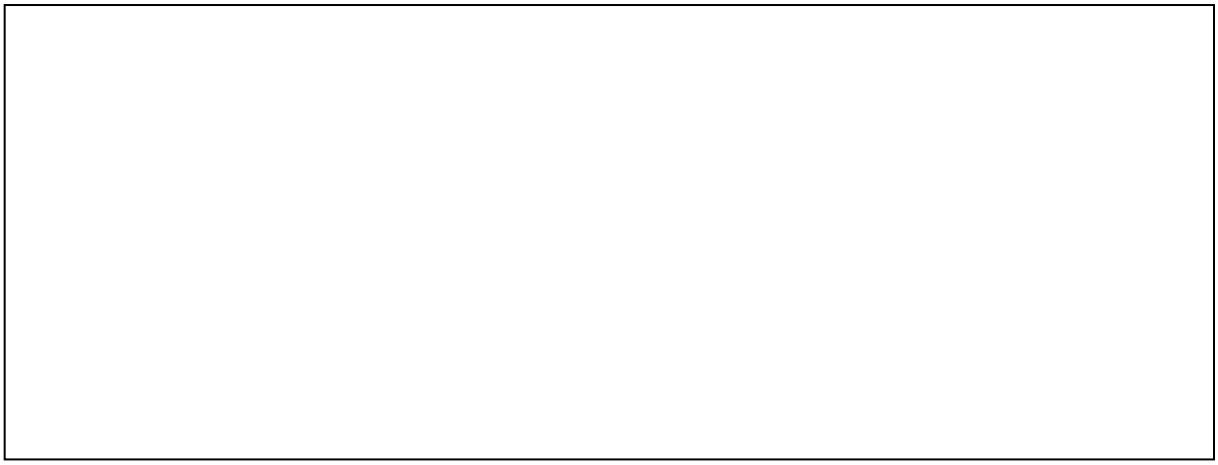
Actividad 4.3: Datos.



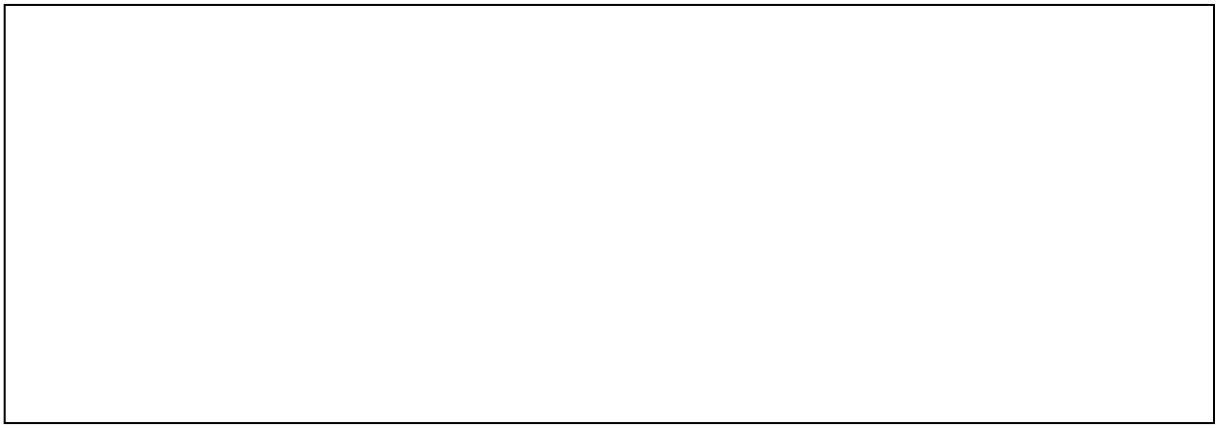
Actividad 4.3: SRound.



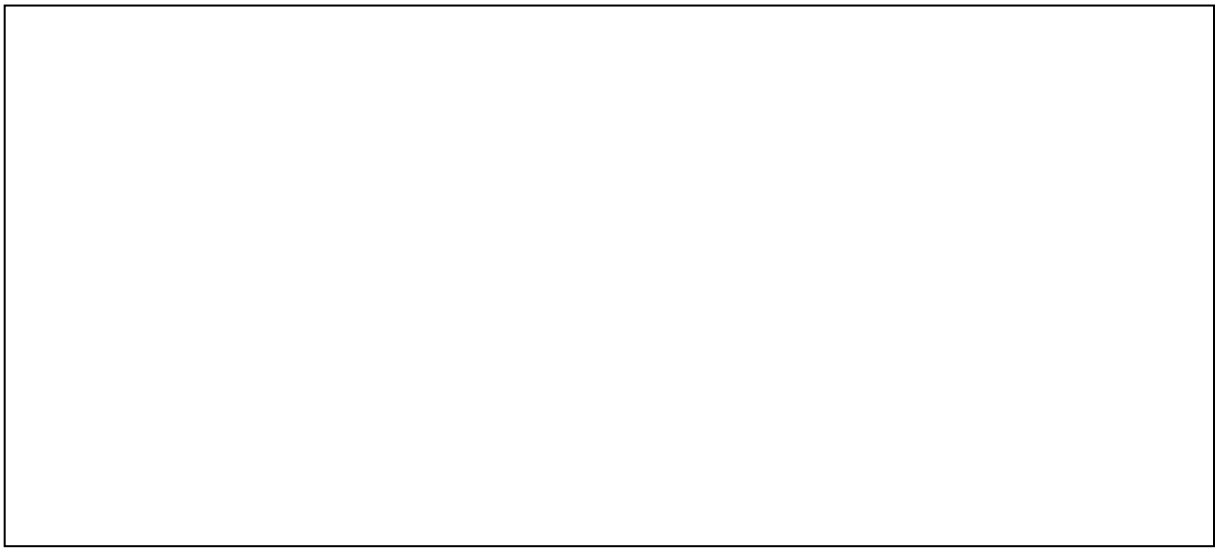
Actividad 4.3: SubBytes.



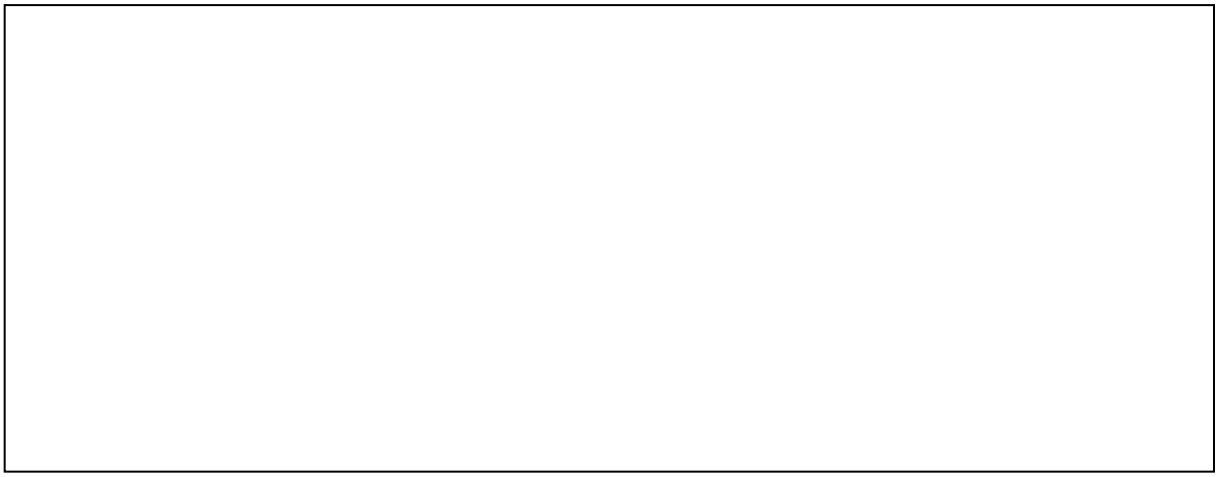
Actividad 4.3: Shift Rows.



Actividad 4.3: Mix Columns.



Actividad 4.3: SubClaves.



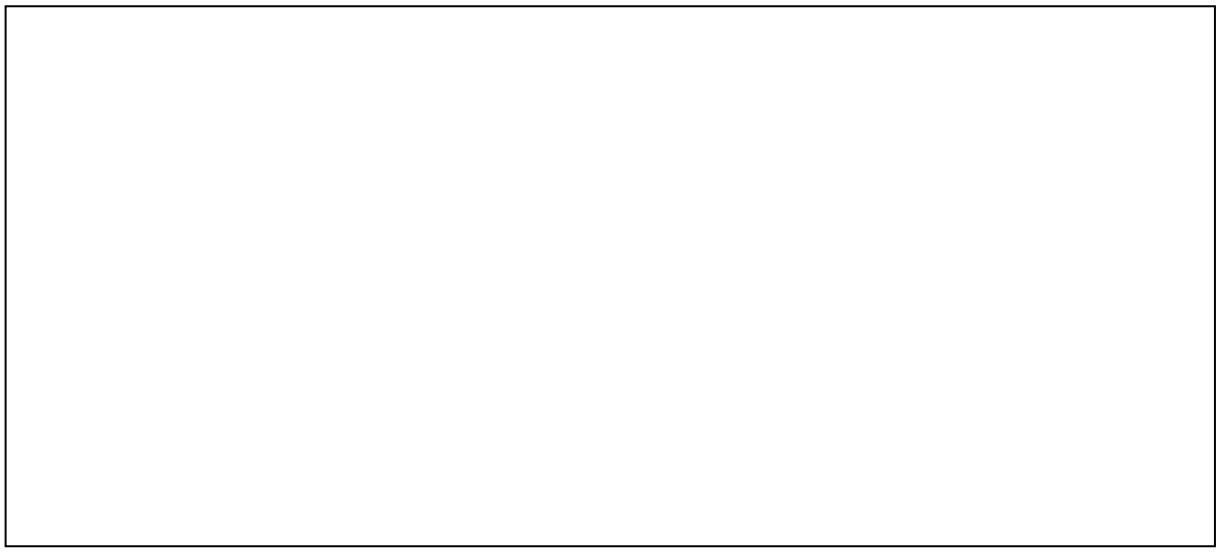
Actividad 4.3: SRound MColumns – SubClave.

	Start of Round				*After SubBytes*				*After ShitfRows*				*After MixColumns*				*Round Key Value*			
Ronda 1																				
Ronda 9																				

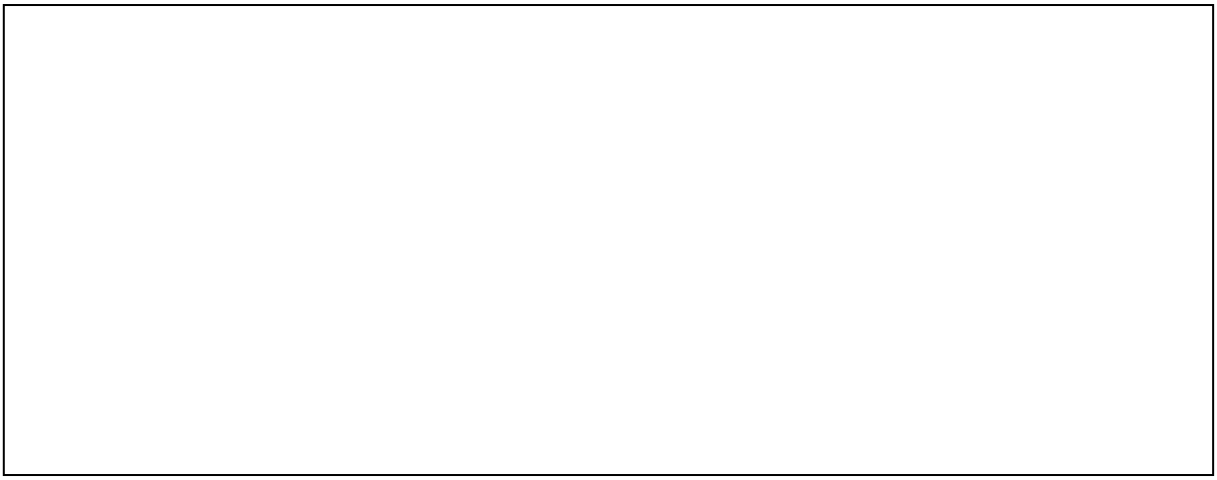
Actividad 4.3: Ronda 1 –Ronda 9.

Actividad 4.3: Ronda Final.

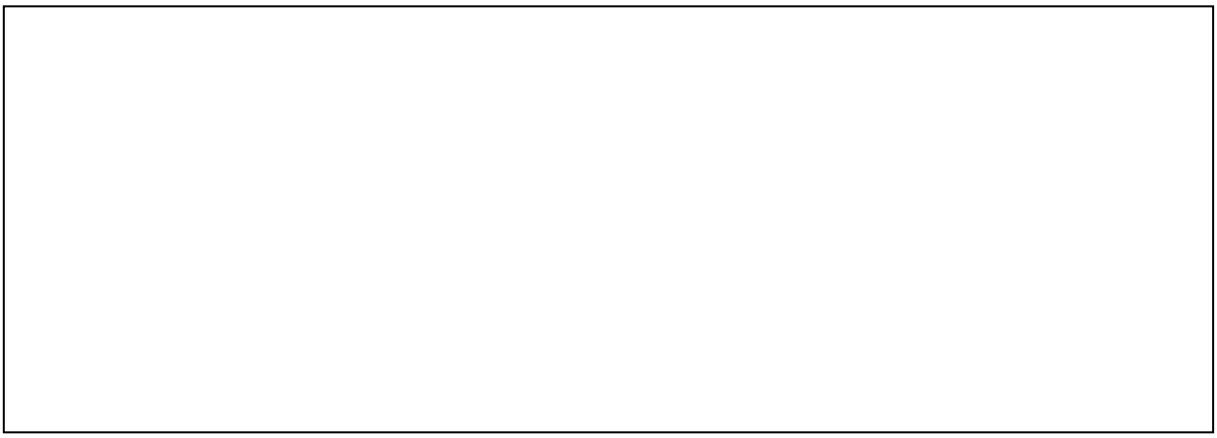
Actividad 4.3: Datos descifrado.



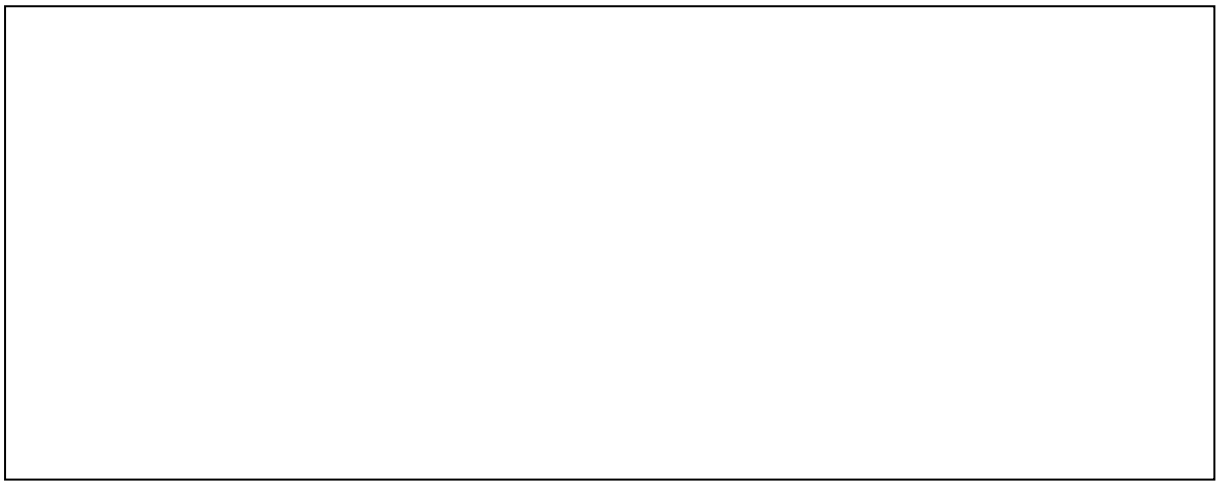
Actividad 4.3: Subclaves descifrado.



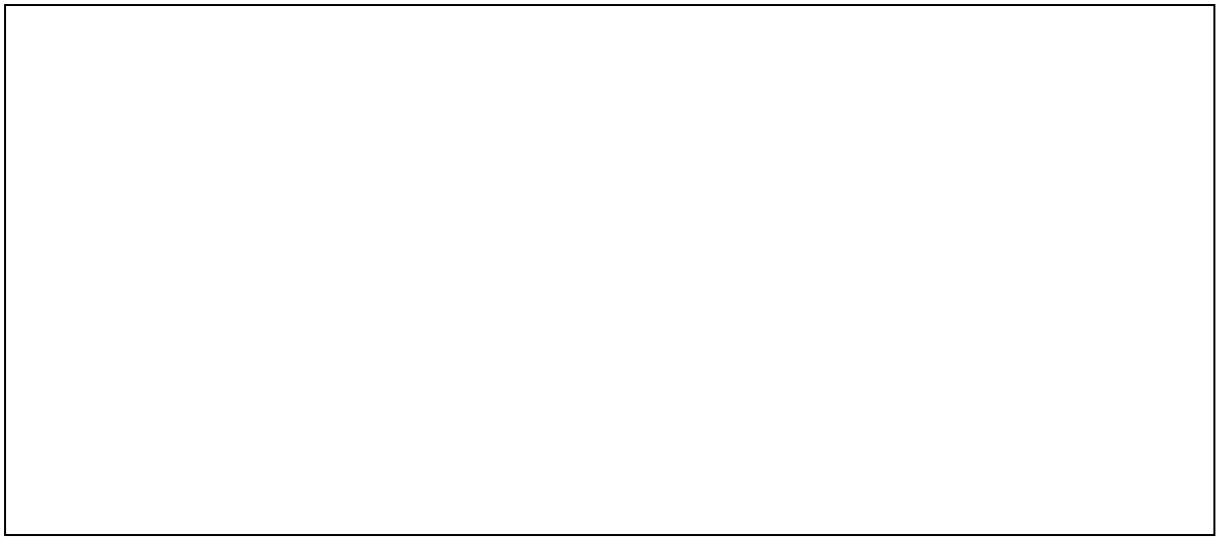
Actividad 4.3: AddRoundKey descifrado.



Actividad 4.3: Inv Shift Rows.



Actividad 4.3: Inv SubBytes.



Actividad 4.3: Inv Mix Columns.

	Round Key Value				AddRoundKey				After MixColumns				After ShitfRows				After SubBytes			
Ronda 9																				
Ronda 1																				

Actividad 4.3: Ronda 9 –Ronda 1.

Actividad 4.3: Ronda inicial descifrado.

ACTIVIDAD 4.4: Cree un script que sea capaz de cifrar y descifrar cualquier tipo de archivo, sea imagen, pdf, texto plano. Se debe hacer uso de las 3 claves, ya sea a 128, 192 o 256 bits, estas deben ser ingresadas como texto.

Se debe hacer uso de un vector de inicialización, por lo cual debe usar el modo CBC. Si el usuario no indica el vector, debe crearse uno aleatorio. Todo debe guardarse en archivos, ya sea cifrado o descifrado.

A continuación, se muestran ejemplos de script, el script fue desarrollado en python y se hace uso de banderas para el manejo de la información.

En la Figura N°4. 1 se muestra el cifrado para un archivo txt, un *exe, un jpg y un pdf*.

En la Figura N° 4.2 se muestra el descifrado de estos archivos, como puede observar dado que el vector puede o no ser colocado, no es solicitado al descifrar, ya que en dado caso que se haya otorgado de manera aleatoria, el usuario no sabrá cuál es este vector.

En el caso de las claves, se solicita dar las 3 opciones de uso, por lo cual el uso de Fernet no es adecuado, este solo hace el cifrado en 128 bits.

Las claves las puede manejar de la misma manera que en la práctica pasada, es decir con el uso de hash y adecuar los caracteres de acuerdo con la longitud solicitada.

Figura N° 4. 1: Cifrado de distintos archivos con el uso del script.

Cd ..

Figura N° 4. 2: Descifrado de archivos usando script.

En la Figura N° 4.3, se muestra la verificación entre el archivo original y el archivo descifrado, haciendo uso de shas256sum, para confirmar que es el mismo archivo y el descifrado se realizó con éxito sin afectar su integridad.

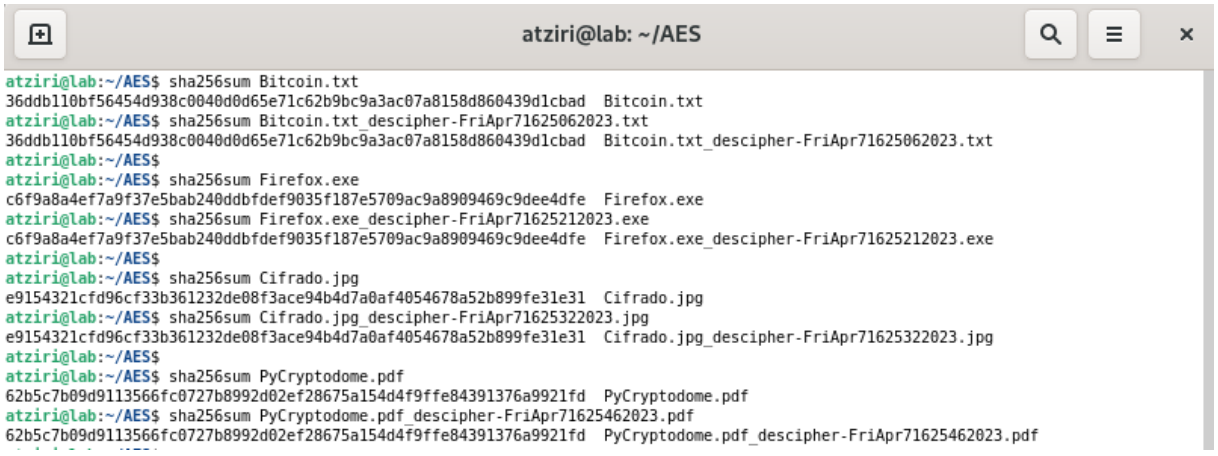


Figura N° 4. 3: Verificación de los archivos entre el original y el descifrado.

En la Figura N° 4.4, se muestra cómo se observa el directorio con todos los archivos, los originales, los cifrados y los descifrados.

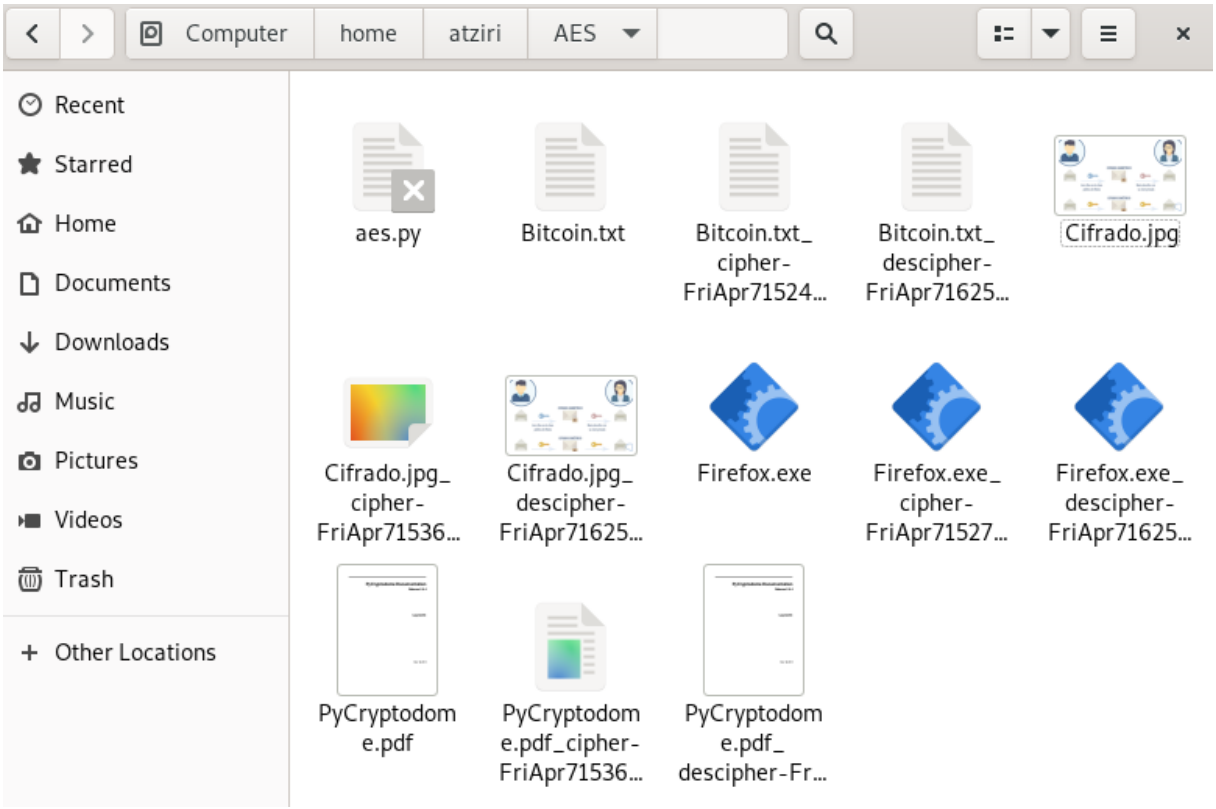
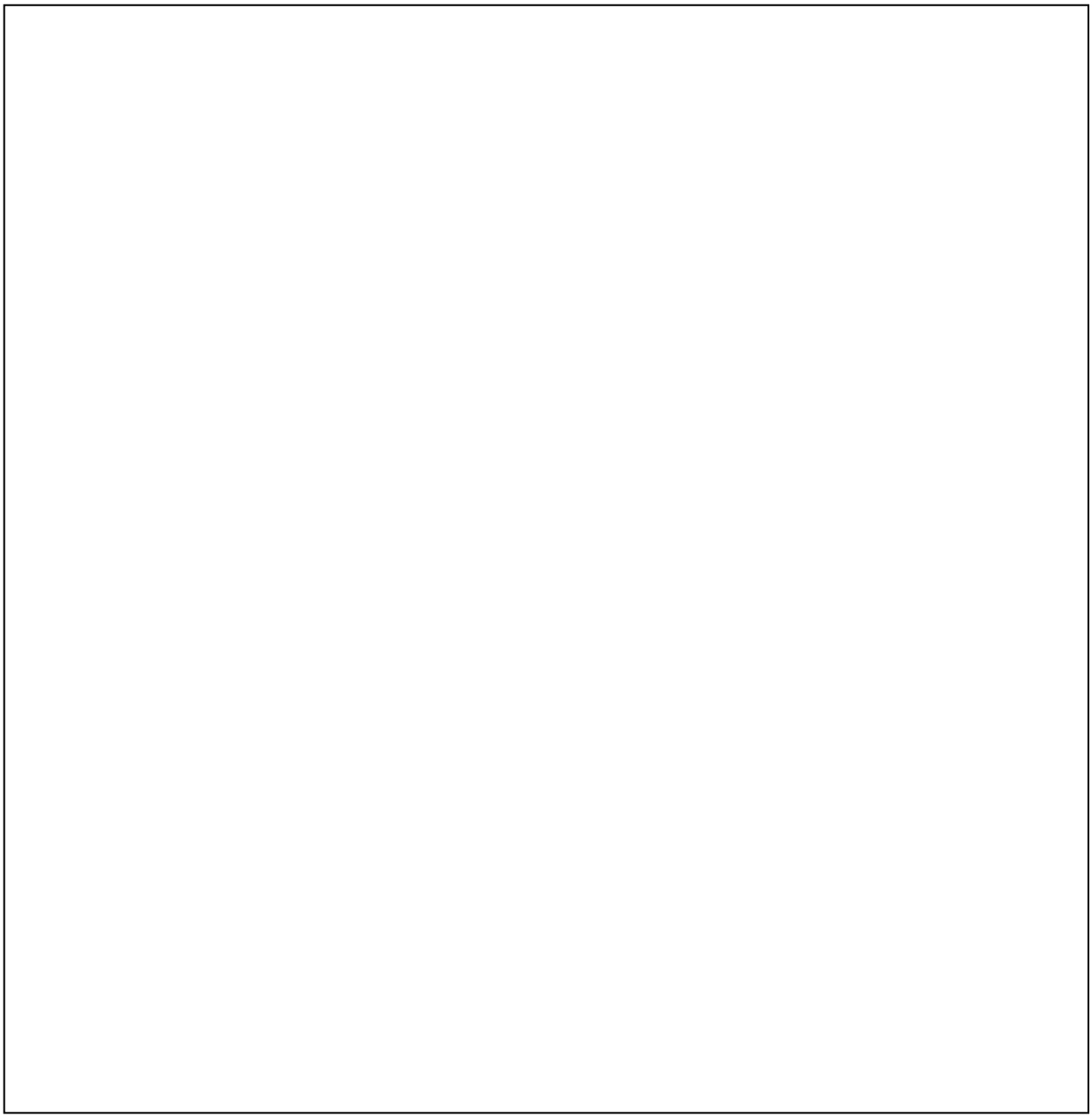


Figura N° 4. 4: Vista de archivos en el directorio.

Para la realización de su script tome en cuenta lo siguiente:

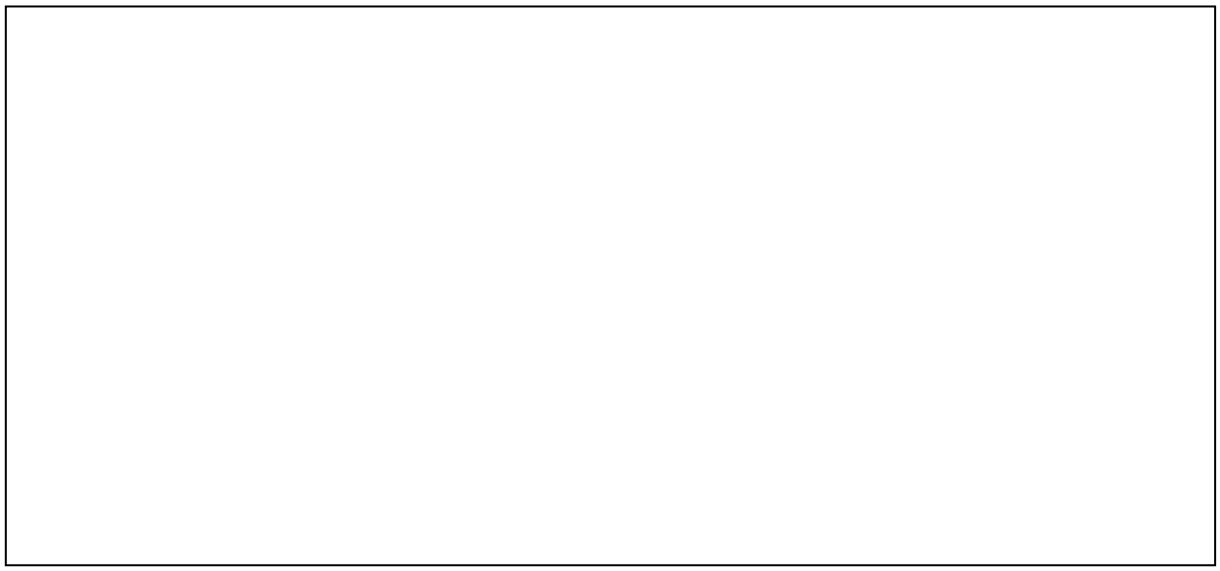
- Recuerde, se debe dar la opción de poder cifrar con cualquier tipo de clave.
- Debe cifrar y descifrar cualquier tipo de archivo, como se observa en los ejemplos se puede cifrar cualquier tipo de archivo conocido.
- Tome en cuenta todos los posibles errores, ya sea que no se ingrese un archivo, que no se indique la clave, o que no se cumpla con las longitudes establecidas por el algoritmo.
- Recuerde que debe comentar todo su código y hacerlo llegar a su docente como se lo indique.
- En caso de usar Python puede hacer uso de la PyCryptodome, este ya tiene implementación para AES-CBC.

En los siguientes recuadros, coloque capturas de pantalla haciendo uso de su script.

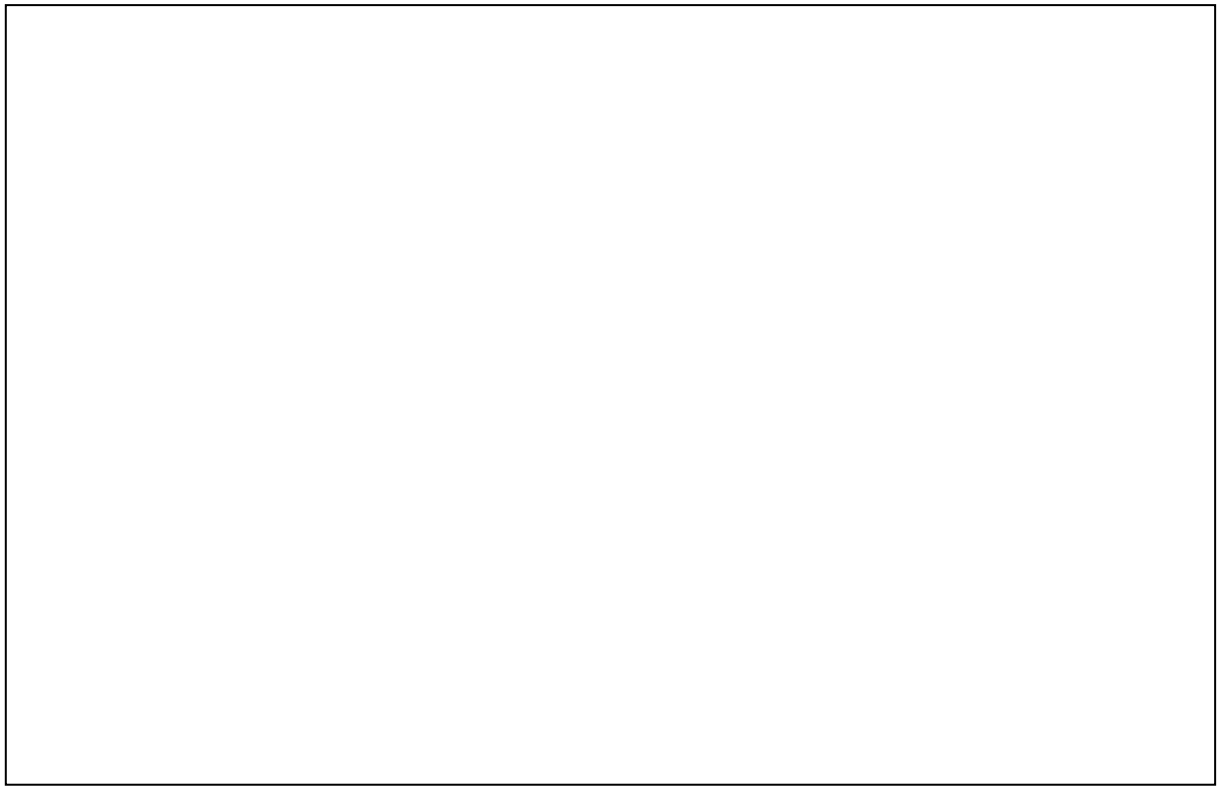


Actividad 4.4: Cifrado de archivos.

Actividad 4.4: Descifrado de archivos.



Actividad 4.4: Verificación de archivos original y descifrado mediante hash.



Actividad 4.4: Archivos originales, cifrado y descifrado en directorio.

ACTIVIDAD 4.5: Así como OpenSSL ayuda con el cifrado de archivos para DES, también lo puede realizar con AES entre otros algoritmos simétricos.

La manera de realizar el cifrado es muy similar a lo realizado en la Actividad 3.4 de la «*Práctica N°3: Data Encryption Standard*». Recuerde que AES funciona con 3 modos distintas longitudes de clave, y distintos modos de cifrado (como ejemplo en la Actividad 4.4 se usó el modo CBC) por lo cual, en OpenSSL se pueden utilizar estas distintas variantes.

El comando a ejecutar para realizar el cifrado AES con una clave de 256 bits en modo CBC es: *openssl enc -aes-256-cbc -in «nombre_de_archivo_a_cifrar» -out «nombre_archivo_cifrado»* -pbkdf2 tal como se muestra en la Figura N° 4.5.

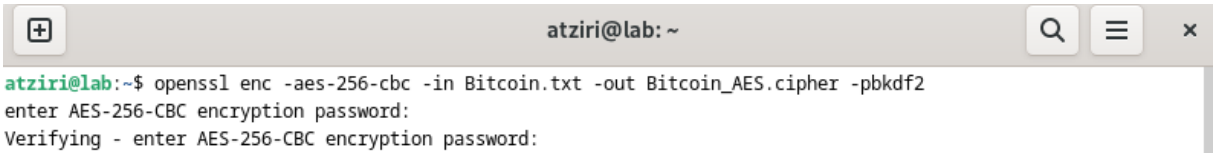


Figura N° 4. 5: Cifrado de un archivo con OpenSSL.

Para descifrar el archivo, se hace uso del comando: *openssl enc -aes-256-cbc -d -in «nombre_de_archivo_a_descifrar» -pbkdf2*, del mismo modo, al final se puede agregar la bandera *-out nombre_archivo* para guardar el contenido del descifrado en un archivo, el uso de esto dependerá del manejo que dé a la información cifrada. El ejemplo se muestra en la Figura N° 4.6

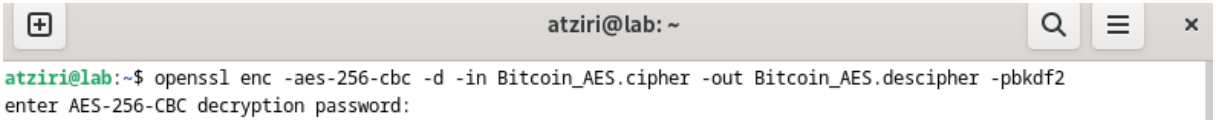


Figura N° 4. 6: Descifrado de un archivo con OpenSSL.

Conclusiones:

1. De forma resumida, explique el algoritmo de cifrado AES.

2. Para usted, ¿Cuál es el proceso más complejo en la realización del cifrado AES? Argumente su respuesta

3. ¿Se puede realizar el cifrado AES si el mensaje es menor a 128 bits? Justifique su respuesta.

4. ¿Cuál es el modo de cifrado AES que hace uso del vector de inicialización?

5. Comentario y conclusiones adicionales.

Actividades prácticas

Sección 3: Criptografía Asimétrica

UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

ACTIVIDADES PRÁCTICAS: CRIPTOGRAFÍA

5. DIFFIE - HELLMAN

DOCENTE

DIFFIE - HELLMAN

Objetivo

Conocer el algoritmo Diffie y Hellman desde el desarrollo matemático y su uso.

Justificación

El principal problema que plantea el uso de algoritmos simétricos es el intercambio seguro de las claves simétricas, problema que fue estudiado y resuelto por Diffie y Hellman a través del algoritmo desarrollado por dichos autores cuyo trabajo es el intercambio de claves de manera segura y con esto sientan las bases para el desarrollo de los algoritmos asimétricos.

En esta práctica, se introduce el concepto de los cifrados asimétricos, permitiendo al estudiante conocer los fundamentos de la criptografía asimétrica. A través de este enfoque, se explicarán las diferencias clave entre los algoritmos de cifrado simétrico y asimétrico, especialmente en el contexto del intercambio de claves utilizando el protocolo Diffie-Hellman.

Introducción

En las prácticas anteriores se presentó a los algoritmos Data Encryption Standard (DES) y Advanced Encryption Standard (AES) como parte de los algoritmos de cifrado simétrico, es decir, algoritmos que ocupan la misma clave tanto para cifrar, como para descifrar. A partir de esta práctica se conocerán algoritmos de cifrado asimétrico, es decir que cada uno de los usuarios cuente con dos claves, una pública y una privada, en donde la clave privada es estrictamente confidencial y no debe ser compartida con nadie.

El algoritmo Diffie y Hellman (RFC 2631), que lleva dicho nombre en honor a sus creadores Whitfield Diffie y Martin Hellman. Fue dado a conocer en 1976 y publicado en el periódico New Directions in Cryptography.

El articulo puede encontrarlo en New directions in cryptography | IEEE Journals & Magazine | IEEE Xplore

El objetivo de este algoritmo, en palabras de sus propios creadores, es que sea imposible obtener la clave privada a partir de la clave pública, sin embargo, con el paso de los años se ha demostrado que obtener la clave privada matemáticamente y con números pequeños es posible, mas no a nivel de bits, por lo tanto, en aplicaciones reales se recomienda hacer uso de números primos de al menos 2048 bits, que son equivalentes a 256 caracteres, sin embargo para efectos académicos en esta práctica se utilizarán valores más pequeños.

El algoritmo en sí es sencillo. Suponga que se tienen dos individuos, Atziri y Balam, ellos se pondrán de acuerdo por un medio inseguro en los siguientes valores:

- $p \rightarrow Número\ primo\ grande$
- $\gamma \rightarrow Número\ aleatorio\ menor\ a\ p$

Teniendo estos valores, tanto Atziri como Balam toman un número aleatorio menor a p, éste será la clave privada llamada a para Atziri y b para Balam, y para ello se debe realizar la siguiente operación:

$$clave\ pública\ \ = \gamma^{clave\ privada}\ mod\ p$$

Esa operación obtiene un valor, que será conocido como clave pública, y es enviada por un medio inseguro al receptor, es decir Atziri mandará su clave a Balam y Balam mandará su clave a Atziri.

Ahora, ambos obtendrán una clave en común, realizando la siguiente operación:

$$clave\ simetrica\ =\ clave\ publica\ receptor^{clave\ privada\ propia}\ \ mod\ p$$

Ambos deben obtener la misma clave simétrica y dicha clave es la usada para realizar el cifrado y descifrado.

Como ejemplo visualice la Figura N° 5.1:

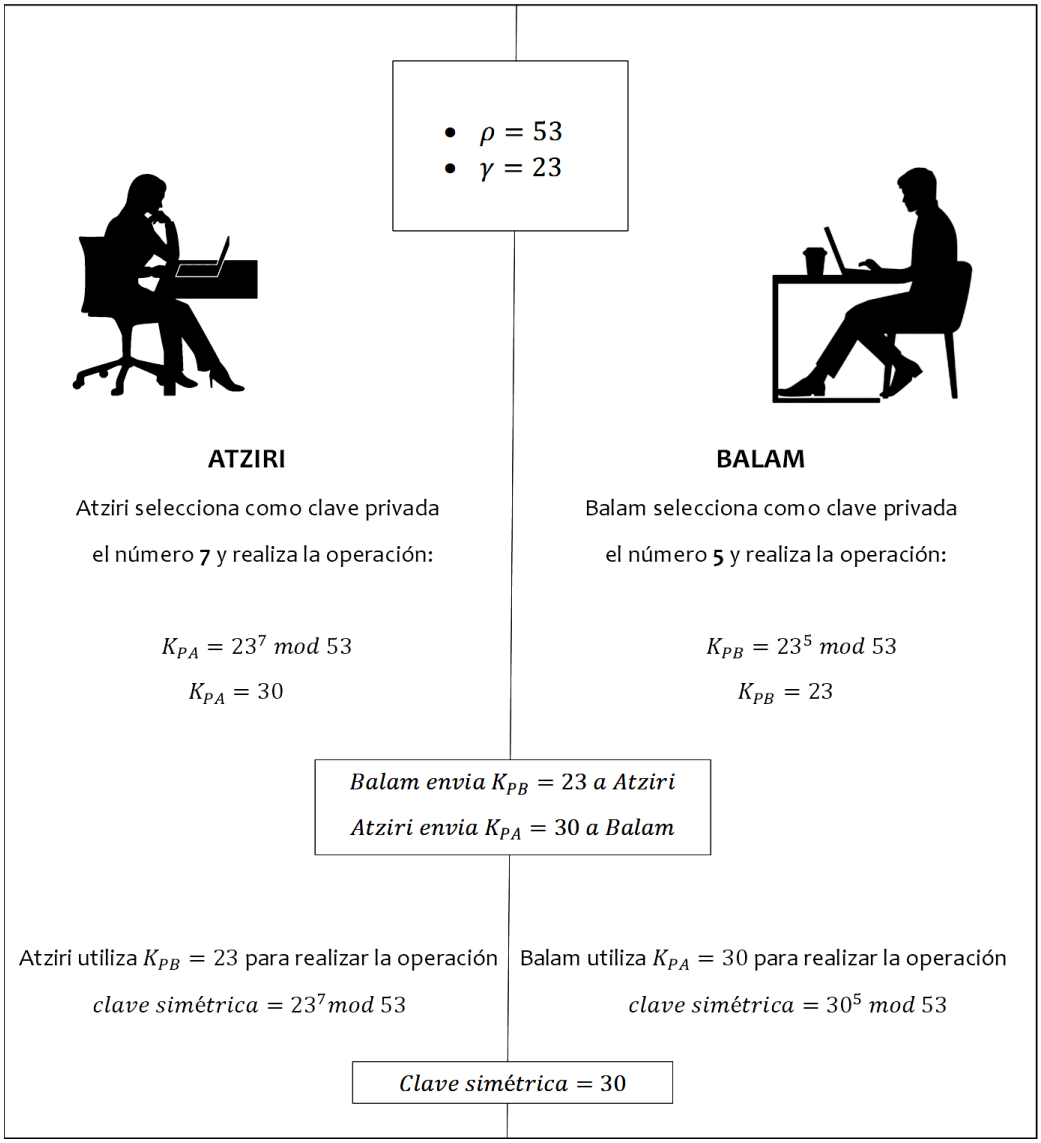


Figura N° 5. 1: Intercambio de Claves.

Diffie y Hellman fue el primer algoritmo que utilizó claves públicas y privadas para dar seguridad en los sistemas de información, a este conjunto es al que se le conoce como uso de claves asimétricas y dio paso al desarrollo de más algoritmos de este tipo altamente utilizados hoy en día, por lo cual es importante conocer su funcionamiento.

Puede encontrar la documentación del algoritmo en: https://tools.ietf.org/html/rfc2631

Equipo y material necesario

Máquina Virtual Debian, para montarla en su equipo puede seguir el «Montaje de máquina virtual», el cual es un manual de Instalación MV, en el que se encuentran las instrucciones tanto para VMware como para VirtualBox. Así mismo puede consultar el Anexo D que contiene ejemplos de cómo debe colocar sus respuestas en la práctica.

Desarrollo

R*ecuerde que debe hacer uso del usuario indicado en la Actividad 4.1 de la práctica N° 4 «Advanced Encryption Standard».*

ACTIVIDAD 5.1: Cree un directorio **DH** en su directorio home, tal como se ha venido trabajando. La actividad se realiza en equipos de dos integrantes.

Uno de los integrantes realizará las actividades correspondientes a **Atziri** y el otro a **Balam**. El objetivo es buscar un archivo en el cual va a encontrar los datos necesarios para realizar el intercambio de claves, cada uno tendrá las instrucciones que debe seguir. Ambos deben contestar una pregunta, la respuesta sólo es el número, es decir si la pregunta es ¿Cuántos bits utiliza como clave el algoritmo DES? La respuesta sólo es «64».

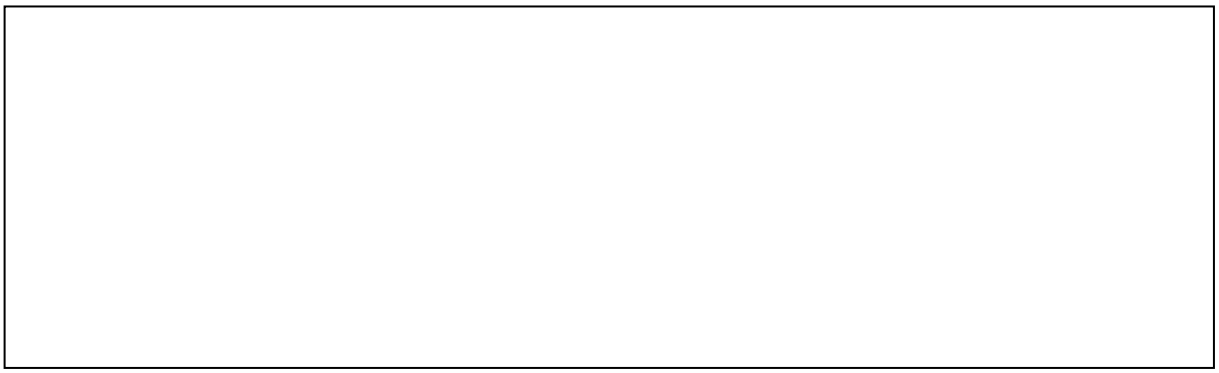
Pista: Para obtener resultados más rápidos busque en los directorios correspondientes al usuario crypto.

Para Atziri

Nombre de quien realizó esta parte:	

Conteste: ¿En qué año fue dado a conocer el algoritmo Diffie y Hellman? Obtenga el valor hash en SHA256 de su respuesta y busque un archivo que contenga dicha cadena. En ese archivo encontrará información que compartirá con Balam.

En el siguiente recuadro coloque la respuesta a la pregunta, el resultado del valor hash, explique lo realizado para encontrar la cadena como contenido de un archivo, y en qué directorio se encuentra dicho archivo.



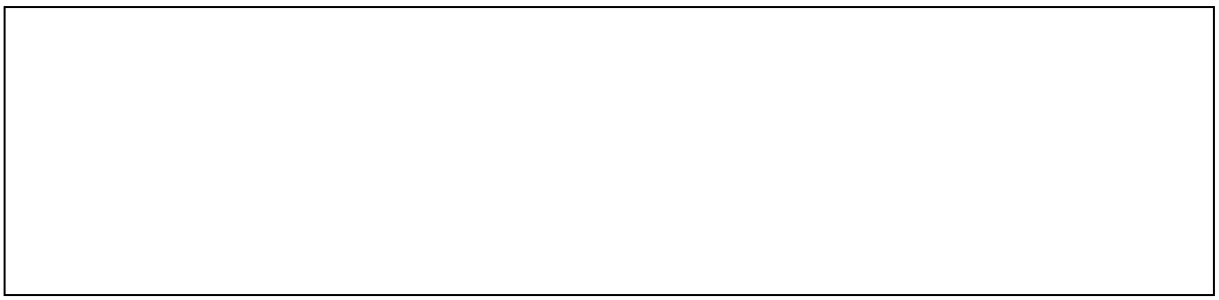
Actividad 5.1: Búsqueda de archivos.

Para Balam

Nombre de quien realizó esta parte:

Conteste: ¿Cuál es la cantidad de bits mínimos que se recomienda utilizar en los números primos del algoritmo Diffie - Hellman? Obtenga SHA-1 dé su respuesta y busque un archivo que tengo como nombre la cadena obtenida, el archivo es un archivo oculto. En ese archivo encontrará información que compartirá con Atziri.

En el siguiente recuadro coloque la respuesta a la pregunta, el resultado del valor hash, explique lo realizado para encontrar el archivo y en que directorio se encuentra el archivo.



Actividad 5.1: Búsqueda de archivos.

Para Atziri y Balam

Cada uno de ustedes tiene instrucciones en el archivo que encontró con los cuales puede realizar el intercambio de claves Diffie y Hellman, pero los datos son incompletos para cada uno, hagan el intercambio de los valores $\boldsymbol{p}$ y $\boldsymbol{\gamma}$. Puede realizar el intercambio por el medio que desee, sin embargo, en los ejemplos se realiza mediante la creación de un canal de comunicación en donde uno será el servidor y el otro el cliente, harán uso del comando «netcat», por lo cual deben conocer y compartir su dirección IP en caso de ser el servidor y por fines prácticos, encontrarse en la misma red.

Quien sea servidor deberá ejecutar: nc –p «puerto» -l

Quien sea cliente deberá ejecutar: nc «ip del servidor» «puerto»

Donde *ip del servidor* debe ser la dirección IP de quien tome el papel de servidor, y *puerto* debe ser el mismo puerto para ambos. En el caso del servidor, *-l* indica que estará a la escucha.

El canal de comunicación debe verse algo parecido a lo mostrado en la Figura N° 5.2, en este caso el servidor es Atziri.

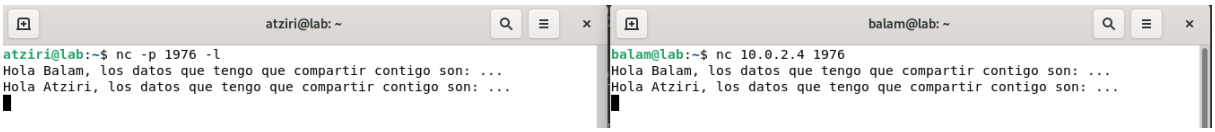


Figura N° 5. 2: Establecer canal de comunicación.

Coloquen cada uno en donde corresponda la captura de pantalla de su conversación, en donde también se muestre el intercambio de las claves públicas, recuerde que las todas las capturas deben realizarse con el usuario creado en la práctica N° 4, es decir, en lugar de utilizar Atziri y Balam, use su usuario.

Actividad N° 5.1: Conversación del lado de quien realizó actividades referentes a Atziri.

Actividad N° 5.1: Conversación del lado de quien realizó actividades referentes a Balam

Ahora, usando la clave simétrica, obtendrán el valor hash usando la función que decidan, de dicha clave, y el hash será usado como clave para cifrar un archivo de texto o imagen, usando el script

creado en *la Practica N° 4 «Advanced Encryption Standard» en la Actividad 4.4,* o bien OpenSSL como lo realizado en la *Actividad 4.5*

Cada participante debe enviar el archivo cifrado a su receptor para que pueda descifrarlo. Dependiendo del método de cifrado, serán los datos que deba compartir con su receptor. Para el caso de los ejemplos, se hace uso del script.

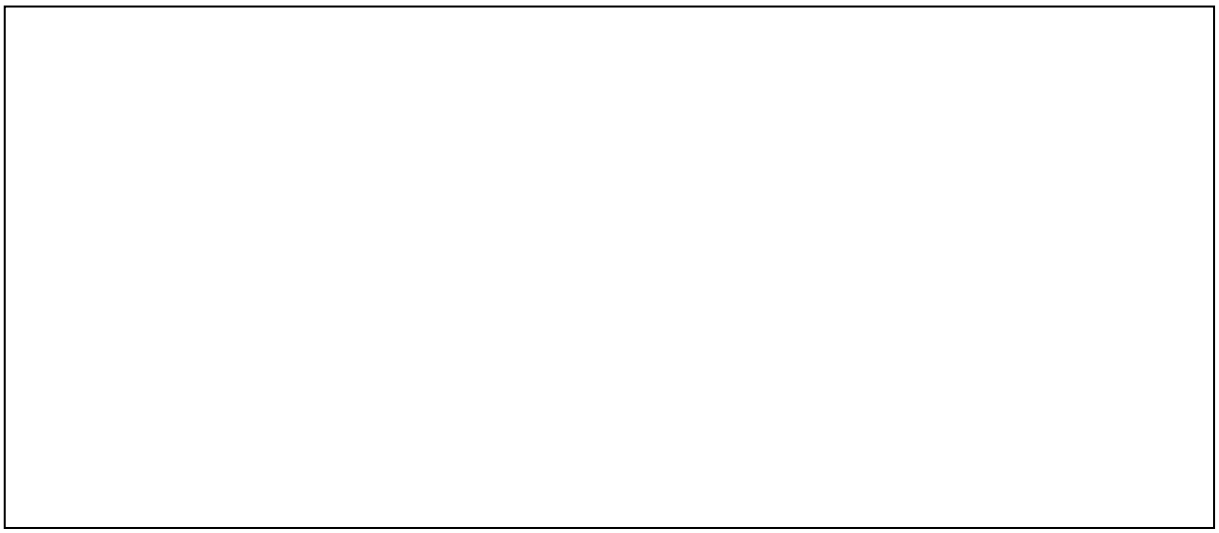
Como ejemplo, suponiendo que Atziri y Balam tienen como clave simétrica «2967» deciden utilizar RMD160, por lo cual obtienen RMD160 de «2967» y dicha cadena la utilizan como clave. Atziri cifrará una imagen y Balam un archivo de texto, al tener los elementos cifrados, deben enviárselos, para esto también pueden hacer uso de «nc» en donde uno será receptor y el otro emisor. (Hermoso, R., 2011)

El receptor hace uso del comando: nc –l –p «puerto» > «nombre a guardar el archivo»

El emisor hace uso del comando: nc «ip del receptor» -q 0 «puerto» < «nombre del archivo»

En donde puerto debe ser un número de puerto que ambos dispongan, deben usar el mismo, *-l* se refiere que está a la escucha y –q 0(cero) se refiere a que en cuanto reciba el archivo, el canal de comunicación se cierra.

Indique: ¿Cuál es el método utilizado? ¿Cuál es la clave simétrica?, ¿Qué hash será utilizado? Y ¿Cuál es la longitud de clave a utilizar para el cifrado? es decir, dado que debe hacer uso del cifrado AES debe indicar si es clave de 128, 192 o 256 bits. Además, indicar el tipo de archivo que será enviado por cada uno.



Actividad 5.1: Instrucciones para cifrado de clave simétrica y su valor.

En la Figura N° 5.3 se muestra cómo deberían enviarse y descifrarse los archivos recibidos, en este caso es Balam quien cifra el archivo, lo envía y Atziri lo recibe y descifra.

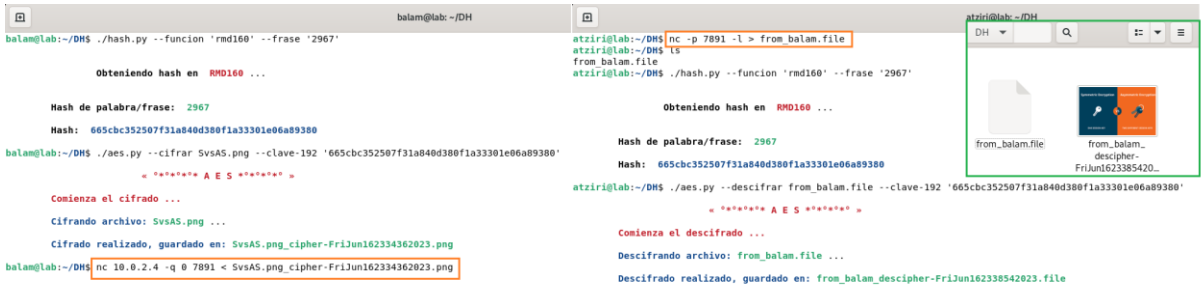
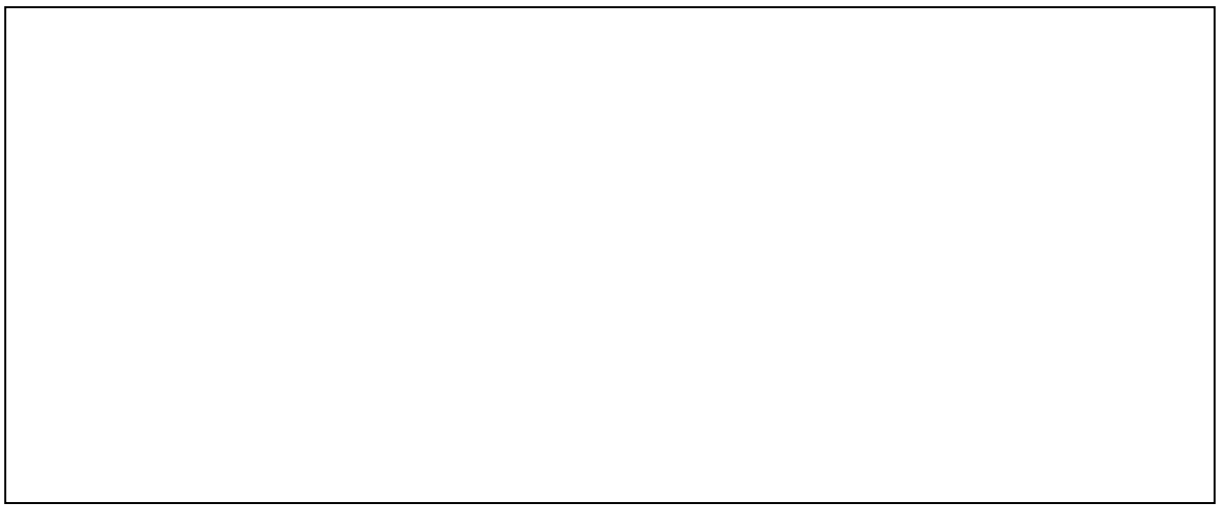


Figura N° 5. 3: Envío, cifrado y descifrado de archivos.

En los siguientes recuadros coloque capturas de pantalla en donde se observe cómo se transfieren los archivos, cómo los descifra de acuerdo al método seleccionado y mostrar el contenido del archivo descifrado.

Actividad N° 5.1: Descifrado del archivo que 'Balam' envió a 'Atziri'.



Actividad N°5.1: Descifrado del archivo que 'Atziri' envió a 'Balam'.

ACTIVIDAD 5.2: (Gondon, S., s.f.) Como se mencionó anteriormente, para que el algoritmo funcione de la mejor manera, se debe hacer uso de números extremadamente grandes. Se realizará nuevamente la obtención de claves, pero en esta ocasión haciendo uso de OpenSSL.

Para lo cual, primero debe generarse los parámetros públicos, es decir el equivalente a $\boldsymbol{p}$ y $\boldsymbol{\gamma}$, ya sea que los realice Atziri o Balam y los intercambien como lo hizo en la Actividad 5.1.

Indique quien realizará la generación de los parámetros:

Ejecute: «*openssl genpkey -genparam -algorithm DH -out pab.pem*» donde **pab.pem** se refiere al archivo en donde guardará los datos y en su caso cambie las iniciales **ab** por las iniciales de usted y su compañera/compañero.

Envíe el archivo a su compañera/compañero, para que ambos cuenten con dicho archivo.

Para observar los parámetros, del archivo puede hacer uso del comando «*cat*», «*more*» o «*less*» donde mostrará los resultados de forma codificada, o usando el comando «*openssl pkeyparam –in pab.pem - text*» recordando cambiar *pab.pem* por el archivo que usted creo con sus iniciales.

En la Figura N° 5.4 se muestra como observar los parámetros.

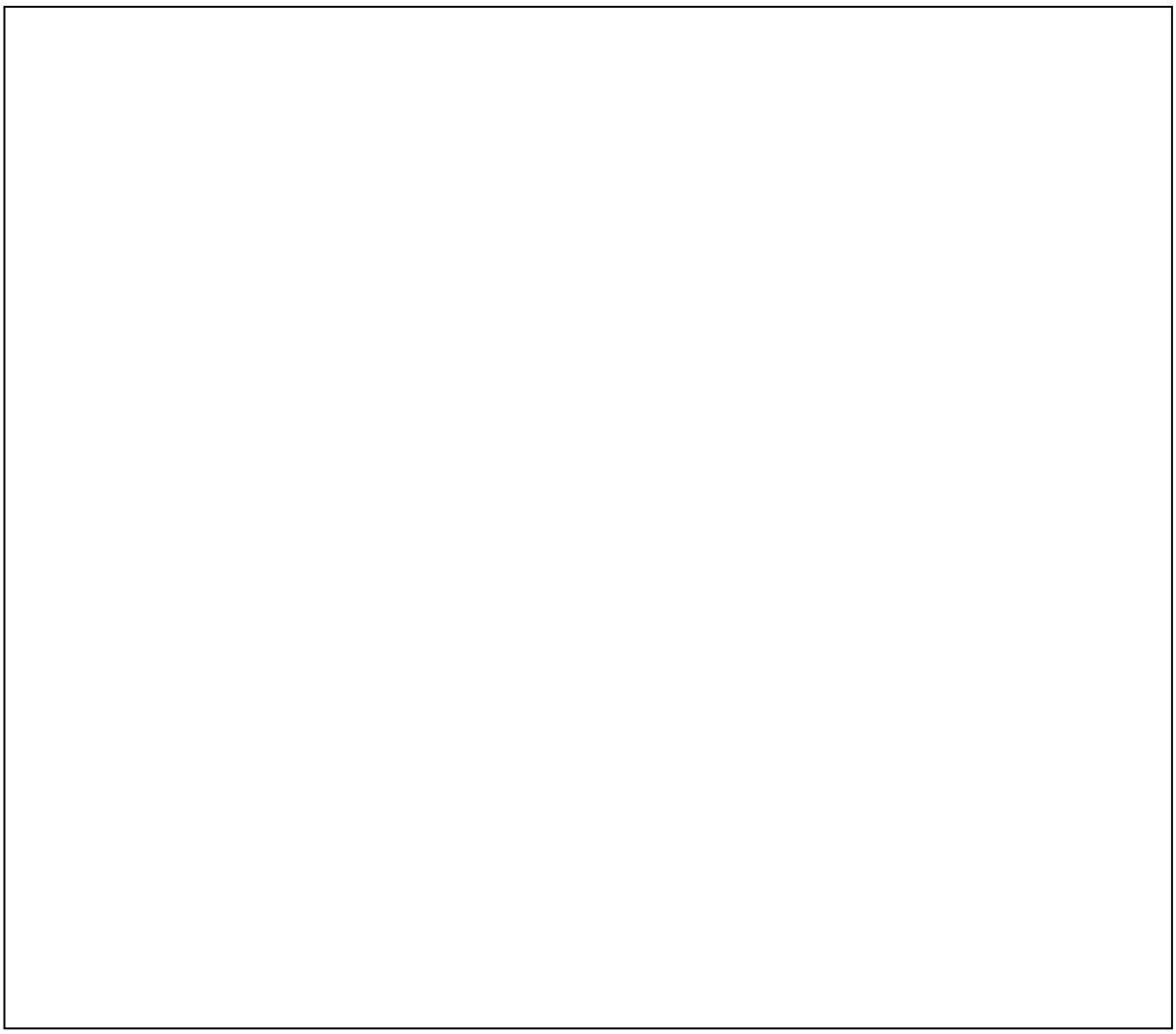
```
atziri@lab:~/DH$ openssl pkeyparam -in pab.pem -text
-----BEGIN DH PARAMETERS-----
MIIBCAKCAQEA2uYus9yMddh3iPy4JY0rddMT85bwLN90jDpCQ6V4ZwmTDulU7P2K
Ja9R8h5XUQ2bj91AXdn39ZK4ZxqqgRdRmcuCF74Wl4FJNQqtAhrUXhcN+aaYowTh
/WyF/gwEkGjK+sI9OJuqGlSImw14M7QMNykb7RkU3uE+xyWMooB6mymDE3QGZFgo
YLSDl1RmLACWuR2A6TAk4hFFLzy0Nafi2fpFABgmnYGoXVt3OiQ2dZS0MRIwZ0No
LiGOaU45k23ijsdLr/Pq1ceb9itTqBvI9gLcut5cXKBMl/s5gkBl9myVBeyEgwKI
1AMQ1jE6PTn9q0cFcdNOmkdddXsKmEAeQwIBAg==
-----END DH PARAMETERS-----
DH Parameters: (2048 bit)
    prime:
        00:da:e6:2e:b3:dc:8c:75:d8:77:88:fc:b8:25:8d:
        2b:75:d3:13:f3:96:f0:2c:df:74:8c:3a:42:43:a5:
        78:67:09:93:0e:e9:54:ec:fd:8a:25:af:51:f2:1e:
        57:51:0d:9b:8f:dd:40:5d:d9:f7:f5:92:b8:67:1a:
        aa:81:17:51:99:cb:82:17:be:16:97:81:49:35:0a:
        ad:02:1a:d4:5e:17:0d:f9:a6:98:a3:04:e1:fd:6c:
        85:fe:0c:04:90:68:ca:fa:c2:3d:38:9b:aa:1a:54:
        88:9b:0d:78:33:b4:0c:37:29:1b:ed:19:14:de:e1:
        3e:c7:25:8c:a2:80:7a:9b:29:83:13:74:06:64:58:
        28:60:b4:83:97:54:66:2c:00:96:b9:1d:80:e9:30:
        24:e2:11:45:2f:3c:b4:35:a7:e2:d9:fa:45:00:18:
        26:9d:81:a8:5d:5b:77:3a:24:36:75:94:b4:31:12:
        30:67:43:68:2e:21:8e:69:4e:39:93:6d:e2:8e:c7:
```

Figura N° 5. 4: Parámetros creados para la creación de claves Diffie – Hellman con OpenSSL.

En el siguiente recuadro explique qué fue lo que realizó para enviar los datos a su compañera/compañero.

Actividad N° 5.2: Envió del archivo con los parámetros públicos.

Además, el receptor debe mostrar el contenido codificado del archivo que recibió en el siguiente recuadro.



Actividad N° 5.2: Revisión del archivo con los parámetros.

Ahora, tanto Balam como Atziri deberán crear sus propias claves privada y pública, a partir de los parámetros en común, por lo cual deberán ejecutar:

«*openssl genpkey -paramfile pab.pem -out Atzirik.pem*» donde deben cambiar el nombre del archivo *pab.pem* por el que crearon en el paso anterior y en el archivo *Atzirik.pem* cambiar el nombre Atziri por el suyo.

Puede visualizar los parámetros con el comando «*openssl pkey -in Atzirik.pem -text -noout*» debe ver algo parecido a lo que mostrado en la Figura N° 5.5:

```
atziri@lab:~/DH$ openssl genpkey -paramfile pab.pem -out Atzirik.pem
atziri@lab:~/DH$ openssl pkey -in Atzirik.pem -text -noout
DH Private-Key: (2048 bit)
    private-key:
        54:68:ea:6e:89:69:cc:1b:cb:67:87:7b:8e:70:6a:
        c7:93:8c:9e:da:93:c6:cf:f5:4d:b7:f7:57:1a:87:
        fc:e5:22:cd:a1:a5:f9:17:a0:7e:06:a0:b3:e1:0e:
        04:6c:5f:02:db:ce:0b:36:ea:8e:b8:5a:39:da:00:
        18:e1:49:e0:b6:ee:70:03:8c:50:36:ec:e9:49:dd:
        ec:99:b4:60:40:e9:3f:25:e9:d9:8f:10:b8:3e:d6:
        b4:ed:e2:f3:15:93:b6:00:33:41:d6:05:4b:9b:44:
        10:2e:78:17:da:11:79:45:b2:06:2a:dd:ad:f8:a0:
        03:c1:c5:7c:f0:49:ea:11:c0:7d:e2:3f:1e:d9:82:
        aa:41:da:85:11:06:72:e5:49:dc:5e:86:16:d6:c9:
        40:a4:7f:26:d2:38:03:8e:0e:0d:02:3b:34:38:6c:
        7e:3f:e6:f6:bb:b7:66:be:b5:33:3c:a7:64:4b:26:
        94:90:bf:df:11:10:01:7c:6e:33:a5:35:9a:8c:c9:
        c9:27:27:75:c6:57:29:ce:10:6d:9c:da:63:c5:e7:
        44:8c:6f:8e:7a:c0:10:d1:e3:0e:8d:62:15:76:67:
        7f:b0:6b:8b:6e:cd:c5:65:21:7a:9d:e5:cb:ce:58:
        ea:5a:3e:21:b4:c4:1b:9b:95:00:d9:d5:da:ac:35:
        30
    public-key:
        00:a7:e1:05:06:60:a2:3c:e0:f0:39:33:a5:36:29:
        10:7b:b6:35:22:c9:68:09:12:b9:d8:96:1b:28:81:
        8e:ff:86:12:86:10:1d:4a:e7:2b:a0:b1:cd:ad:41:
        f0:0a:a0:9c:dd:b9:c4:a3:40:d0:a4:64:df:da:72:
        dc:fb:a0:97:99:ea:ba:01:88:da:5b:60:b9:ef:72:
        3d:19:6e:b5:c9:fe:a7:41:54:2a:68:7f:bc:a7:d6:
        bd:d4:ae:dc:a5:0c:9a:8d:35:73:df:d3:e5:fa:d2:
        79:0b:c3:0f:9f:54:c6:42:d8:fc:65:e1:14:8a:65:
        bd:84:df:25:41:3b:a5:b2:6c:fe:42:99:ff:dd:08:
```

Figura N° 5. 5: Visualización de parámetros correspondientes a clave privada y pública.

Ahora, debe extraer su clave pública de la generada anteriormente, para esto cada uno debe ejecutar: «*openssl pkey -in Atzirik.pem -pubout -out Atziripub.pem*», en donde *Atzirik.pem* se refiere a su archivo creado anteriormente y *Atziripub.pem* es el archivo donde guardará su clave pública. Recuerde cambiar el nombre de Atziri por el suyo.

Ejecute: «*openssl pkey -pubin -in Atziripub.pem -text*» para mostrar los datos de la clave pública, tal como se muestra en la Figura N° 5.6:

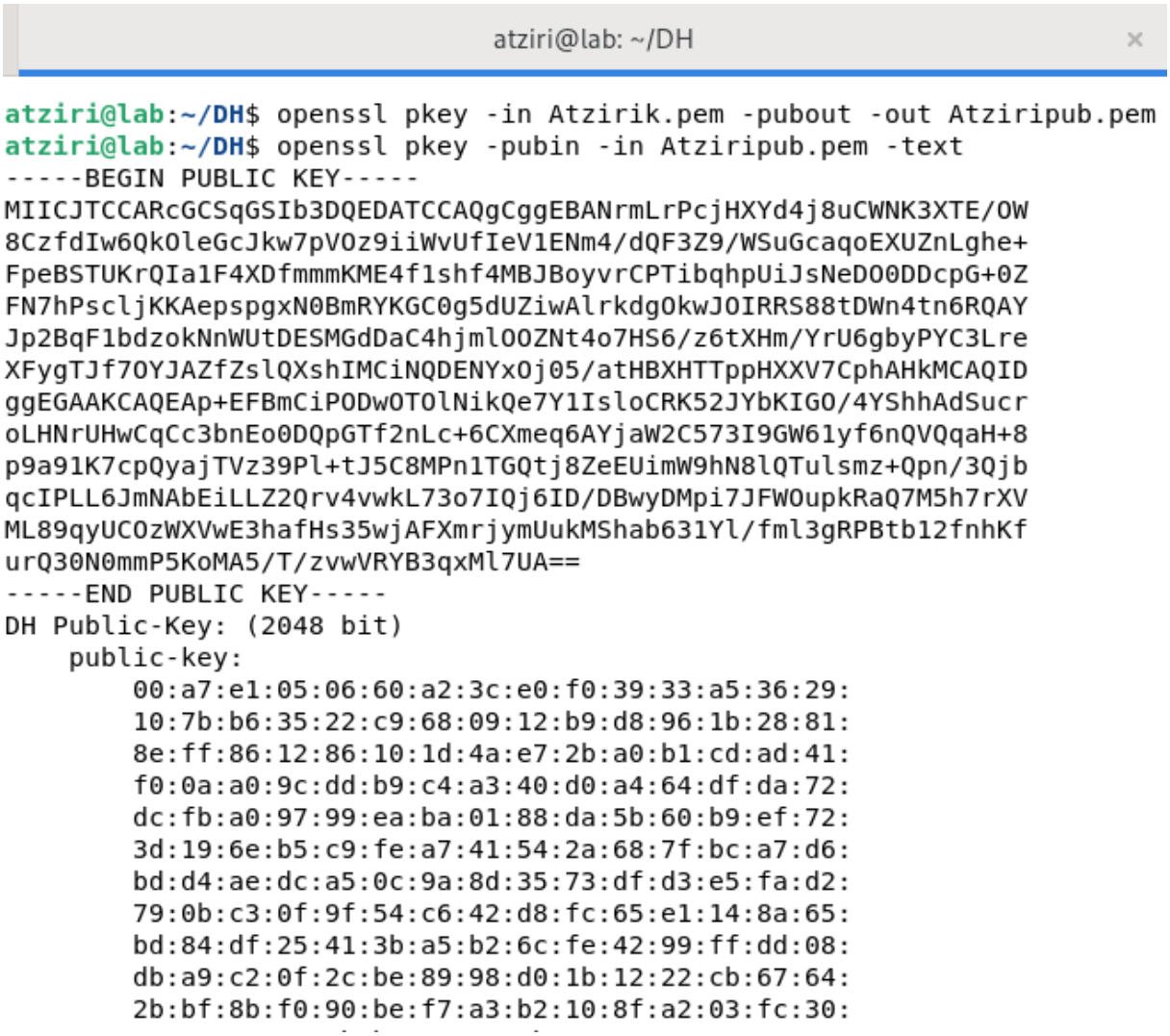


Figura N° 5. 6: Visualización de clave pública.

Intercambien su clave pública como lo han realizado hasta el momento. ¡IMPORTANTE! Solo se comparte la clave pública, ponga mucha atención en el archivo que compartirá.

Ahora, que cada uno conoce la clave pública del otro, y cuenta con los parámetros iniciales, deben crear lo que anteriormente se conoció como clave simétrica. Para eso, ambos ejecuten:

«*openssl pkeyutl -derive -inkey Atzirik.pem -peerkey Balampub.pem -out SimetricAtziri.bin*»

Recuerde utilizar su equivalente a *Atzirik.pem* que se refiere al archivo de creación de claves para Atziri, y *Balampub.pem* es la clave pública de Balam y nuevamente en *SimetricAtziri.bin* cambie *Atziri* por su nombre, en este archivo se guarda la clave simétrica.

Ahora, verifiquen que los archivos que acaban de crear sean iguales, para esto ejecute el comando: «*xxd SimetricAtziri.bin*»

Debe visualizar algo parecido a lo mostrado en la Figura N° 5.7, en donde se observa como Atiziri y Balam crean la clave simétrica y ésta es la misma.

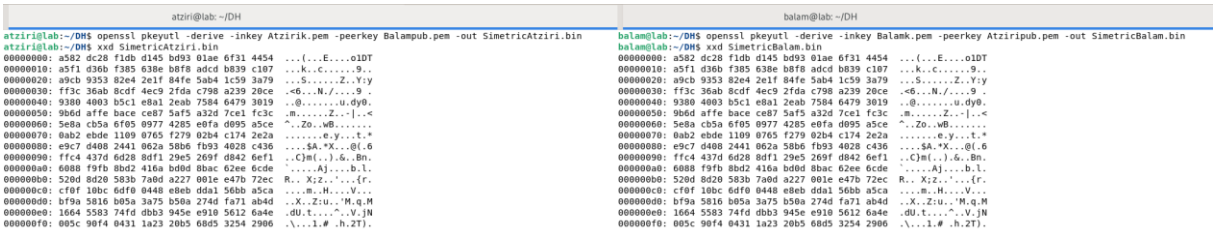


Figura N° 5. 7: Creación y verificación de clave simétrica.

En el siguiente recuadro, coloque una captura de pantalla en donde se muestre la ejecución del comando que verifica que sean iguales, tal como se mostró en la Figura N° 5.7:

Actividad N° 5.2: Creación y verificación de clave simétrica.

Finalmente, solo queda que cada uno cifre su mensaje y se envíe a su respectivo receptor, en este caso se va a realizar el cifrado haciendo uso de AES con una clave de 192 bits en modo CBC, para cifrar se puede hacer uso de OpenSSL, por lo cual, tanto Atziri como Balam deben ejecutar lo siguiente (En el ejemplo, Atiziri es quien envía los archivos a Balam):

«*openssl enc -aes-192-cbc -kfile SimetricAtziri.bin -in archivo_a_cifrar -out archivo_cifrado -pbkdf2*» donde la bandera *-kfile* hace referencia al archivo con la clave, en este caso *SimetricAtziri.bin,* recuerde cambiar esta por su correspondiente. Las banderas *-in* indica el archivo a cifrar y *-out* el nombre del archivo con el cual se guardará el archivo ya cifrado.

Nuevamente envíe el archivo a su correspondiente receptor y descifré su archivo con el siguiente comando (En este caso es Balam quien recibió, por lo tanto, es quien descifra):

«*openssl enc -d -aes-192-cbc -kfile SimetricBalam.bin -in from_atziri.cipher -out from_atziri.descipher -pbkdf2*» donde *SimetricBalam.bin* debe ser su equivalente así como los archivos *from_atziri.cipher y*

from_atziri.descipher. En el descifrado es importante no olvidar la bandera *-d* ya que justo esta, indica que el proceso a realizar es el descifrado.

En la Figura N° 5.8 se visualizan los pasos descritos anteriormente, tanto el cifrado por parte de Atziri, el envío del archivo y el descifrado por parte de Balam.

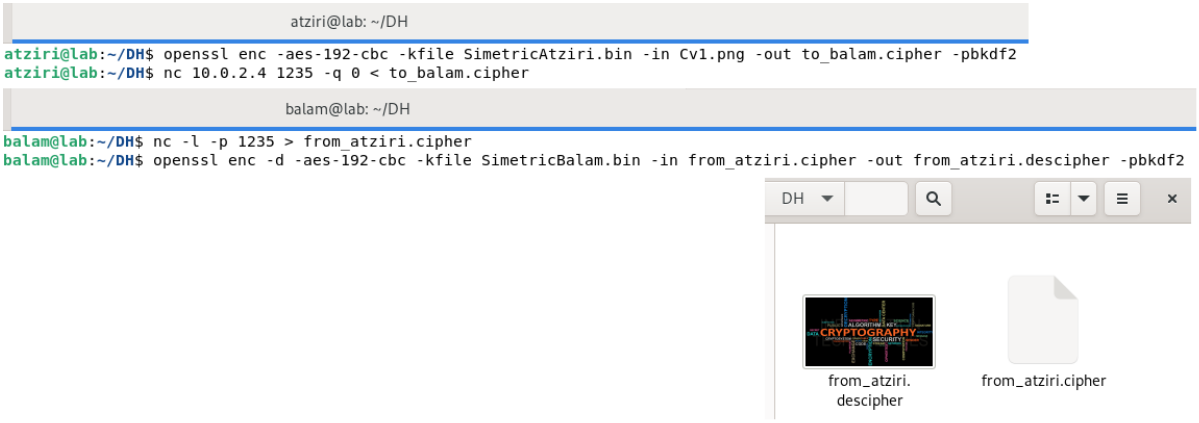
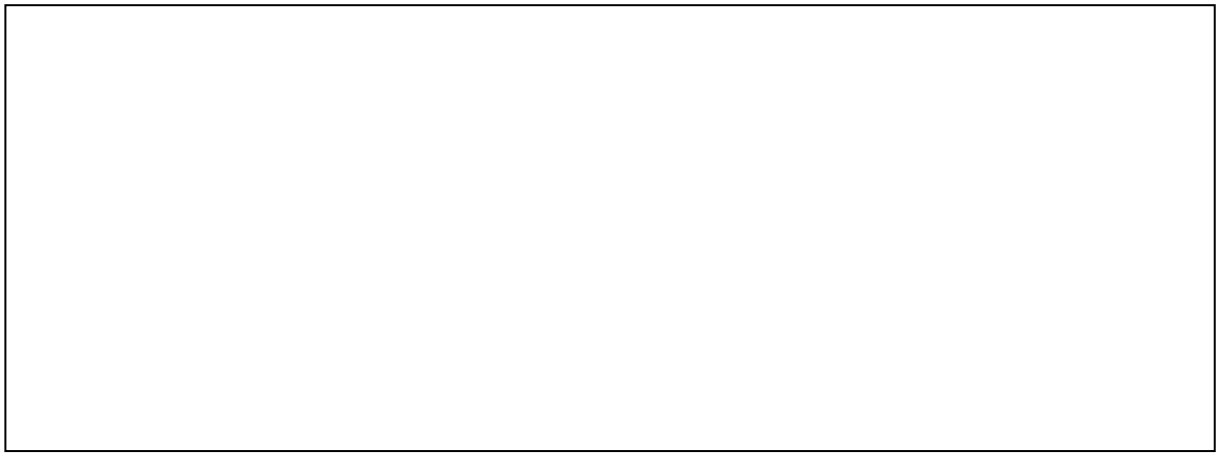


Figura N° 5. 8: Cifrado y descifrado de archivos con clave simétrica.

En los siguientes recuadros, coloque en donde le corresponda una captura de pantalla en donde se visualice que descifra su archivo.

Actividad N° 5.2: Descifrado del archivo que 'Atziri' envió a 'Balam'.



Actividad N° 5.2: Descifrado del archivo que 'Balam' envió a 'Atziri'.

ACTIVIDAD 5.3: En la introducción se comentó que es posible obtener la clave privada a partir de los datos que son compartidos por los medios inseguros y para efectos de la práctica estos son números pequeños. Las matemáticas dicen que para obtener el inverso de:

$$clave\ p\acute{u}blica = \gamma^{clave\ privada}\ mod\ \rho$$

Se obtiene mediante:

$$clave\ privada = \left(\log_{\gamma} clave\ p\acute{u}blica\right) mod\ \rho$$

Si se piensa de otra manera tanto γ como su $clave\ privada$ deben ser menores a ρ por lo cual si de algún modo se conoce el valor de ρ, la clave privada es un valor comprendido entre 1 < $clave\ privada$ < ρ y puede ser fácilmente obtenida.

En la Actividad 5.1, usted y su compañera/compañero compartieron los datos necesarios para lograr obtener la clave privada uno del otro.

Cada uno de los integrantes del equipo, debe crear un script en el cual se muestre:

- Comprobación de que el número «ρ» utilizado sea primo, recuerde que «γ» es un número aleatorio menor a «ρ», es por eso, que no es necesario comprobarlo, del mismo modo recuerde que la clave secreta cumple con el mismo requisito, es decir menor a «ρ». Tome esto en cuenta para la realización de su programa.
- Clave pública de la víctima

El programa es sencillo, no es necesario que haga uso de banderas, puede colocar los datos directo en el programa o pedir ingresarlos en pantalla, recuerde que debe comentarlo y enviarlo a su docente como se le solicite. Puede realizarlo en Python, en el leguaje que su docente le indique o bien, en el lenguaje de su preferencia

El script debe funcionar algo parecido a lo mostrado en la Figura N° 5.9:

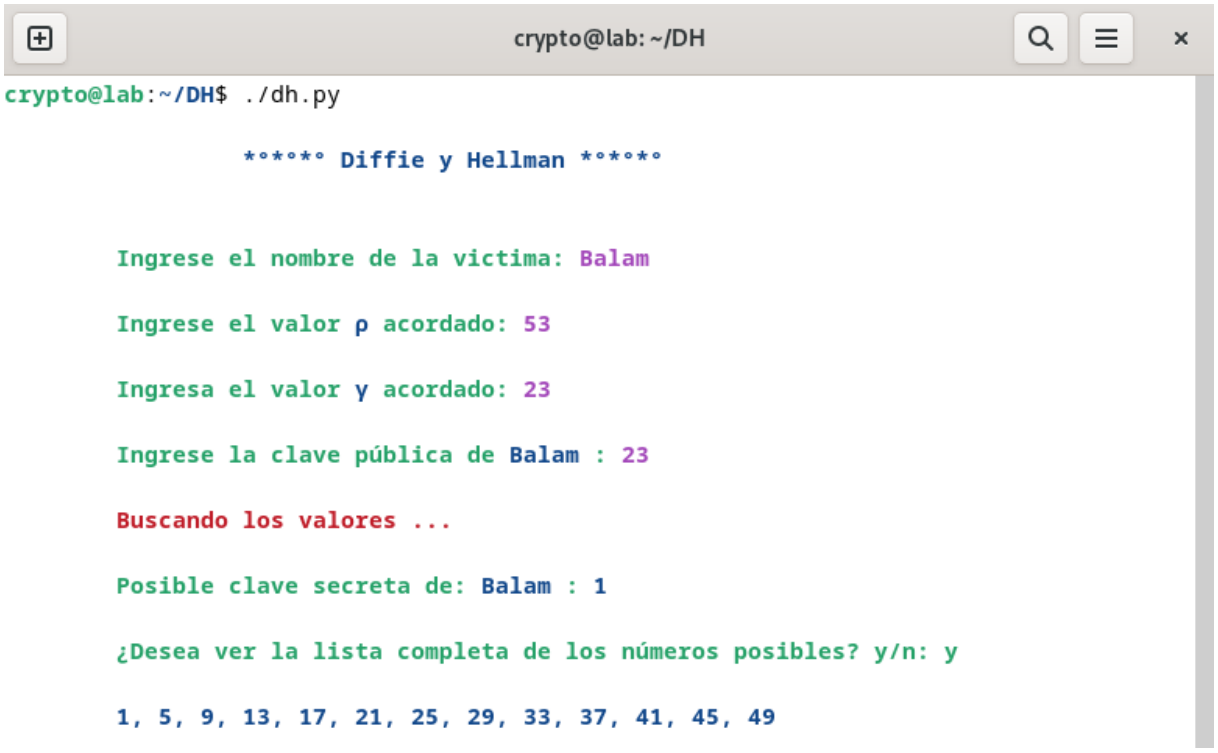


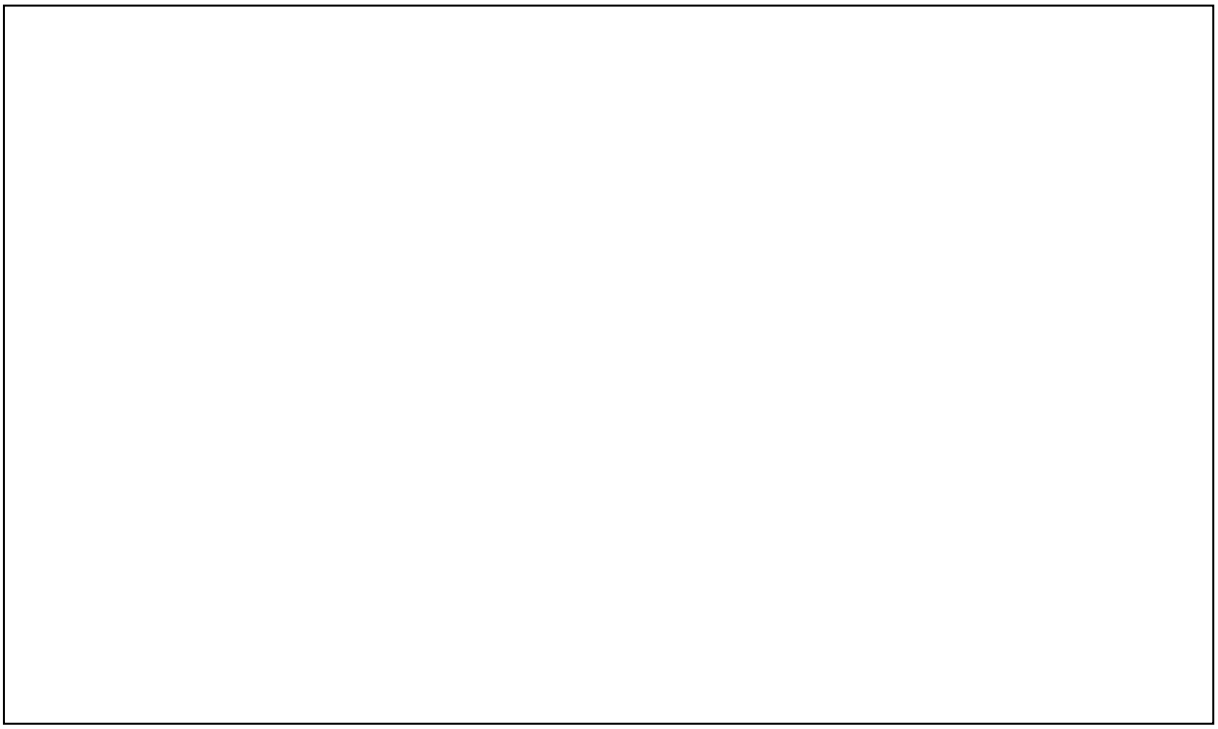
Figura N° 5. 9: Ejemplo de funcionamiento del script DH.

En la Figura N° 5.9 se muestra un ejemplo para los valores mostrados en la Figura N° 5.1, observe que al ser valores tan pequeños los que se escogieron, la clave privada puede ser distintos valores, es por eso que se deben usar números grandes.

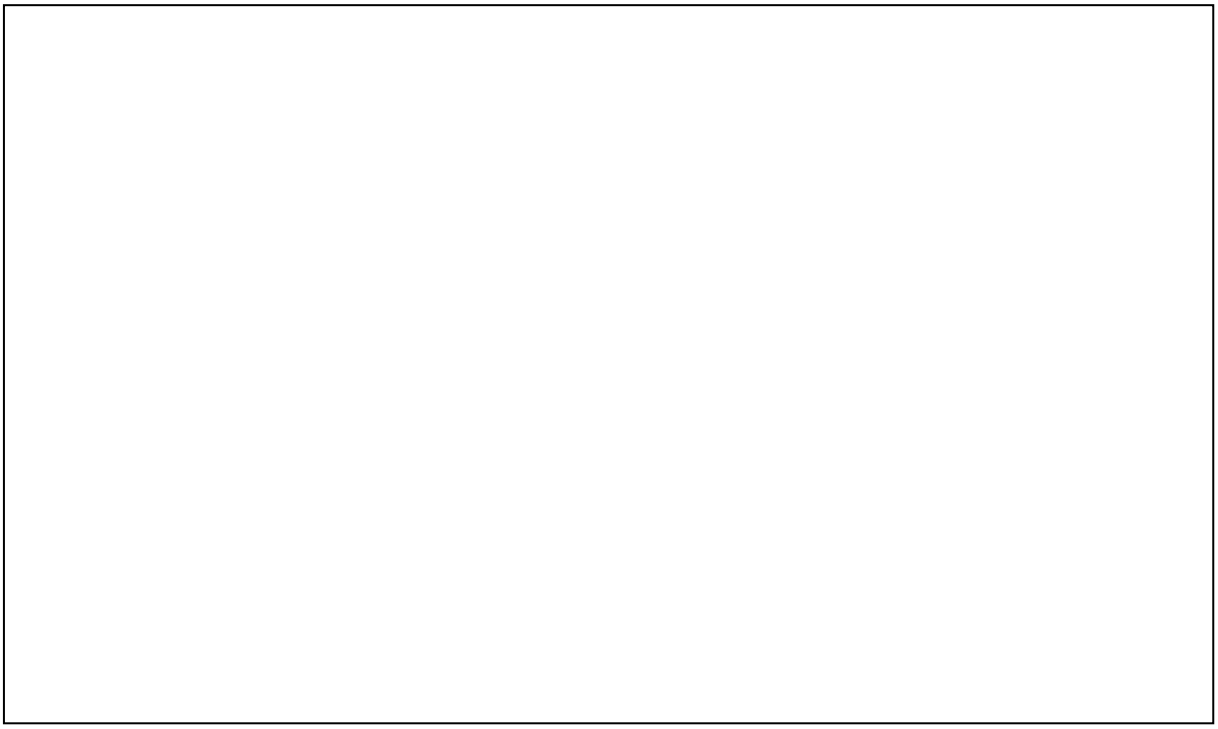
En este caso se obtienen trece datos que cumplen con los criterios, es decir, con esos trece valores se obtiene la misma clave pública y también la misma clave simétrica, he aquí otro punto importante para hacer uso de número grandes, puesto que además de ser sencillo obtener la clave, muchos números cumplen con el criterio.

Utilice su script para obtener la clave privada de su compañera/compañero de la Actividad 5.1, de esta Actividad ya tiene todos los datos, por lo cual solo faltaría encontrar esa clave privada.

En el siguiente recuadro coloque la captura donde corresponda:



Actividad N° 5.3: Obtención de clave privada de 'Atziri'.



Actividad N° 5.3: Obtención de clave privada de 'Balam'.

ACTIVIDAD 5.4: Imagine el siguiente escenario: Atziri y Balam se encuentran en la biblioteca, se ponen de acuerdo para realizar el proyecto de criptografía, a lo cual Atziri le dice a Balam: «*Anota los siguientes datos, estos serán nuestros valores* $\boldsymbol{p}$ y $\boldsymbol{\gamma}$» Atziri dicta a Balam los datos mostrados en la Figura N° 5.10:

$$p = 7333$$

$$\gamma = 4001$$

Figura N° 5. 10: Datos iniciales.

Mientras Balam anotaba los datos, cerca de ellos se encontraba Canek, el cual también anotaba los datos que Atziri dictaba. Teniendo los datos, cada uno por separado escoge su clave privada y realiza la operación necesaria para obtener la clave pública.

Ya que cada uno tuvo su clave pública, se la comparten, los valores que obtuvieron son los mostrados en la Figura N° 5.11:

$$Clave\ pública\ de\ Atziri: 4457$$

$$Clave\ pública\ de\ Balam: 3554$$

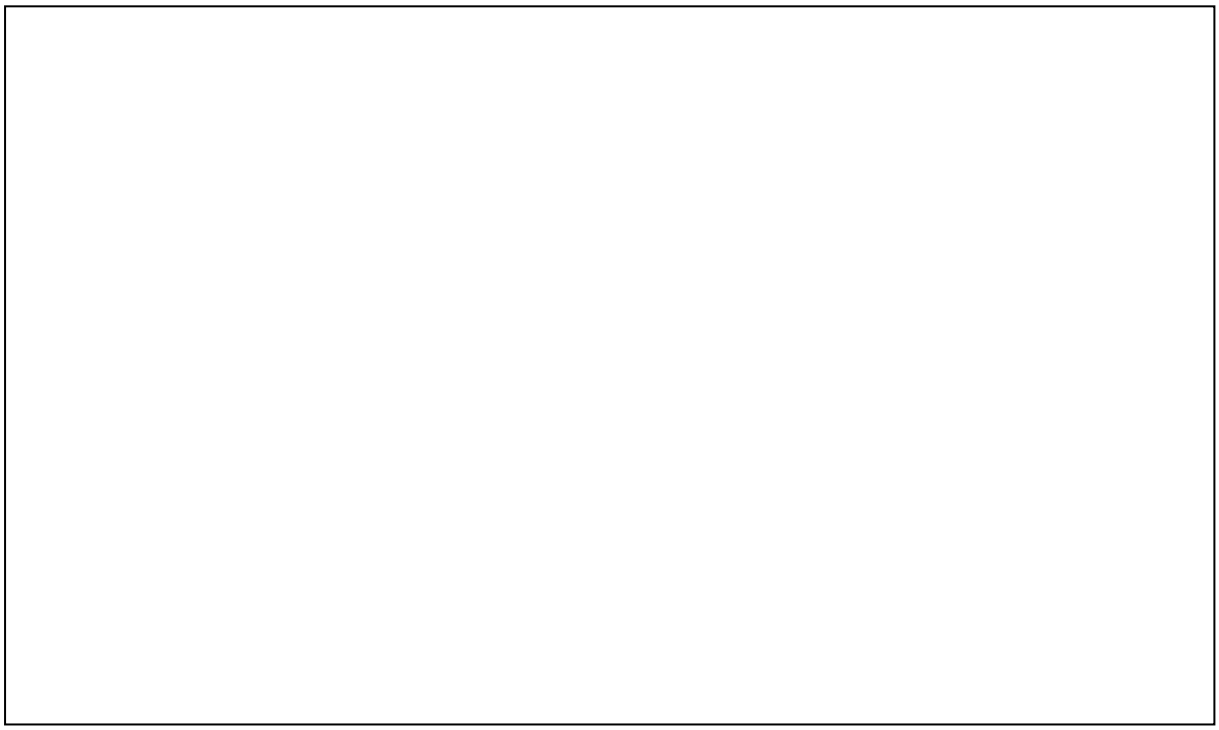
Figura N° 5. 11: Clave pública de Atziri y Balam.

Canek, seguía atento a los datos que se compartían, sin embargo, sabe que tiene los datos necesarios para obtener los datos de las claves privadas de cada uno.

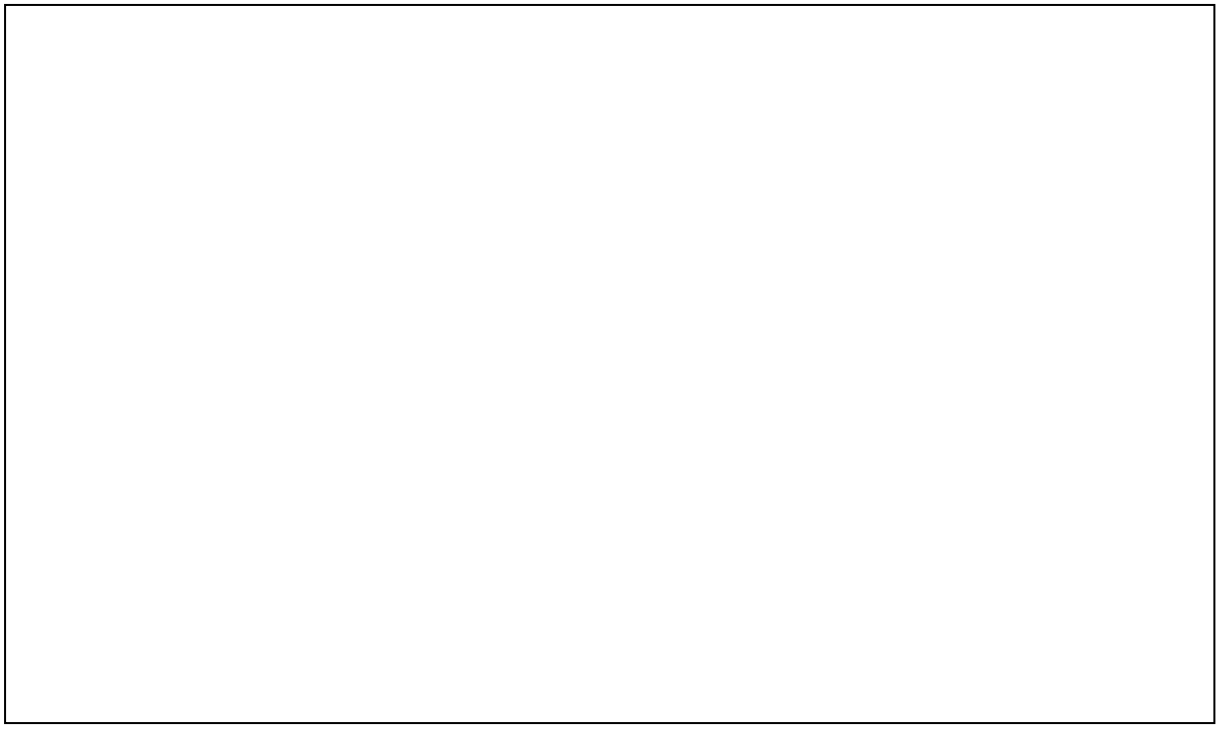
Obtenga la clave privada de cada uno, es decir, la clave pública de Atziri y Balam, lo ideal es que haga uso de su script.

Escriba, ¿Cuál es la clave simétrica? [ ]

Coloque en los siguientes recuadros donde corresponda la obtención de las claves privadas:



Actividad N° 5.4: Obtención de clave privada de 'Atziri'.



Actividad N° 5.4: Obtención de clave privada de 'Balam'.

ECDH - Elliptic Curve Diffie-Hellman (RFC 7748)

Existe una variante del algoritmo Diffie-Hellman que utiliza curvas elípticas.

Para entender mejor este concepto, primero es necesario definir: ¿Qué es una curva elíptica en criptografía?

La criptografía de curva Elíptica (ECC) se definen como «una variante de la criptografía asimétrica o de clave pública basada en las matemáticas de las curvas elípticas. Sus autores argumentan que la CCE puede ser más rápida y usar claves más cortas que otros métodos, como RSA, al tiempo que proporcionan un nivel de seguridad equivalente. La utilización de curvas elípticas en criptografía fue propuesta de forma independiente por Neal Koblitz y Victor Miller en 1985» (Wikipedia, s.f.).

Con lo anterior, el protocolo de intercambio de claves ECDH, es una variante del protocolo Diffie-Hellman, en donde aprovecha las propiedades matemáticas que tienen las curvas elípticas para proporcionar un método de intercambio de claves más eficiente y seguro (EITCA,s.f.).

Puede consultar la documentación en RFC 7748: Elliptic Curves for Security

Conclusiones:

1. ¿Con qué objetivo fue que se desarrolló el algoritmo Diffie – Hellman?

2. ¿Por qué es importante hacer uso de números primos grandes en la creación de las claves?

3. ¿Cómo realizó la obtención de la clave privada en las actividades 5.3 y 5.4? Especifique lo que realiza su script

4. En la Actividad 5.4 está simulando un ataque conocido como *man in the middle*, ¿De qué se trata dicho ataque?

5. Comentarios o conclusiones adicionales.

UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

ACTIVIDADES PRÁCTICAS: CRIPTOGRAFÍA

6. RSA

DOCENTE

RSA

Objetivo

Conocer el algoritmo de cifrado asimétrico RSA, su estructura, funcionamiento y algunas herramientas que trabajan con el algoritmo.

Justificación

Como se dio a conocer en la práctica anterior, los algoritmos de cifrado asimétricos, usan una clave para cifrar y otra para descifrar. En esta práctica se da la introducción al algoritmo asimétrico RSA, con la finalidad de entender sus procesos de cifrado y descifrado.

Introducción

El algoritmo RSA (RFC 8017) es un algoritmo que se dio a conocer en el año 1977 publicado por Ron **R**iverst, Adi **S**hamir y Leonard **A**dleman del Instituto Tecnológico de Massachusetts. El algoritmo lleva dicho nombre ya que son cada una de las iniciales de los apellidos de los creadores.

La seguridad de este algoritmo se basa en la dificultad computacional de factorizar números muy grandes, generados a partir de la multiplicación de dos números primos grandes (mínimo 512 bits), ya que dicha multiplicación genera 1024 bits, si bien es un número elevado de bits, hoy día se recomienda usar claves de al menos 2048 bits (López Barrientos, M.J., 2016).

Para entender cómo funciona el algoritmo se seguirán cada uno de los pasos que lo componen (SCIC-UNS, s.f.) y para ello el primer paso es que los usuarios Atziri y Balam seleccionen cada uno dos números primos llamados p y q, y a partir de ellos cada uno generará sus claves pública y privada.

1. Con los valores p y q se realiza la multiplicación: $n = p * q$
2. Ahora se debe calcular la función de Euler $\varphi(n) = (p-1)(q-1)$
3. Se debe seleccionar un número e aleatorio que cumpla con lo siguiente: $1 < e < \varphi(n)$ y que además el mínimo común divisor de e y $\varphi(n)$ sea igual a 1, es decir: $mcd(e, \varphi(n)) = 1$. Dicho número e forma parte de la clave pública.
4. Se obtiene el valor d, el cual es el inverso multiplicativo de e y representa la clave privada. Dicho número debe cumplir con $1 < d < \varphi(n)$ y $ed \equiv 1 \; mod \; \varphi(n)$. Se obtiene usando el algoritmo extendido de Euclides.
5. La clave pública es: (e, n)
6. La clave privada es: (d)
7. Para realizar el cifrado el mensaje debe cumplir con: $1 < mensaje < n$.
   Consideré que Balam enviará un mensaje cifrado a Atziri, el mensaje cifrado se obtiene con:

$$cifrado = mensaje^{e_{Atziri}} \; mod \; n_{Atziri}$$

En donde como se observa Balam utiliza los valores de la clave pública de Atziri para realizar el cifrado lo que significa que Atziri es la única que podrá descifrar el mensaje haciendo uso de su clave privada.

8. El descifrado se realiza haciendo uso del valor $\boldsymbol{d}$. Es decir, si Atziri quiere descifrar el mensaje que Balam le envió, realizará:

$$descifrado = mensaje^{d_{Atziri}} mod\ n_{Atziri}$$

En la Figura N° 6.1 se observa el procedimiento antes descrito:

ATZIRI

OBTENCIÓN DE CLAVES

Atziri selecciona los valores $p = 103$ y $q = 107$

$n = p * q = 103 * 107 = 11021$
$\varphi(n) = (p-1)(q-1) = (103-1)(107-1)$
$\varphi(n) = 10812$

Atziri selecciona el valor $e = 31$, con este valor obtiene d

$$31d \equiv 1\ mod\ 10812$$
$$d = 3139$$

Atziri guarda su número d y los números primos p y q, estos no los comparte con nadie.

DESCIFRADO

Atziri recibe el valor **3396** y aplica el descifrado:

$$descifrado = 3396^{3139}\ mod\ 11021$$
$$descifrado = \mathbf{9798}$$

Atziri descifra satisfactoriamente el mensaje enviado por Balam.

BALAM

OBTENCIÓN DE CLAVES

Balam selecciona los valores $p = 109$ y $q = 113$

$n = p * q = 109 * 113 = 12317$
$\varphi(n) = (p-1)(q-1) = (1039-1)(113-1)$
$\varphi(n) = 12096$

Balam selecciona el valor $e = 17$, con este valor obtiene d

$$17d \equiv 1\ mod\ 12096$$
$$d = 10673$$

Del mismo modo, Balam guarda su número d y los números primos p y q, estos no los comparte con nadie.

CIFRADO

Balam recibe de Atziri su número $\boldsymbol{n}$ y su número $\boldsymbol{e}$. Balam va a cifrar **9798** que cumple con ser menor a $\boldsymbol{n}$.

$$cifrado = 9798^{31}\ mod\ 11021$$
$$cifrado = \mathbf{3396}$$

El cifrado es **3396** , ese valor es enviado a Atziri Para que ella lo pueda descifrar.

Figura N° 6. 1: Explicación del algoritmo RSA.

La documentación del algoritmo la puede encontrar en RFC 8017: PKCS #1: RSA Cryptography Specifications Version 2.2

Equipo y material necesario

Máquina Virtual Debian, para montarla en su equipo puede seguir el «Montaje de máquina virtual», el cual es un manual de Instalación MV, en el que se encuentran las instrucciones tanto para VMware como para VirtualBox. Así mismo puede consultar el Anexo D que contiene ejemplos de cómo debe colocar sus respuestas en la práctica.

Desarrollo

Recuerde que debe hacer uso del usuario indicado en la Actividad 4.1 de la práctica N° 4 «Advanced Encryption Standard».

ACTIVIDAD 6.1: Tomando como base la explicación de la Figura N° 6.1, cree un script en el lenguaje que indique su docente o bien, el de su preferencia. El script debe realizar la generación de claves, el cifrado y descifrado utilizando el algoritmo RSA. El número primo menor para utilizar debe ser mayor a **101**, considérelo para su programa. El programa debe ser un menú con las siguientes tres opciones:

a) **Generador de claves:** Dados los valores $\boldsymbol{p}$ y $\boldsymbol{q}$ puede obtener la clave pública y privada. En este apartado debe dar la opción para que el usuario ingrese su número $\boldsymbol{e}$ o el programa pueda proporcionárselo, debe verificar que todos los números ingresados cumplan con los requisitos, es decir que sea número primo y que cumpla con la condición de **MCD**.

b) **Cifrado:** Se pedirá el mensaje a cifrar, los valores públicos $\boldsymbol{n}$ y $\boldsymbol{e}$. En este apartado su programa debe comprobar que el valor del mensaje codificado sea menor a $\boldsymbol{n}$. Se hará uso del siguiente alfabeto para la creación de los mensajes, así como para cifrar y descifrar: «ABCDEFGHIJKLMNÑOPQRSTUVWXYZ0123456789» donde las posiciones de cada elemento van del 1 al 37, como se muestra en la Tabla N° 6.1:

Tabla N° 6. 1:Posición de los elementos del alfabeto.

Letra	A	B	C	D	E	F	G	H	I	J	K	L	M	…	9
Posición	1	2	3	4	5	6	7	8	9	10	11	12	13	…	37

Para realizar la explicación del proceso de cifrado considere que Balam cifrará un mensaje, para Atziri. Para que obtenga el valor del mensaje codificado, se realizará lo siguiente:

1. Obtener la posición de la letra, para obtener su equivalente en número, para lo cual, si el mensaje a cifrar es «CD» de acuerdo con el alfabeto, las posiciones son «$3, 4$».

2. Teniendo los valores correctamente, se realiza la siguiente operación:

$$Mensaje_{3,4} = (3 * 37^0) + (4 * 37^1)$$

En donde el número $\mathbf{37}$ corresponde a la longitud del alfabeto a utilizar, por lo cual este número es fijo para todos, y las potencias se realizan de acuerdo con la longitud del mensaje a enviar comenzado en **cero**, en este caso el mensaje consta de dos caracteres, es por eso que hay dos potencias, cero y uno. Si el mensaje tuviera cuatro caracteres las potencias serían: cero, uno, dos y tres. Recuerde, siempre se comienza desde cero y la potencia siempre aplica al número $\mathbf{37}$. Para el ejemplo, el resultado que representa el valor del mensaje codificado es: $Mensaje_{3,4} = \mathbf{151}$.

3. Se comprueba que el mensaje sea menor que el valor de $\boldsymbol{n}$, si esto no es así, su programa debe terminar. Recuerde que se debe usar la siguiente expresión para obtener el cifrado en RSA:

$$cifrado = mensaje^{e_{Atziri}} mod\ n_{Atziri}$$

En este caso, el valor de mensaje es $\mathbf{151}$, el valor de la clave pública ($\boldsymbol{e}$) de Atziri es $\mathbf{31}$ y el valor de $\boldsymbol{n}$ para Atziri es $\mathbf{11021}$. Realizando la operación, el resultado del cifrado es:

$$cifrado = \mathbf{7387}$$

4. Ahora, para obtener las equivalencias en el alfabeto, realizará divisiones, de la siguiente manera:

$$\frac{7387}{37} = \mathbf{199}, \qquad residuo = \mathbf{24}$$

$$\frac{199}{37} = \mathbf{5}, \qquad residuo = \mathbf{14}$$

Las divisiones se realizarán hasta que el resultado sea menor a $\mathbf{37}$, es decir, en este caso se realiza una segunda división ya que el primer resultado fue $\mathbf{199}$, si en la segunda división el resultado hubiese sido $\mathbf{55}$ en lugar de $\mathbf{5}$, se realizaría una división más y así sucesivamente.

Ahora, deberá tomar los números en el siguiente orden: primero, el último resultado, es decir el número $\mathbf{5}$, después el último residuo, es decir el número $\mathbf{14}$ y de las demás divisiones solo se toman los residuos, en este caso solo hay una división más cuyo residuo es $\mathbf{24}$. Por lo cual, los números quedan: «$\mathbf{5}, \mathbf{14}, \mathbf{24}$», cuya equivalencia por posición de acuerdo con el alfabeto establecido es: «$\boldsymbol{ENW}$», sin embargo, es conveniente dejar el resultado en números, para el siguiente paso.

5. Adicionalmente se hará uso de Cifrado César. Dicho cifrado pertenece a los cifrados simétricos y su procedimiento es sencillo. Consiste en realizar un corrimiento a cada uno de los caracteres fijamente $\mathbf{3}$ posiciones, pero bien ese puede ser $\boldsymbol{n}$ posiciones a la derecha, es decir, si tenemos el carácter «$\boldsymbol{C}$», cuya posición es «$\mathbf{3}$» y se aplica el cifrado césar usando $\boldsymbol{n} = \mathbf{3}$, el carácter «$\boldsymbol{C}$» cifrado será «$\boldsymbol{F}$» cuya posición es «$\mathbf{6}$». Para la realización del programa, el valor de $\boldsymbol{n}$ será la longitud original de la cadena a cifrar. En este caso la cadena original es «$\boldsymbol{CD}$», cuya longitud es $\mathbf{2}$, por lo cual, a los valores obtenidos anteriormente «$\mathbf{5}, \mathbf{14}, \mathbf{24}$» → «$\boldsymbol{ENW}$» se les aplicará el cifrado César, haciendo el corrimiento dos posiciones, es decir, los valores finales serán «$\mathbf{7}, \mathbf{16}, \mathbf{26}$» → «$\boldsymbol{GOY}$».
6. Al final se va a concatenar el valor de la longitud de la cadena original, pero tomando su equivalente numérico en el alfabeto dado, es decir, en este caso, la longitud es $\mathbf{2}$, la posición $\mathbf{2}$ en el alfabeto la ocupa la letra «$\boldsymbol{B}$» , por lo cual, el cifrado final quedará «$\boldsymbol{GOYB}$».

Se debe tener en cuenta que si, por ejemplo, se quiere cifrar el número «$\mathbf{8}$» cuya posición es «$\mathbf{36}$» y se quiere hacer el corrimiento haciendo uso de $\boldsymbol{n} = \mathbf{3}$, el corrimiento debe ser circular, es decir, si llega al tope debe regresar al inicio, para cumplir con el valor establecido en $\boldsymbol{n}$, en el ejemplo, al recorrer $\mathbf{3}$ posiciones, estas serían «$\mathbf{37}, \mathbf{1}, \mathbf{2}$» siendo que «$\mathbf{8}$» cifrado es «$\boldsymbol{B}$».

Su programa debe ser capaz de aceptar los valores en minúscula y para evitar errores, pasarlos a mayúsculas.

c) **Descifrado:** El descifrado va a recibir la cadena que debe estar compuesta por elementos del alfabeto establecido, el valor de la clave privada, $\boldsymbol{d}$ y $\boldsymbol{n}$ para poder realizar la operación.

Para realizar el descifrado se continúa con el ejemplo, Atziri va a descifrar el mensaje que Balam le envió, el mensaje es: «$\boldsymbol{GOYB}$».

1. Como se indicó anteriormente, el último carácter es la longitud del mensaje original, por lo cual, se aísla este valor del resto, quedando los valores de la siguiente manera: «$\boldsymbol{GOY}$», «$\boldsymbol{B}$», de estos se obtiene sus posiciones equivalentes, por lo cual los valores quedan: «$\boldsymbol{GOY}$» → «$\mathbf{7}, \mathbf{16}, \mathbf{26}$», $\boldsymbol{y}$ «$\boldsymbol{B}$» → «$\mathbf{2}$».

2. Ahora, teniendo las equivalencias numéricas, se realiza el descifrado César, para esto, se realiza el corrimiento a la izquierda $\boldsymbol{n}$ veces, que como se indicó anteriormente $\boldsymbol{n}$ tanto para el cifrado y descifrado césar, será la longitud del mensaje, es por eso que en el paso anterior se separó dicho valor. Entonces, al aplicar el descifrado césar a «$\boldsymbol{GOY}$» $\rightarrow$ «$\mathbf{7, 16, 26}$» con $\boldsymbol{n} = \mathbf{2}$, los nuevos valores serán «$\mathbf{5, 14, 24}$».

   Del mismo modo debe tener en cuenta que si el elemento a descifrar es «$\boldsymbol{B}$» con $\boldsymbol{n} = \mathbf{3}$, debe realizar el corrimiento a la izquierda de forma circular, por lo cual, al recorrer las $\mathbf{3}$ posiciones, estas serían «$\mathbf{1, 37, 36}$» siendo que «$\boldsymbol{B}$» descifrado es «$\mathbf{8}$».

   Ahora, con los valores obtenidos, después del descifrado César «$\mathbf{5, 14, 24}$», se realiza la siguiente operación:

$$cifrado = (5 * 37) + 14 = 199$$

$$cifrado = (199 * 37) + 24 = 7387$$

   El valor $\mathbf{37}$ es fijo, recuerde que es el valor de la longitud del alfabeto. Si hubiese un valor más, se toma el resultado de la operación anterior, se vuelve a multiplicar por $\mathbf{37}$ y se suma el valor faltante, es decir si los valores fueran «$\mathbf{5, 14, 24, 7}$» quedaría de la siguiente manera:

$$cifrado = (7387 * 37) + 7 = 273326$$

   Si hubiese obtenido más valores en el descifrado César, debe volver a realizar la operación anterior y así hasta no tener más valores.

   Sin embargo, para este caso no fue necesario realizar más que las dos primeras operaciones.

3. Con este nuevo valor, se realiza el descifrado con RSA recordando que se realiza con la siguiente operación:

$$descifrado = mensaje^{d_{Atziri}} mod\ n_{Atziri}$$

   En donde los valores son: para *mensajes* es $\mathbf{7387}$, que es el valor obtenido en el paso anterior, *clave privada (d)* de Atziri es $\mathbf{3139}$ y el valor de $\boldsymbol{n}$ para Atziri es $\mathbf{11021}$
   Realizando la operación, el resultado es: $descifrado = \mathbf{151}$.

4. Ahora, con el nuevo valor obtenido se deben obtener las equivalencias en el alfabeto, se realizarán divisiones de la siguiente manera:

$$\frac{\mathbf{151}}{37} = \mathbf{4}, \qquad residuo = \mathbf{3}$$

Nuevamente el valor **37** es la longitud del alfabeto, por lo cual este será fijo. En esta ocasión solo bastó una división para obtener los valores, sin embargo, si el resultado hubiese sido **53** en lugar de **4**, se debería realizar una división más y así sucesivamente hasta que el resultado sea menor a **37**.

5. Teniendo los valores: «**4, 3**», recuerde que son las posiciones, y obteniendo su equivalente en el alfabeto, se obtiene el mensaje «***DC***», que como se observa es el mensaje en orden inverso, si está trabajando en Python, solo aplique la función *inverse*, que se aplica a listas, para lograr el acomodo y así el mensaje original que es «***CD***»

Ejemplificando lo anterior, su programa debe ser capaz de realizar lo mostrado en las Figuras N° 6.2, 6.3 y 6.4:

a) **Generador de claves:** Atziri obtiene sus claves, solo comparte las públicas.

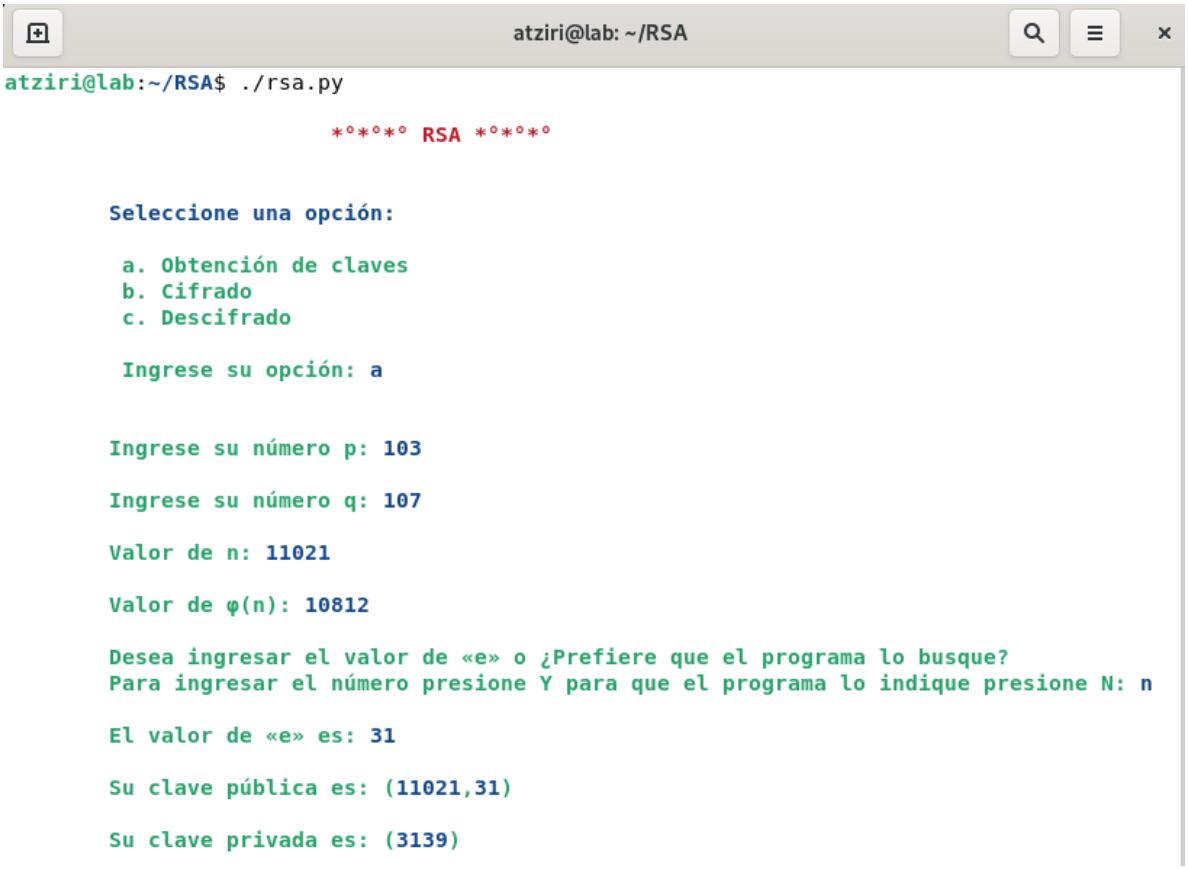


Figura N° 6. 2: Creación de claves RSA.

b) **Cifrado:** Balam cifra el mensaje «CD» para Atziri y comparte los datos con ella, para que pueda realizar el descifrado.

Figura N° 6. 3: Cifrado de mensaje.

c) **Descifrado:** Atziri descifra el mensaje que Balam envió.

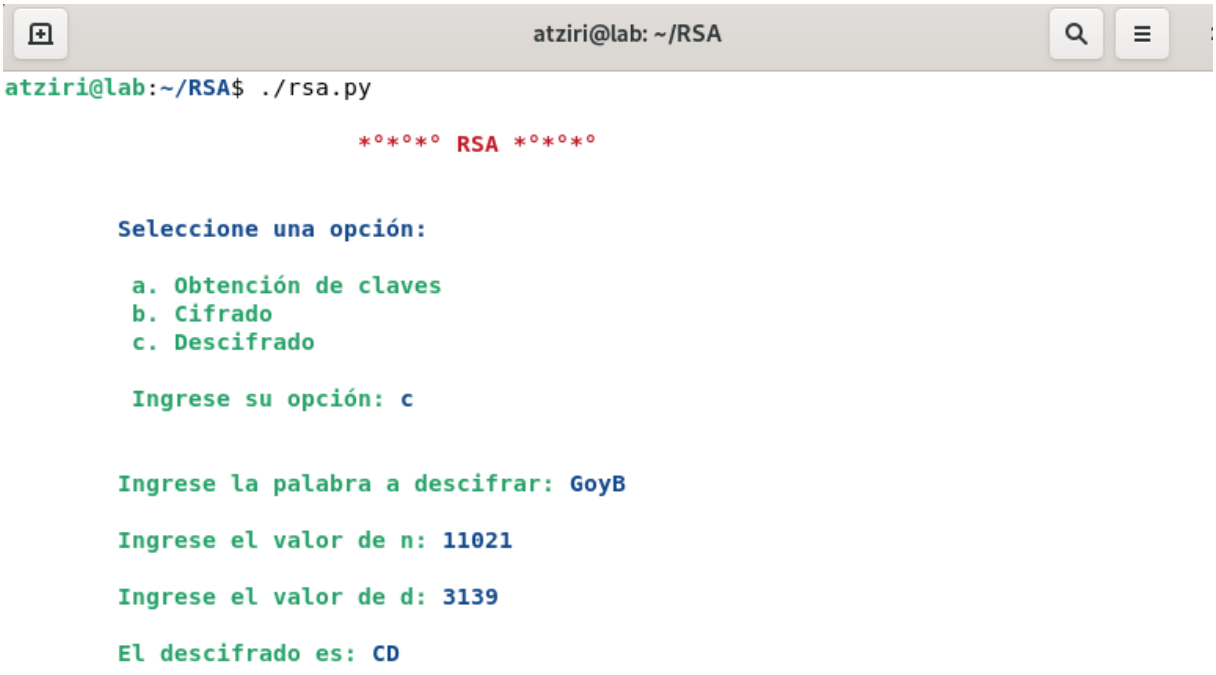


Figura N° 6. 4: Descifrado de mensaje.

Ejemplos de errores a considerar (Véase Figura N° 6.5)**:**

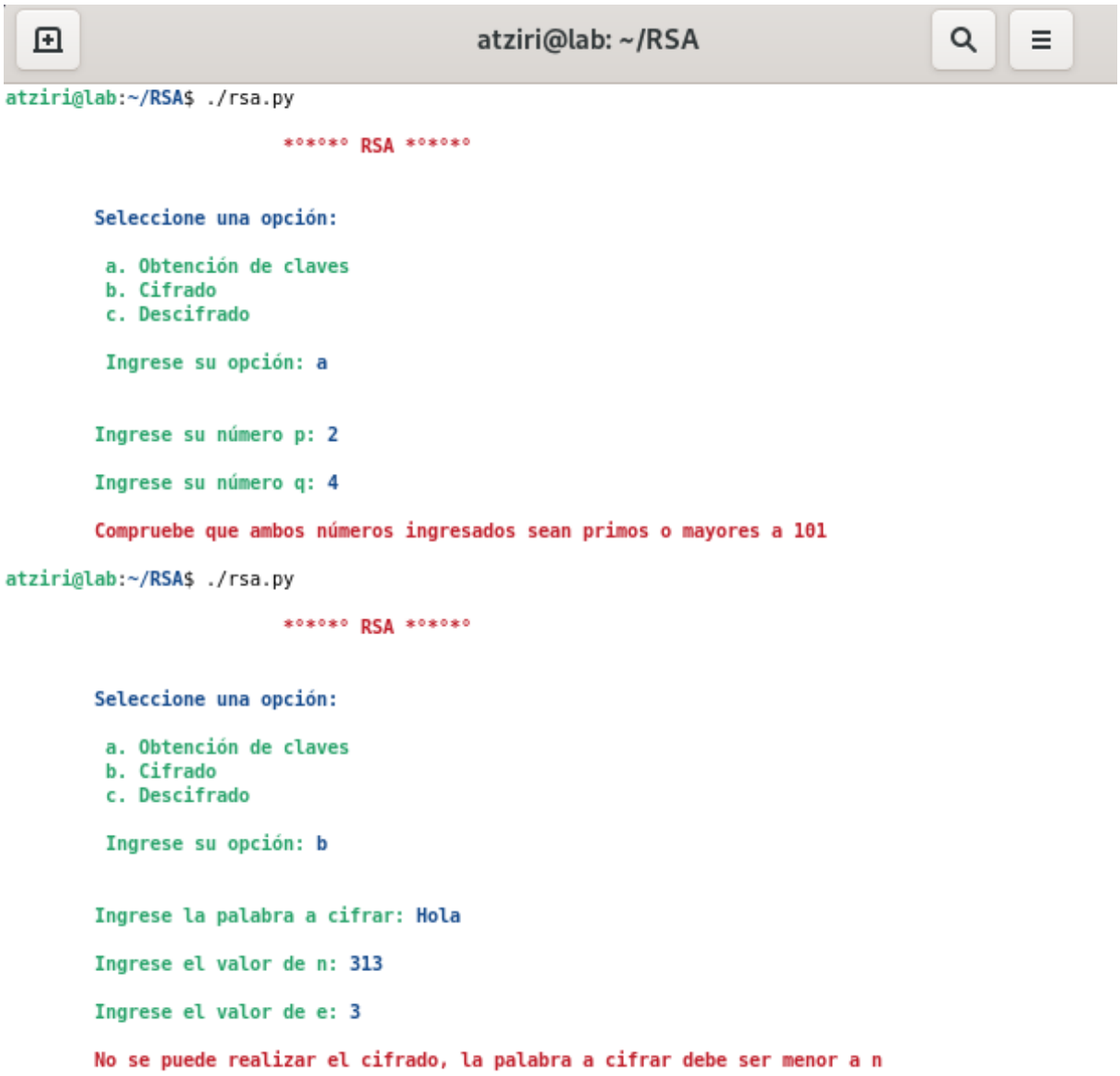


Figura N° 6. 5: Ejemplo de errores.

La siguiente actividad se realiza en parejas. Recuerden que su código debe estar debidamente comentado.

Tomen en cuenta todos los posibles errores a obtener en su programa, en caso de que el usuario ingrese valores erróneos, el programa debe identificarlos, mostrar un mensaje en la pantalla y salir.

En los siguientes recuadros cada uno debe colocar una captura donde se aprecie la obtención de sus claves, Balam cifrando un mensaje y Atziri descifrándolo. *Recuerden hacer uso del usuario indicado en la Actividad 4.1 de la práctica N° 4 «Advanced Encryption Standard».* Si hace uso de Python, se

recomienda hacer uso de listas, puede buscar información de cómo usar: upper, reverse. join, pueden ser de gran ayuda para la realización de su programa.

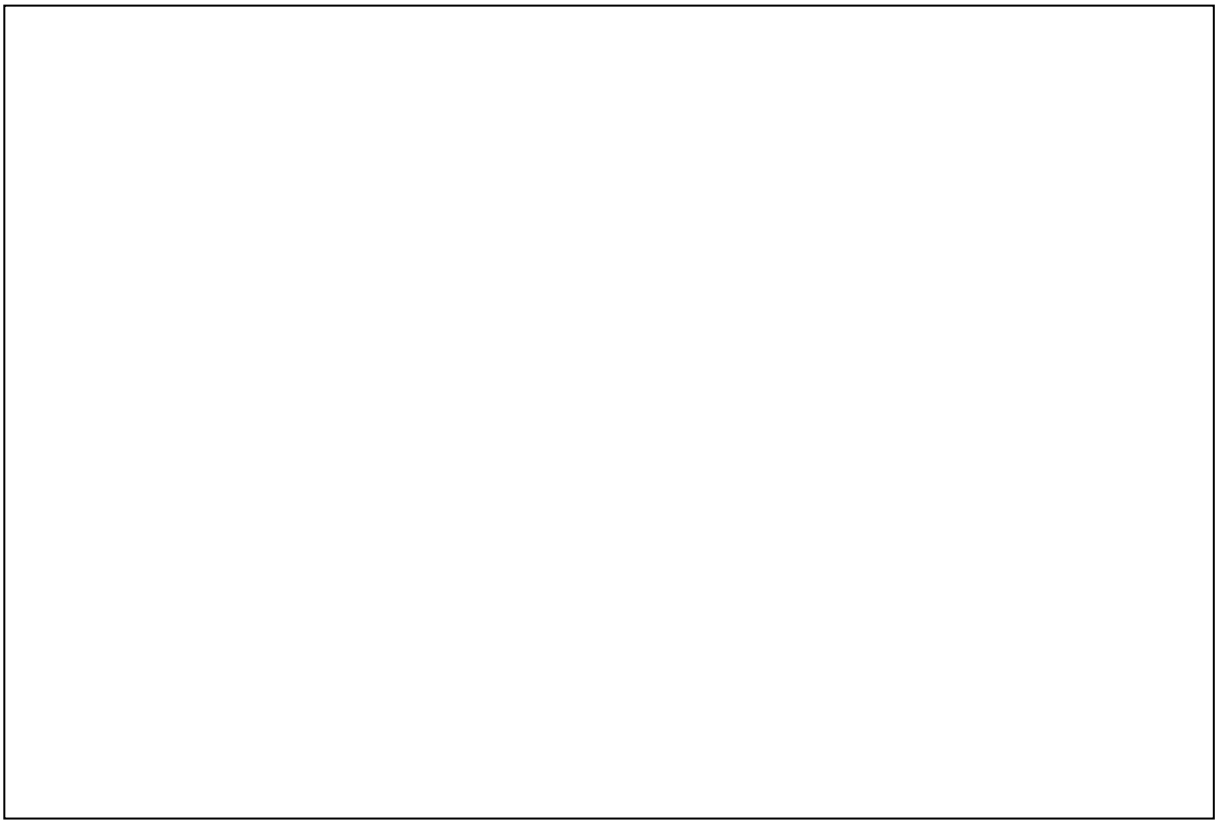
El script se entrega según lo indicado por su docente.

Indique qué integrante será Atziri y quién será Balam:

Atziri:

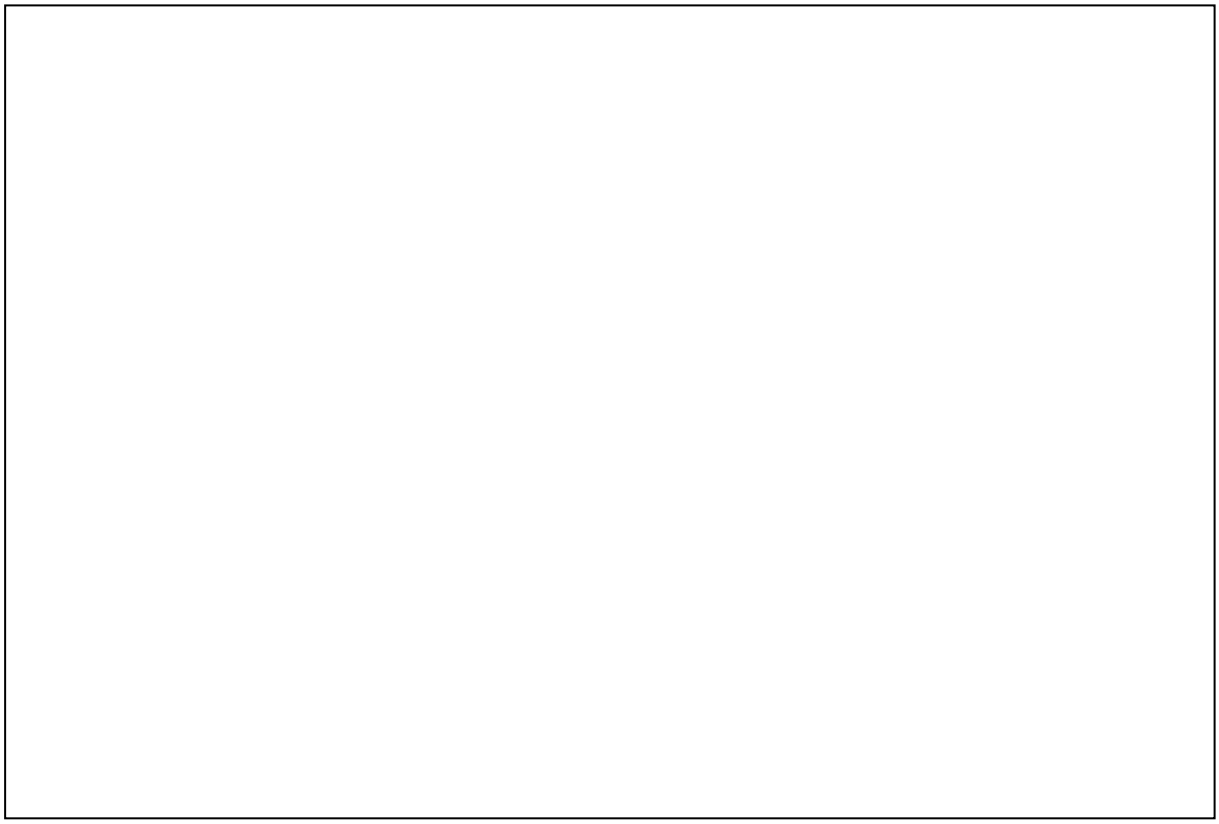
Balam:

Obtención de claves para Atziri.



Actividad 6.1: Creación de claves por parte de 'Atziri'.

Obtención de claves para Balam.

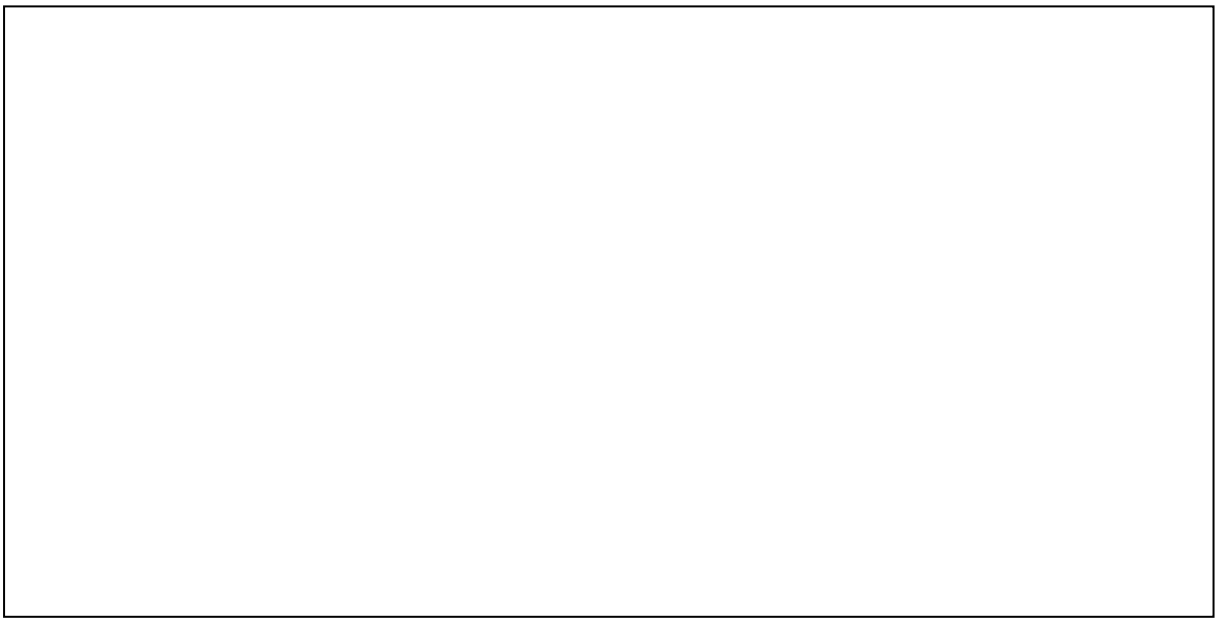


Actividad 6.1: Creación de claves por parte de 'Balam'.

Cifrado de mensaje por parte de Balam

Actividad 6.1: Cifrado de mensaje enviado por parte de 'Balam' a 'Atziri'.

Descifrado de mensaje enviado a Atziri por parte de Balam



Actividad 6.1: Descifrado mensaje enviado por parte de 'Balam' a 'Atziri'.

ACTIVIDAD 6.2: OpenSSL es una herramienta que no solo ayuda con la obtención de funciones hash, o cifrado simétrico, también puede usarse con cifrado asimétrico. En esta actividad, cada uno va a realizar un par de claves de 2048 bits, por lo cual esta actividad se va a trabajar en parejas.

Para crear la clave, ejecute: «*openssl genrsa -out nombre_rsa.pem 2048*» donde *nombre_rsa.pem* es el nombre con que guardar la clave, y *2048* es la longitud de la clave, en este caso 2048 bits, la creación de la clave se muestra en la Figura N° 6.6, no siempre se muestra la creación de clave, bien solo puede aparecer el prompt nuevamente cuando esta se haya creado. Recuerde cambiar *atziri* por su nombre:

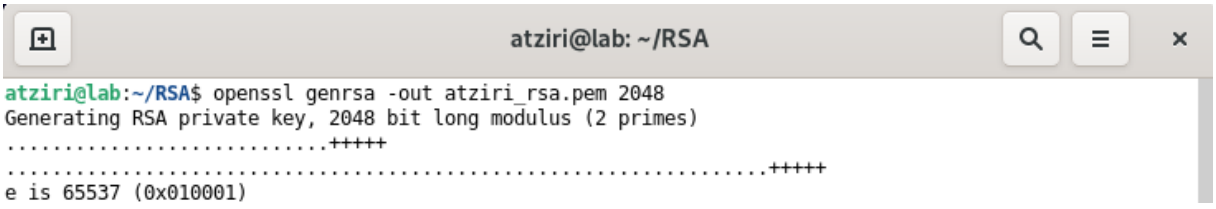


Figura N° 6. 6: Creación de clave RSA.

De esta clave, se obtiene la clave pública. Para eso, ejecute: «*openssl rsa -in nombre_rsa.pem -pubout -out nombre_publica.pem*» donde primero se indica la clave de donde se obtendrá la clave pública, para el caso de los ejemplos es *atziri_rsa.pem*, posteriormente se indica que se obtendrá la clave pública y esta será guardada en un archivo, para el ejemplo *atziri.publica.pem*. La ejecución del comando muestra en la Figura N° 6.7:

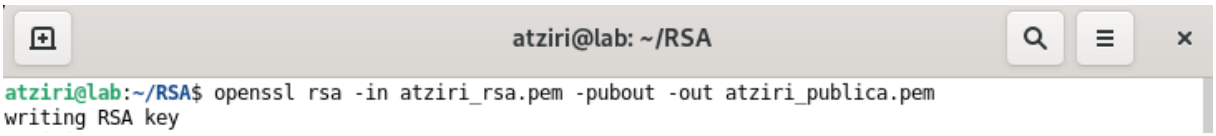


Figura N° 6. 7: Obtención de clave pública.

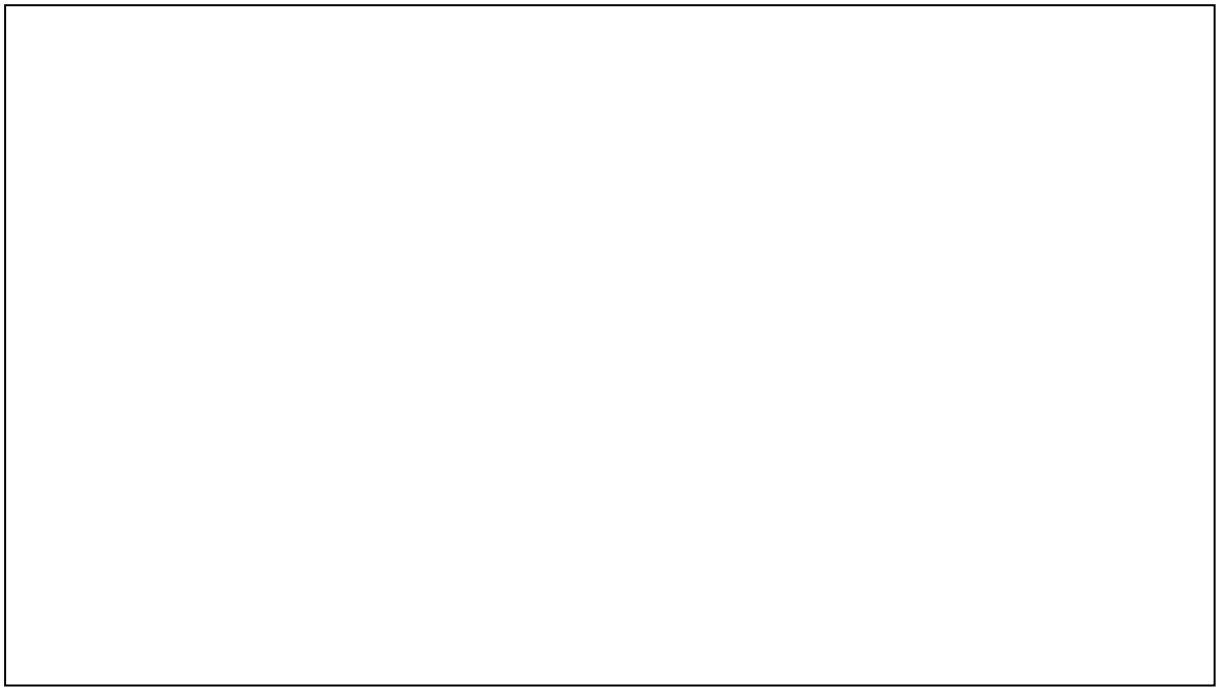
Puede revisar los datos de su clave pública con el comando: «*openssl rsa -in nombre_publica.pem -pubin -text*» tal como se muestra en la Figura N° 6.8:

Figura N° 6. 8: Revisión de datos de clave pública.

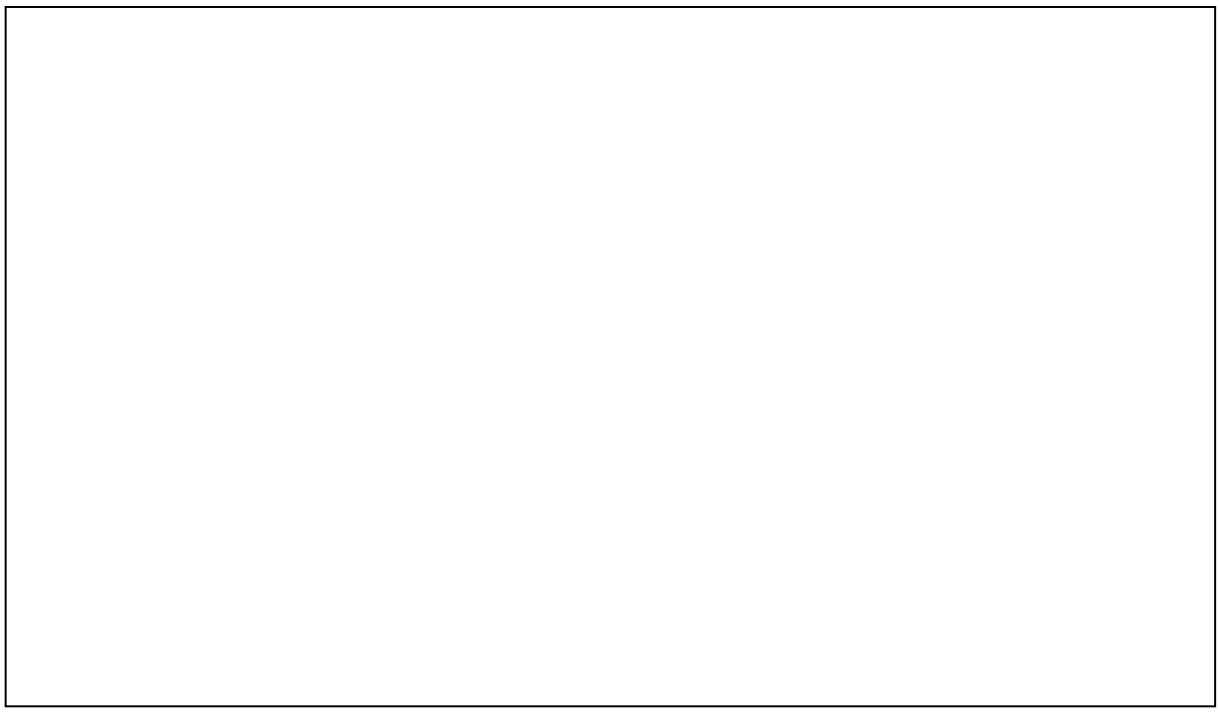
Como se observa en la Figura N° 6.8, hay un número bastante elevado en hexadecimal, este número es el módulo, el resultado de multiplicar los números primos p y q, tal como se vio en la Figura N° 6.1.

Los valores de estos números los pueden revisar si así lo desean, estos se encuentran en la clave privada, para revisar los valores ejecute el siguiente comando: «*openssl rsa -in clave_publica -text -modulus*»

En los siguientes recuadros, coloque una captura en donde corresponda, de los datos de su **clave pública**.



Actividad 6.2: Datos de clave pública creada por 'Atziri'.



Actividad 6.2: Datos de clave pública creada por 'Balam'.

Envíe a su compañera/compañero su clave pública, puede realizarlo por netcat como se vio antes o bien por el medio de su preferencia. Guarde la clave pública de su compañera/compañero en su directorio RSA.

ACTIVIDAD 6.3: La siguiente actividad se realizará en parejas, (para los ejemplos Atziri y Balam) los cuales cada uno tendrá que buscar un archivo, dicho archivo tendrán que cifrarlo ambos usando OpenSSL.

Para buscar el archivo a cifrar, cada uno tendrá que contestar una pregunta, la respuesta a dicha pregunta es el directorio en donde se encuentra el archivo.

Es decir, si la pregunta es: *¿Directorio en donde se alojan los binarios importantes del sistema?* La respuesta es «/*sbin*», por lo cual se tiene que dirigir a dicho directorio.

Dentro de ese directorio encontrará el archivo, en donde el nombre dirigido a los usuarios, es decir el archivo que sea para Atziri se llama **RSA_Atziri#** y el dirigido a Balam se llama **RSA_Balam#**, donde # es un número, es decir, en el directorio encontrará distintos archivos en el cual lo único que cambiará será el último digito.

¿Cómo sabrá cuál es el archivo correcto? Investigue de qué manera puede buscar archivos que tengan determinado tamaño usando el comando *find*. El archivo correcto es cuyo tamaño sea exactamente el número de bytes que se indiquen, por ejemplo **80 bytes**.

RECUERDE que debe realizar las actividades haciendo uso del usuario creado en la práctica N° 4.

Para Atziri:

Nombre de quien realizó esta parte:

Conteste: ¿Directorio que sirve como punto de montaje automático para dispositivos extraíbles? El archivo correcto tiene **94 bytes.**

¿Cuál es el nombre correcto del archivo?

Para Balam:

Nombre de quien realizó esta parte:

Conteste: ¿Directorio que sirve el punto de montaje temporal para sistemas de archivos? El archivo correcto tiene **87 bytes.**

¿Cuál es el nombre correcto del archivo?

Para ambos cursantes:

Escriban, ¿Cuál es el comando completo que utilizaron para buscar el archivo correcto?

Copien el archivo a su directorio RSA.

Cifren el archivo que acaban de encontrar con la clave pública de su compañera/compañero, para cifrar el archivo ejecute: «*openssl pkeyutl -encrypt -pubin -inkey nombre_publica.pem -in RSA_nombre# -out from_nombre_cipher*». En la Figura N° 6.9 se muestra el cifrado del mensaje por parte de Atziri, mensaje que enviará a Balam.

Figura N° 6. 9: Cifrado de archivo encontrado por 'Atziri' y cifrado con clave pública de 'Balam'.

En los siguientes recuadros, coloque una captura de pantalla donde corresponda, el cifrado del archivo que encontró.

Actividad 6.3: Cifrado de archivo encontrado por 'Atziri' y cifrado con clave pública de 'Balam'.

Actividad 6.3: Cifrado de archivo encontrado por 'Balam' y cifrado con clave pública de 'Atziri'.

Envíe el mensaje cifrado a su compañera/compañero, ya sea por netcat o por el método que prefiera.

ACTIVIDAD 6.4: Ahora, van a descifrar el mensaje que recibieron por parte de su compañera/compañero, es importante recordar que al cifrar el mensaje usaron la clave pública de su compañera/compañero, por lo cual para el proceso de descifrado es necesario utilizar su propia clave privada.

En los algoritmos asimétricos, para cifrar se utiliza la **clave pública**, dado que esa es la clave que cualquier persona puede conocer de usted y que usted puede conocer de cualquier persona, y los mensajes privados se descifran con su **clave privada**.

Para descifrar el mensaje, ejecute: *«openssl pkeyutl -decrypt –inkey nombre_rsa.pem -in from_nombre_cipher».*

En la Figura N° 6.10 se descifra el mensaje que Atziri envió a Balam y el contenido de este.

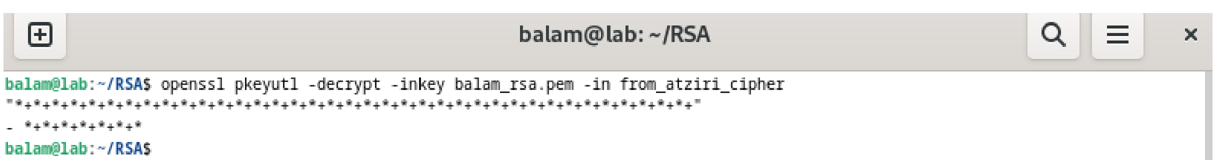


Figura N° 6. 10: Descifrado de archivo enviado por 'Atziri' y descifrado con clave privada de 'Balam'.

En caso de que quiera guardar el contenido del descifrado en algún archivo al comando agregue *–out nombre_archivo*, quedando el comando de la siguiente manera: *«openssl pkeyutl -decrypt –inkey nombre_rsa.pem -in from_nombre_cipher –out nombre_archivo».*

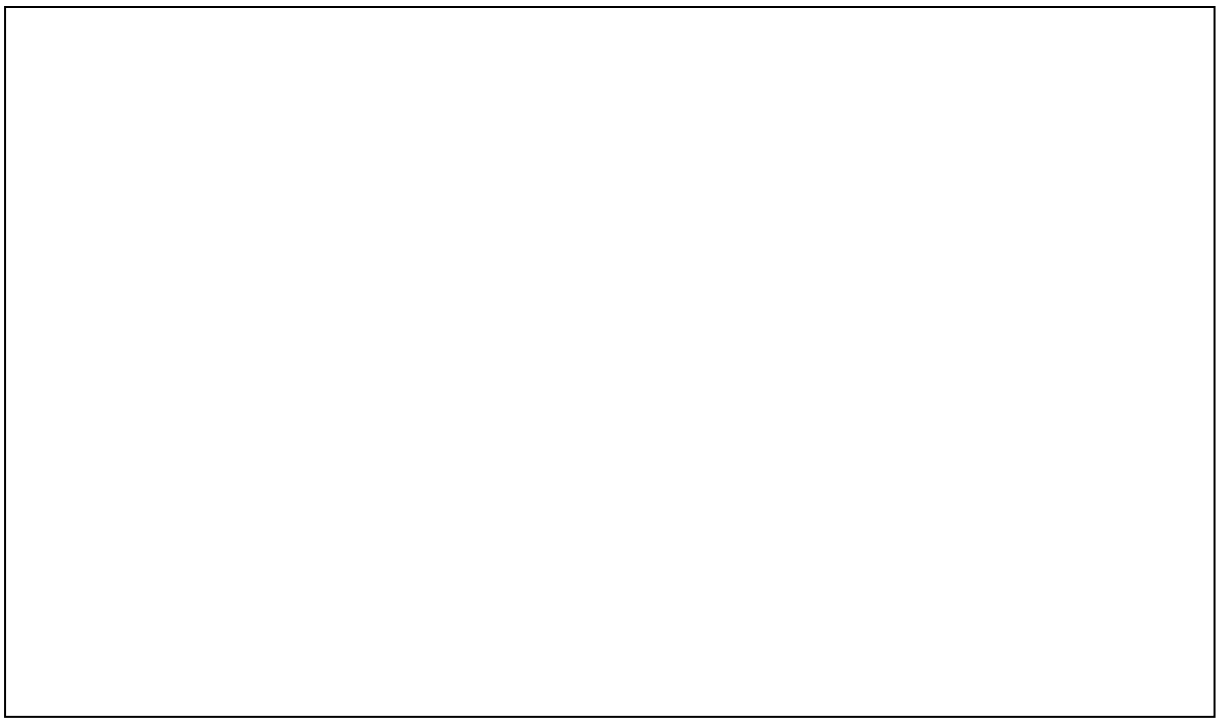
En los siguientes recuadros, coloque una captura de pantalla donde corresponda, del descifrado del archivo que le enviaron.

Actividad 6.4: Descifrado de archivo enviado por 'Atziri' y descifrado con clave privada de 'Balam'.

Actividad 6.4: Descifrado de archivo enviado por 'Balam' y descifrado con clave pública de 'Atziri'.

ACTIVIDAD 6.5: En el año 2006, Daniel Lerch Hostalot, realizó un ataque de factorización a RSA. El ataque se pudo llevar a cabo dado que las claves eran sumamente pequeñas. Es por eso por lo que se recomienda e incluso ahora por default, el uso de OpenSSL que tiene la capacidad de crear claves de mínimo 2048 bits.

Su trabajo fue publicado en una revista llamada Haking con el nombre Ataque de factorización a RSA. (Lech-Hostalot, D., 2006) El PDF de la publicación se encuentra en el directorio /home/crypto/Desktop/.Ataque_RSA. Lean el trabajo, y realicen en el siguiente recuadro un breve comentario resaltando los puntos principales del porqué se pudo realizar el ataque:



Actividad 6.5: Comentario sobre trabajo realizado por Daniel Lerch Hostalot.

Cabe mencionar que, si desea realizar el ejercicio mostrado en el documento, es importante tener presente que se puede llevar a cabo en versiones anteriores a Debian 12 ya que a partir de esta versión el propio sistema no permite la creación de claves pequeñas.

Conclusiones:

1. ¿Por qué el algoritmo RSA es considerado de los más seguros?

2. ¿Considera que la longitud con que se crearon las claves en la Actividad 6.2 fue insegura? ¿Sí o no y por qué?

3. En el trabajo de Daniel Lerch Hostalot, ¿Para qué es utilizado Msieve?

4. ¿Cuál fue el principal reto en la realización del script solicitado en la Actividad 6.1?

5. Comentarios o conclusiones adicionales

UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

ACTIVIDADES PRÁCTICAS: CRIPTOGRAFÍA

7. ELGAMAL

DOCENTE

ELGAMAL

Objetivo

Conocer el algoritmo de cifrado asimétrico ElGamal desde el desarrollo matemático y su uso.

Justificación

El conocer el algoritmo ElGamal se considera importante debido a que toma como base para su desarrollo el algoritmo de Diffie – Hellman y la robustez de los algoritmos discretos.

Introducción

En 1985 el investigador egipcio Taher ElGamal, propuso el algoritmo para cifrar y firmar conocido como «ElGamal», el cual está basado en el algoritmo de Diffie – Hellman en el que cada usuario calcula sus claves pública y privada.

La documentación la puede encontrar en A public key cryptosystem and a signature scheme based on discrete logarithms | IEEE Journals & Magazine | IEEE Xplore.

Para comenzar con la generación de claves se deben elegir o producir dos valores $\boldsymbol{p}$ y $\boldsymbol{\alpha}$ y para ello es necesario tomar en cuenta lo siguiente:

$$\boldsymbol{p \rightarrow Número\ primo\ grande}$$

$$\boldsymbol{\alpha \rightarrow Número\ aleatorio\ menor\ a\ p}$$

Asimismo, es necesario generar un número aleatorio que cumpla dos condiciones:

1. Debe ser número primo.
2. Debe ser menor a p;

este valor será llamado $\boldsymbol{\lambda}$, y cabe mencionar que es el que corresponderá a la clave privada del usuario, teniendo estos datos, se debe llevar a cabo la siguiente expresión para obtener la clave pública:

$$\boldsymbol{Clave\ pública = \beta = \alpha^{clave\ privada}\ mod\ p}$$

Por lo cual, los valores que serán compartidos serán: $\boldsymbol{p}$, $\boldsymbol{\alpha}$ y $\boldsymbol{\beta}$. Recuerde que la clave privada $\boldsymbol{\lambda}$ jamás debe compartirse.

Para la práctica, la comunicación se realizará entre los usuarios A (Atziri) y B (Balam) Atziri realiza lo anteriormente descrito y comparte sus valores $\boldsymbol{p}$, $\boldsymbol{\alpha}$ y $\boldsymbol{\beta}$ con Balam, estos son:

$\boldsymbol{p = 1709}$, $\boldsymbol{\alpha = 701}$ y $\boldsymbol{\beta = 821}$ el valor de $\boldsymbol{\lambda = 1451}$ lo conserva solo para ella.

Balam va a enviarle a Atziri un mensaje secreto el cual consta de dos letras para este ejercicio, las letras son $\boldsymbol{CD}$ y debe cifrarlas con los datos que Atziri le compartió, para eso realiza el siguiente procedimiento:

1. Obtener el equivalente numérico de $\boldsymbol{CD}$, se obtendrá de la misma manera que se obtuvo en la *Práctica N° 6. RSA*, por lo cual se usará el alfabeto en español, dicho valor es $\mathbf{151}$ y será llamado $\boldsymbol{N}$.
2. Debe generar un número aleatorio dentro de $\boldsymbol{p}$ el cual representa un número de sesión y debe generarse uno para cada comunicación que se realice, en este caso el número es $\mathbf{115}$, este valor será llamado $\boldsymbol{v}$.
3. Realice la operación $\boldsymbol{N_1 = \alpha^v \bmod p}$ cuyo resultado es: $\boldsymbol{N_1 = 701^{115} \bmod 1709 = 66}$
4. Además, se debe obtener $\boldsymbol{N_2 = N\beta^v \bmod p}$ cuyo resultado es:

$$\boldsymbol{N_2 = (151 * 821^{115}) mod\ 1709 = 512}$$

5. $\boldsymbol{N_1}$ y $\boldsymbol{N_2}$ representan el mensaje cifrado que Balam le va a enviar a Atziri, así que se los envía para que ella pueda realizar el descifrado.

Atziri realizará el descifrado con los valores que Balam le envió:

1. Realizar la operación $\boldsymbol{N_3 = N_1^{\lambda} \bmod p}$ cuyo resultado es: $\boldsymbol{N_3 = 66^{1451} \bmod 1709 = 1022}$
2. Obtener el inverso multiplicativo entre $\boldsymbol{N_3}$ *y* $\boldsymbol{p}$, esto es: $\boldsymbol{N_4 = inv(1022, 1709) = 1005}$

   (Se puede hacer uso del algoritmo extendido de Euclides)
3. Finalmente realizar: $\boldsymbol{N = (N_2 * N_4) \bmod p}$ el resultado es:

$$\boldsymbol{N = (512 * 1005) mod\ 1709 = 151}$$

   el cual representa el mensaje que Balam le envió a Atziri, el mensaje ya está descifrado sólo que está codificado.
4. De este valor se obtienen los valores equivalentes como se obtuvieron en la *Práctica N° 6. RSA*, por lo cual el descifrado es $\boldsymbol{CD}$.

Lo descrito anteriormente se puede observar en la Figura N° 7.1:

ATZIRI

CREACIÓN DE CLAVES

$p = 1709$, $\alpha = 701$ y $\lambda = 1451$

Realiza la operación:

$$\beta = 701^{133}\ mod\ 1709$$
$$\beta = 821$$

Comparte los datos p, α y β con Balam

DESCIFRADO

Atziri recibe $N_1 = 66, N_2 = 512$ con los cuales realizará el descifrado:

$$N_3 = 66^{1451}\ mod\ 1709 = 1022$$
$$N_4 = inv(1022{,}1709) = 1005$$
$$N = (1022 * 1005)\ MOD\ 1709 = 151$$

Mensaje descifrado = CD

BALAM

CIFRADO

Balam recibe de Atziri p, α y β los cuales utilizara Para cifrar CD cuyo equivalente numérico es 151

$$N = 151, v = 115$$
$$N_1 = 701^{115}\ mod\ 1709$$
$$N_1 = 66$$

$$N_2 = (151 * 821^{115})\ mod\ 1709$$
$$N_2 = 512$$

Comparta estos valores con Atziri para que realice el descifrado.

Figura N° 7. 1: Generación de claves, cifrado y descifrado.

El valor de $\boldsymbol{p}$ debe ser mayor al de $\boldsymbol{N}$ para obtener el cifrado y descifrado de manera correcta.

Equipo y material necesario

Máquina Virtual Debian, para montarla en su equipo puede seguir el «Montaje de máquina virtual», el cual es un manual de Instalación MV, en el que se encuentran las instrucciones tanto para VMware como para VirtualBox. Así mismo puede consultar el Anexo D que contiene ejemplos de cómo debe colocar sus respuestas en la práctica.

Desarrollo

Recuerde que debe hacer uso del usuario indicado en la Actividad 4.1 de la práctica N° 4 «Advanced Encryption Standard».

ACTIVIDAD 7.1: Busque el archivo llamado «*elgamal.py*», indique con qué comando fue buscado y la ruta absoluta en donde fue encontrado:

Actividad 7.1: Búsqueda del script.

Copie el archivo a un nuevo directorio llamado «ElGamal» dentro del home de su usuario.

Visualice y comente todas las funciones del script, este fue desarrollado en Python 3. Debe hacer funcionar el script indicando qué líneas, funciones, valores, variables etcétera faltan. El script debe funcionar de manera que cuente con las siguientes opciones, como se muestra en la Figura N° 7.2:

1. *Creación de clave pública.*
2. *Cifrado.*
3. *Descifrado.*

Figura N° 7. 2: Opciones disponibles del script.

a) Creación de clave pública

1. Cree la clave pública, aquí debe darle al usuario la opción de ingresar su clave privada o que sea asignada dentro de los valores de p, este valor debe ser mayor a 101.
2. p debe ser un número primo, se debe comprobar que lo sea.
3. Debe dar la opción de ingresar la clave privada o generar una aleatoria, en ambos casos debe verificar que el número sea primo.
4. Debe devolver el valor de la clave pública.

El funcionamiento de esta opción, asignando una clave privada, se muestra en la Figura N° 7.3:

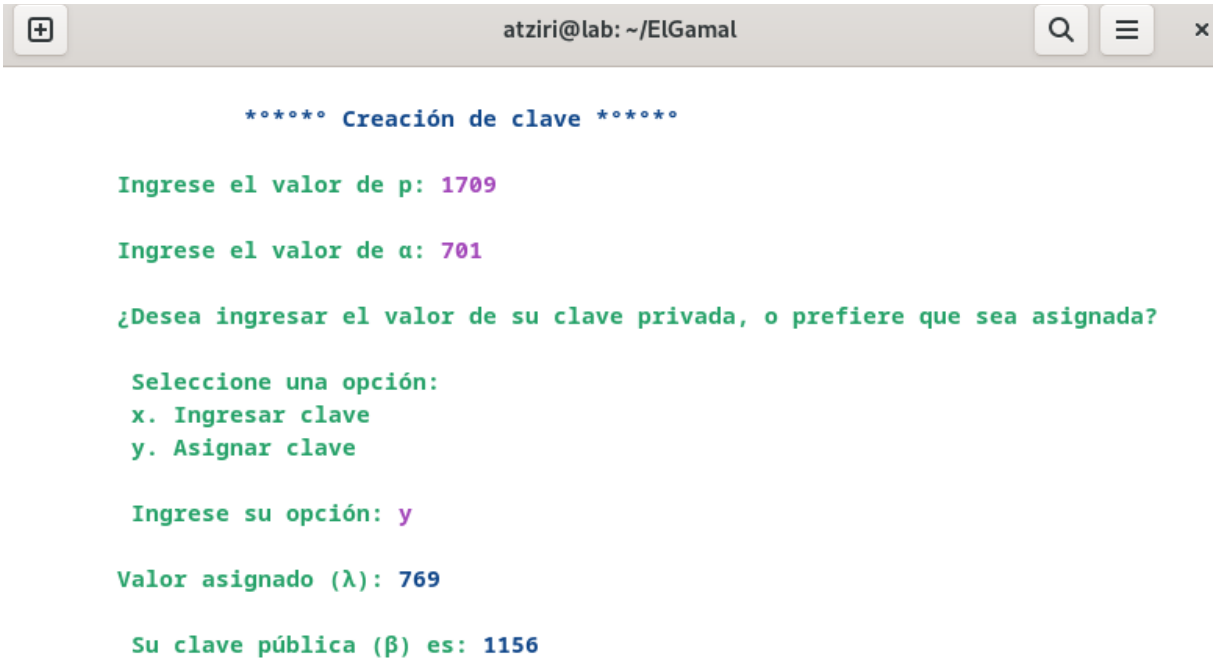


Figura N° 7. 3: Creación de claves.

b) Cifrado

1. Esta opción debe devolver los valores N_1 y N_2.
2. Se debe ingresar la palabra o letra(s) a cifrar, para esto se hace uso del alfabeto en español usado en la *Práctica N° 6. RSA*, así como el mismo proceso para obtener los equivalentes numéricos *(véase Actividad 6.1 apartado Cifrado de práctica N° 6. RSA).*
3. Para su realización debe pedir los valores p, α y β.
4. Debe verificar que el valor p sea mayor a N.
5. Debe dar la opción para generar aleatoriamente el valor υ o que el usuario lo ingrese.

Un ejemplo de esta opción se muestra en la Figura N° 7.4

c) Descifrado

1. Esta opción debe devolver la palabra o letra(s) descifradas.

2. Se deben ingresar los valores N_1 y N_2.
3. Se deben ingresar el valor de la clave privada λ.
4. Se debe ingresar el valor p.
5. Debe obtener la palabra o letra(s) siguiendo el proceso usado en la *Práctica N° 6. RSA (véase Actividad 6.1 apartado Descifrado de práctica N° 6. RSA).*

El ejemplo de esta opción se muestra en la Figura N° 7.5

Los posibles errores son mostrados en la Figura N° 7.6, debe tomar en cuenta para las modificaciones realizadas en el script.

Recuerde que todo lo introducido en el script debe ser comentado. El objetivo de esta actividad es que además de conocer el funcionamiento del algoritmo ElGamal, usted logre identificar errores en código.

El script está desarrollado en Python 3.

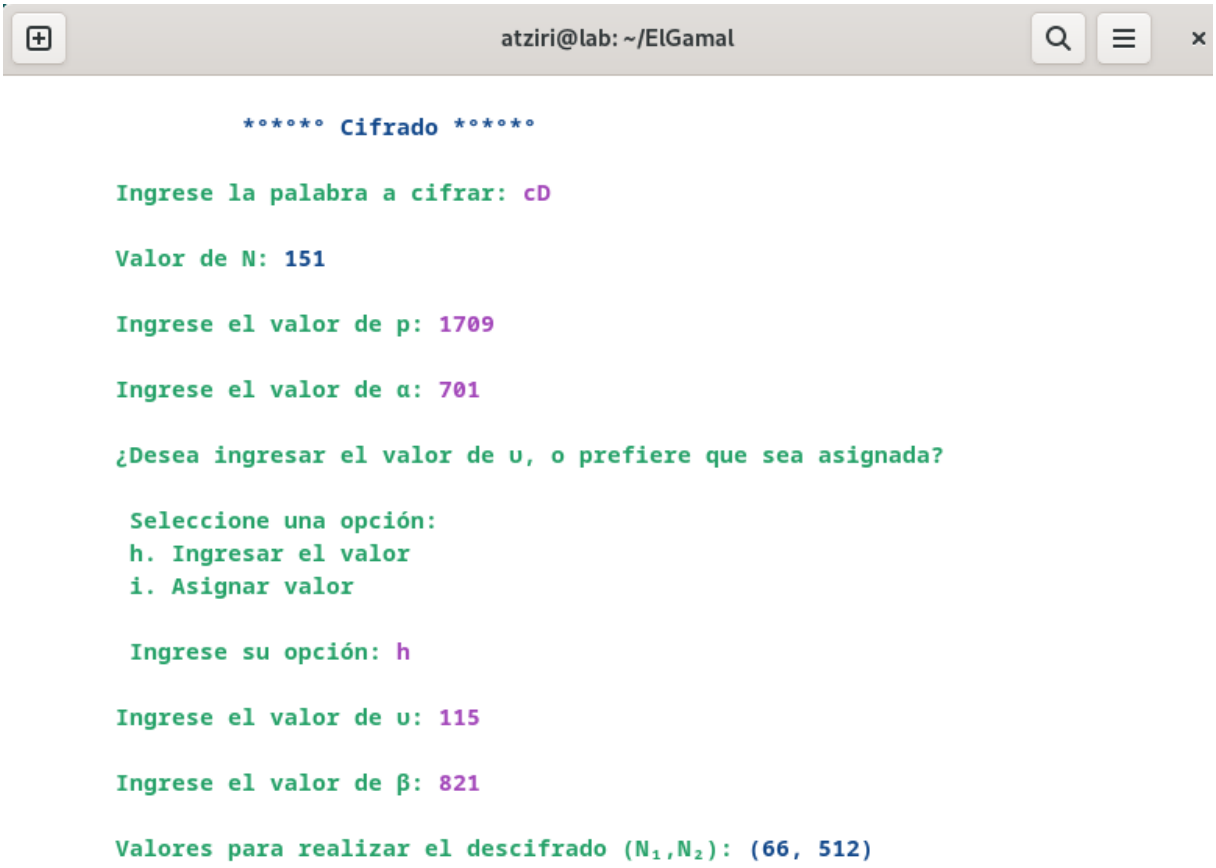


Figura N° 7. 4: Ejemplo de cifrado.

Figura N° 7. 5: Ejemplo de descifrado.

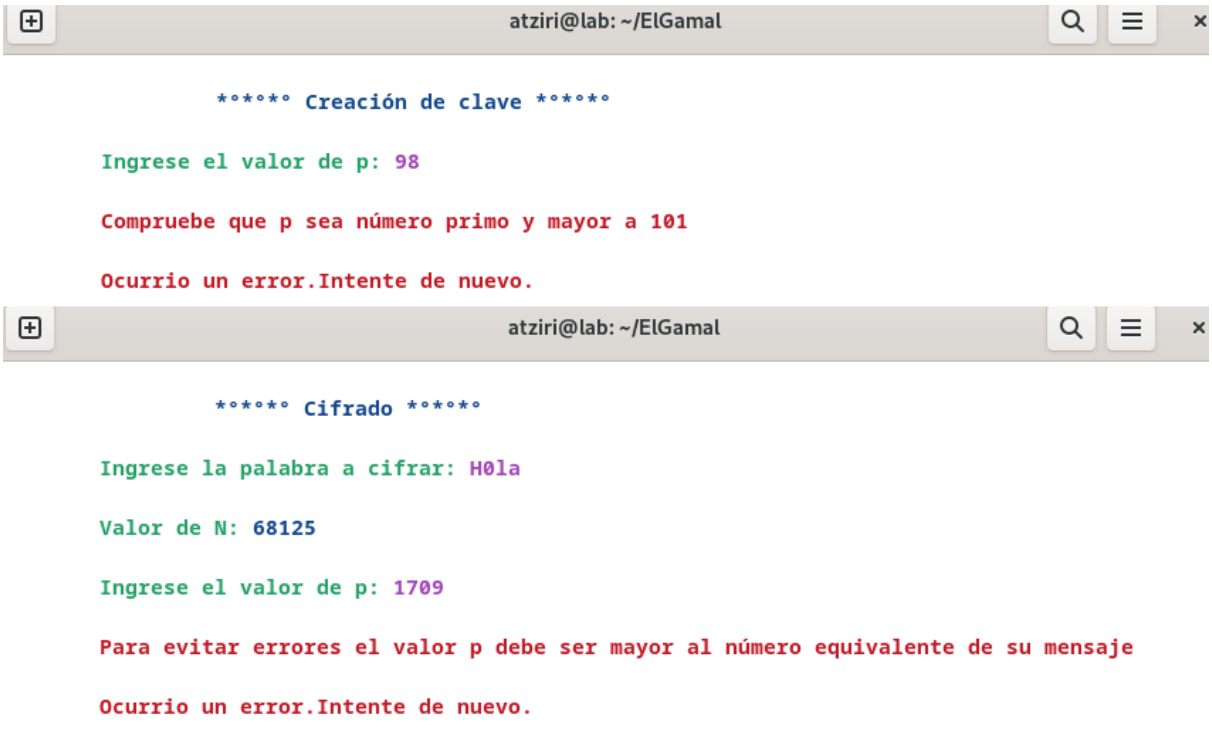


Figura N° 7. 6: Posibles errores.

Conteste: ¿Cuáles fueron los elementos que tuvo que corregir para que el script funcionara correctamente?

Actividad 7.1: Explicación de elementos corregidos en el script.

No olvide colocar su nombre en el script y hacerlo llegar a su docente como se le indique.

ACTIVIDAD 7.2: Formen parejas, Atziri y Balam, cada uno va a crear sus claves con el algoritmo ElGamal. Cada quien compartirá los datos necesarios con su compañera/compañero para que cada uno pueda cifrar un breve mensaje (escriba sus iniciales), mensaje que se enviarán entre sí y que cada uno de ustedes descifrará al recibirlo.

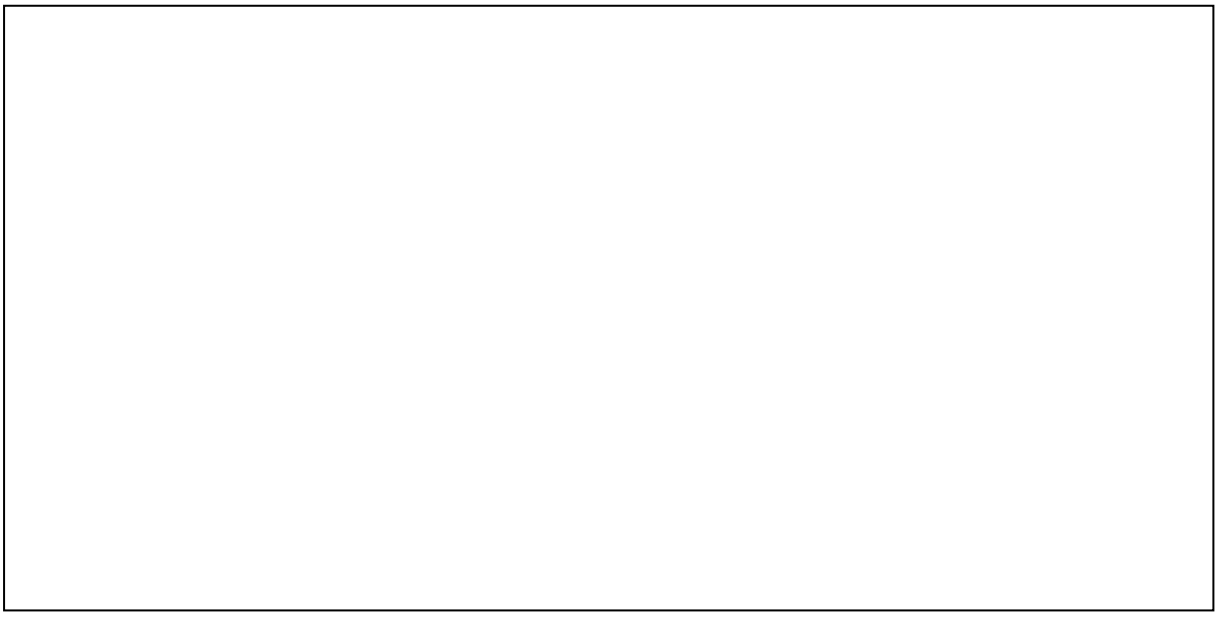
Es necesario hacer uso del script que acaban de arreglar, por lo cual será necesario colocar capturas de pantalla en donde se indique.

Anoten los nombres de: el alumno(a) que será Atziri y el alumno(a) que será Balam:

Atziri:

Balam:

1. Creación de claves para Atziri

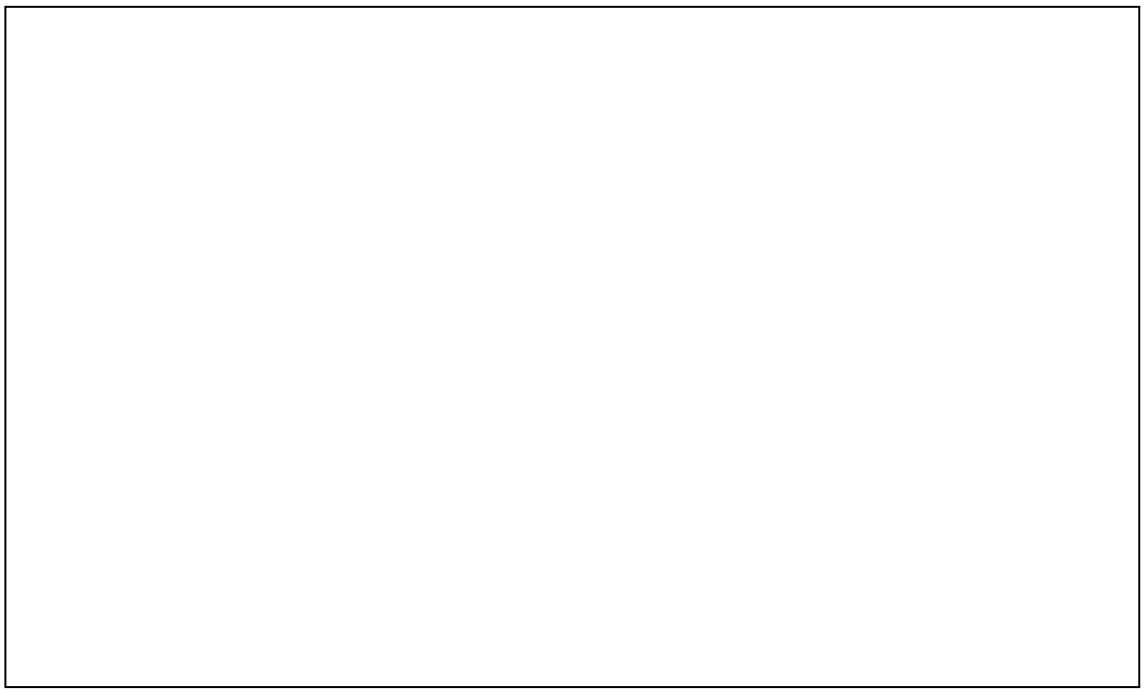


Actividad 7.2: Creación de claves para Atziri.

2. Creación de claves para Balam

Actividad 7.2: Creación de claves para Balam.

3. Cifrado de mensaje para Atziri por parte de Balam

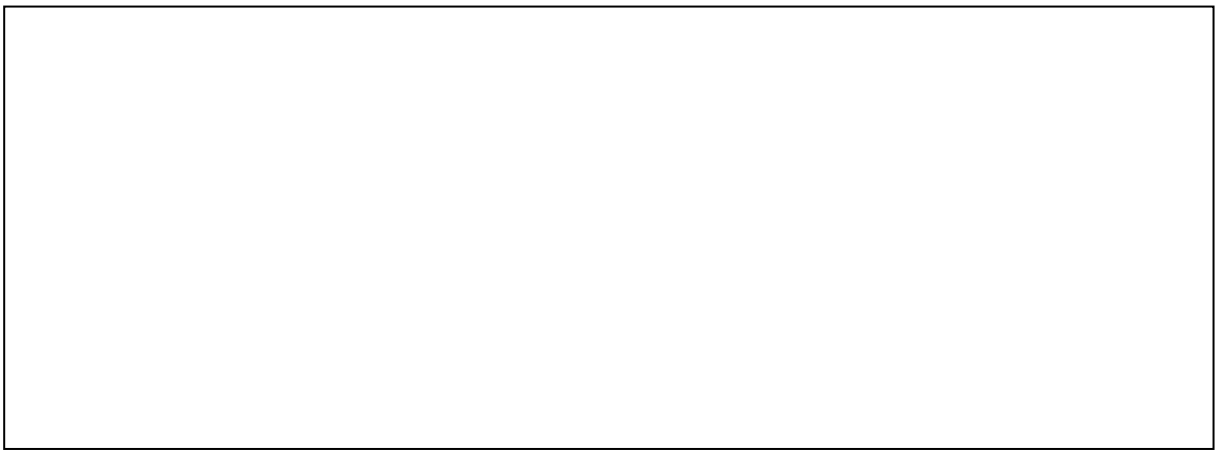


Actividad 7.2: Cifrado de mensaje para Atziri.

4. Cifrado de mensaje para Balam por parte de Atziri

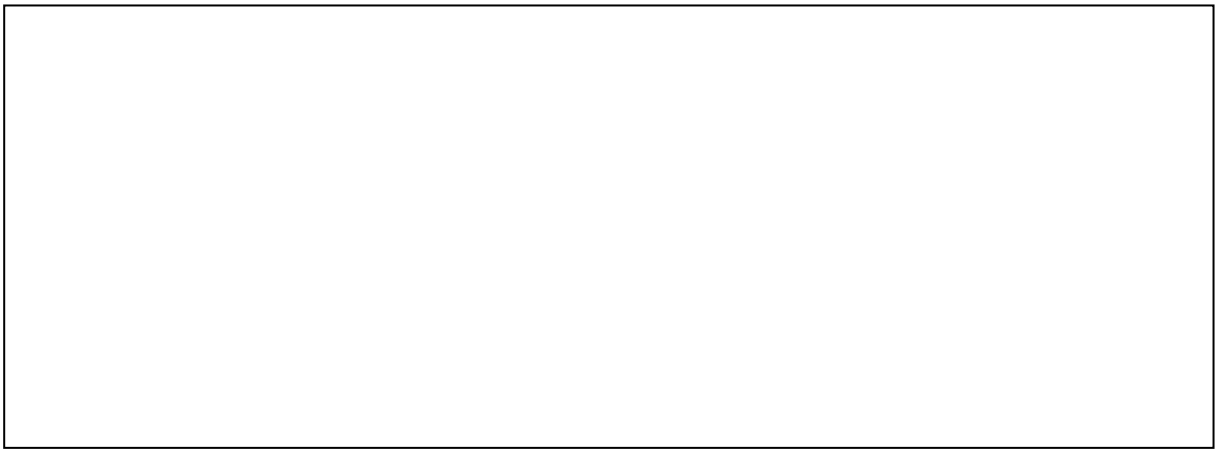
Actividad 7.2: Cifrado de mensaje para Balam.

5. Descifrado de mensaje que recibió Atziri



Actividad 7.2: Atziri descifra el mensaje.

6. Descifrado de mensaje que recibió Balam



Actividad 7.2: Balam descifra el mensaje.

ACTIVIDAD 7.3: Realice las modificaciones necesarias en el script creado en la Practica N° 5. DH Actividad 5.3, para obtener la clave privada de Balam.

Balam comparte los siguientes datos:

$$p = 536513$$

$$\alpha = 103$$

$$\beta = 192427$$

Figura N° 7. 7: Datos de Balam para obtener su clave privada.

Con estos datos, debe obtener el valor de λ.

La ejecución del script debe verse como el mostrado en la Figura N° 7.8.

Figura N° 7. 8: Obtener clave privada de Balam.

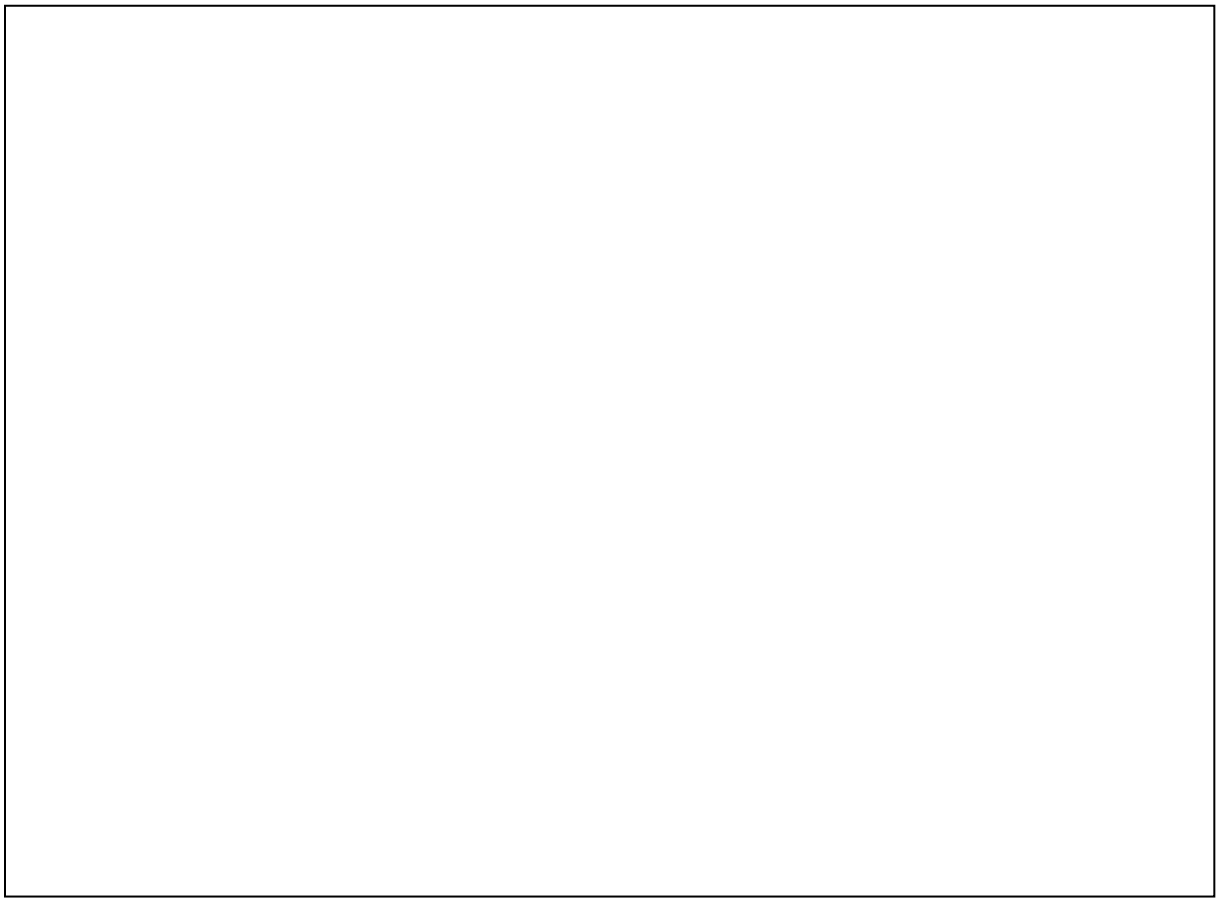
Como puede observar en la Figura 7.8 el script tardó aproximadamente 1 hora 33 minutos y solo llegó al número 103974, quedando aún lejano del valor de la clave privada, esto es debido a que los números son de cifras considerables, es por eso la importancia de usar número primos grandes.

Su script debe ser capaz de calcular el tiempo que tarde en obtener el valor de la clave, o bien el tiempo que estuvo trabajando tal como se muestra en la Figura N° 7.8.

Se recomienda comenzar desde el valor 420000 para evitar que la ejecución tarde tanto tiempo, puede tardar aproximadamente 35 minutos, todo dependerá de los recursos de su equipo.

Como extra para su script, si el usuario no ingresa ningún valor, como se muestra en la Figura N° 7.8, debe comenzar desde 1.

Coloque una captura de pantalla donde se observe la ejecución de su script y la obtención de la clave privada de Balam:



Actividad 7.3: Obtención de la clave privada de Balam.

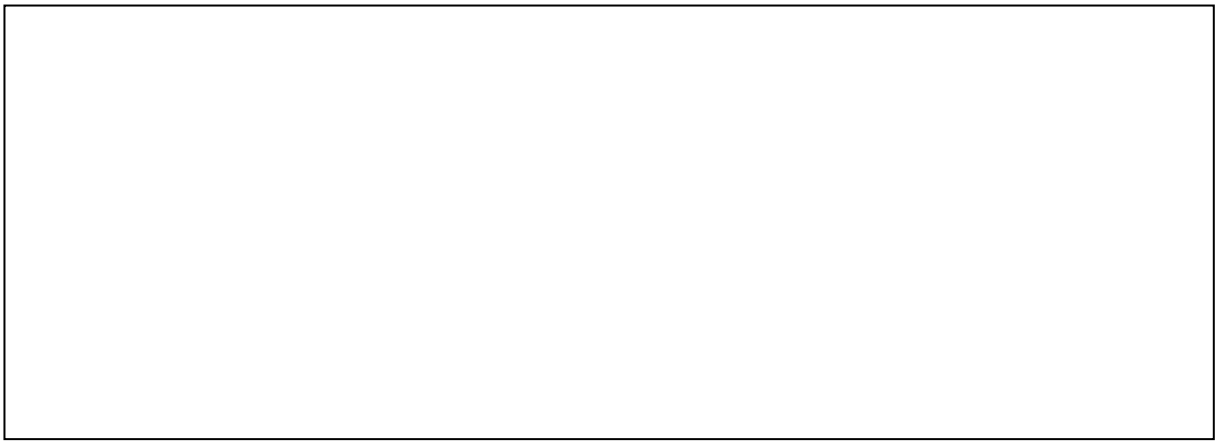
ACTIVIDAD 7.4: Con el valor obtenido en la actividad anterior, y haciendo uso del script de la Actividad 7.1, realice el descifrado con los siguientes valores:

$$N_1 = 486885$$

$$N_2 = 5544$$

Figura N° 7. 9: Datos para realizar el descifrado.

Coloque una captura de pantalla en donde se visualice la obtención del resultado:



Actividad 7.4: Descifrado de mensaje.

Conclusiones:

1. ¿Con qué otro algoritmo tiene similitud el algoritmo ElGamal?

2. En la actualidad, ¿Dónde se hace uso del algoritmo ElGamal?

3. ¿Cuál es la diferencia entre los algoritmos simétricos y los algoritmos asimétricos?

4. Además de ser usado para cifrar y descifrar, ¿Con qué otro objetivo fue creado el algoritmo ElGamal?

5. Comentarios y conclusiones adicionales.

Actividades prácticas

Sección 4: Aplicaciones

UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

ACTIVIDADES PRÁCTICAS: CRIPTOGRAFÍA

8. FIRMAS DIGITALES

DOCENTE

FIRMAS DIGITALES

Objetivo

Conocer los algoritmos usados para la creación y comprobación de firmas digitales con ElGamal y RSA.

Justificación

La criptografía tiene distintas aplicaciones, entre ellas se encuentran las firmas digitales, las cuales tienen como fin identificar y autenticar la identidad del creador o emisor de un mensaje, asimismo permite verificar la integridad de la información contenida en el mensaje, documento o similar. Así, en esta práctica conocerá la manera de generar firmas digitales y comprobar que las mismas sean auténticas.

Introducción

Las firmas digitales son elaboradas con base en algoritmos cuya función principal reside en comprobar **la autenticidad, integridad y no repudio**[17] en el archivo firmado. Es decir, las firmas digitales funcionan en los documentos electrónicos de manera similar a las firmas manuscritas, esto es, a la firma que una persona imprime con su puño y letra sobre un documento impreso en papel (SingPlus, s.f.).

En México, el Servicio de Administración Tributaria (SAT), emite firmas electrónicas a cada uno de los ciudadanos o empresas que así lo soliciten, llamadas *e.firmas* la cual en definición del SAT es: «el conjunto de datos y caracteres que te identifica al realizar trámites y servicios por Internet en el SAT, así como en otras Dependencias, Entidades Federativas, Municipios y la iniciativa privada. Tu e.firma es única, es un archivo seguro y cifrado, que tiene la validez de una firma autógrafa. Por sus características, es segura y garantiza tu identidad»[18]

Por lo cual, la *e.firma* es justamente una aplicación en la vida real de las firmas digitales.

En esta práctica se abordan dos tipos de firmas utilizando los algoritmos: ElGamal y RSA. Ambos fueron vistos en prácticas pasadas, sin embargo, realizando algunos pasos más se pueden utilizar para realizar firmas digitales con ellos. La parte clave de las firmas digitales es

17 **Autenticidad:** Confirmación del firmante como persona que firma el documento.

Integridad: La firma digital permite garantizar que el contenido no se ha cambiado ni se ha manipulado desde que se firmó digitalmente.

No repudio: Prueba a todas las partes el origen del contenido firmado. Por rechazo se entiende el acto de un firmante de negar cualquier asociación con el contenido firmado.

18 Preguntas frecuentes. ¿Qué es la e.firma (antes firma electrónica)? . Recuperado de https://www.sat.gob.mx/aplicacion/44275/descarga-de-manera-directa-tu-certificado-de-e.firma#

comprobar que el archivo es auténtico, esto es, que efectivamente es de quien lo envía. El documento puede ser enviado en claro o cifrado y hace uso de la función hash para realizar la firma.

Firma digital con ElGamal. (Mauro_88, 2013)

Recuerde, ElGamal es un algoritmo de criptografía asimétrica, sin embargo, cuando este fue creado se pensó su uso para realizar firmas digitales. Los pasos para obtener y comprobar una firma digital con ElGamal son muy similares a los realizados anteriormente como se puede observar:

1. **Creación de firma digital.**
    a) Debe contar con los valores $\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\lambda}$ y $\boldsymbol{p}$ y el archivo que será enviado.
    b) Obtener el valor hash del archivo a enviar.
    c) Teniendo el hash, debemos obtener el valor numérico de este hash, por lo cual se hace uso del alfabeto usado en la *Práctica N° 6: RSA*, obteniendo el equivalente de cada uno de los caracteres del hash y sumando estos, para así lograr tener el resultado, este número será llamado $\boldsymbol{h}(\boldsymbol{M})$.
    d) Se debe obtener un número aleatorio $\boldsymbol{H}$ que cumpla con la condición de $\boldsymbol{mcd}(\boldsymbol{H}, \boldsymbol{\phi_p}) = \mathbf{1}$, donde $\boldsymbol{H}$ es un número aleatorio menor a $\boldsymbol{p}$ y $\boldsymbol{\phi_p}$ es $\boldsymbol{p} - \mathbf{1}$.
    e) Se debe obtener el valor $\boldsymbol{H^{-1}}$ de la siguiente manera: $\boldsymbol{H^{-1}} = \boldsymbol{inv}(\boldsymbol{H}, \boldsymbol{\phi_p})$ (Se puede realizar usando el algoritmo extendido de Euclides)
    f) Obtener el valor $\boldsymbol{r}$ de la siguiente manera: $\boldsymbol{r} = \boldsymbol{\alpha^H \ mod \ p}$
    g) Obtener el valor $\boldsymbol{s}$ de la siguiente manera: $\boldsymbol{s} = ((\boldsymbol{h}(\boldsymbol{M}) - (\boldsymbol{\lambda} * \boldsymbol{r})) * \boldsymbol{H^{-1}}) \boldsymbol{mod} \ \boldsymbol{\phi_p}$
    h) Se hace envío del archivo, además de los valores $(\boldsymbol{r}, \boldsymbol{s})$, esto para que el receptor compruebe que el archivo es el auténtico.
    i) Aunque se da la opción de enviar el archivo en claro, lo recomendable es cifrarlo para aumentar la seguridad.
2. **Verificación de firma digital:**
    a) Debe contar con los valores $\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{p}$ del emisor, además de los valores $(\boldsymbol{r}, \boldsymbol{s})$ y el archivo recibido.
    b) Obtener $\boldsymbol{N_1}$ de la siguiente manera: $\boldsymbol{N1} = \boldsymbol{r^s \ mod \ p}$
    c) Obtener $\boldsymbol{N_2}$ de la siguiente manera: $\boldsymbol{N_2} = \boldsymbol{\beta^r \ mod \ p}$
    d) Con estos nuevos valores, obtener el valor $\boldsymbol{K_1}$ de la siguiente manera: $\boldsymbol{K_1} = (\boldsymbol{N_1} * \boldsymbol{N_2}) \ \boldsymbol{mod \ p}$

e) Se obtiene el hash del archivo recibido, del mismo modo se obtiene su equivalente numérico haciendo uso del alfabeto usado en la *Práctica N° 6: RSA.*

f) Con el valor obtenido en el punto anterior obtener $\boldsymbol{K_2}$ de la siguiente manera: $\boldsymbol{K_2 = \alpha^{h(M)} \bmod p}$

g) Si $\boldsymbol{K_1}$ y $\boldsymbol{K_2}$ son iguales, el archivo es auténtico, lo que quiere decir que no sufrió ninguna modificación y es enviado por el emisor esperado.

Los pasos con un ejemplo se muestran en la Figura N° 8.1:

ATZIRI

CREACIÓN DE FIRMA

$p = 90001, \alpha = 13, \beta = 64146, \lambda = 12781$

Se obtiene el hash en SHA256 del archivo a.pdf es:
62b5c7b09d9113566fc07 … a154d4f9ffe84391376a9921fd

Se obtiene su equivalente numérico de acuerdo con el alfabeto usado desde la Práctica N° 6: RSA:

6	2	b	5	c	7	…	1	f	D
34	20	2	33	3	35	…	29	6	4

Se suman todos los valores, y de este modo se obtiene $h(M) = \mathbf{1555}$

El número $H = 7$ cumple con la condición $mcd(7{,}90000) = 1$ por lo cual $H^{-1} = inv(7{,}9000)$ el resultado es $H^{-1} = \mathbf{77143}$

Se obtiene el valor: $r = 13^7 mod\ 90001 = \mathbf{17820}$

Se obtiene el valor:
$s = \big(1555 - (12781 * 17820)\big) * 77143 \ \ mod\ 90000 = \mathbf{56305}$

Debe enviar los valores (r,s) = (17820,56305) además del archivo, para que Balam compruebe que el archivo es autentico Y fue enviado por Atziri.

BALAM

COMPROBACIÓN DE FIRMA

$p = 90001, \alpha = 13, \beta = 64146, r = 17820, s = 56305$

Se obtiene el valor $N_1 = 17820^{56305} mod\ 90001 = \mathbf{42815}$
Se obtiene el valor $N_2 = 64146^{17820}\ mod\ 90001 = \mathbf{7869}$

Se obtiene el valor:
$K_1 = (42815 * 7869)\ mod\ 90001 = \mathbf{37492}$

Debe obtener el hash SHA256 del archivo recibido y del mismo modo, obtener el equivalente numérico de acuerdo al alfabeto:

6	2	b	5	c	7	…	1	f	D
34	20	2	33	3	35	…	29	6	4

Al sumar todos los valores se obtiene $h(M) = \mathbf{1555}$

Se obtiene el valor $K_2 = 13^{1555} mod\ 90001 = \mathbf{37492}$

Como $K_1 = K_2$ se comprueba la firma digital.

Figura N° 8. 1: Firmar digital con ElGamal.

Firma digital con RSA.

RSA, también es un algoritmo de cifrado asimétrico, de hecho, es considerado el más seguro hoy en día. Los valores de la clave pública $(\boldsymbol{n}, \boldsymbol{e})$ y la clave privada $(\boldsymbol{d})$ sirven para realizar firmas digitales y realizar la comprobación de estas. (Comunicación de datos, 2004)

Los pasos a seguir para la obtención de firmas digitales con RSA son:

1. **Creación de firma digital**

    a) Debe contar con los valores $\boldsymbol{n}$, $\boldsymbol{e}$, $\boldsymbol{d}$ y el archivo que será enviado.

    b) Obtener el valor hash del archivo a enviar.

    c) Teniendo el hash, se debe obtener el valor numérico de ese hash, por lo cual se hace uso del alfabeto usado en la *Práctica N° 6: RSA*, obteniendo el equivalente de cada uno de los caracteres del hash y sumando estos, para así lograr tener el resultado, este número será llamado $\boldsymbol{h}(\boldsymbol{M})$.

    d) Obtener el valor $\boldsymbol{r}$ de la siguiente manera: $\boldsymbol{r = h(M)^d \bmod n}$

    e) Se hace envío del archivo, además del valor $(\boldsymbol{r})$, esto para que el receptor compruebe que el archivo es auténtico.

    f) Aunque se da la opción de enviar el archivo en claro, lo recomendable es cifrarlo para aumentar la seguridad.

2. **Verificación de firma digital:**

    a) Debe contar con los valores $\boldsymbol{n}$, $\boldsymbol{e}$ del emisor, además del valor $(\boldsymbol{r})$ y el archivo recibido.

    b) Obtener $\boldsymbol{r_1}$ de la siguiente manera: $\boldsymbol{r_1 = r^e \bmod n}$

    c) Se obtiene el hash del archivo recibido, del mismo modo se obtiene su equivalente numérico haciendo uso del alfabeto usado en la *Práctica N° 6: RSA*, este es llamado $\boldsymbol{h}(\boldsymbol{M})$

    d) Si $\boldsymbol{r_1 = h(M)}$ se comprueba la integridad del archivo, lo que quiere decir que no sufrió ninguna modificación y es enviado por el emisor esperado.

En la Figura N° 8.2 se observa un ejemplo haciendo uso de esta firma:

ATZIRI

CREACIÓN DE FIRMA

$n = 869107, e = 17, d = 408113$

Se obtiene el hash en SHA256 del archivo a.pdf es:
62b5c7b09d9113566fc07 … a154d4f9ffe84391376a9921fd

Se obtiene su equivalente numérico de acuerdo con el alfabeto usado desde la Práctica N° 6: RSA:

6	2	b	5	c	7	…	1	f	D
34	20	2	33	3	35	…	29	6	4

Se suman todos los valores, y de este modo se obtiene $h(M) = \mathbf{1555}$

Se obtiene el valor:
$r = 1555^{408113} \ mod \ 869107 = \mathbf{841443}$

Debe enviar el valor (r) = (841443) además del archivo, para que Balam compruebe que el archivo es autentico Y fue enviado por Atziri.

BALAM

COMPROBACIÓN DE FIRMA

$n = 869107, e = 17, r = 841443$

Se obtiene el valor $r_1 = 841443^{17} \ mod \ 869107 = \mathbf{1555}$

Debe obtener el hash SHA256 del archivo recibido y del mismo modo, obtener el equivalente numérico de acuerdo al alfabeto:

6	2	b	5	c	7	…	1	f	D
34	20	2	33	3	35	…	29	6	4

Al sumar todos los valores se obtiene $h(M) = \mathbf{1555}$

Como $r_1 = h(M)$ se comprueba la firma digital.

Figura N° 8. 2: Firmar archivo con RSA.

Equipo y material necesario

Máquina Virtual Debian, para montarla en su equipo puede seguir el «Montaje de máquina virtual», el cual es un manual de Instalación MV, en el que se encuentran las instrucciones tanto para VMware como para VirtualBox. Así mismo puede consultar el Anexo D que contiene ejemplos de cómo debe colocar sus respuestas en la práctica.

Desarrollo

Recuerde que debe utilizar el usuario indicado en la Actividad 4.1 de la práctica N° 4 «Advanced Encryption Standard».

ACTIVIDAD 8.1: Cree un script para realizar la firma digital con ElGamal siguiendo los pasos descritos en la introducción y tomando como base la Figura N° 8.1

El programa debe tener dos opciones:

1. Firma del archivo.
2. Verificar firma del archivo.

Tal como se muestra en la Figura N° 8.3:

Figura N° 8. 3: Opciones disponibles en script para firma digital con ElGamal.

1. **Firma del archivo:** Para esta opción debe solicitar lo siguiente:
    a) Nombre del archivo a firmar: Debe verificar que el archivo exista, si existe debe obtener el valor hash del archivo y obtener su equivalente numérico, en pantalla solo muestra el nombre del archivo y su hash, en este caso se hace uso de SHA256, pero es libre de elegir la función hash que desee.
    b) Debe preguntar al usuario si desea cifrar el archivo, se recomienda realizar el cifrado con **AES**, ya que usted cuenta con código que realiza esta operación.

        i. Si la respuesta es sí, debe pedir la longitud del archivo, y la clave de cifrado, el vector IV debe crearse de manera aleatoria, es decir no debe solicitar dicho valor al usuario. En pantalla debe mostrarse el nombre del archivo ahora cifrado y el hash de este, y a continuación comenzará a pedir los valores para la firma, tal como se muestra en la Figura N° 8.4.

        ii. Si la respuesta es no, comenzará a pedir los valores para la firma, tal como se muestra en la Figura N° 8.5.

Figura N° 8. 4: Cifrar archivo a firmar.

Figura N° 8. 5: Se elige no cifrar el archivo a firmar.

c) Pida que ingresen los valores $p, \alpha, \beta, \lambda$. Para el valor H, muestre la opción de ingresarlo, o generarlo aleatoriamente, para lo cual se sugiere que tome en cuenta que H debe cumplir con $mcd(H, \phi_p) = 1$, posteriormente debe obtener $H^{-1} = inv(H, \phi_p)$.

Calcule los valores de (r, s) y muéstrelos. Recuerde que debe compartir estos valores a su receptor para la verificación de la firma, además del nombre del archivo y la clave de descifrado en el caso de haber cifrado el archivo. En la Figura N° 8.6 puede observar lo descrito en este punto:

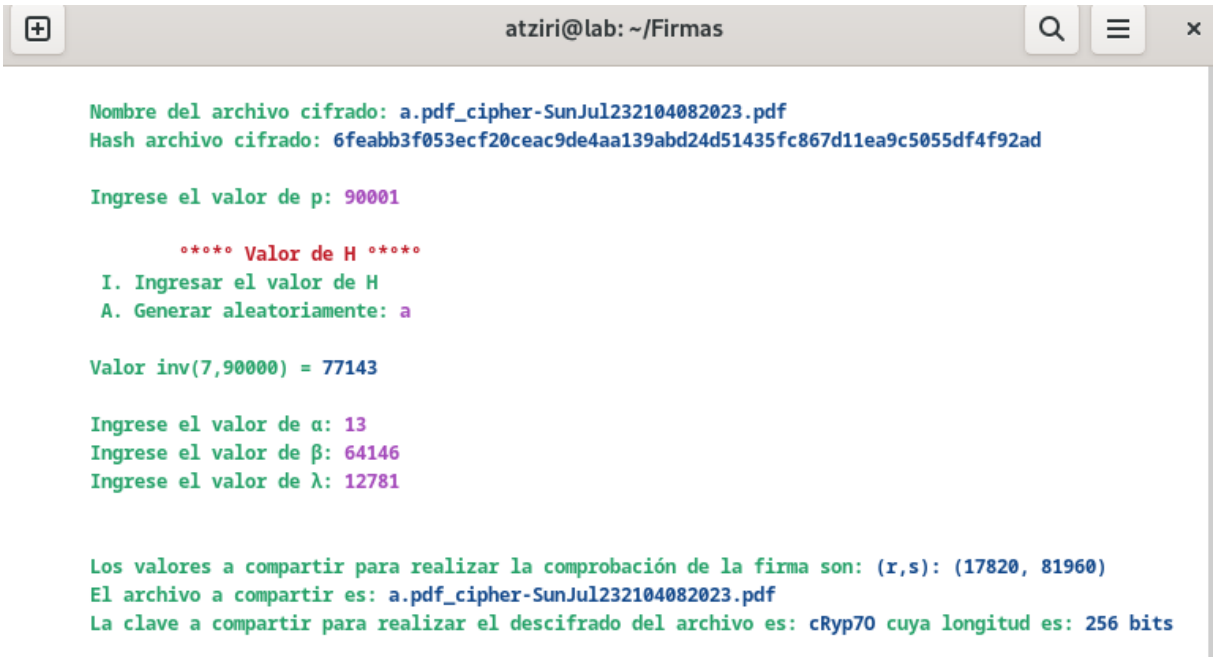


Figura N° 8. 6: Se elige generar aleatoriamente el valor H.

O bien, el usuario puede ingresar el valor H si ya lo conoce, tal como se muestra en la Figura N° 8.7:

Figura N° 8. 7: Se elige ingresar el valor de H.

En las Figuras mostradas anteriormente, se cifró el archivo, por lo cual los resultados son diferentes a los mostrados en la Figura N° 8.1.

El resultado para cuando el archivo no es cifrado se muestra en la figura N° 8.8:

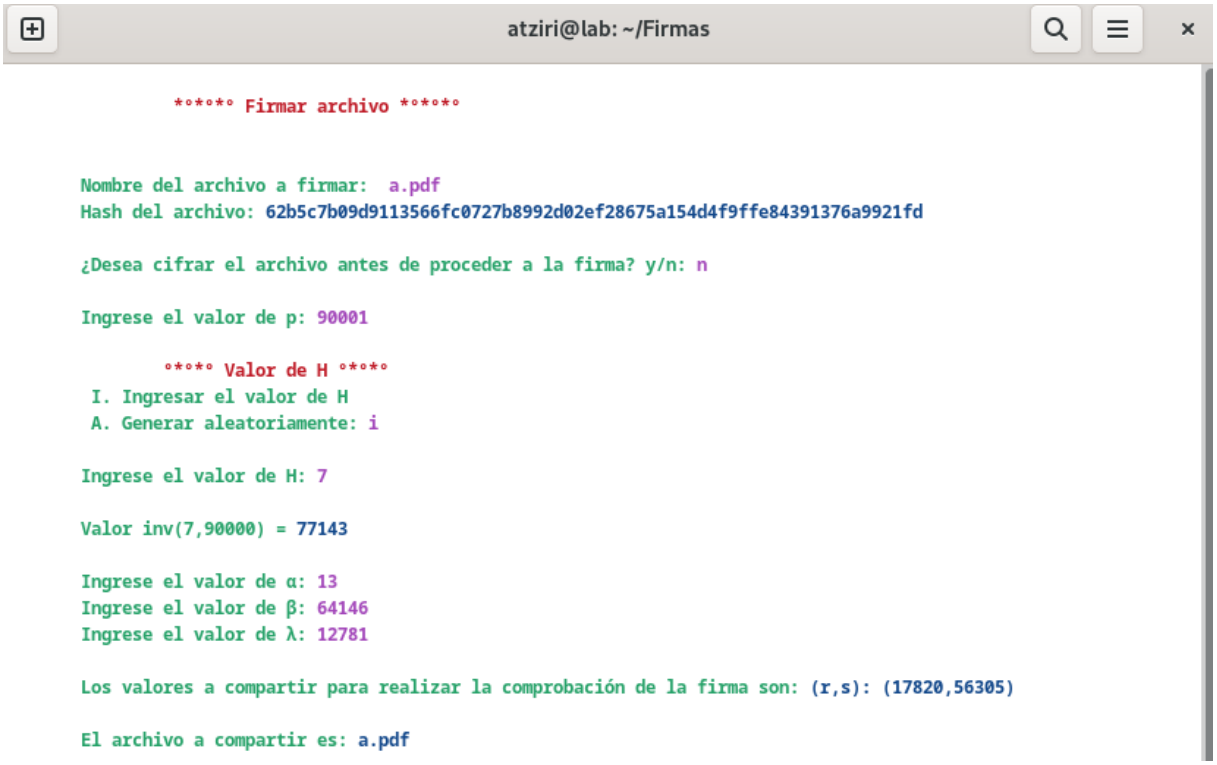


Figura N° 8. 8: Firma de archivo no cifrado.

2. **Verificar firma de archivo:** para esta opción pida que el usuario ingrese lo siguiente:
   a) Los valores p,α, β, r, s además del nombre del archivo en el cual se va a verificar la firma.
   b) Si la firma es correcta, debe mostrar un mensaje con el resultado, tal como se muestra en la Figura N° 8.9
   c) Si la firma no es correcta, debe mostrarse un mensaje indicando el resultado como se muestra en la Figura N° 8.10

Figura N° 8. 9: Verificar firma de archivo no cifrado, donde la firma es correcta.

Figura N° 8. 10: Verificar firma de archivo no cifrado, donde la firma no es correcta.

d) Sin embargo, si la firma es correcta, se debe preguntar si el archivo está cifrado, de manera que si está cifrado entonces debe darse la opción para descifrar el archivo.

   En esta opción debe pedir la longitud de la clave y la clave. Se muestra el resultado de este punto en la Figura N° 8.11:

Figura N° 8. 11: Verificar firma de archivo cifrado, donde la firma es correcta.

Tome en cuenta todos los posibles errores, ya sea: el incumplimiento de mcd, archivos no existentes, y opciones inválidas.

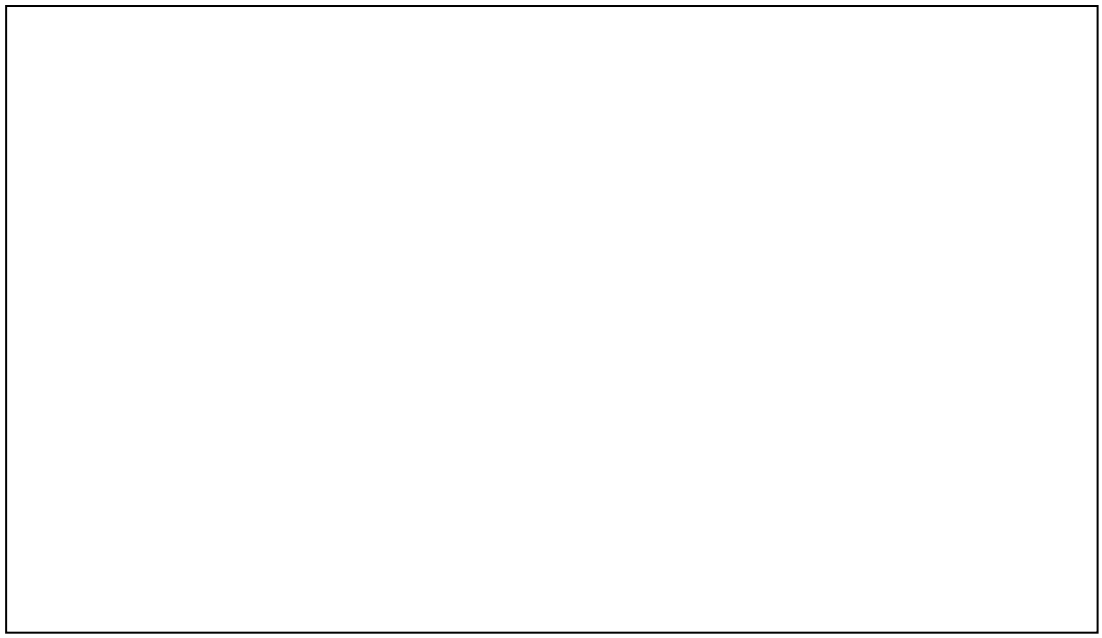
Envíe su script a su docente de la manera que se le indiqué.

Coloque en los siguientes recuadros un ejemplo de la firma y la comprobación de firma de un archivo, el archivo debe encontrarse cifrado.

Firmar archivo *(La captura debe verse como lo mostrado en la Figura N° 8.6):*

Actividad 8.1: Firmar un archivo.

Verificar firma de archivo *(Debe verse como lo mostrado en la Figura N° 8.11):*



Actividad 8.1: Firmar un archivo.

ACTIVIDAD 8.2: Busque un archivo llamado *Desafio_ElGamal.txt*, y que además en su interior tenga el siguiente fragmento de texto: «*0VsR2FtYWxfY2lwaGVyL*». Indique, ¿Cuál fue el comando o comandos que usó para la búsqueda del archivo? ¿En qué directorio está localizado? Intente usar una sola línea para localizar el archivo. Se recomienda buscar desde el directorio raíz.

Actividad 8.2: Búsqueda de archivo.

Revise el contenido del archivo, debe visualizar algo parecido a lo mostrado en la Figura N° 8.12:

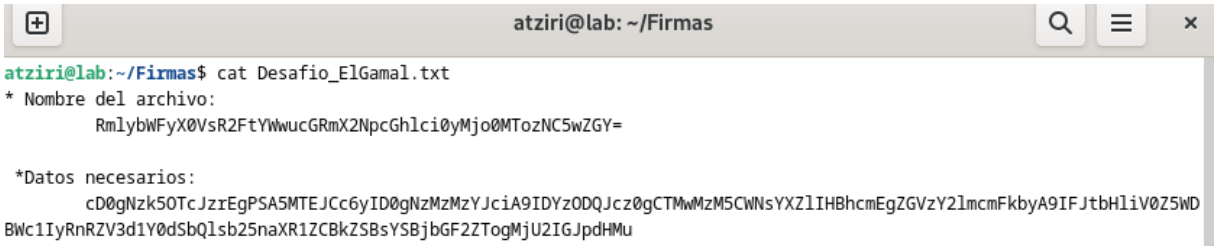


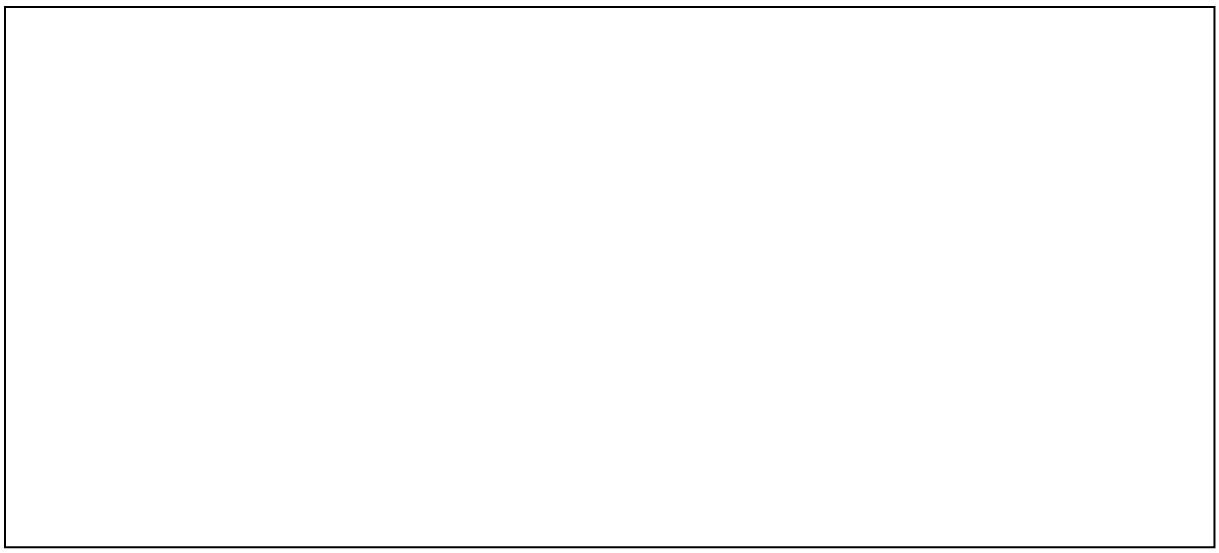
Figura N° 8. 12: Ejemplo de archivo con datos a utilizar en la verificación de firma.

Como puede observar, hay dos cadenas que se encuentran en base64, decodifique la primera cadena, puede realizarlo usando el comando *echo -n «cadena» | base64 -d* en la terminal de Linux, o con un pequeño script en python, ¿Cuál es el resultado? Debe obtener el nombre de un archivo.

Actividad 8.2: Nombre de archivo.

El archivo se encuentra en el mismo directorio donde localizó el archivo Desafio_ElGamal.txt, decodifique la segunda línea, use los datos para comprobar la firma y descifrar el archivo. En el mismo directorio encontrará un archivo llamado *verificar_elgamal*, utilícelo para realizar la actividad.

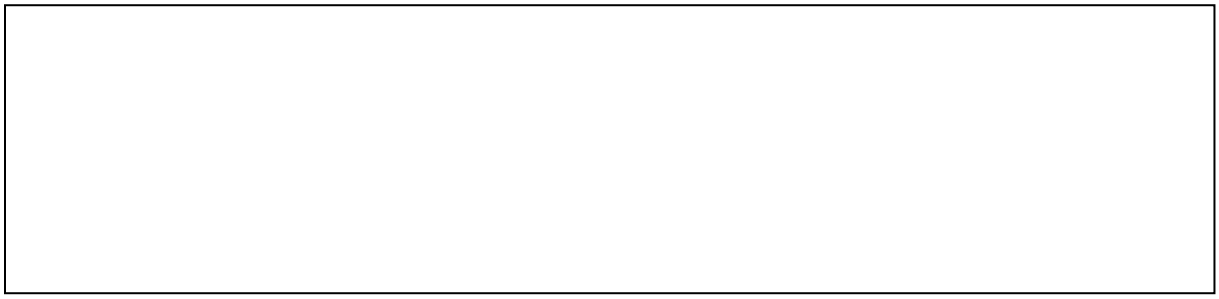
Coloque una captura de pantalla donde se visualice la comprobación de la firma y descifrado del archivo.



Actividad 8.2: Comprobar firma y descifrar el archivo.

Realice lo indicado en el archivo que descifró. Recuerde ***hacer uso de valores diferentes a los mostrados en los ejemplos.***

Coloque una captura de pantalla de archivo FirmaElGamal_INICIALES (Recuerde cambiar la palabra INICIALES por sus iniciales comenzando por apellido).



Actividad 8.2: Contenido del archivo FirmaElGamal_INICIALES.

Envíe a su docente como le indique, su archivo con los datos necesarios y el archivo con su reseña para verificar la firma, la cual debe coincidir con la imagen colocada en el recuadro anterior.

ACTIVIDAD 8.3: Realice las modificaciones necesarias en el script creado en la Actividad 8.1 para Firmar y verificar firma de un archivo con RSA. Puede consultar la Figura N° 8.2.

El programa debe tener dos opciones:

1. Firmar archivo
2. Verificar firma del archivo

1. **Firmar archivo:** Debe pedir el nombre del archivo a firmar, se debe comprobar que el archivo exista. Igual que en el script de ElGamal, debe dar la opción para que el usuario decida cifrar o no el archivo. Finalmente debe pedir los valores n, d y obtener el valor r, mostrar dicho valor, así como el archivo que se firmó y si es el caso, la clave para descifrar. En la Figura N° 8.13 se muestra el proceso de firma sin usar el cifrado.

Figura N° 8. 13: Firmar archivo con RSA.

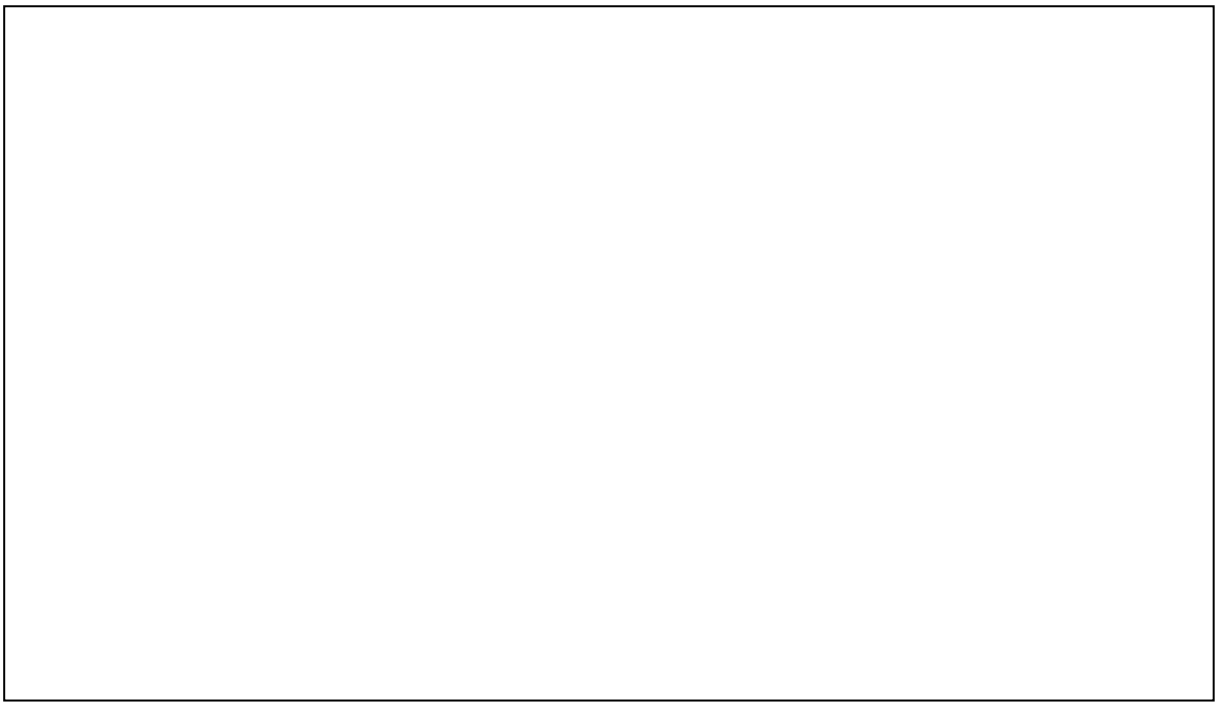
2. **Verificar la firma del archivo:** Debe pedir los valores n, r y e, además del archivo a verificar la firma. Si la firma es correcta, se debe preguntar si el archivo está cifrado, si está cifrado debe darse la opción para descifrarlo. En la Figura N° 8.14 se muestra el proceso de comprobación de la firma sin usar el cifrado.

Figura N° 8. 14: Verificar firmar de archivo con RSA.

Tome en cuenta todos los posibles errores, como: ingresar opciones inválidas, o verificar que el archivo exista.

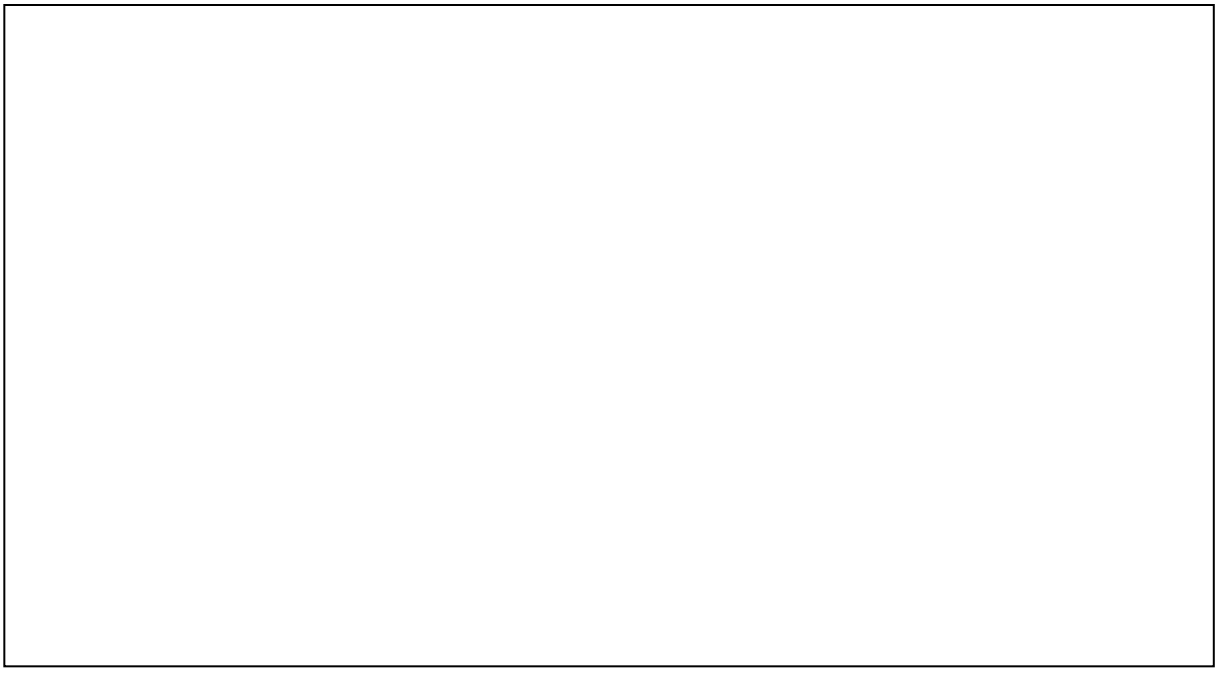
En los siguientes recuadros, coloque captura de pantalla en donde firme y verifique la firma de un archivo. Recuerde que en RSA la firma es correcta, si al verificarla se obtiene el hash del archivo, el archivo debe encontrarse cifrado.

Firmar archivo: *(En la captura basta con que solo se visualice el nombre del archivo cifrado, el ingreso de los datos n, d, el nombre del archivo a compartir y la clave para el descifrado).*



Actividad 8.3: Firmar un archivo.

Verificar firma de archivo: (En la captura basta con que solo se visualice el nombre del archivo cifrado, el ingreso de los datos n, d, el nombre del archivo a compartir y la clave para el descifrado)



Actividad 8.3: Firmar un archivo.

ACTIVIDAD 8.4: Del mismo modo que en la Actividad 8.2, busque un archivo llamado Desafio_RSA.txt, y que además en su interior tenga el siguiente fragmento de texto: «*yX1JTQV9jaXBoZXI*». Indique, ¿Cuál fue el comando o comandos que usó para la búsqueda del archivo? ¿En qué directorio está localizado? Intente usar una sola línea para localizar el archivo.

Actividad 8.4: Búsqueda de archivo

.

Revise el contenido del archivo, debe visualizar algo parecido a lo mostrado en la Figura N° 8.12, como puede observar, nuevamente encontrará dos cadenas que se encuentran en base64, decodifique la primera cadena, puede realizarlo usando el comando *echo -n «cadena» | base64 -d* en la terminal de Linux, o con un pequeño script en python.

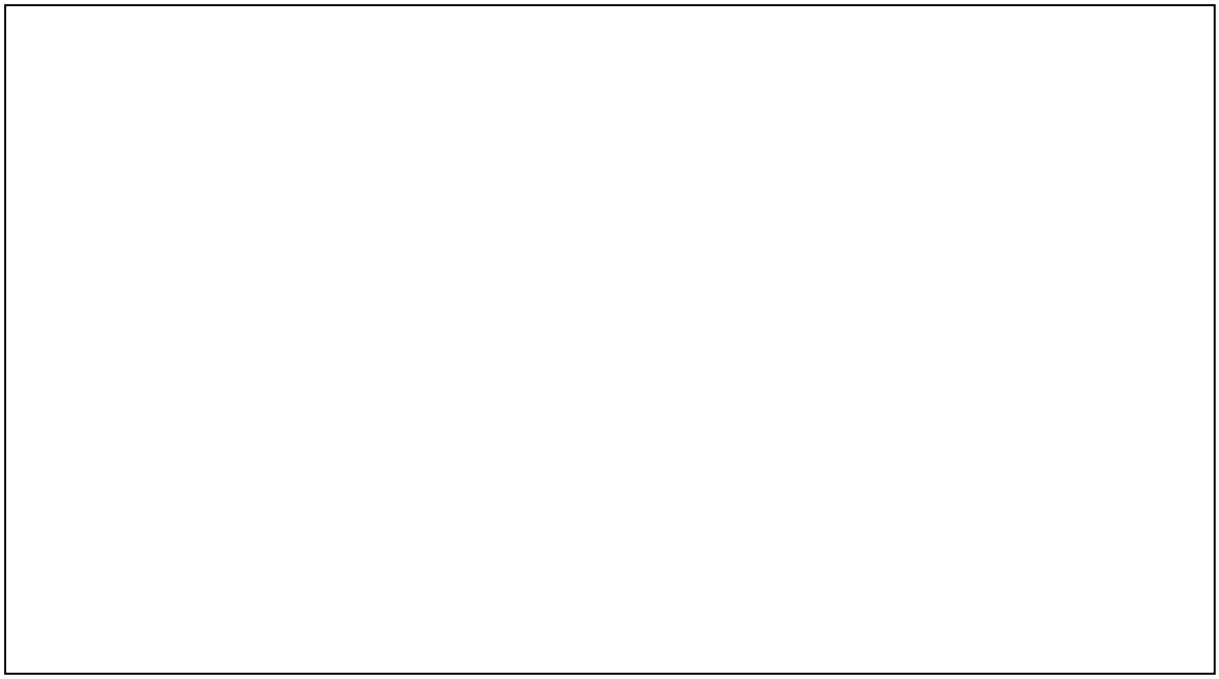
¿Cuál es el resultado? Debe obtener el nombre de un archivo. En el siguiente recuadro coloque una captura de pantalla donde se observe el contenido de Desafio_RSA.txt y el resultado de cada una de las líneas decodificadas.

Actividad 8.4: Decodificación de líneas en base64.

El archivo se encuentra en el mismo directorio donde localizó el archivo Desafio_RSA.txt.

En el mismo directorio se encuentra una archivo llamado *verificar_rsa* úselo para realizar la actividad.

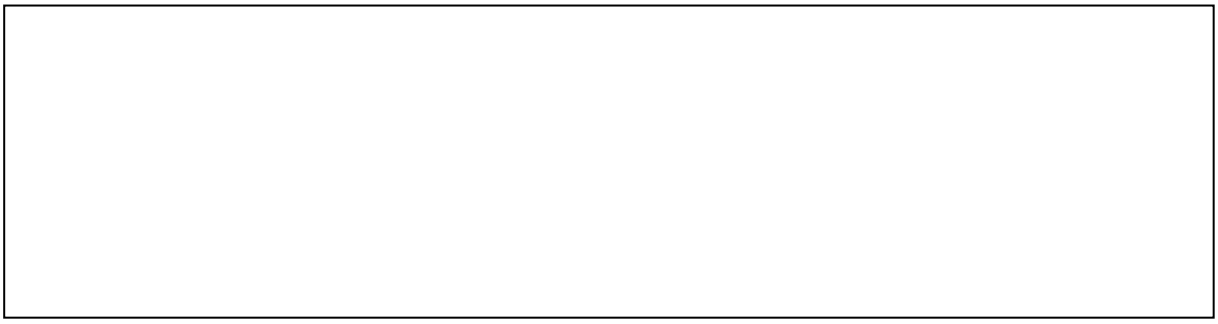
Coloque una captura de pantalla donde se visualice la comprobación de la firma y descifrado del archivo.



Actividad 8.4: Comprobar firma y descifrar el archivo.

Realice lo indicado en el archivo que descifró. Recuerde que debe entregar los datos tal cual como le fueron dados. ***Haga uso de valores diferentes a los mostrados en los ejemplos.***

Coloque una captura de pantalla de archivo FirmaRSA_INICIALES (Recuerde cambiar la palabra INICIALES por sus iniciales comenzando por apellido):



Actividad 8.4: Contenido del archivo FirmaRSA_INICIALES

Envíe a su docente el archivo con los datos necesarios y el archivo con su reseña para verificar la firma, la cual debe coincidir con la imagen colocada en el recuadro anterior.

Firma Digital de Curva Elíptica (ECDSA)

Además de los algoritmos de firma con ElGamal y RSA, el uso de curvas elípticas para la firma digital ha ganado mucha relevancia en la actualidad.

Este algoritmo, basado en las propiedades matemáticas de las curvas elípticas, se utiliza para generar firmas digitales que permiten verificar la autenticidad e integridad de mensajes o documentos digitales.

Las firmas digitales con curvas ofrecen un nivel de seguridad comparable al de RSA pero con claves más pequeñas, lo que mejora la eficiencia y reduce los requisitos de almacenamiento y procesamiento. Este enfoque se ha vuelto especialmente popular en aplicaciones de alta seguridad, como comunicaciones seguras, transacciones financieras, entre otras (VPN Unlimited, s.f.).

Existen diversas documentaciones según el uso que se desee dar a ECDSA, las cuales pueden ser consultadas en los siguientes enlaces

RFC 6637 RFC 6637: Elliptic Curve Cryptography (ECC) in OpenPGP

RFC 5758 RFC 5758: Internet X.509 Public Key Infrastructure: Additional Algorithms and Identifiers for DSA and ECDSA

RFC 5656 RFC 5656: Elliptic Curve Algorithm Integration in the Secure Shell Transport Layer

Conclusiones:

1. ¿Qué valor no es compartido para comprobar la firma de archivo en ElGamal?

2. ¿Qué valor no es compartido para comprobar la firma de archivo en RSA?

3. ¿En qué otro caso puede ser utilizada la Firma Digital?

4. ¿Qué fue lo más complicado de la realización de sus scripts?

5. Comentarios y conclusiones adicionales.

UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

ACTIVIDADES PRÁCTICAS: CRIPTOGRAFÍA

9. ESTEGANOGRAFÍA Y PGP

DOCENTE

ESTEGANOGRAFÍA | PRETTY GOOD PRIVACY

Objetivo

Conocer distintos servicios que ayudan al cifrado de correo electrónico mediante el uso de claves PGP (Pretty Good Privacy).

Como identificar la diferencia entre criptografía y esteganografía, así como el uso de ésta.

Justificación

Una aplicación más de la criptografía es PGP, cuya funcionalidad más popular es el cifrado y descifrado de correo electrónico. Actualmente, el correo electrónico se ha vuelto de suma importancia para la transferencia de información sensible, es por ello la importancia de conocer herramientas que ayuden a darle seguridad a la información.

Por otro lado, la esteganografía es una herramienta para ocultar información y de este modo pase inadvertida.

Introducción

El nombre PGP responde a las siglas Pretty Good Privacy y se trata de un proyecto iniciado a principios de los 90 por Phil Zimmermann.

La gran ausencia de aquel entonces de herramientas sencillas, potentes y económicas que acercaran la criptografía a los usuarios, animó a su autor a desarrollar dicha aplicación.

Con el paso de los años, PGP se ha convertido en uno de los mecanismos más populares y fiables para mantener la seguridad y privacidad en las comunicaciones, especialmente a través del correo electrónico.

Una clave PGP es una clave pública de cifrado que se puede usar para firmar y cifrar correos electrónicos e incluso, archivos (GNOME HELP, s.f.). Cuando un usuario crea una clave PGP, se genera un par de claves: clave pública y clave privada. Puede compartir la clave pública con cualquier persona de quien desee recibir mensajes o archivos cifrados, pero recuerde, en todos los casos del uso de criptografía asimétrica, la clave privada sólo debe conocerla el usuario para poder descifrar los mensajes recibidos.

Por otro lado, la esteganografía, es un método que permite ocultar información dentro de otro objeto, principalmente, imágenes, pero también se usan archivos de audio o video.

Se debe tener clara la diferencia entre esteganografía y criptografía, pues si bien, como se ha visto anteriormente, la criptografía transforma la información de manera que sea incomprensible, la esteganografía oculta la información, pues bien, se puede creer que solo se ve una imagen, sin sospechar que ésta tiene oculta información.

Hay distintas herramientas para realizar la esteganografía, durante esta práctica se verán 3 de ellas.

Equipo y material necesario

Máquina Virtual Debian, para montarla en su equipo puede seguir el «Montaje de máquina virtual», el cual es un manual de Instalación MV, en el que se encuentran las instrucciones tanto para VMware como para VirtualBox. Así mismo puede consultar el Anexo D que contiene ejemplos de cómo debe colocar sus respuestas en la práctica.

Desarrollo

Recuerde que debe hacer uso del usuario indicado en la Actividad 4.1 de la práctica N° 4 «Advanced Encryption Standard».

ACTIVIDAD 9.1: En la máquina virtual, se encuentra un directorio en el directorio raíz llamado *Esteganografía*, dentro de este hay algunas imágenes y archivos que se usarán para realizar Esteganografía.

En el directorio home de su usuario, realice un directorio llamado *Esteganografía*, dentro de él cree cuatro directorios llamados *Steghide*, *Stegosuite, Cat* y Stenography, por ahora todos los directorios están vacíos, conforme avance la práctica se van a ir colocan archivos. El directorio *Esteganografía* se debe ver como se muestra en la Figura N° 9.1.

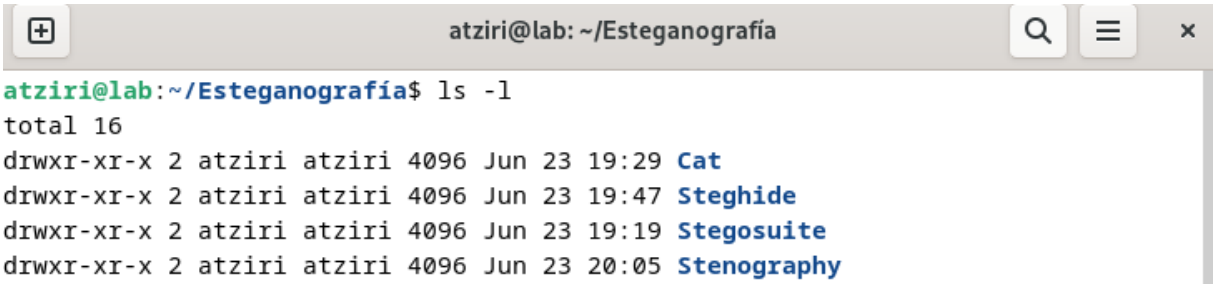


Figura N° 9. 1: Ejemplo de directorio a trabajar para esteganografía.

Dentro de cada directorio va a copiar archivos de su preferencia del directorio */Estenografía*, este directorio tiene imágenes, archivos de texto y archivos PDF. La copia se hace como super usuario, ya que de otra manera no dejará realizar la copia.

Aspectos por considerar: Intente que la imagen elegida sea de mayor tamaño que el archivo de texto o PDF a ocultar, ya que algunas de las herramientas marcan errores si detecta que la imagen no es suficientemente grande para ocultar la información.

A todos los archivos que copió, *debe cambiar al dueño*, para que el dueño sea su usuario, esto para evitar errores en el manejo de los archivos.

ACTIVIDAD 9.2: *Steghide* ayuda a ocultar información, sobre todo archivos de texto, dentro de imágenes en formato JPG, BMP o incluso archivos de sonido. En este caso se hará uso solo de imágenes. Steghide solicita contraseña para poder esconder la información, misma que será solicitada cuando se quiera extraer. La información oculta la guarda en la misma imagen, no crea un nuevo archivo. El uso de la herramienta se hace desde terminal. (ESGEEKS, s.f.)

Copie del directorio /Estenografía a su directorio Steghide una imagen, de esta cree una copia, es decir tendrá dos veces la misma imagen, y un archivo de texto, su directorio debe verse algo parecido a lo mostrado en la Figura N° 9.2:

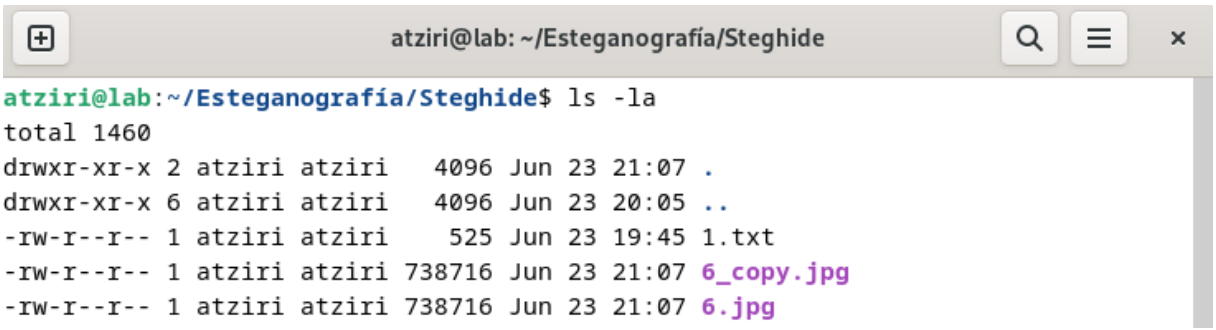


Figura N° 9. 2: Ejemplo de directorio Steghide.

La manera de ocultar la información se realiza ejecutando: «*steghide embed -ef archivo_a_ocultar -cf imagen_donde_ocultar*». Esto se muestra en la Figura N° 9.3:

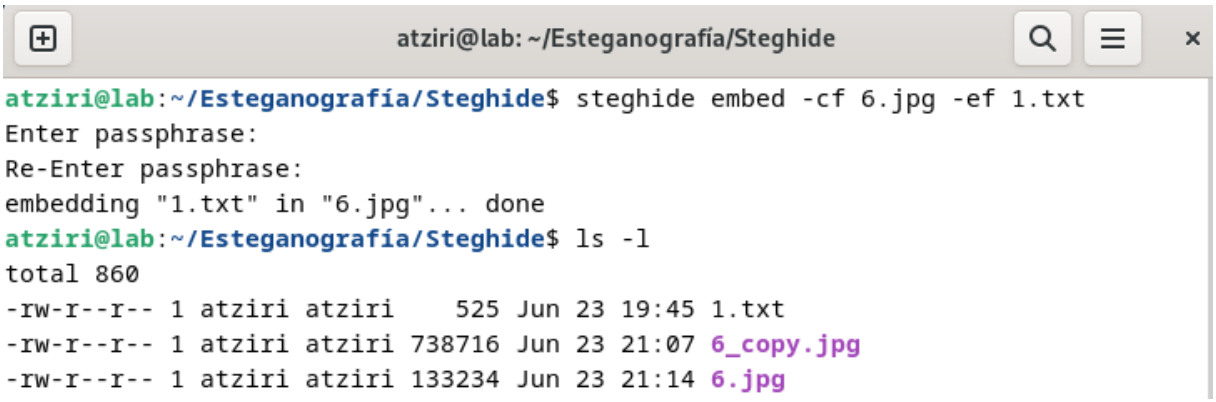


Figura N° 9. 3: Ocultar la información en una imagen con Steghide.

Como se muestra en la Figura N° 9.3, el archivo al que se le ocultó la información fue al archivo *6.jpg*, por lo cual el valor de su tamaño cambia.

En el siguiente recuadro, coloque una captura en donde se observe que se esconde el archivo la diferencia en cuanto a tamaño, tal como en la Figura N°9.3.

Actividad 9.2: Ocultar un archivo con Steghide.

Copie la imagen con el archivo oculto al directorio *Stenography*. Colóquese en dicho directorio.

Para extraer el archivo oculto de la imagen con Steghide, ejecute: «*steghide extract -sf imagen_con_información_oculta*», tal como se muestra en la Figura 9.4:

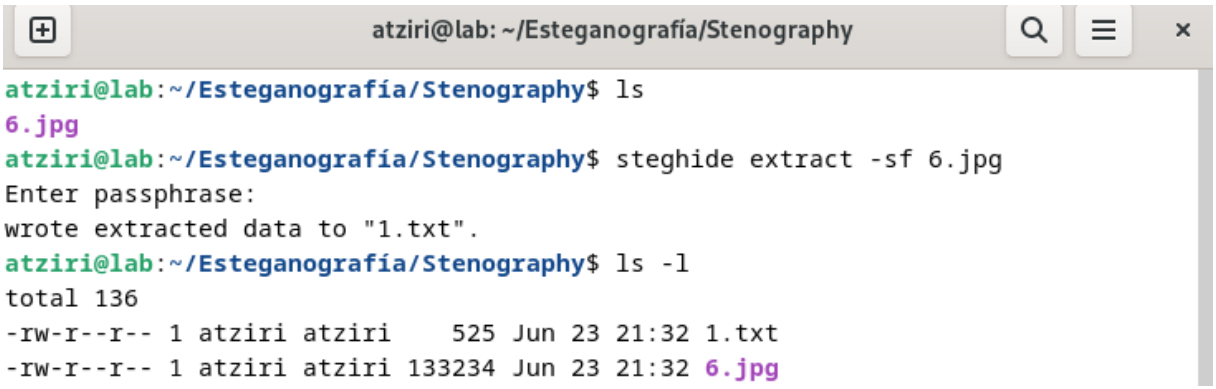


Figura N° 9. 4: Obtener información oculta en imagen con Steghide.

En el siguiente recuadro, coloque una captura en donde se muestre el proceso para extraer el archivo de la imagen con el uso de Steghide.

Actividad 9.2: Obtener información oculta en imagen con Steghide.

ACTIVIDAD 9.3: Stegosuite (GeekforGeeks, s.f.) a diferencia de Steghide, este tiene interfaz gráfica, para abrirla puede buscarla, tal como inicia la terminal, verá algo como lo mostrado en la Figura N° 9.5:

Figura N° 9. 5: Inicio de Stegosuite.

Al iniciarlo, verá una pantalla como lo mostrado en la Figura N° 9.6:

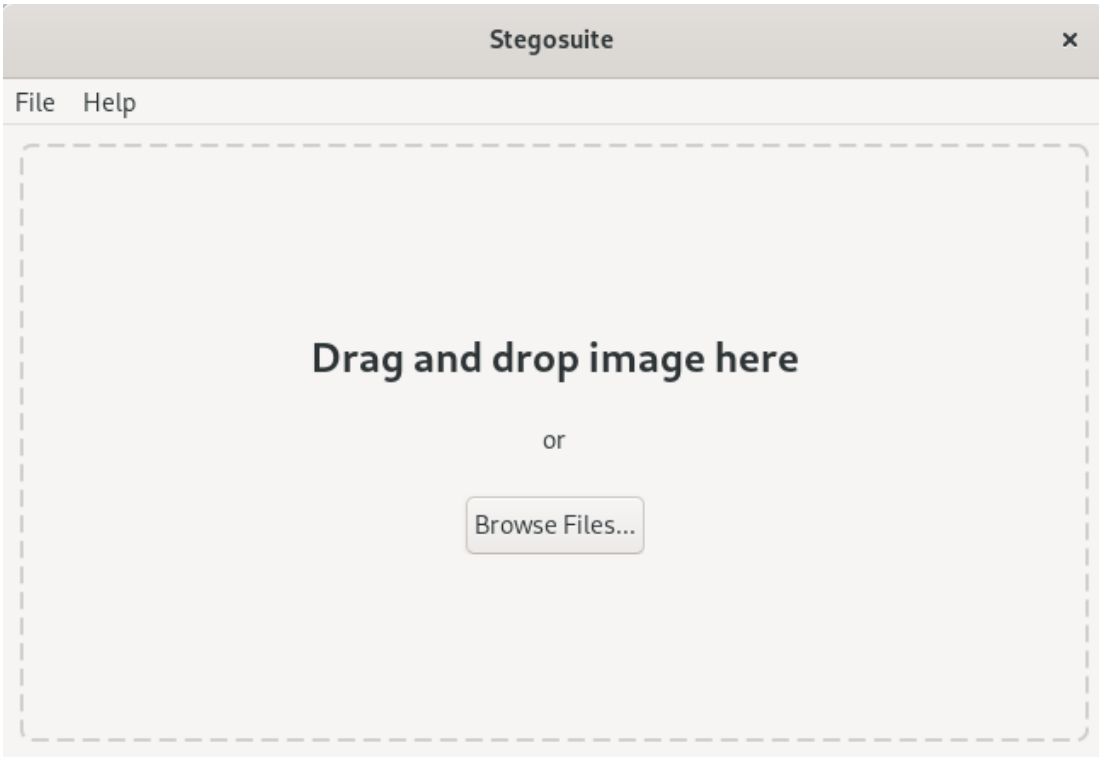


Figura N° 9. 6: Ventana de inicio Stegosuite.

Stegosuite oculta la información a resguardar en imágenes del tipo JPG, GNP, BMP Y GIF. Permite ocultar un mensaje o desde uno a varios archivos.

Del mismo modo que con Steghide, copie a su directorio Stegosuite una imagen y un archivo o PDF de su preferencia. Recuerde cambiar al dueño de los archivos para evitar algún error.

El ejemplo para el directorio Stegosuite se muestra en la Figura N° 9.7:

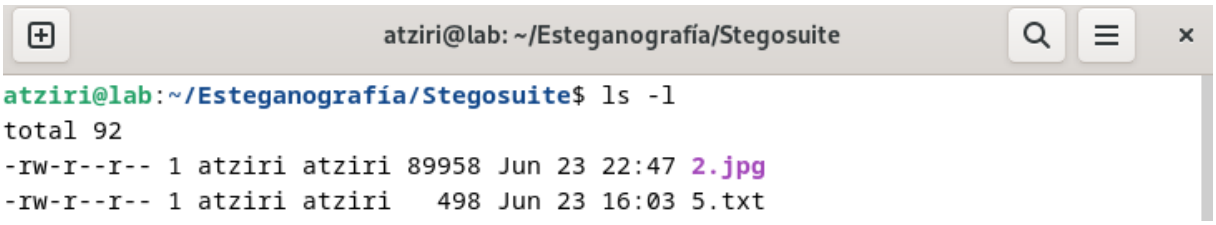


Figura N° 9. 7: Directorio ejemplo para Stegosuite.

Teniendo los archivos seleccionados, regrese a la ventana de inicio de Stegosuite y de clic en *Browse Files* esto le permitirá seleccionar una imagen, en este caso se selecciona la imagen que se copió en el directorio *Stegosuite*.

Posteriormente, en el apartado que dice *o embedded files*, dé clic derecho y abrirá un menú con dos opciones, tal como se muestra en la Figura N° 9.8.

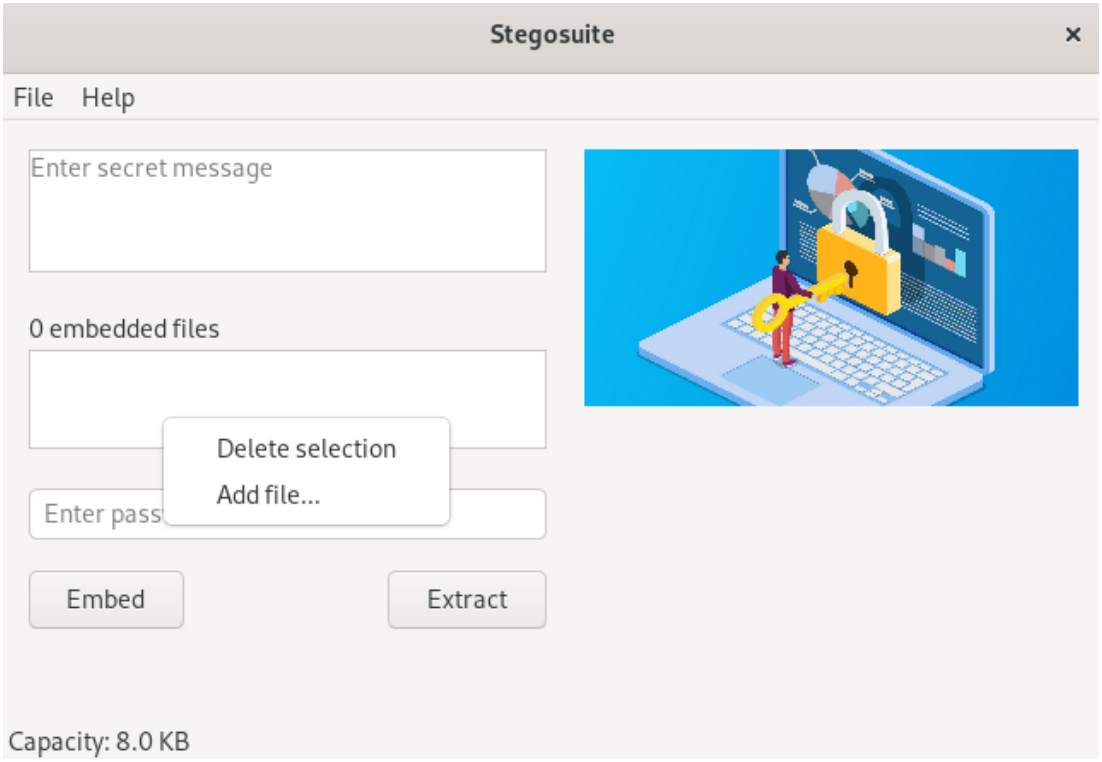


Figura N° 9. 8: Agregar archivo a esconder en la imagen.

Dé clic en *Add file*, busque el archivo a guardar en este caso es el mismo que se colocó en el directorio Stegosuite. Por último, coloque una contraseña en este caso la contraseña si se muestra en claro. Antes de ocultar la información debe ver algo como lo parecido en la Figura N° 9.9.

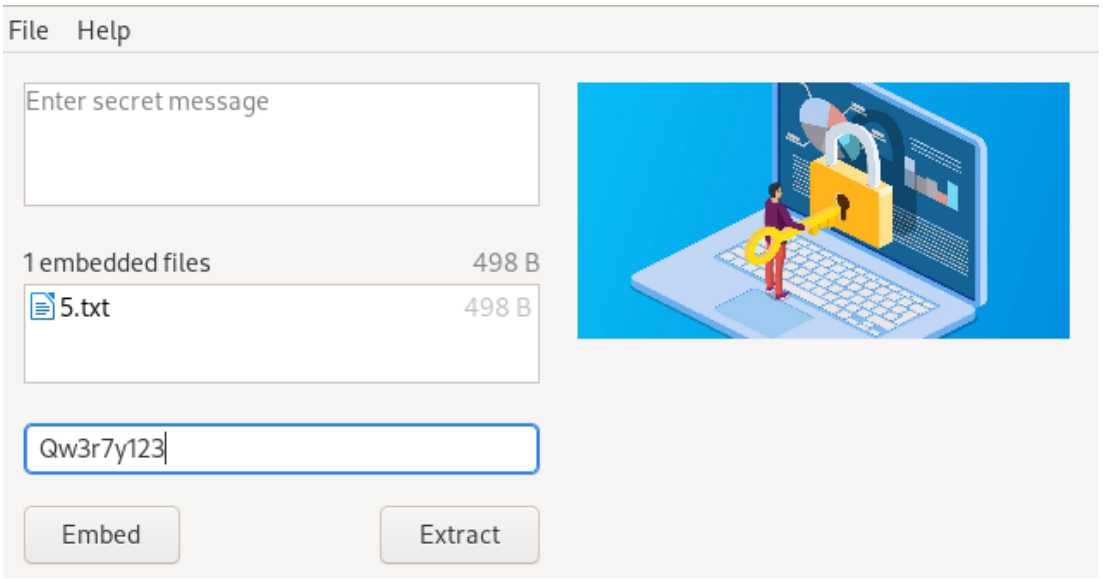


Figura N° 9. 9: Selección de archivo a ocultar y contraseña.

Finalmente dé clic en *Embed*. Al finalizar en la parte inferior indica en donde se guardó la imagen, a diferencia de Steghide en Stegosuite se guarda en un archivo de imagen distinto, tal como se muestra en la Figura N° 9.10.

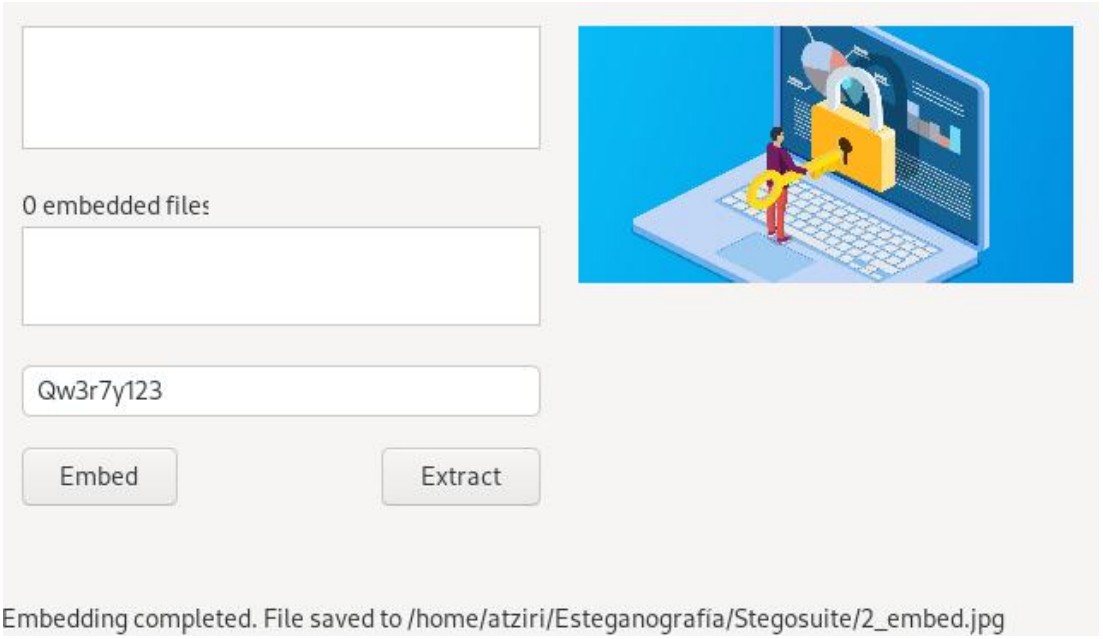


Figura N° 9. 10: Finaliza el proceso de ocultar información en Stegosuite.

El directorio de Stegosuite se muestra en la Figura N° 9.11:

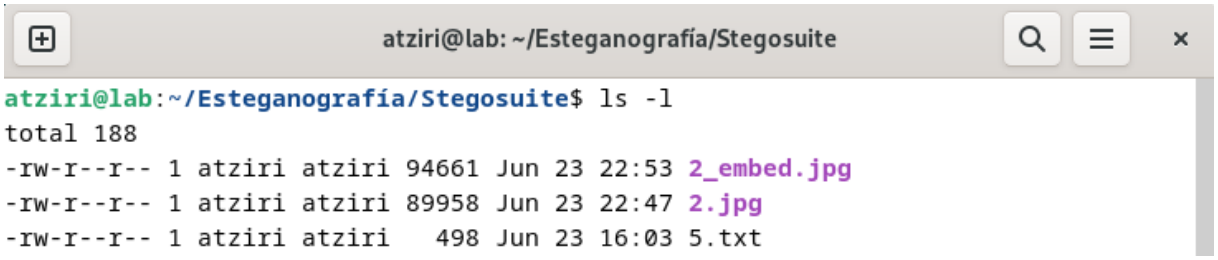


Figura N° 9. 11: Directorio Stegosuite con imagen de archivo oculto.

En el siguiente recuadro, coloque una captura donde se visualice su directorio Stegosuite, tal como en la Figura N° 9.11.

Actividad 9.3: Directorio Stegosuite con imagen de archivo oculto.

Copie la imagen con el archivo oculto al directorio Stenography.

Para obtener el archivo oculto, abra Stegosuite, y en la ventana principal (Véase Figura N° 9.6) dé clic en *Browse Files* y busque el archivo correspondiente en el directorio Steganography.

Coloque la contraseña y dé clic en *Extract.* Al finalizar, en la parte inferior se muestra dónde se guardó el archivo, y además lo muestra en el apartado de *Embedded files*, tal como se muestra en la Figura N° 9.12.

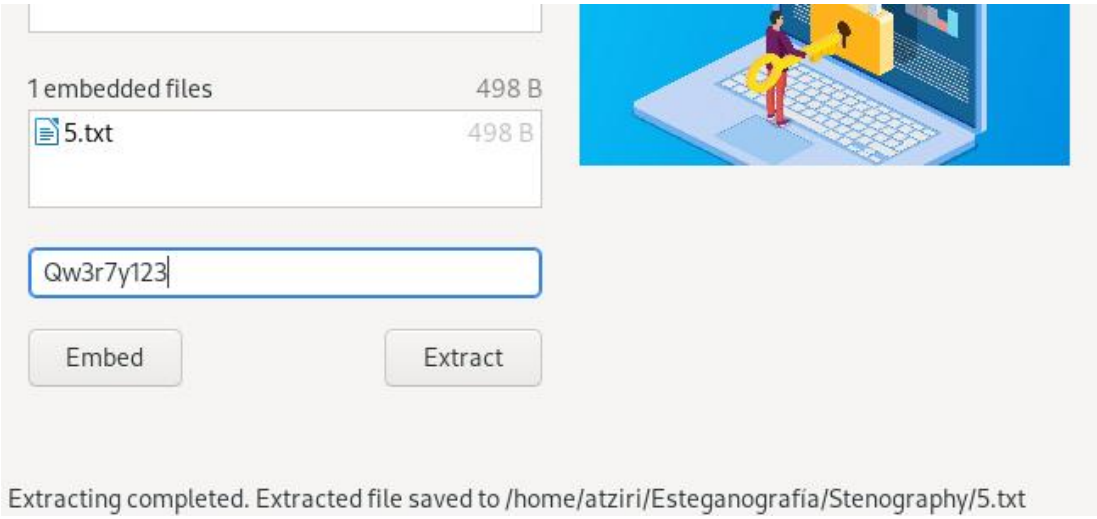


Figura N° 9. 12: Extracción del archivo mediante Stegosuite.

En el siguiente recuadro coloque una captura en donde se muestre la extracción del archivo mediante el uso de Stegosuite, tal como se mostró en la Figura N° 9.12.

Actividad 9.2: Extracción del archivo mediante Stegosuite.

ACTIVIDAD 9.4: Cat, uso más común es para la visualización de archivos, o bien, para la concatenación de archivo de texto mediante el redireccionamiento, sin embargo, en esta ocasión se le dará uso para ocultar un archivo. (Rock, D., 2014)

Del mismo modo, del directorio /Esteganografía copie una imagen y un archivo de texto o PDF a su directorio Cat. Su directorio se debe ver tal como en la Figura N° 9.13.

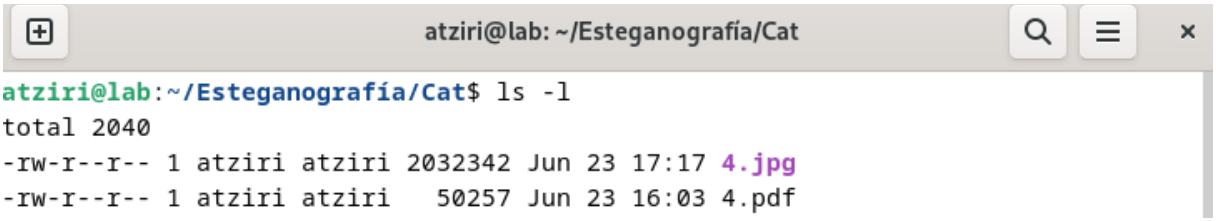


Figura N° 9. 13: Directorio de trabajo Cat.

Además de *cat*, se usará el comando *zip*, el proceso es: *comprimir* el archivo a ocultar en la imagen, en el caso del ejemplo, se va a comprimir el archivo *4.pdf*, y para agregar seguridad, se va a colocar contraseña, para eso, ejecute: «*zip -e nombre_archivo_comprimido archivo_a_comprimir*», tal como se muestra en la Figura N° 9.14:

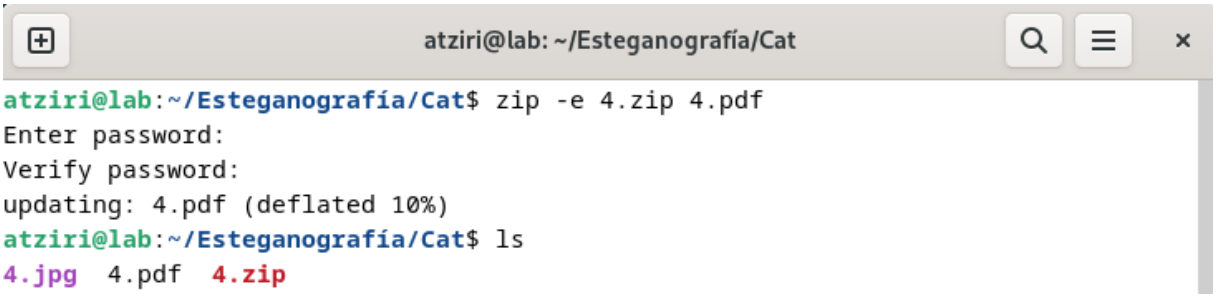


Figura N° 9. 14: Creación de archivo zip.

Posteriormente, se usa *cat* para direccionar el contenido de la imagen y el archivo comprimido a un nuevo archivo, en este caso llamado *cat_graphy.jpg*. Es importante tener presente que el archivo en donde se va a copiar debe tener la misma extensión, en este caso, el archivo es *jpg* por lo tanto el archivo *cat_graphy* también es *jpg*. Lo descrito se muestra en la Figura N° 9.15.

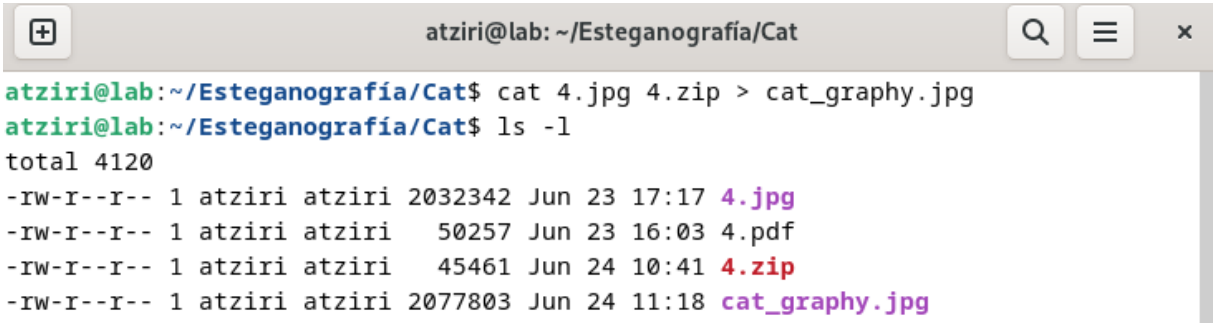


Figura N° 9. 15: Creación del archivo con la imagen y el archivo zip.

En el siguiente recuadro, coloque una captura en donde se muestre el proceso descrito en la Figura N° 9.14:

Actividad 9.4: Creación del archivo con la imagen y el archivo zip.

Copie la imagen creada a su archivo Stenography.

Colóquese en el directorio Stenography. Para extraer el archivo, debe cambiar la extensión de su imagen, en el caso del ejemplo de *jpg* a *zip*. Después de realizar el cambio, debe extraer el archivo, con el comando «*unzip nombre_archivo_comprimido*», al ejecutarlo le mostrará un mensaje indicando que el archivo tiene bytes de más, pero que el proceso continuará, coloque la contraseña, el archivo se descomprimirá. Lo descrito se muestra en la Figura N° 9.16.

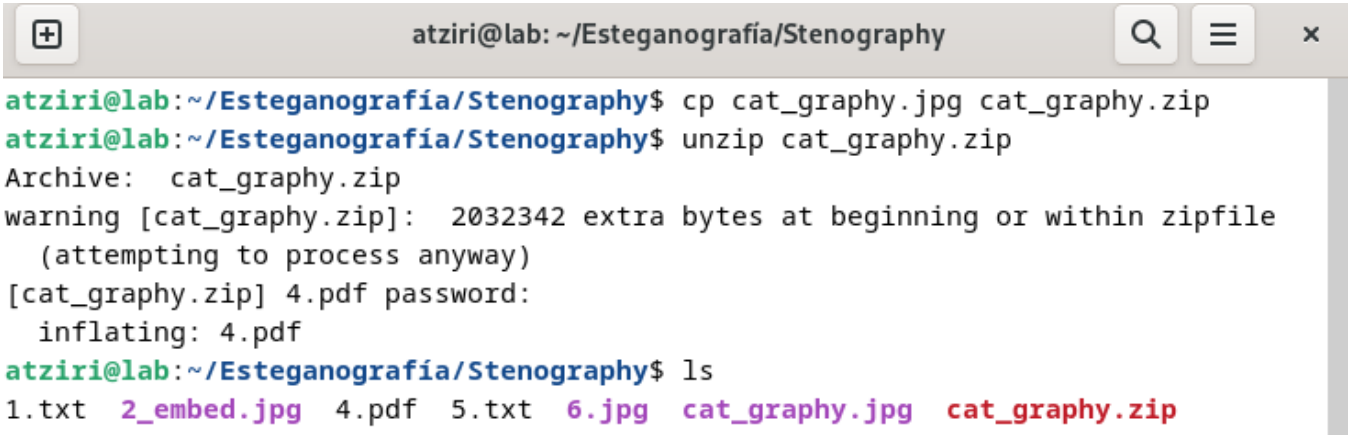


Figura N° 9. 16: Cambio de extensión del archivo y descompresión para obtener el archivo.

Como se observa, el archivo descomprimido se guardó con el nombre 4.pdf.

En el siguiente recuadro, coloque una captura en donde se muestre el proceso descrito en la Figura N° 9.16.

Actividad 9.4: Cambio de extensión del archivo y descompresión para obtener el archivo.

Algo característico de la esteganografía es que a simple vista no se ve diferencia alguna entre la imagen original y la imagen que ha sido sometida a este proceso, tal como se muestra en la Figura N° 9.17.

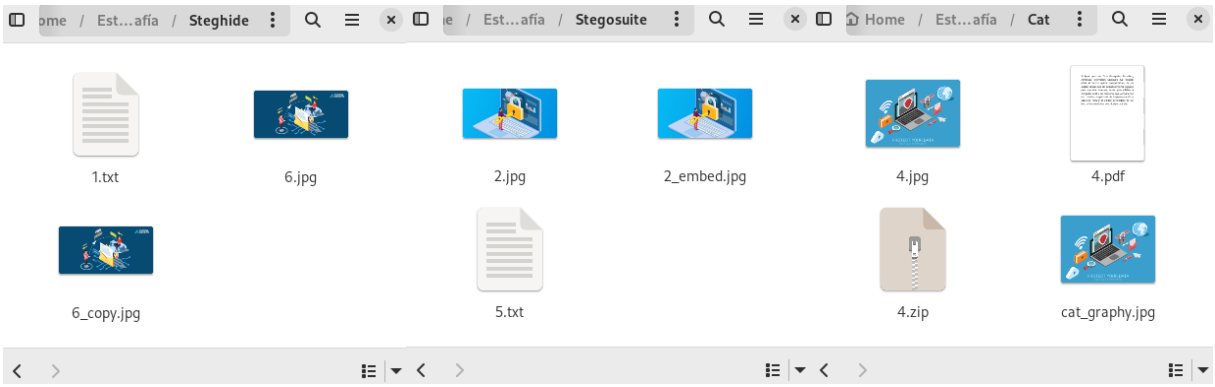


Figura N° 9. 17: Imágenes originales e imágenes sometidas a esteganografía.

Una forma de observar esta diferencia es mediante el tamaño de las imágenes, como se observó anteriormente en las Figuras N° 9.3, 9.11 y 9.14, los tamaños de las imágenes que habían sido sometidas aumentaron en comparación a las originales.

Otra forma de apreciar esa diferencia es mediante el uso de funciones hash, ya que estas serán distintas, recuerde que las funciones hash al mínimo cambio, el resultado del hash cambia completamente.

ACTIVIDAD 9.5: Como se mencionó en la introducción, PGP se creó en un inicio para el cifrado de correo electrónico. En la actualidad existen distintos servicios que apoyan en el cifrado de correo. Esta actividad se va a trabajar en parejas, dicha actividad consiste en realizar una investigación sobre algún servicio que permita el cifrado de correo.

En el siguiente recuadro, coloquen el nombre del servicio que eligieron, el porqué de la elección, una breve descripción de éste y cómo se instala o crea una cuenta.

Actividad 9.5: Explicación de servicio elegido para cifrado de correo.

Realicen la instalación del servicio que eligieron, y envíense mutuamente por correo cifrado alguna de las imágenes que sometieron a esteganografía, es decir de la **Actividad 9.2, 9.3 o 9.4**, para el envío de la contraseña, colóquenla en el propio correo, de manera que **NO** se vea en claro. Como ejemplo se muestra el uso de Mailvelope, un servicio para cifrar correo, su uso se realiza instalando una extensión en el navegador, creando el par de claves y funciona para distintos servicios de correo.

En el correo debe indicar a su compañera/compañero qué es lo que debe realizar para obtener la contraseña en claro y de este modo obtener el mensaje oculto.

En la Figura N° 9.18 se observa cómo tanto Atziri como Balam reciben por correo la imagen correspondiente.

Figura N° 9. 18: Uso de Mailvelope para enviar correo cifrado.

En el siguiente recuadro coloquen una captura en donde se muestre que recibieron el correo, tal como en la Figura N° 9.17:

Actividad 9.5: Uso de algún servicio para el cifrado de correo.

En los siguientes recuadros coloquen donde corresponda, la obtención del mensaje oculto:

Actividad 9.5: Obtención del mensaje oculto enviado de 'Atziri' a 'Balam'.

Actividad 9.5: Obtención del mensaje oculto enviado de 'Balam' a 'Atziri'.

Conclusiones:

1. ¿Cuál es la principal diferencia entre Esteganografía y Criptografía?

2. ¿Qué método de los vistos para ocultar información considera el más robusto? ¿Por qué?

3. ¿Considera que es útil hacer uso de correos cifrados? ¿Sí o no y por qué?

4. ¿Qué tan factible es que ustedes hagan uso del correo cifrado en su día a día?

5. Comentarios o conclusiones adicionales.

UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

ACTIVIDADES PRÁCTICAS: CRIPTOGRAFÍA

10. BLOCKCHAIN

DOCENTE

BLOCKCHAIN

Objetivo

Conocer los principales procesos de consenso con los que trabaja blockchain. En la práctica se darán a conocer los protocolos de consenso *Proof of Work* que es el protocolo inicial de la blockchain donde destacan los *mineros* y el protocolo *Proof of Stake*, que viene a ser para muchos el sustituto del protocolo anterior y cambian los mineros por los *validadores*.

Justificación

Blockchain se ha vuelto muy popular desde el surgimiento de las criptomonedas, dado que, con su uso se evitan los intermediarios pues son los involucrados quienes aceptan los bloques, es decir, es la misma comunidad que conforma la blockchain la que acepta el ingreso del siguiente bloque.

Por otro lado, la seguridad que ofrece al ser una cadena de datos hace que cambiar un bloque resulte imposible pues su encadenación se debe al ingreso del hash del bloque anterior en el bloque a agregar.

Introducción

Blockchain, o conocido como *cadena de bloques* se compone justamente de bloques con información, la cual debe ser validada y al ser verificada es agregada a la cadena.

Un protocolo de consenso es la manera en que los participantes se ponen de acuerdo para la validación de los bloques y que estos, puedan ser agregados a la cadena.

Existen principalmente dos protocolos de consenso, Proof of Work *(Prueba de trabajo)* y Proof of Stake *(Prueba de participación)*, en donde la primera, es la utilizada por la mayor parte de las criptomonedas, entre ellas: Bitcoin, Etherium y Monero.

Proof of Work, consiste en resolver un problema computacional de gran dificultad, es por eso que este protocolo necesita de equipos con muchos recursos computacionales y por lo tanto de alto valor económico. Los participantes son conocidos como *mineros*, ya que ellos son los que realizan la solución del problema, y para ello es que se vuelve necesario agregar un bloque a la vez cabe mencionar que quien agrega el bloque recibe una recompensa. El problema de este protocolo es el gran consumo energético que lleva contar con equipos que trabajen hasta encontrar la solución al problema dado.

Las cadenas de bloques contienen además de la información del bloque, la cadena hash del bloque anterior, es por eso por lo que se vuelve difícil poder romper el bloque y de aquí su seguridad. La función mayormente utilizada es la correspondiente al hash **SHA256**.

Por otro lado, Proof of Stake, tiene un método distinto, ya que sus participantes son conocidos como *validadores* y en lugar de resolver un problema computacional para tener la oportunidad de agregar un bloque nuevo, deben contar con la mayor cantidad de criptomonedas de la que se esté trabajando, esas monedas las ponen en un *Stake* que viene a ser algo parecido a su apuesta, y esta queda guardada, ya que no se puede mover hasta que termine la ronda. Si el validador es elegido, debe verificar el bloque y solo en ese caso se agrega a la cadena. En caso de verificar un bloque corrupto, el validador tiene un castigo, y puede perder parte o todo su stake.

Equipo y material necesario

Máquina Virtual Debian, para montarla en su equipo puede seguir el «Montaje de máquina virtual», el cual es un manual de Instalación MV, en el que se encuentran las instrucciones tanto para VMware como para VirtualBox. Así mismo puede consultar el Anexo D que contiene ejemplos de cómo debe colocar sus respuestas en la práctica.

Desarrollo

Recuerde que debe hacer uso el usuario indicado en la Actividad 4.1 de la Práctica N° 4 «Advanced Encryption Standard».

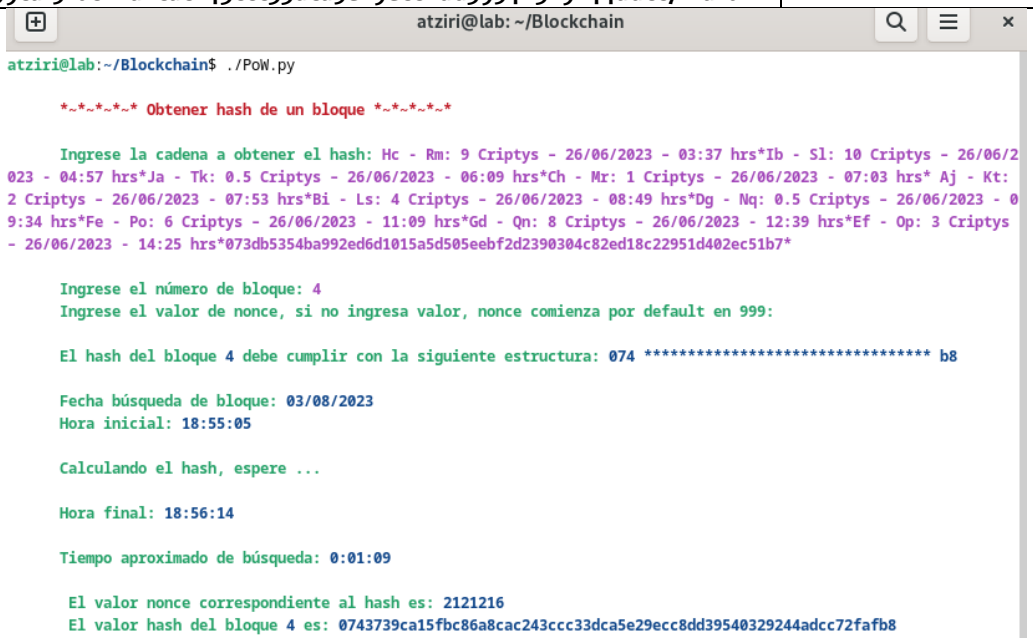
ACTIVIDAD 10.1: Se realizará la simulación de Proof of Work (Bit2Me Academy, s.f.), en donde a continuación se le muestra la tarea que debe realizar de modo que pueda agregar el siguiente bloque.

En /Criptografia/Blockchain se encuentra un archivo llamado *Proof-of-Work_Actividad1.pdf* que tiene ya la cadena de bloques formada, y la siguiente cadena a agregar, así como la manera en la que trabaja la criptomoneda.

Siga las instrucciones que el documento le indica para la realización de un script, recuerde debe comentar todo su script y hacerlo llegar a su docente como se le indique.

En la Figura N° 10.1 se muestra un ejemplo de cómo debe llenar los recuadros, con los datos del cuarto bloque.

Hash:	Valor nonce:
0743739ca15fbc86a8cac243ccc33dca5e29ecc8dd39540329244adcc72fafb8	**2121216**



Captura

Figura N° 10. 1: Llenado de recuadros para la actividad 10.1.

En los siguientes recuadros coloque la captura adecuada a la ejecución de su script, el valor nonce o la cadena hash según corresponda de acuerdo con el bloque tal como se mostró en la Figura N° 10.1

Quinto bloque:

Hash:	Valor nonce:
Captura	

Actividad 10.1: Datos quinto bloque.

Sexto bloque:

Hash:	Valor nonce:
Captura	

Actividad 10.1: Datos sexto bloque.

Séptimo bloque:

Hash:	Valor nonce:
Captura	

Actividad 10.1: Datos séptimo bloque.

Octavo bloque:

Hash:	Valor nonce:
Captura	

Actividad 10.1: Datos octavo bloque.

Noveno bloque:

Hash:	Valor nonce:
Captura	

Actividad 10.1: Datos noveno bloque.

Décimo bloque:

Hash:	Valor nonce:
Captura	

Actividad 10.1: Datos décimo bloque.

ACTIVIDAD 10.2: En el protocolo Proof Of Work, como observó en la actividad anterior, los mineros buscan la manera de realizar la tarea y de esta manera tener una recompensa, pero, hay un paso importante a realizar después de haber encontrado en el caso de la criptomoneda Cripty, el valor de nonce y el valor de hash correspondiente, recuerde que este hash debe ser SHA256.

Cuando el minero ha resuelto la tarea, debe indicar a los demás mineros que ya encontró el valor de NONCE y ellos deben validar que los datos sean correctos para así agregar el bloque a la cadena.

Esta actividad se realizará entre al menos 2 integrantes, simulando que son una red de blockchain y son mineros buscando la recompensa.

Deben crear un canal de comunicación, bien pueden realizarlo mediante algún método visto en las prácticas anteriores o de la manera que ustedes prefieran, bien puede ser un chat en alguna plataforma o una videollamada para que todo sea llevado en tiempo real.

Lo que van a realizar es: cada participante con la utilización de su script va a buscar el valor del siguiente bloque, el primero que lo encuentre lo va a decir en el medio de comunicación dando los datos necesarios para que los demás mineros puedan corroborar. Si el minero acertó, se lleva la recompensa.

Tomen en cuenta el tiempo que les puede llevar la actividad, considerando el tiempo que les llevó realizar la actividad anterior.

En caso de ser necesario, realice las modificaciones pertinentes a su script.

Algo a tomar en cuenta, es que ahora los bloques, cuentan con un campo de datos más, el valor de la recompensa. Los ejemplos se muestran para la búsqueda del cuarto bloque con la recompensa, el bloque está constituido tal como se muestra en la Figura N° 10.2

Hc - Rm: 9 Criptys – 26/06/2023 – 03:37 hrs*	Bi - Ls: 4 Criptys – 26/06/2023 - 08:49 hrs*
Ib - Sl: 10 Criptys – 26/06/2023 - 04:57 hrs*	Dg - Nq: 0.5 Criptys – 26/06/2023 - 09:34 hrs*
Ja - Tk: 0.5 Criptys – 26/06/2023 - 06:09 hrs*	Fe - Po: 6 Criptys – 26/06/2023 - 11:09 hrs*
Ch - Mr: 1 Criptys – 26/06/2023 - 07:03 hrs*	Gd - Qn: 8 Criptys – 26/06/2023 - 12:39 hrs*
Aj - Kt: 2 Criptys – 26/06/2023 - 07:53 hrs*	Ef - Op: 3 Criptys – 26/06/2023 - 14:25 hrs*
073db5354ba992ed6d1015a5d505eebf2d2390304c82ed18c22951d402ec51b7*	
Recompensa: 10 Criptys*	
NONCE*	
074.. …… **b8**	

Figura N° 10. 2: Estructura del bloque con recompensa.

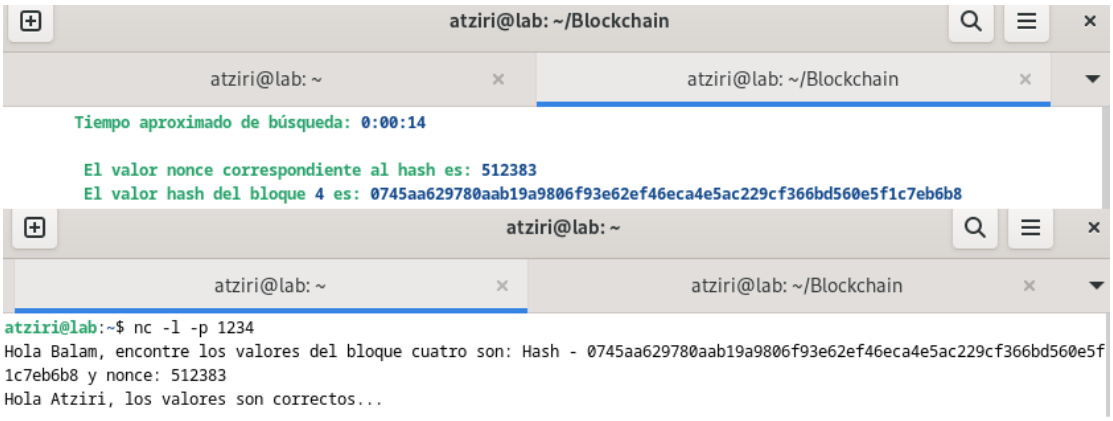
Los bloques completos de datos para esta actividad con su recompensa, más ejemplos, están en el documento *Proof-of-Work_Actividad2.pdf* en el mismo directorio de la actividad anterior.

En la Figura N° 10.3 se muestra un ejemplo de cómo debe llenar los recuadros, con los datos del cuarto bloque, la captura debe incluir el fragmento del script donde muestra que encontró los datos del bloque y la parte de la conversación donde se indica quién encontró los datos y quién los comprueba.

Nota: En un entorno real, en blockchain nunca se conoce la identidad del minero, ya que todo se maneja mediante una cartera digital. En caso de que el minero obtenga la recompensa, esta se enviará a su cartera digital.

Sin embargo, por fines académicos y para poder identificar quién está realizando la actividad, se solicita que se indique el nombre.

Hash: **0745aa629780aab19a9806f93e62ef46eca4e5ac229cf366bd560e5f1c7eb6b8**	Valor nonce: **512383**	Nombre del minero(a) ganador(a): **Atziri**



Captura

Figura N° 10. 3: Llenado de recuadros para la actividad 10.2.

En los siguientes recuadros coloque la captura adecuada a la ejecución de su script, el valor nonce o la cadena hash según corresponda y la captura donde se muestre quién encontró primero los valores de acuerdo con el bloque tal como se mostró en la Figura N° 10.1

Quinto bloque:

Hash:	Valor nonce:	Nombre del minero(a) ganador(a):
Captura		

Actividad 10.2: Datos quinto bloque.

Sexto bloque:

Hash:	Valor nonce:	Nombre del minero(a) ganador(a):
Captura		

Actividad 10.2: Datos sexto bloque.

Séptimo bloque:

Hash:	Valor nonce:	Nombre del minero(a) ganador(a):
Captura		

Actividad 10.2: Datos séptimo bloque.

Octavo bloque:

Hash:	Valor nonce:	Nombre del minero(a) ganador(a):
Captura		

Actividad 10.2: Datos octavo bloque.

Noveno bloque:

Hash:	Valor nonce:	Nombre del minero(a) ganador(a):
Captura		

Actividad 10.2: Datos noveno bloque.

Décimo bloque:

Hash:	Valor nonce:	Nombre del minero(a) ganador(a):
Captura		

Actividad 10.2: Datos décimo bloque.

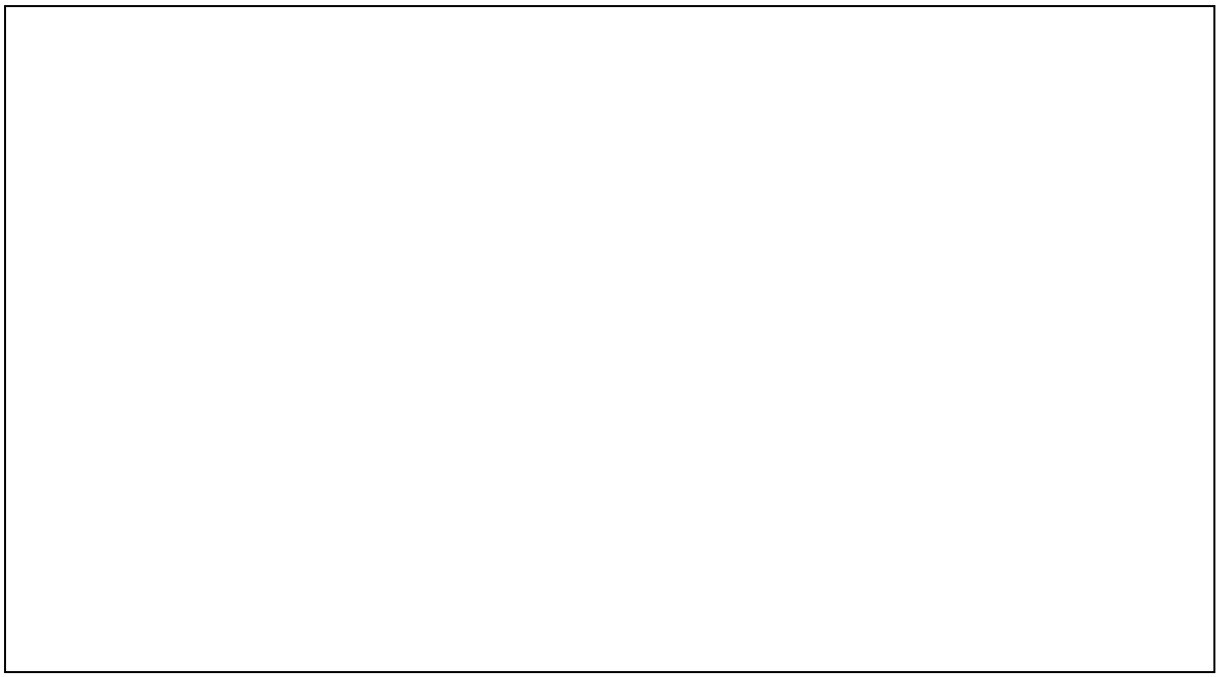
ACTIVIDAD 10.3: En el siguiente recuadro, coloque una breve explicación sobre su entendimiento del protocolo de conceso Proof of Work y liste al menos el nombre de 3 criptomonedas que trabajen con este protocolo.

Actividad 10.3: Explicación Proof of Work.

ACTIVIDAD 10.4: Investigue como trabaja el protocolo de consenso Proof of Stake y de criptomonedas que trabajen sobre este protocolo, cree una criptomoneda que trabaje como lo hace Proof of Stake.

En el siguiente recuadro, coloque brevemente como trabaja el protocolo de consenso Proof of Stake, el nombre de su criptomoneda y su precio en comparación de las criptomonedas que investigó.

Además, liste 3 criptomonedas que trabajen con este protocolo y sus precios.



Actividad 10.4: Explicación Proof of Work.

Conclusiones:

1. ¿Qué es blockchain?

2. ¿Cuál es la principal diferencia entre Proof of Work y Proof of Stake?

3. ¿Cuál y en qué año se dio a conocer la primera criptomoneda?

4. ¿Cuál y en qué año se dio a conocer la primera criptomoneda que trabaja bajo el protocolo de consenso Proof of Stake?

5. Conclusiones adicionales.

UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

ACTIVIDADES PRÁCTICAS: CRIPTOGRAFÍA

11. CONEXIONES REMOTAS

DOCENTE

CONEXIONES REMOTAS

Objetivo

Conocer las distintas maneras de realizar conexiones remotas, haciendo uso de ello para la autenticación mediante SSH a un servidor identificando los elementos de la criptografía.

Justificación

La conexión a servidores de manera remota es una actividad muy realizada en el ámbito laboral y comercial, por ello es importante conocer las distintas maneras en las que se puede realizar la conexión y la seguridad que aporta cada una de ellas.

Introducción

Las conexiones remotas, se realizan regularmente para modificar servicios instalados en los servidores, o incluso para realizar mantenimiento a estos sin necesidad de estar de manera física en donde se encuentren.

La manera más común de realizar las conexiones es mediante el uso del nombre de usuario y su contraseña que el servidor otorga para realizar la conexión, sin embargo, del mismo modo puede ser la más insegura, al hacer uso de diccionarios y fuerza bruta, se podría obtener la contraseña y de este modo realizar la conexión remota.

Existen otras formas como el uso de claves asimétricas o bien, el uso de certificados, de esta manera se otorga mayor seguridad.

En la práctica se hará uso del protocolo Secure Shell (SSH) que es un protocolo destinado a la realización de conexiones entre máquinas en donde toda la información pasa de manera cifrada. Este protocolo trabaja sobre el puerto 22.

Equipo y material necesario

Máquina Virtual Debian, para montarla en su equipo puede seguir el «Montaje de máquina virtual», el cual es un manual de Instalación MV, en el que se encuentran las instrucciones tanto para VMware como para VirtualBox. Así mismo puede consultar el Anexo D que contiene ejemplos de cómo debe colocar sus respuestas en la práctica.

Desarrollo

Recuerde que debe hacer uso del usuario indicado en la Actividad 4.1 de la Práctica N° 4 «Advanced Encryption Standard».

ACTIVIDAD 11.1: La forma más común de realizar la conexión es mediante el usuario y la contraseña.

La forma de realizar la conexión es mediante el siguiente comando: «ssh usuario@ip» donde usuario es el nombre del usuario en el servidor al que se va a conectar, ip se refiere a la IP del servidor.

En la Figura N° 11.1 se muestra la conexión que realiza Balam al servidor de Atziri, mediante el uso de usuario y contraseña. Para esto, Atziri creó un usuario con el nombre balam_ssh y Balam usa la IP del servidor de Atziri.

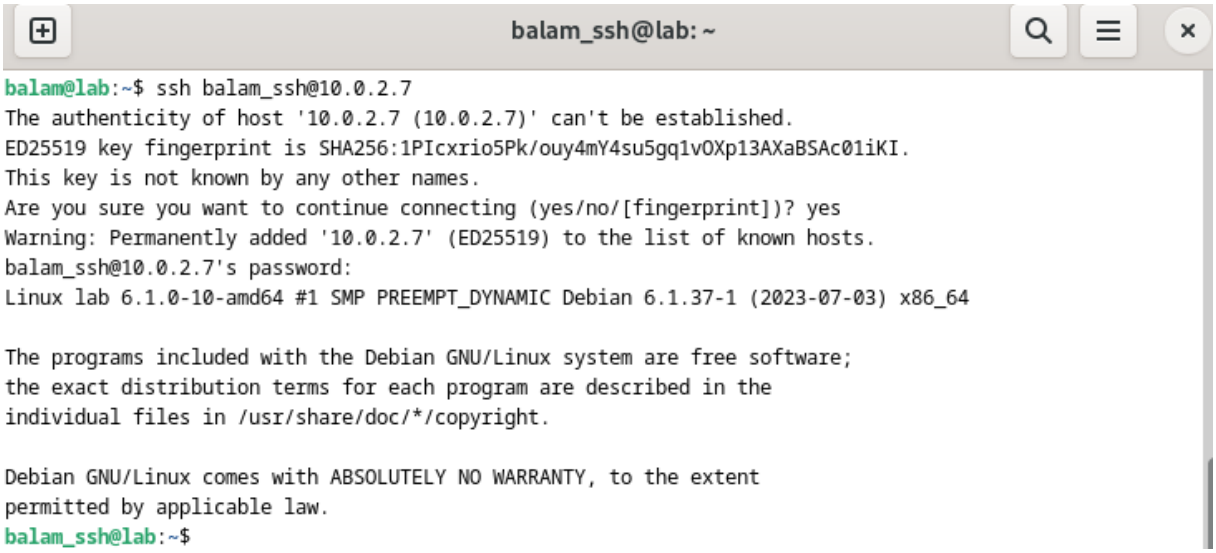


Figura N° 11. 1: Conexión con SSH usando usuario y contraseña.

Dado que la actividad se realizará en parejas y necesitan conocer su IP deben estar conectados a la misma red. En un ambiente real, es suficiente con que el usuario conozca la IP real de su servidor, sin embargo, estando en un ambiente académico, lo ideal es que ambos participantes se encuentren conectados a una misma red.

ACTIVIDAD 11.2: En esta actividad se realizará la autenticación SSH usando una clave pública. Esta actividad se seguirá trabajando en parejas, no olvide indicar en el recuadro de Integrante(s) el nombre de usted y de la persona con quien trabaja en el equipo.

Pasos para quien tome el rol de cliente. En este caso el cliente será el usuario identificado como Balam, usando el usuario balam_ssh. Coloque el nombre de quien será el cliente:

Para el cliente:

Debe generar un par de claves, una pública y una privada para poder realizar la conexión mediante SSH, para eso:

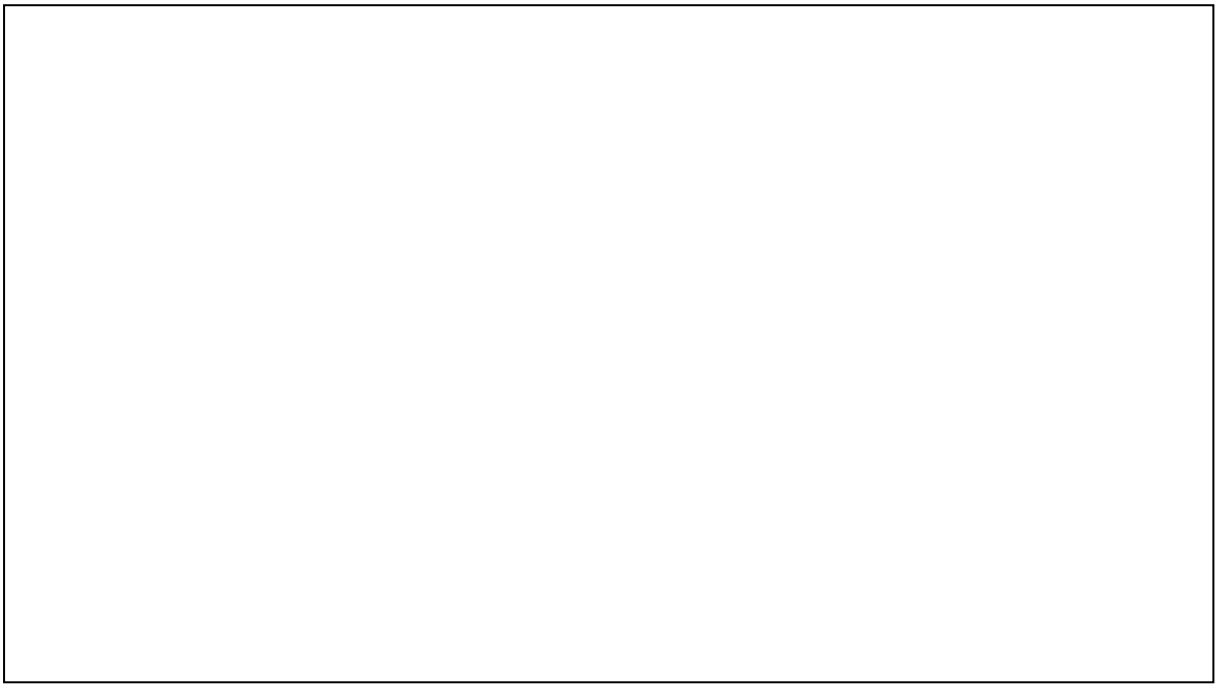
1. Verifique que en el directorio home de su usuario se encuentre el directorio *.ssh*, si no se encuentra, créelo.
2. Posicionándose en el directorio *.ssh*, ejecute: «*ssh-keygen -t rsa -b 4096 -C conexion_ssh_NOMBRE -f NOMBRE_claves*», cambie la palabra NOMBRE, por sus iniciales. Del comando: **-t** se refiere al tipo de clave, en este caso RSA, **-b** el tamaño, en este caso de 4096 bits, **-C** se usa para indicar un comentario y **-f** indica el nombre de las claves, o bien la ruta completa en donde se desea guardar las claves, en este caso como se quieren guardar en el directorio *.ssh* solo basta con colocar el nombre de guardado.

   Al ejecutar el comando, le pedirá que confirme que se guarden sus claves en el directorio *.ssh*, oprima la tecla Enter, posteriormente le pedirá que ingrese una contraseña, ingrésela y confírmela, esta es para proteger su clave privada. Finalmente muestra la huella digital[19]de su clave. La ejecución de los pasos se muestra en la Figura N° 11.2:

Figura N° 11. 2: Creación de claves.

19 Huella Digital (Fingerprint): Secuencia de caracteres lo suficientemente larga como para que sea única. La huella digital se emplea para asegurar la autenticidad de una clave. La huella digital es el valor hash de un archivo.

Coloque una captura de pantalla en donde se observe que creó sus claves, tal como se muestra en la Figura N° 11.2:



Actividad N° 11.2: Creación de claves.

Revise el directorio .ssh, verifique que tenga dos archivos uno llamado *NOMBRE_claves*, que es su clave privada, y un archivo más llamado *NOMBRE_claves.pub* que es su clave pública, esta es la que debe compartir con el servidor.

ACTIVIDAD 11.3: Comparta su clave pública con el servidor, para esto, debe conocer la IP del servidor, y ambos deben encontrarse en la misma red. Coloque la dirección IP del servidor:

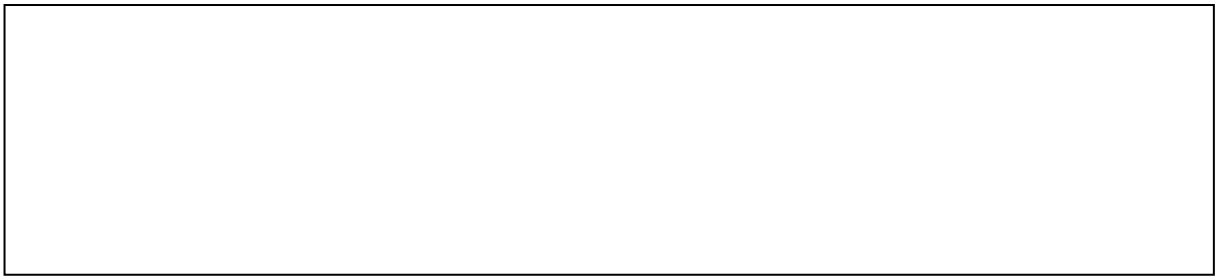
Coloque el nombre del alumno(a) que será el servidor:

Para el servidor: Antes de enviar la clave, el servidor debe verificar que exista un usuario para el cliente, en este caso, *Atziri* es el servidor, y ella debe comprobar que exista el usuario balam_ssh en el servidor. ¿Cómo verificaría qué existe el usuario? ¿En qué archivo se muestran los usuarios del sistema? En el siguiente recuadro escriba sus respuestas:

Actividad 11.3: Verificar datos en el servidor.

Coloque una captura de pantalla en donde se muestre que existe un usuario para el cliente, esto lo debe realizar el integrante del equipo que sea servidor.

Actividad 11.3: Verificar datos en el servidor.



En caso de que el cliente no conozca su nombre de usuario y contraseña para ingresar al servidor, compártalo.

Para el cliente: Confirmando que tiene acceso al servidor mediante usuario y contraseña (Actividad 11.1), en el siguiente recuadro describa con qué comando puede verificar que el servidor se encuentre disponible y por tanto realizar la conexión exitosa:

Actividad 11.3: Verificar conexión con el servidor.

Copie su clave pública al servidor, para esto ejecute lo siguiente: «*ssh-copy-id -i clave_pública usuario@IP_servidor*» tal como se muestra en la Figura N° 11.3, para copiar la clave, le pedirá la contraseña del usuario creado para usted en el servidor.

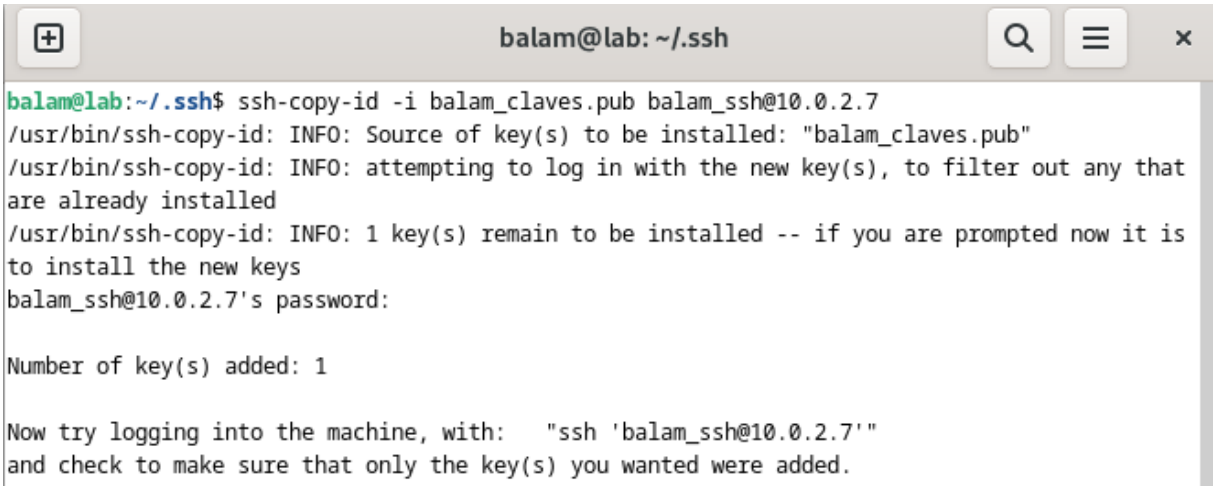


Figura N° 11. 3: Copiar clave pública al servidor.

En el siguiente recuadro, coloque una captura de pantalla en donde se visualice que copió su clave pública tal como se muestra en la Figura N° 11.3:

Actividad 11.3: Copiar clave pública al servidor.

ACTIVIDAD 11.4: Realice las configuraciones de seguridad necesarias.

Para el servidor: El archivo de configuración de SSH es */etc/ssh/sshd_config*, realice una copia de este archivo, ya que si se llega a cometer algún error se tendrá un respaldo disponible. Puede copiarlo en el mismo directorio, y solo agregando al final del nombre del archivo, la fecha en que se realizó la copia.

Hecha la copia del archivo, ingrese a él con su editor de preferencia, verá el archivo tal como se muestra en la Figura N° 11.4

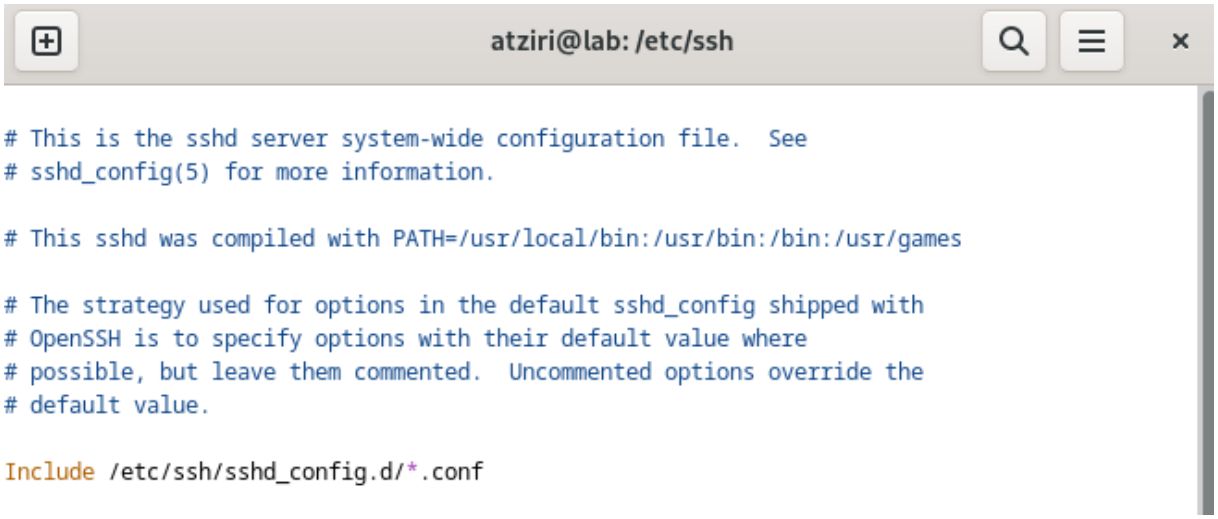


Figura N° 11. 4: Archivo sshd_config.

Localice la línea «*#Port* 22», descomente la línea y cambie el puerto, es decir, cambie el número 22 por uno diferente, solo asegúrese de que este no sea un puerto bien conocido[20]. Recuerde, por defecto SSH trabaja sobre el puerto 22. El cambio de puerto se recomienda para evitar que alguien ajeno acceda al servidor.

Además del cambio de puerto, realice las modificaciones descritas a continuación:

PasswordAuthentication: Se coloca en no, esto evitará conexiones mediante credenciales, es decir, el uso de una contraseña.

PermitRootLogin: Se coloca en no, para evitar ataques de fuerza bruta y que usuarios no autorizados quieran ingresar. De esta manera se bloquea la opción para ingresar ya sea con usuario y contraseña o con claves asimétricas.

PubKeyAuthentication: Se coloca en yes, para permitir el inicio haciendo uso de claves o certificado.

AllowUsers: Para asegurar que los usuarios tienen permiso de ingresar al servidor, se crea una lista blanca, esto quiere decir, que se colocan los nombres de los usuarios permitidos, estos deben encontrarse separados por un espacio.

Lo descrito anteriormente se muestra en la Figura N° 11.5:

20 Puertos bien conocidos: Son puertos que van desde el 1 al 1023, muchos de estos puertos son utilizados por distintos protocolos para realizar las conexiones de sus servicios.

Figura N° 11. 5: Configuraciones necesarias para la autenticación en SSH.

Revise que las modificaciones sean correctas, sobre todo que el nombre del usuario sea correcto para evitar inconvenientes. Guarde el archivo, y ejecute «*sudo service ssh reload*», esto para que se apliquen los cambios que acaba de realizar.

Realizado lo anterior, indique a su cliente que ya puede realizar la conexión mediante SSH usando su clave privada.

Para el cliente: Para ingresar al servidor, posiciónese en el directorio .ssh ejecute lo siguiente: «*ssh -i clave_privada usuario@IP_Servidor -p puerto*» tal como se muestra en la Figura N° 11.6, tome en cuenta que en la Figura se coloca el nombre de la clave hasta que Balam se encuentre en su directorio *.ssh*, si usted se encuentra en un directorio diferente, coloque la ruta a seguir para llegar a su clave privada.

Tenga presente que se cambió el puerto por defecto, por lo que es necesario hacer uso de la bandera **-p**, sin embargo, si se utiliza el puerto por defecto no es necesario hacer uso de dicha bandera.

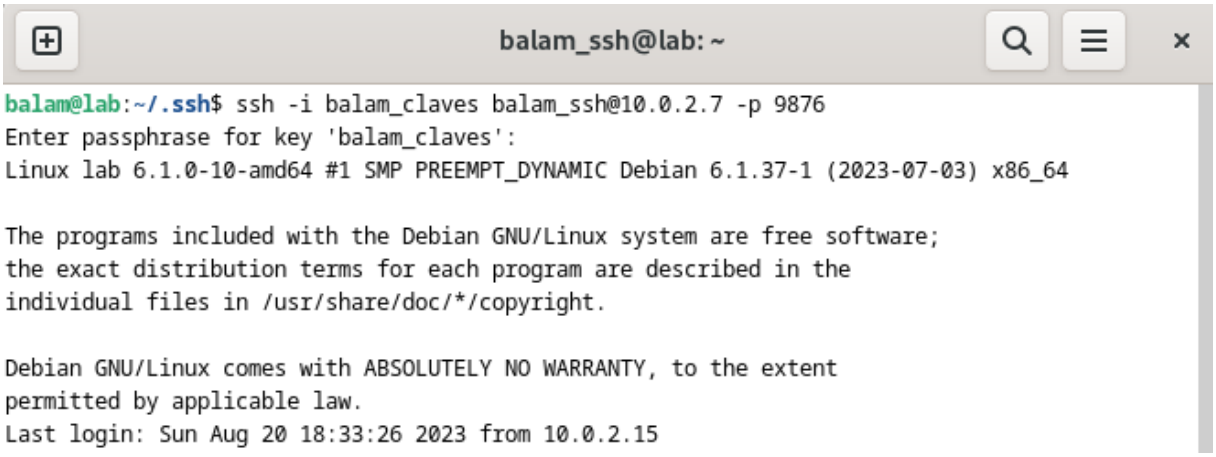


Figura N° 11. 6: Comando para ingresar al servidor mediante SSH usando clave privada.

Al ejecutar el comando se pedirá que coloque el *Passphrase*, se refiere a la contraseña que colocó cuando creó sus claves *(Figura N° 11.2)*, ingrese su clave y de ese modo podrá realizar la conexión.

Coloque una captura de pantalla en donde se observe que ha logrado ingresar a la máquina servidor, como se muestra en la Figura N° 11.6:

Actividad N° 11.4: Cliente se conectó al servidor haciendo uso de su clave privada.

Si intenta ingresar con un usuario distinto, la conexión no será exitosa, tal como se muestra en la Figura N° 11.7, Canek intenta ingresar y se le indica que no se puede conectar con el puerto 22, cuando coloca el puerto, tampoco puede ingresar ya que aún no tiene sus claves y este no se encuentra en el servidor. Además, no se encuentra la lista blanca que se creó en el archivo sshd_config.

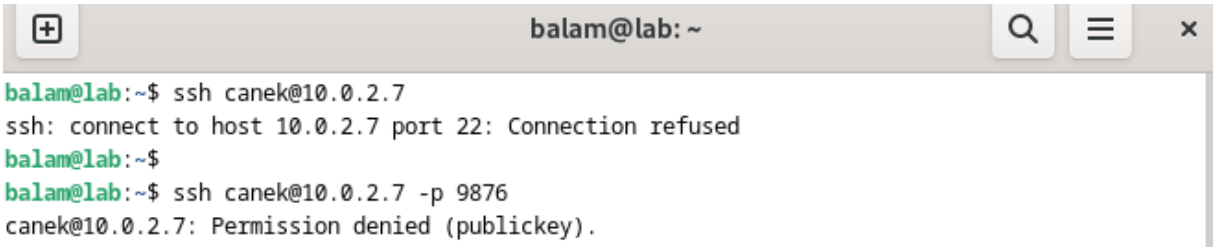


Figura N° 11. 7: Usuarios distintos no pueden realizar la conexión al servidor.

ACTIVIDAD 11.5: En la actividad anterior logró realizar la conexión mediante claves, sin embargo, se necesitan algunos pasos más para lograr la autenticación con un certificado. Para tener un certificado, es necesario que una autoridad certificadora lo firme. (WildUnix, 2016)

Para eso, el servidor será una CA que firmará la clave pública del cliente para lograr la autenticación por certificado.

En esta ocasión el servidor será Balam y el cliente Atziri, se pide que ustedes también hagan el cambio de roles.

Para el servidor: Cree un par de claves pública y privada, tal como lo realizo el cliente *(Figura N° 11.2)* pero en esta ocasión tanto cliente como servidor deben tener su par de claves.

En el caso del ejemplo, Balam será el servidor, él ya cuenta con sus claves.

Para el cliente: Atziri será el cliente, ella tendrá que crear sus claves y hacer llegar a Balam su clave pública de la manera que desee, ya sea por correo cifrado o mediante nc.

Para el servidor: Ahora, firme la clave pública que recibió de parte del cliente, guarde la clave pública en un directorio llamado CERT-Nombre, y cambie *Nombre* por el nombre del cliente.

Para llevar a cabo la firma y obtener el certificado ejecute: «*ssh-keygen -s clave_privada_del_servidor -I ID_del_servidor -V _tiempo_de_vida_para_el_certificado -n usuario clave_pública_del_cliente*», para el tiempo de vida, siempre inicia con el símbolo de más «+» indicando semanas, días. En la Figura N° 11.8 se muestra la firma realizada por Balam para Atziri. Expira en 54 semanas 5 días, es decir su tiempo de validez es de la fecha que se creó: *21 de agosto 2023 al 7 de septiembre de 2024*. El tiempo que sea válido su certificado dependerá de las necesidades que requiera, para el ejercicio se recomienda seguir el mismo tiempo, aproximadamente 1 año.

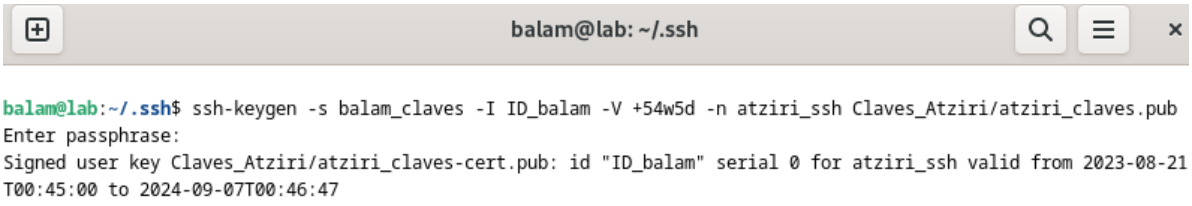


Figura N° 11. 8: Firma de claves por parte del servidor.

Envíe el certificado por correo electrónico cifrado al cliente.

Ahora, haga copia del certificado del cliente al directorio /etc/ssh.

Además de realizar las configuraciones mostradas en la *Figura N° 11.5* debe agregar la siguiente línea:

TrustedUserCAKeys: (sebelk, 2019) Aquí, se debe indicar la ruta en donde se encuentra la clave pública del servidor, ya que esta es la que verifica si el certificado usado para realizar la conexión es válido, tal como se muestra en la Figura N° 11.9

Figura N° 11. 9: Indicar el uso de certificado en sshd_config

Guarde el archivo, y ejecute «*sudo service ssh reload*», esto es para que se guarden los cambios que acaba de realizar.

Para el cliente: Guarde su certificado en el directorio *.ssh*. Haga revisión de su certificado, para esto ejecute: «*ssh-keygen -L -f certificado*», tal como se muestra en la Figura N° 11.10

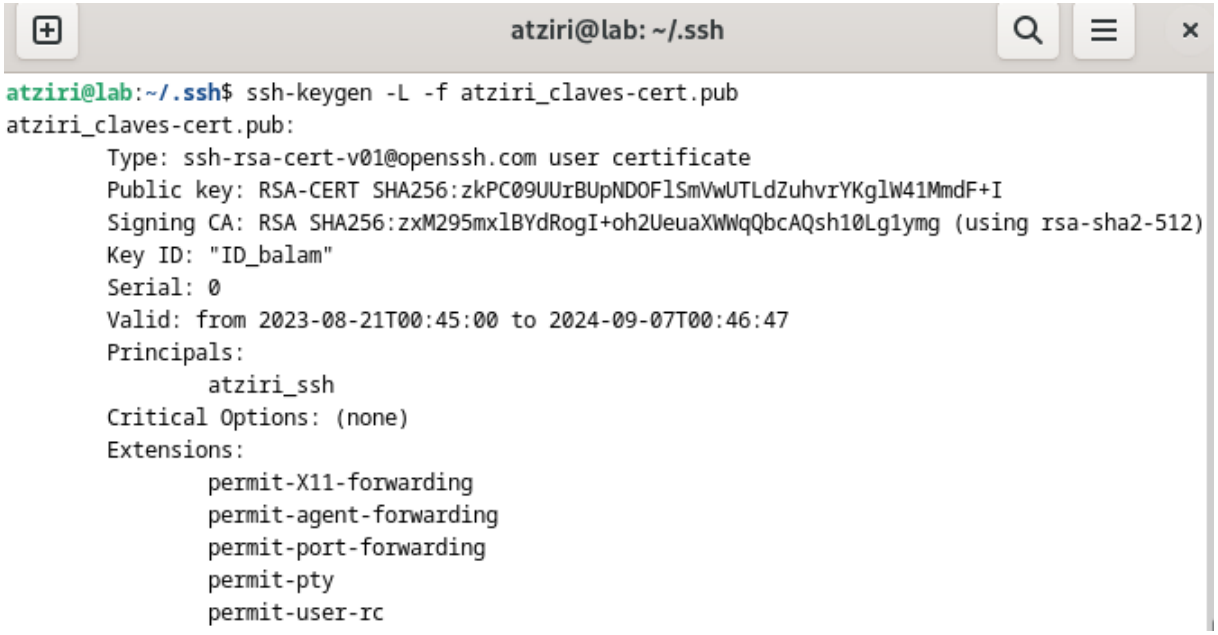


Figura N° 11. 10: Revisión de certificado.

Coloque una captura de pantalla donde se muestren los datos de su certificado, tal como en la Figura N° 11.10.

Actividad N° 11.5: Revisión de certificado.

Por último, realice la conexión al servidor, para esto hará uso de su clave pública y el certificado, ejecute: «*ssh –i clave_privada usuario@IP_Servidor -p puerto*» le pedirá la contraseña de su clave, tal como en la Figura N° 11.6, colóquela y tendrá acceso al servidor, tal como se muestra en la Figura N° 11.11:

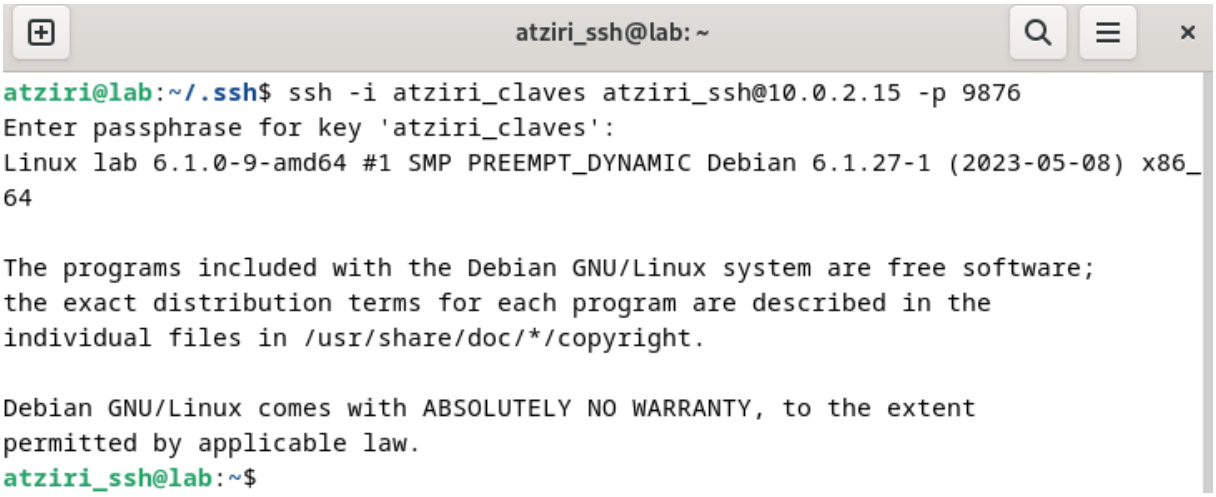


Figura N° 11. 11: Conexión SSH mediante certificado.

Coloque una captura de pantalla donde se muestre que realiza la conexión al servidor mediante certificado, tal como se muestra en la Figura N° 11.11.

Actividad N° 11.5: Ingreso al servidor mediante certificado.

El servidor puede revisar la conexión mediante certificado en el archivo auth.log, localizado en /var/log y se puede verificar con los datos del certificado, tal como se muestra en la Figura N° 11.12:

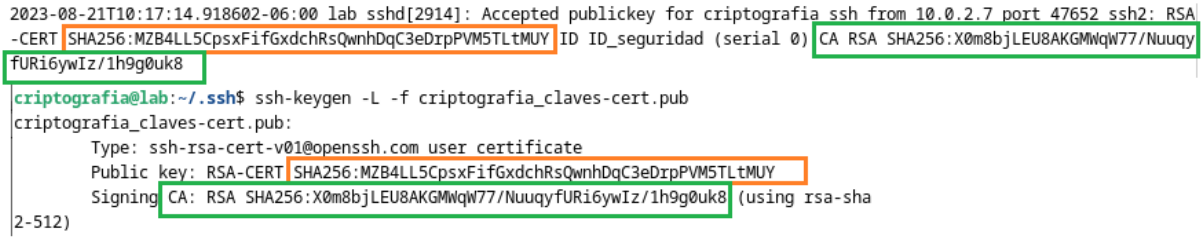


Figura N° 11. 12: Comprobación de conexión mediante certificado.

ACTIVIDAD 11.6: Anteriormente, todas las conexiones se realizaron mediante Linux – Linux, sin embargo, también se pueden realizar de Windows a Linux, haciendo uso de un software llamado PuTTy. (Rocío, G.R.,2023)

Uso de usuario y contraseña: Para esta actividad se hará uso de las dos máquinas virtuales, tanto Linux que será el servidor, como Windows que será el cliente, por lo cual, las conexiones se realizarán desde la máquina Windows mediante el uso de PuTTy.

PuTTy es principalmente un cliente SSH, el cual está disponible para Windows.

Para lograr realizar exitosamente la conexión, en la máquina Linux, debe realizar configuraciones en el archivo */etc/ssh/sshd_config* donde regrese el valor del puerto al *puerto 22*, elimine la línea de *AllowUsers* o bien, ingrese el nombre del usuario de Windows, comente la línea de *PubkeyAuthentication* y en *PasswordAuthentication* coloque *yes*.

En el escritorio de Windows tiene un acceso directo a PuTTY, inicie el servicio, le aparecerá una ventana como la mostrada en la Figura N° 11.13:

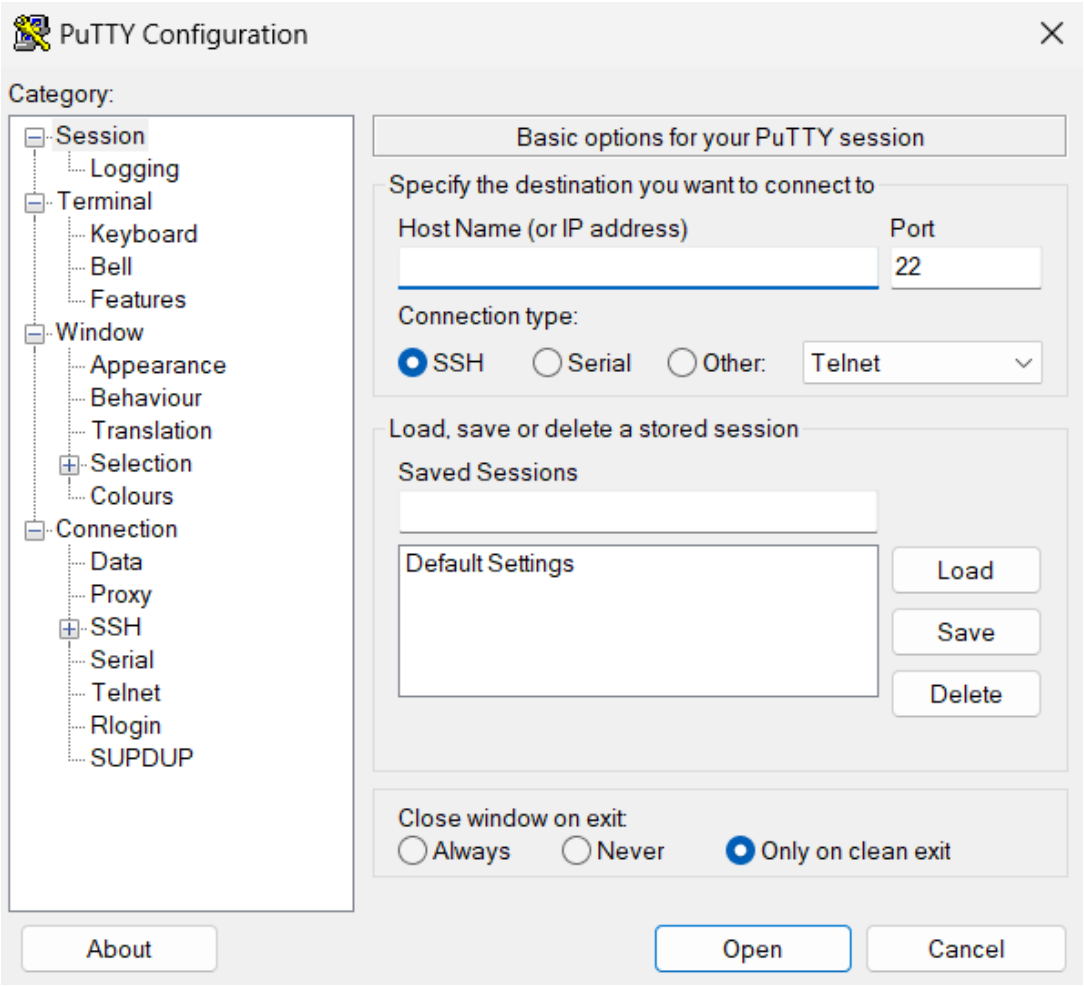


Figura N° 11. 13: Incio de PuTTY.

Verifique que tenga conexión con la máquina servidor, en el siguiente recuadro, coloque la manera en que puede realizar dicha verificación:

Actividad N° 11.6: Verificar conexión a la máquina servidor.

Al haber realizar la verificación, en puTTY, en el recuadro Host Name, coloque el nombre del usuario que le creo su compañero, y la IP de la máquina servidor, es decir de la máquina Linux de su compañera/compañero, tal como se muestra en la Figura N° 11.14:

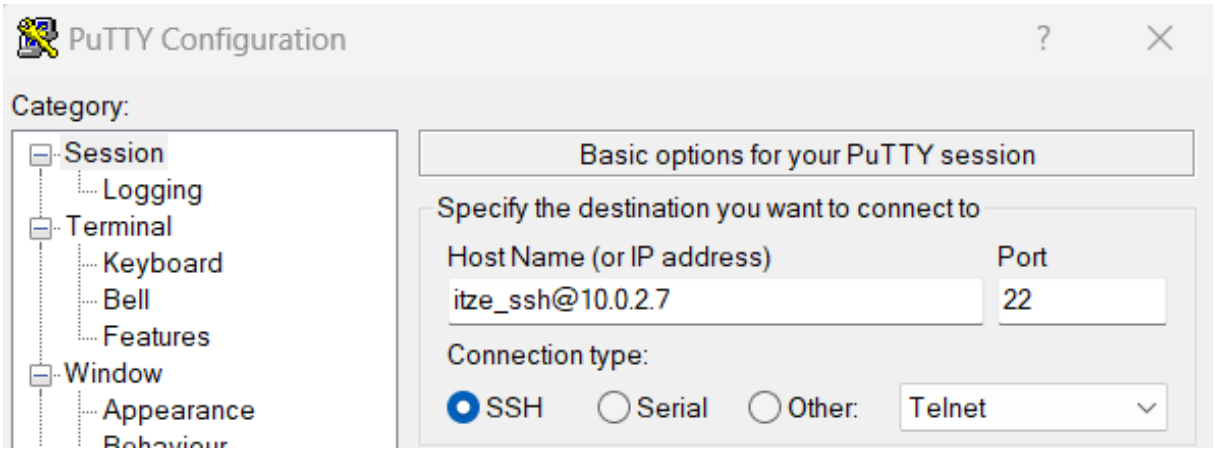


Figura N° 11. 14: Datos para realizar conexión por SSH.

Al terminar de llenar dichos datos, de clic en Open, se iniciará la conexión, esto le abrirá algo parecido a una terminal, en donde se le solicitará la contraseña del usuario, colóquela, y así logrará la conexión, tal como se muestra en la Figura N° 11.15:

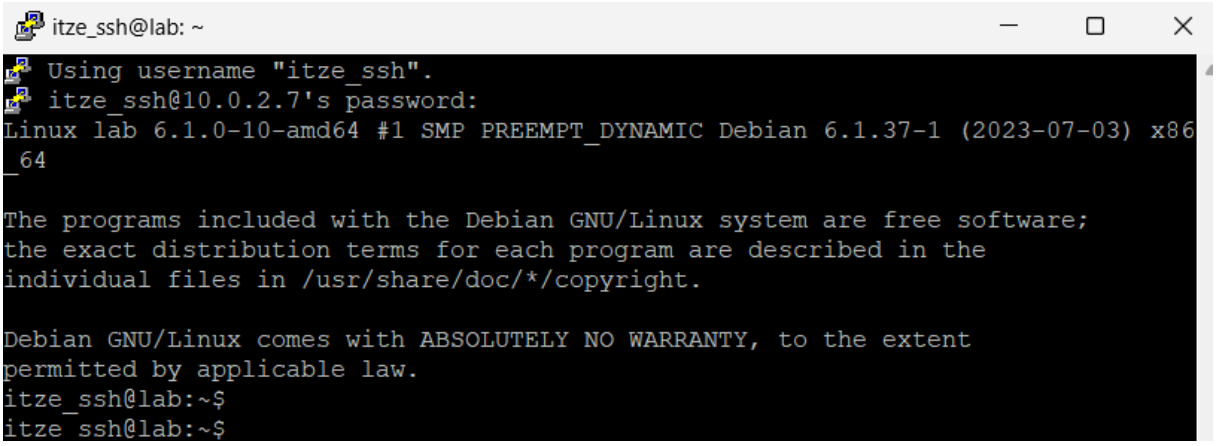
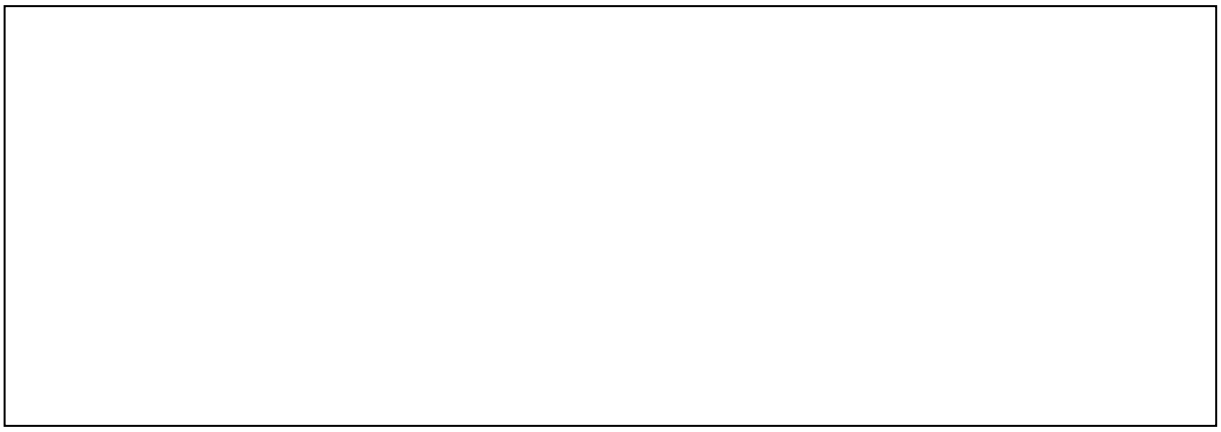


Figura N° 11. 15: Conexión por SSH mediante el uso de PuTTY.

En el siguiente recuadro, coloque una captura de pantalla en donde se muestre que se ha realizado la conexión exitosamente.



Actividad 11.6: Conexión mediante claves.

ACTIVIDAD 11.7: Imagine que se necesita la conexión a su usuario de la misma manera que se realizó anteriormente, solo que ahora no le permite el acceso mediante solo el usuario y la contraseña, y el acceso es específicamente mediante el uso de claves.

En la máquina de Linux, que es a donde quiere ingresar, en la sesión de su usuario, verifique que su usuario tenga el directorio .ssh, por lo cual, posiciónese en el directorio home de su usuario con el comando cd /home/«nombre de usuario» para visualizar qué archivos se encuentran en dicho directorio, ejecute el comando ls -a.

Si el directorio .ssh no se encuentra, debe crearlo con el comando «mkdir .ssh» y finalmente se muestra nuevamente que el directorio ya se encuentra. Si usted si cuenta con este directorio, omita lo anterior.

Dentro del directorio, cree un archivo llamado *authorized_keys* y cambie los permisos del archivo de manera que queden: de lectura y escritura para el dueño, escritura para grupo, y ninguno para otros.

Explique en el siguiente recuadro, cómo realizó lo anterior:

Actividad N° 11.7: Creación de archivo y aplicación de permisos.

Desde la máquina Windows, inicien puTTYgen, lo puede encontrar en el escritorio. PuTTygen creará las claves, al iniciarlo, verán una nueva ventana, donde se mostrarán las opciones para generar las claves, en esta ocasión se dejan los valores por defecto, y se da clic en Generate,

para generar las claves, mueva el cursor en la parte blanca superior, para que las claves se genere, el resultado final debe ser parecido a la Figura N° 11.16, coloque una contraseña, la cual debe recordar, no se pueden recuperar, y salve las claves en donde guste.

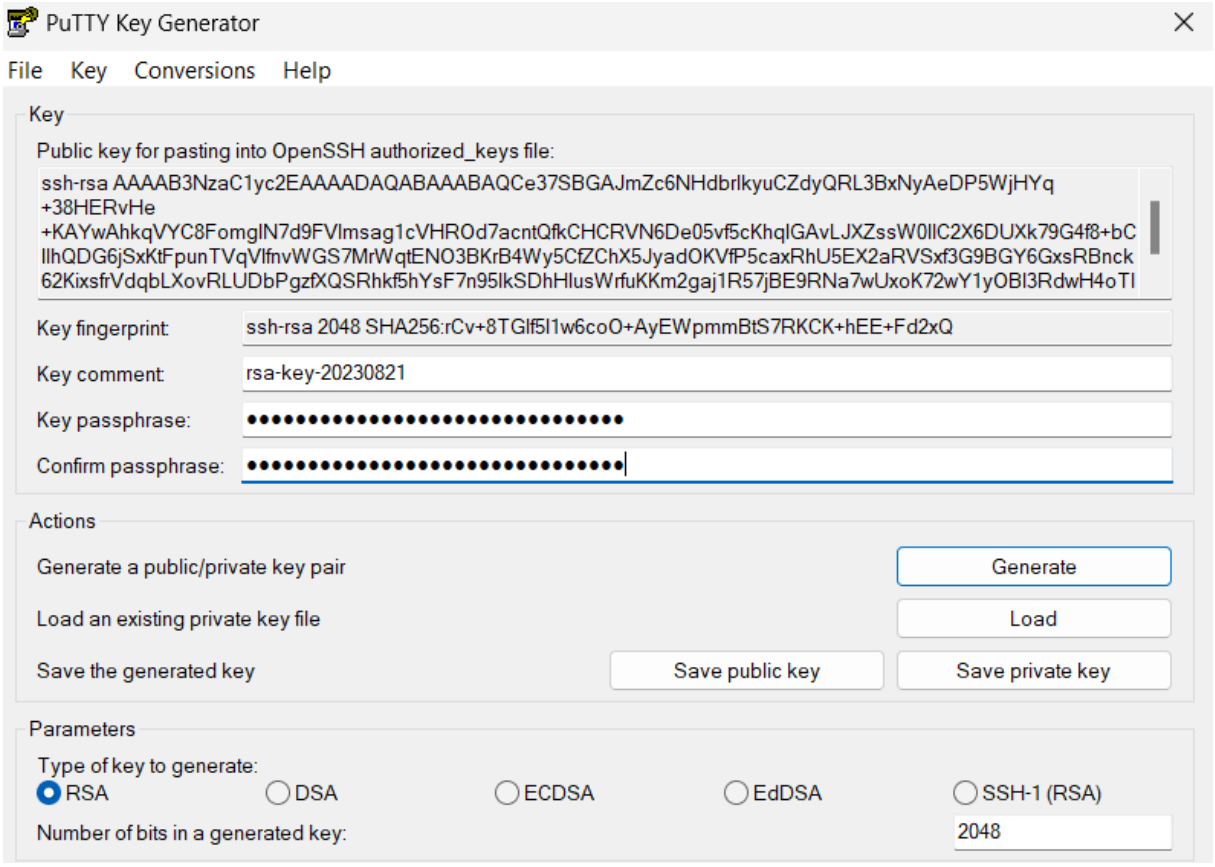


Figura N° 11. 16: Creación de claves con PuTTYgen.

Copien el texto que aparece al inicio, que comienza con ssh-rsa …

El texto copiado, debe colocarse en el archivo authorized_keys, por lo cual debe iniciar sesión con el uso de PuTTY, para que el copiado sea exitoso, edite el archivo con nano, y de clic derecho sobre el espacio del archivo, se copiará. Guarde el archivo, al visualizar el archivo debe verse como en la Figura N° 11.17:

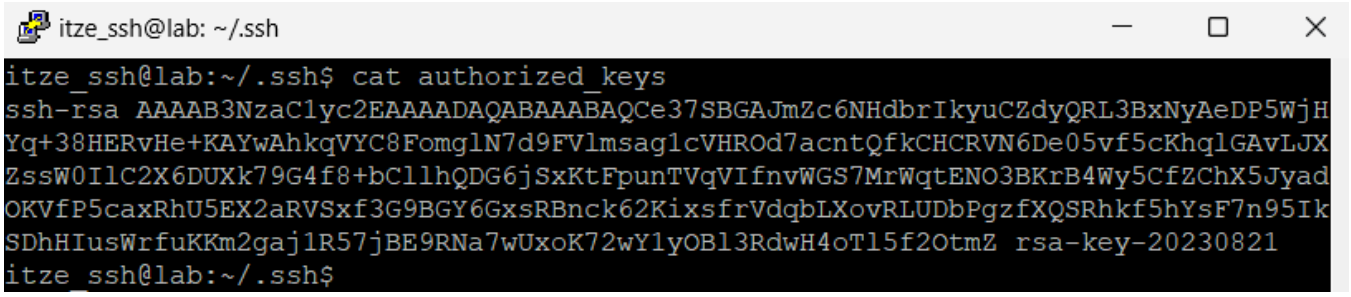


Figura N° 11. 17: Copia de la clave al archivo authorized_keys.

Máquina Linux.

Ahora, en la máquina Linux, hay que realizar las modificaciones pertinentes para establecer la autenticación mediante las claves.

Las modificaciones son las mismas realizadas en la Figura N° 11.2, solo tome en cuenta ahora al usuario que este utilizando para esta actividad.

Realice las modificaciones pertinentes y guarde el archivo, ejecute *«sudo service ssh reload»*, esto para que se guarden los cambios que acaba de realizar.

Describa en el siguiente recuadro cuales fueron las modificaciones realizadas y por qué se realizaron.

Actividad N° 11.7: Modificación del archivo de configuración de SSH.

Con las modificaciones que acaba de realizar, podrá realizar la conexión de Windows a Linux, intente realizar la conexión mediante el uso de usuario y contraseña,

Ingrese una captura de pantalla donde se visualice lo ocurrido

Actividad N° 11.7: Realizando conexión con usuario y contraseña.

Ahora, para realizar la conexión con el uso de claves, debe hacer uso de la clave creada anteriormente, en la ventana de puTTY, llene como realizo anteriormente, el campo de Host Name, después, de clic en el + del apartado que dice SSH, posteriormente en Auth y posteriormente en Certificate en esta ventana se cargará la clave, por lo cual debe dar clic en Browse, para buscar la clave y agregarla, el proceso se muestra en la Figura N° 11.18:

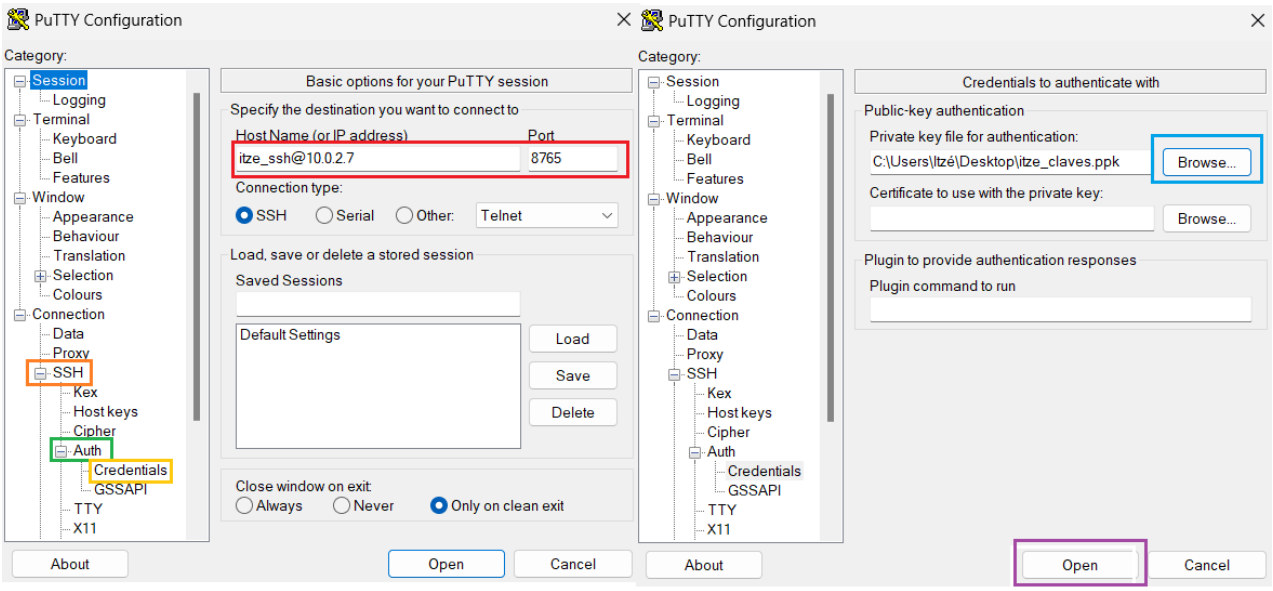


Figura N° 11. 18: Proceso para realizar la conexión mediante claves.

Finalmente, de clic en Open, se abrirá la terminal y le solicitará la contraseña que le coloco a sus claves, colóquela y se realizará la conexión exitosamente.

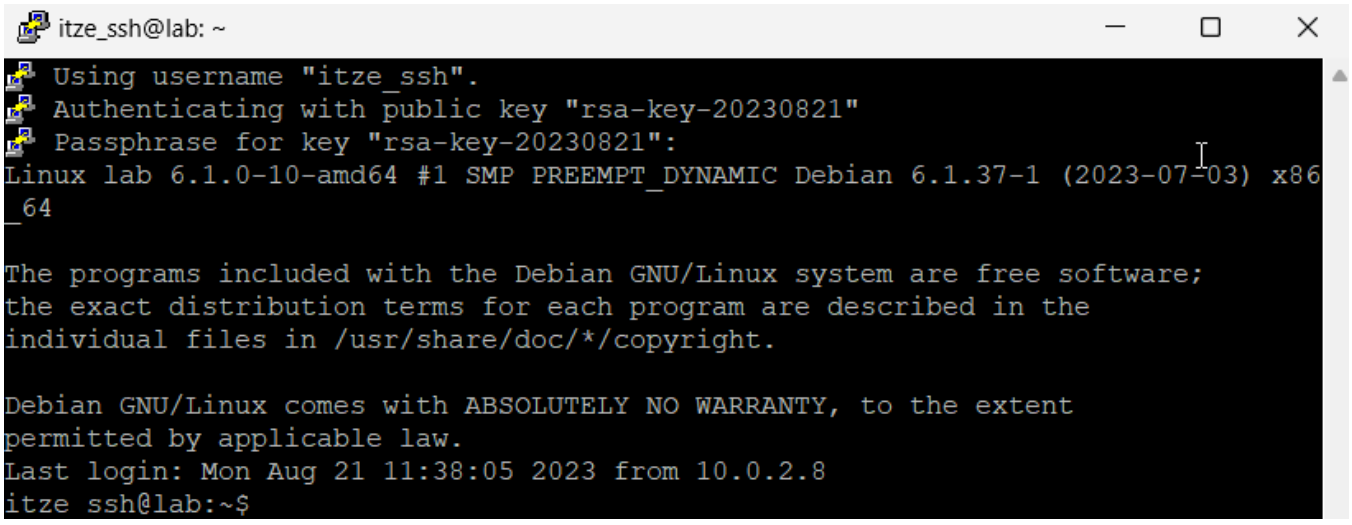


Figura N° 11. 19: Conexión mediante claves.

En el siguiente recuadro coloque una captura donde se muestre la conexión:

Actividad N° 11.7: Conexión mediante el uso de claves.

En Linux, puede checar el archivo auth.log, localizado en el directorio /var/log. El archivo, es un archivo en donde se muestran todos los inicios de sesión o intentos de inicio al servidor, puede

consultarlo ya sea con cat, more o less, El comando completo usando cat es: «sudo cat /var/log/auth.log»

En el siguiente recuadro coloque una captura donde se muestre la conexión en el archivo auth.log tal como en la Figura N° 11.12

Actividad 11.7: Comprobación de conexión mediante certificado.

Conclusiones:

1. ¿Qué es SSH?

2. ¿Qué otros usos tienen los Certificados Digitales?

3. ¿Qué otros controles de seguridad colocaría usted en el archivo sshd_config?

4. ¿Cuál fue el mayor desafío durante la práctica?

5. Comentario y conclusiones adicionales.

UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

ACTIVIDADES PRÁCTICAS: CRIPTOGRAFÍA

PRÁCTICA FINAL

DOCENTE

Práctica Final

Creación de autoridad certificadora y uso del certificado en una página web (Nguyen, J., 2015)

Objetivo

Conocer y elaborar certificados de manera que se identifique una autoridad certificadora raíz y una autoridad certificadora intermedia para utilizar el certificado en servidor apache en una página web.

Especificaciones del proyecto

El proyecto será realiza en equipos de dos o como máximo tres integrantes.

Realizar la documentación de lo realizado para este proyecto. Dicha documentación debe contar con:

a) Portada
b) Índice
c) Introducción
d) Desarrollo
e) Conclusiones personales

Debe incluir capturas de pantalla de cada uno de los puntos realizados, durante el desarrollo del proyecto, cada una de estas debe contar con pie de imagen.

Se debe indicar que alumno(a) es la autoridad certificadora raíz, la autoridad certificadora intermedia y el servidor web, para esto deben usar idealmente tres máquinas virtuales, si esto no es posible, puede usar mínimo dos de ellas.

Debe asignar un nombre a su equipo, dicho nombre y los integrantes del equipo, deben ser enviados en la fecha, hora y por el medio que indique su docente.

Debe hacer usos del usuario creado en la práctica N° 4.

Todos los certificados: raíz, intermedio, servidor y el certificado para verificación de los certificados (concatenación de certificado raíz e intermedio) para clientes y servidores, así como claves públicas deben ser enviados por correo electrónico cifrado entre los integrantes del equipo.

La documentación debe ser enviada a su docente como se le indique.

Introducción

Un certificado digital es esencialmente una clave pública y un identificador, firmados digitalmente por una autoridad certificadora (CA), y su utilidad es demostrar con toda certeza que una clave pública pertenece a un usuario en particular. Evidentemente, la CA debe encargarse de verificar previamente que la clave pública es auténtica. (Universidad Autónoma de Ciudad Juárez, s.f.)

Para emitir el certificado, las CA, validan previamente la identidad del usuario y firman digitalmente el certificado, incorporando su firma digital a éste.

La autoridad certificadora calcula la firma digital del certificado con una función hash, y lo firman con su llave privada.

Los certificados digitales contienen: Llave pública del dueño, nombre, fecha de expiración, nombre del emisor, es decir, la autoridad certificadora que genero el certificado, número de serie, firma digital del emisor.

Los certificados de llave pública permiten asegurar que una llave pública pertenece a la entidad identificada y que dicha identidad posee la correspondiente llave privada.

La confianza que un certificado proporciona respecto a la identidad del usuario depende de qué tan confiable sea la CA y de los mecanismos que la entidad certificada utilice para proteger su llave privada.

Desarrollo

Basándose en la Figura N° pf.1:

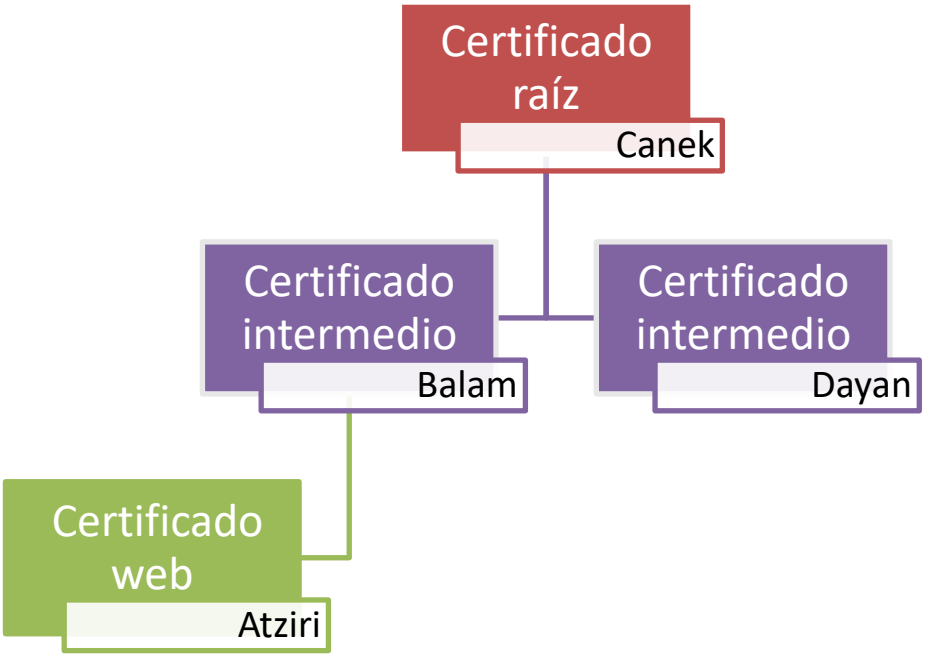


Figura N° pf. 1: Jerarquía para la creación y firma de certificados.

El certificado raíz, autoriza a los certificados intermedios para poder firmar certificados a clientes y servidores. En un ambiente real, lo ideal es que el certificado raíz sea un equipo aislado, sin acceso a la red, y que solo sea utilizado para autorizar a los certificados intermedios para firmar los certificados finales.

Los certificados intermedios, como ya se indicó, son los certificados en los que la CA raíz confía para poder firmar certificados finales, es decir de clientes, en este caso, de un servidor web.

Los certificados finales pueden ser utilizados con distintos fines, como lo son tramites, autenticación de usuarios en SSH, como se vio en la práctica N° 11, o su uso en páginas web.

Basándose en la Figura N° pf.1, deben crear un certificado raíz, un certificado intermedio y un certificado para un servidor web.

Elementos necesarios para certificados raíz e intermedio

Para realizar el certificado raíz y certificado intermedio, realice lo siguiente:

1. Cree un directorio, en el cual guardará los certificados y claves creadas

   Ejemplo: ca_canek, intermedio_balam

2. Dentro del directorio creado en el punto anterior, cree los siguientes directorios:

   certs, newcerts, csr y private

3. Cambie los permisos para el directorio *private* de modo que solo el dueño cuente con todos los permisos, en este caso, el usuario que cree los certificados correspondientes.
4. Cree el archivo *index.txt*, en este se irán numerando los certificados que se firmen.
5. Ejecute *echo 1000 > serial*, esto se utiliza como ID para los certificados firmados.
6. Hagan uso del archivo *openssl.cnf* (para certificado raíz) y *iopenssl.cnf* (para certificado intermedio) estos se encuentran en /.Certificados, visualice y modifique las siguientes líneas según le convenga:
   - dir
   - private_key
   - certificate

7. el archivo correspondiente en el directorio creado en el punto 1.

Creación de claves:

Tanto ca raíz, ca intermedia y certificado para el servidor web, debe crear un par de claves, puede realizarlo ejecutando lo siguiente:

openssl genrsa -aes256 -out private ca_INICIALES.key.pem 4096

Cambie *INICIALES* por las iniciales del usuario que cree el certificado correspondiente.

En caso de las claves para el certificado intermedio cambie ca por intermedia.

En el caso de las claves para el certificado del servidor web, se recomienda omitir *- aes256*, para evitar que cada que entre a su página web, esta pida la contraseña.

En el caso de las claves para el certificado del servidor web, cambie *ca* por el dominio de su página web.

Ejemplo: *www.apc.com_INICIALES.key.pem*

Donde debe cambiar INICIALES por las iniciales del usuario que cree el certificado.

Creación del certificado raíz

Para la creación del certificado raíz, debe posicionarse en el directorio creado en el punto 1 del inciso a, y ejecutar el siguiente comando:

openssl req -config openssl.cnf -key private/ca_INICIALES.key.pem -new -x509 -days 7300 -sha256 -extensions v3_ca -out certs/ca_INICIALES.cert.pem

1. Debe cambiar *INICIALES* por las iniciales del usuario correspondiente.
   - Asegúrese de haber realizado los cambios pertinentes en el archivo openssl.cnf para evitar futuros errores.
   - En la creación del certificado se indican, además del tipo de certificado, los días de validación para este, que en este caso son *7300 días*.
   - Cuando cree el certificado, llene los campos correspondientes, tomando en cuenta las anotaciones.
2. Verifique el certificado creado, para esto, ejecute:

   openssl x509 -noout -text -in certs/ca_INICIALES.cert.pem

   - Recuerde cambiar INICIALES por las iniciales del usuario correspondiente.
   - Al ejecutar este comando visualizará: La validez del certificado, el tipo de clave pública que se utilizó, quien firmo el certificado, por mencionar algunos.
   - Al certificado raíz también se le conoce como certificado autofirmado.

Creación del certificado intermedio

Para la creación del certificado raíz, debe posicionarse en el directorio creado en el punto 1 del inciso a, y ejecutar el siguiente comando:

openssl req -config iopenssl.cnf -new -sha256 -key private/intermedio_INICIALES.key.pem -out csr/intermedio_INICIALES.csr.pem

Tenga cuidado en usar el archivo *iopenssl.cnf*, para el certificado intermedio, y haber realizado las configuraciones correspondientes en dicho archivo para evitar futuros errores.

Recuerde cambiar INICIALES por las iniciales del usuario correspondiente.

El archivo *intermedio_INICIALES.csr.pem*, es el archivo que se debe enviar a la CA raíz, para que lo firme, y realice la creación del certificado valido para realizar firmas.

Recuerde enviar el archivo por correo electrónico cifrado.

La ca raíz, debe firmar el certificado de la ca intermedia, para esto, copie el archivo intermedio_INICIALES.csr.pem en el directorio csr y ejecute:

openssl ca -config openssl.cnf -extensions v3_intermediate_ca -days 3650 -notext -md sha256 -in csr/intermedio_INICIALES.csr.pem -out certs/intermedio_INICIALES.cert.pem

Verifique el certificado de la ca intermedia ejecutando:

openssl x509 -noout -text -in certs/intermedio_INICIALES.cert.pem

Verifique el certificado intermedio con el certificado raíz, esto para validar la confianza del certificado:

openssl verify -CAfile certs/ca_INICIALES.cert.pem certs/intermedia_INICIALES.cert.pem

Si es confiable, tendrá como respuesta un OK

Cree el archivo *ca-chain.cert.pem* para verificar la validez de los certificados que la ca intermedia firme, este archivo crea una cadena de confianza entre la ca raíz y la ca intermedia, para esto concatene los certificados para la ca intermedia y la ca raíz, es decir los archivos *intermedio_INICIALES.cert.pem y ca_INICIALES.cert.pem.*

Cambie los permisos para el archivo *ca-chain.cert.pem*, de modo que tanto el dueño, como los grupos y otros usuarios tengan solo el permiso de lectura.

Comparta el certificado de la ca intermedia *intermedia_INICIALES.cert.pem* y el archivo ca-chain.cert.pem. Para compartirlos, envíelo por correo electrónico cifrado.

Los archivos deben ser guardados en el directorio *certs* del usuario correspondiente a ca intermedia

Creación de certificado para servidor web.

Para crear el certificado, haga uso del archivo /.Certificados/wopenssl.cnf, realice las modificaciones pertinentes, además utilice las claves realizadas en la sección **Creación de Claves**. Ejecute:

openssl req -config wopenssl.cnf -key private dominio_INICIALES.key.pem - new -sha256 -out csr/dominio_INICIALES.csr.pem

Recuerde cambiar INICIALES por las iniciales del usuario correspondiente y dominio por el nombre de su sitio web.

Ahora, comparta el archivo que acaba de crear con la ca intermedia, para que esta lo firme, recuerde enviarlo por correo electrónico cifrado.

La ca intermedia debe firmar el archivo de petición del servidor web para crear el certificado, para lo cual, copie el archivo de petición recibido por correo electrónico al directorio csr. Ejecute:

openssl ca -config iopenssl.cnf -extensions server_cert -days 375 -notext -md sha256 -in csr/dominio_INICIALES.csr.pem -out certs/dominio_INICIALES.cert.pem

Recuerde cambiar INICIALES por las iniciales del usuario correspondiente y dominio por el nombre de su sitio web

Verifique el certificado:

openssl x509 -noout -text -in certs/dominio_INICIALES.cert.pem

Verifique la confianza del certificado, para eso use el archivo ca-chain.cert.pem. Ejecute:

openssl verify -CAfile certs/ca_chain.cert.pem newcerts/dominio_INICIALES.cert.pem

Si el valido, tendrá como respuesta un OK

Envié el certificado y el archivo *ca_chain.cert.pem* al servidor web, ya que estos le serán de ayuda para usar el certificado en su sitio web.

Uso de certificados en el sitio web.

Como servidor web debe contar con su clave privada, el certificado y el archivo ca_chain.cert.pem para la verificación del certificado. Se hará uso de apache para para realizar la configuración del servidor web.

- *Copie por archivos *.cert.pem al directorio /etc/ssl/certs*
- *Copie el archivo *.key.pem al directorio /etc/ssl/private*

Cambie los propietarios de estos archivos por *root:ssl-cert* para los 3 archivos.

Cambie los permisos para los archivos en el directorio */etc/ssl/certs* de modo que queden: lectura y escritura para el dueño, y solo escritura para grupos y otros.

Cambie los permisos para los archivos en el directorio */etc/ssl/private* de modo que queden: lectura y escritura para el dueño, ningún permiso para grupos y otros.

Edite el archivo */etc/apache2/sites-available/default-ssl.conf*, agregue las siguientes líneas:

- SSLEngine on
- SSLCertificateFile ruta_donde_se_encuentra_el_certificado
- SSLCertificateKeyFile ruta_donde_se_encuentra_la_clave_privada
- SSLCertificateChainFile ruta_donde_se_encuentra_el_certificado_de_confianza

Además, agregue:

- ServerName dominio_de_su_pagina_web

Y de ser el caso, realice la modificación:

- DocumentRoot ruta_de_los_archivos_de_su_página_web

Guarde los cambios, ahora, realice modificaciones en el archivo 000-default.conf, las líneas que debe agregar-cambiar son:

DocumentRoot *ruta_de_los_archivos_de_su_página_web*

ServerName *dominio_de_su_pagina_web*

Redirect / https://dominio_de_su_pagina_web

Al finalizar las modificaciones pertinentes en el archivo default-ssl.conf guarde el archivo y debe activar el módulo SSL, para esto, ejecute

- su –
- a2enmod ssl

Debe activar el archivo de configuración default-ssl.conf, para esto ejecute:

- a2ensite default-ssl.conf

Edite el archivo */etc/hosts* y agregue la IP de su servidor, o bien solo coloque la IP 127.0.0.1 seguido del dominio de su sitio, esto para que pueda ver su página web en el navegador, esto se realiza ya que todo se está realizando de manera local.

La línea debe quedar:

- Ejemplo: 127.0.0.1 www.actividades.criptografia

Guarde el archivo con los cambios.

Reinicie el servidor para guardar los cambios, para esto ejecute:

/etc/init.d/apache2 restart

Dado que es un certificado autofirmado, el navegador no va a reconocer al certificado intermedio, quien fue que firmo el certificado para el sitio web, una manera de evitar algún inconveniente es importando en el navegador, para eso realice:

Para esto, vaya a Preferencias → Privacidad y Seguridad → Certificados → Ver Certificados → Autoridades → Importar

Y seleccione el archivo ca_chain.cert.pem, y seleccione la casilla Confiar en esta CA para identificar sitios web.

Puede que, aunque se importe le certificado de confianza, continue apareciendo la alerta de certificado no reconocido, mientras al revisar el certificado en el sitio, aparezca su certificado la actividad estará completa.

Al finalizar todas las configuraciones podrá ver su sitio desde el navegador, como comprobar el certificado, tal como se muestra en la Figura N° pf.2:

Figura N° pf. 2: Sitio web.

Al entrar al sitio, podrá visualizar su certificado, tal como se muestra en la Figura N° pf.3:

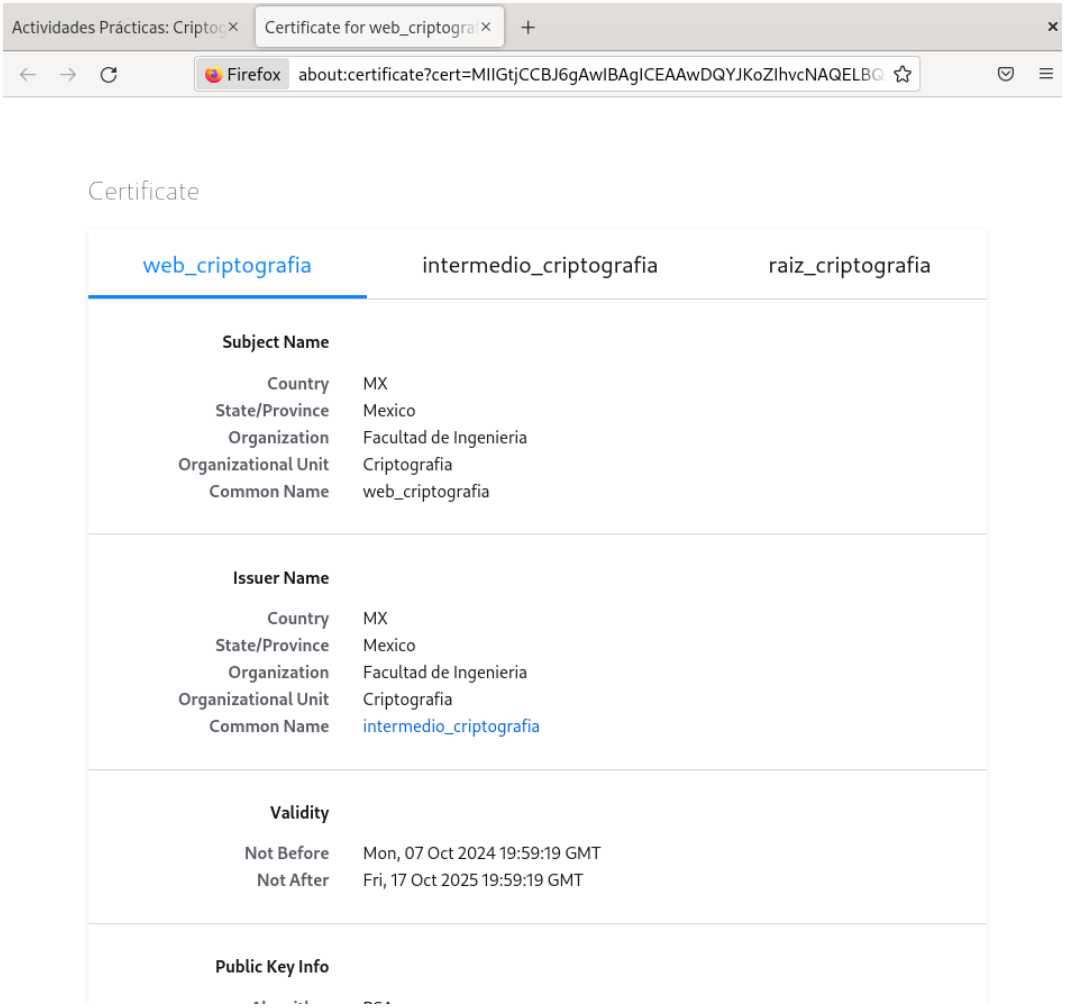


Figura N° pf. 3: Certificados en el sitio.

Sean creativos con el diseño de su página web, este es tema libre.

Puede agregar configuraciones extras a su sitio web, estas podrán ser tomadas en cuenta para su calificación final según lo considere su docente.

UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

ACTIVIDADES PRÁCTICAS: CRIPTOGRAFÍA

PROYECTO FINAL

DOCENTE

Proyecto Final

a. Cifrado de archivos que se encuentre en el directorio - Ransomware

Objetivo

Poner en práctica los conocimientos referentes a la criptografía para la realización del proyecto.

Especificaciones del proyecto

Para la realización del proyecto, debe tener en cuenta el funcionamiento de Ransomware.

Cree un script el cual simule el funcionamiento de Ransomware, en donde debe tener su script junto con demás archivos, el script debe ser capaz de leer todos los archivos y cifrarlos.

El script debe parecer que realizará cualquier otra actividad, debe funcionar como *engaño.*

El script solo debe cifrar los archivos.

A continuación, se muestra un ejemplo de como debe funcionar su script, en la Figura PF1.1 se muestra cómo se listan los archivos del directorio actual, que es el que se analiza, y muestra que los archivos se están analizando, al final se muestra que fue un engaño y los archivos están cifrados.

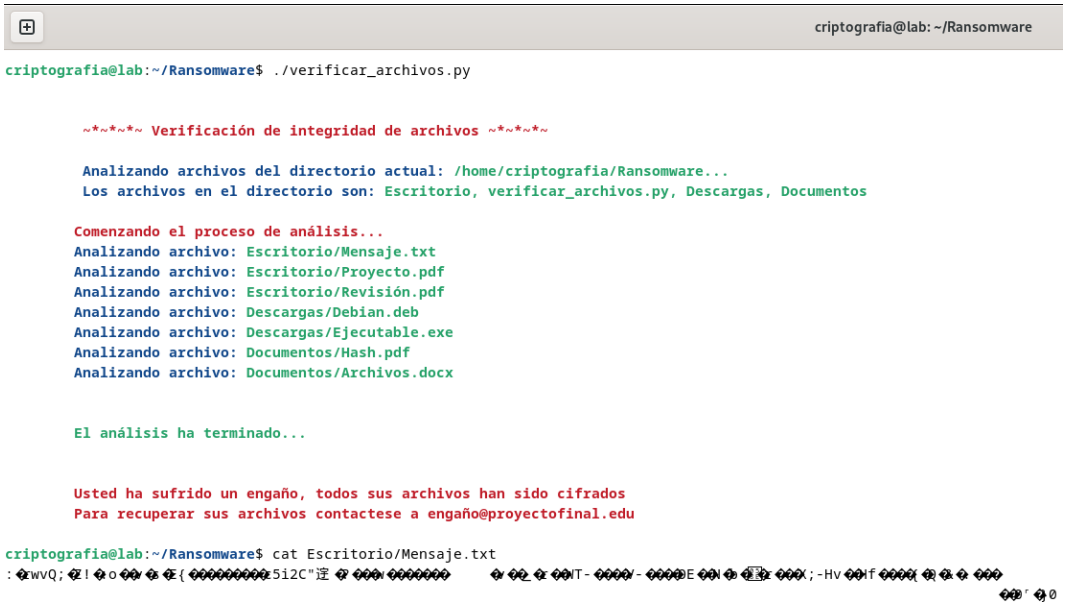


Figura N° PF1. 1: Funcionamiento del script.

Debe realizar la documentación del proyecto. Dicha documentación debe contar con:

a) Portada
b) Índice
c) Introducción
d) Desarrollo
e) Conclusiones personales

Debe incluir capturas de pantalla de cada uno de los puntos realizados, durante el desarrollo del proyecto, cada una de estas debe contar con pie de imagen.

b. Eliminar elementos iguales de un directorio

Objetivo

Poner en práctica los conocimientos referentes a la criptografía para la realización del proyecto.

Especificaciones del proyecto

Para la realización del proyecto, debe tener en cuenta el funcionamiento de las funciones Hash.

Cree un script el cual lea todos los archivos que se encuentran en el directorio, debe mostrar los archivos que sean iguales, y listarlos para que el usuario seleccione aquellos que quiera eliminar. Debe tener su script junto con demás archivos, el script debe ser capaz de leer todos los archivos.

A continuación, se muestra un ejemplo de cómo debe funcionar su script:

En la Figura PF2.1 se muestra cómo se listan los archivos del directorio actual

Figura N° PF2. 1: Uso de script para eliminar elementos duplicados.

En la Figura N° PF2.2 se muestra la primera lista de archivos duplicados, aquí se selecciona que archivos se quieren eliminar, y así sucesivamente.

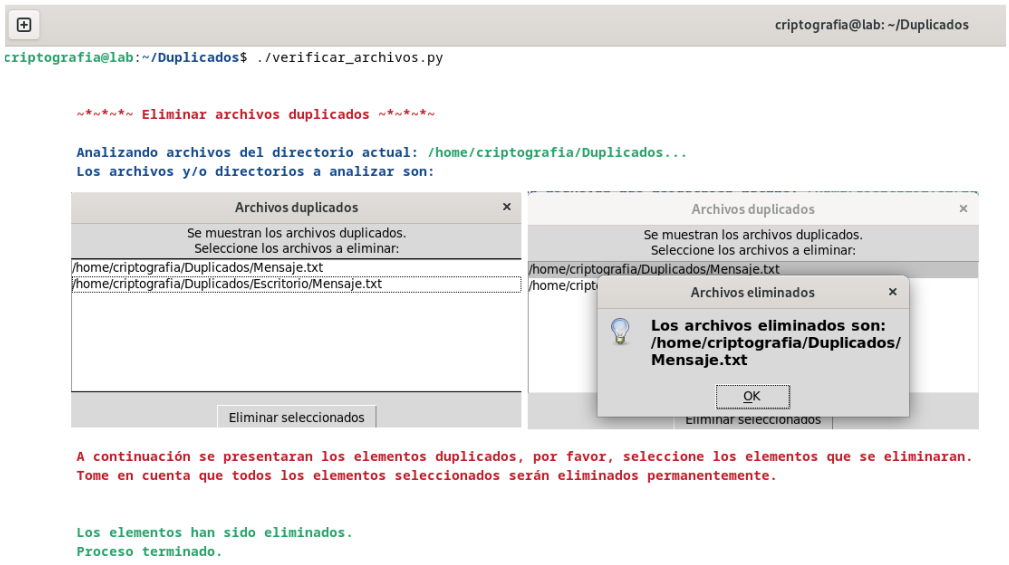


Figura N° PF2. 2: Uso de script para eliminar duplicados.

Debe realizar la documentación del proyecto. Dicha documentación debe contar con:

a) Portada
b) Índice
c) Introducción
d) Desarrollo
e) Conclusiones

Debe incluir capturas de pantalla de cada uno de los puntos realizados, durante el desarrollo del proyecto, cada una de estas debe contar con pie de imagen.

UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

ACTIVIDADES PRÁCTICAS: CRIPTOGRAFÍA

ANEXOS

DOCENTE

Anexo A: Cronograma de inicio del proyecto

Universidad Nacional Autónoma de México
Facultad de Ingeniería
Cronograma de actividades para la elaboración de material didáctico «Actividades Prácticas: Criptografía»
Cronograma 1 - Anexo 2
Elaboró: Patricia Nallely Gómez Flores
Asesora: M. en C. María Jaquelina López Barrientos

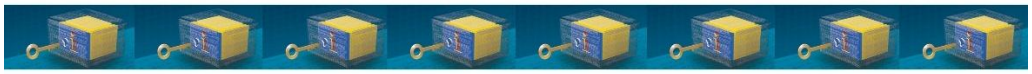
Actividad	Fecha estimada		Tiempo estimado
	Inicio	Fin	
Reunión inicial para planeación	05/06/2019	05/06/2019	2 horas
Reunión, entrega de borrador para prácticas a realizar.	07/08/2019	07/08/2019	2 horas
Elaboración de prácticas.			
Práctica I: Introducción a Linux			
Desarrollo de la práctica	12/08/2019	30/08/2019	80 horas
Revisión de la práctica	04/09/2019	04/09/2019	2 horas
Práctica II: Funciones hash			
Desarrollo de la práctica	06/09/2019	27/09/2019	80 horas
Revisión de la práctica	02/10/2019	02/10/2019	2 horas
Práctica III: Data Encryption Standard			
Desarrollo de la práctica	04/10/2019	30/10/2019	80 horas
Revisión de la práctica	06/11/2019	06/11/2019	2 horas
Práctica IV: Advanced Encryption Standard			
Desarrollo de la práctica	07/11/2019	06/12/2019	80 horas
Revisión de la práctica	08/01/2020	08/01/2020	2 horas
Práctica V: Diffie Hellman			
Desarrollo de la práctica	09/01/2020	31/01/2020	80 horas
Revisión de la práctica	05/02/2020	05/02/2020	2 horas
Práctica VI: RSA			
Desarrollo de la práctica	06/02/2020	28/02/2020	80 horas
Revisión de la práctica	03/06/2020	03/06/2020	2 horas
Práctica VII: ElGamal, Práctica VIII: Firmas Digitales			
Desarrollo de la práctica	04/06/2020	16/06/2020	20 horas
Desarrollo de la práctica	17/06/2020	29/06/2020	20 horas
Revisión de la práctica VII y VIII	01/07/2020	01/07/2020	2 horas
Práctica IX: PGP, Práctica X: Certificados			
Desarrollo de la práctica	03/07/2020	13/07/2020	20 horas
Desarrollo de la práctica	14/07/2020	22/07/2020	20 horas
Revisión de la práctica IX y X	27/07/2020	27/07/2020	2 horas
Proyecto Final			
Desarrollo de la práctica	23/07/2020	03/08/2020	20 horas
Revisión de la práctica	04/08/2020	04/08/2020	2 horas
Taller Virtual Intersemestral			
Planeación de impartición de curso para evaluar prácticas por parte de los alumnos	04/05/2020	29/05/2020	20 horas
Impartición de cursos a un grupo de alumnos para evaluar las prácticas (Parte I)	10/06/2020	06/07/2020	8 horas
Revisión de prácticas entregadas por los cursantes (Parte I)	22/06/2020	13/07/2020	40 horas
Impartición de cursos a un grupo de alumnos para evaluar las prácticas (Parte II)	22/07/2020	17/08/2020	8 horas
Revisión de prácticas entregadas por los cursantes (Parte II)	04/08/2020	26/08/2020	40 horas
Reunión con asesora para resultados de Taller virtual	02/09/2020	02/09/2020	2 horas

Anexo B: Taller Virtual de Criptografía Práctica

El Laboratorio de Redes y Seguridad te invitan al

TALLER VIRTUAL DE CRIPTOGRAFÍA PRÁCTICA

Objetivo: Que los estudiantes interesados en el campo de la Seguridad Informática adquieran conocimientos prácticos que les permitan entender el funcionamiento de diversos algoritmos de cifrado e implementarlos en aplicaciones.



Calendario de actividades:
Bienvenida y presentación del Programa de actividades martes 9 de junio 13h. (Sesión vía Zoom)

Parte I

Prácticas	Envió de las prácticas a los alumnos	Sesión vía Zoom	Entrega de prácticas
1 y 2	10 de junio	16 de junio	22 de junio
3 y 4	17 de junio	23 de junio	29 de junio
5 y 6	24 de junio	30 de junio	6 de julio

Parte II **Las sesiones por Zoom serán** de 13:00 a 14:30 h

Prácticas	Envió de las prácticas a los alumnos	Sesión vía Zoom	Entrega de prácticas
7 y 8	22 de julio	28 de julio	3 de agosto
9 y 10	29 de julio	4 de agosto	10 de agosto
Desarrollo de App	5 de agosto	11 de agosto	17 de agosto

A los alumnos inscritos se les enviarán las prácticas y materiales necesarios vía correo electrónico 4 días antes de la fecha de la sesión por Zoom.

Requisitos indispensables:

-Contar con una computadora con al menos 8 GB de RAM, y 20 GB libres en disco duro, en cuanto a procesador contar mínimo con Celeron o AMD equivalente, es decir la serie E1.

-Conocimientos de programación en Python, preferentemente en su versión 3.

-Disponer de 2 hrs. al día para llevar a cabo las actividades prácticas.

Estructura del Laboratorio

El laboratorio consta de 6 sesiones virtuales para el desarrollo de todas las prácticas.

Durante la Parte I de este taller, se realizarán un total de 6 prácticas a través de las cuales se trabajarán distintos algoritmos criptográficos.

Y en la Parte II se realizarán de 5 prácticas y un proyecto final, todos orientados a aplicaciones de seguridad informática.

En cada sesión se proporcionarán tips y recomendaciones para el éxito de las prácticas, además de asesoría vía correo electrónico.

Inscripciones

Enviar historial académico reciente al correo lab.redyseguridad@gmail.com con fecha límite al 5 de junio del presente año. El cupo máximo es de 20 personas.

Se entregará constancia de participación con valor curricular.

Comentarios en las prácticas realizadas por los alumnos:

1. Conceptos básicos de Linux.

5. Comentarios o conclusiones adicionales:
Es importante realizar esta practica porque no siempre se tienen todos los conceptos básicos de un sistema operativo Linux, pero que son muy útiles para usar el sistema. También se ven conceptos importantes de seguridad, como es un sistema multiusuario es posible que otros usuarios quieran entrar a los archivos de los demás, pero con esto podemos establecer quienes pueden o no acceder a la información.

2. Conceptos de funciones Hash.

5. Comentarios o conclusiones adicionales
Es interesante ver como surgen los hashes y el poder haber realizado un programa que recibe parámetros de opciones es muy útil no solo para estos programas, sino para muchas cosas más. Por otro lado, pude notar que existen varios tipos de hash, yo creía que eran pocos los que existían, pero ahora puedo notar que son varios y no solo eso, que podemos usarlos dependiendo de lo que necesitamos.

3. Data Encryption Standard

5. Comentarios o conclusiones adicionales

Esta práctica (en conjunto con el material adicional) me pareció muy completa en el aspecto práctico y teórico ya que me dio una buena base de información referente al algoritmo DES.

4. Advanced Encrption Standard

5. Comentario y conclusiones adicionales.

Se logró conocer y comprender de manera satisfactoria el funcionamiento básico del cifrado y descifrado del algoritmo AES, y se logró realizar exitosamente un script en python3 (usando el módulo Pycrypto) el cual fuera capaz de cifrar y descifrar cualquier tipo de archivo, incluyendo frases ingresadas directamente en la terminal.
En general, en el desarrollo de la práctica no se presentó ningún inconveniente para poder desarrollarla con éxito, se encuentra muy bien explicada y desarrollada de una manera que no es aburrida su realización. Me agrado bastante la parte de tener que buscar archivos para poder continuar con el desarrollo de la práctica, además, gracias a como esta planteada, se refuerza el conocimiento adquirido en la práctica No. 1, ya que fue necesario crear un usuario nuevo, cambiar permisos de archivos y carpeta para poder trabajar bien la ejecución de los scripts, buscar archivos, crear directorios, etc.

5. Diffie-Hellman

5. Comentario y conclusiones adicionales.

Con esta practica me quedo mas claro el comportamiento del envío de las claves por el algoritmo de Diffie-Hellman, tanto la forma de implementar la generación de claves publicas y privadas, como su vulnerabilidad por medio del ataque man in the middle y como es muy importante el uso de números primos en la criptografía.

6. RSA

5. Comentario y conclusiones adicionales.

En esta practica aprendimos a utilizar y a programar el algoritmo RSA, el cuál nos llena más de historia en el camino de la criptografía, haciendo cada vez más seguras las formas de encriptar información, me gustó que el algoritmo haya sido menos complicado que los anteriores.

7. ElGamal

5. Comentarios y conclusiones adicionales.

En general, la práctica fue muy completa, y comprendí correctamente todos los conceptos y actividades que se desarrollaron, su funcionamiento, y como es posible generar y romper el algoritmo, además claro esta de entender como es más eficiente El Gamal, entre más grande sean los números empleados, más difícil de romper será.

8. Firmas Digitales

5. Comentarios y conclusiones adicionales.

Esta práctica me permitió repasar la teoría referente a los algoritmos ElGamal y RSA, además de entender la importancia y utilidad de las firmas digitales en nuestra actualidad para poder comprobar la integridad, autenticidad y no repudio de archivos con información sensible.

9. PGP

5. Comentario y conclusiones adicionales.

Considero que con esta práctica, logramos observar la importancia que tiene el mantener información cifrada, inclusive si se trata de correos electronicos, los cuales, debemos recordar que son personales y manejan demasiada información sensible, por lo que para una correcta información y de manera más personal, sería preferible y recomendable, cifrar cada uno de los correos que contenga información confidencial.

10. Certificados

5. Comentario y conclusiones adicionales.

En lo personal, la práctica resulta interesante y sencilla, dado que en mi curso de criptografía en el semestre anterior realizamos actividades similares a estas, no tuve dudas o cuestionamientos sobre lo que estabamos realizando, en general, las explicaciones mejoraron, y son más claras las actividades.

11. Proyecto Final

Con la elaboración de este trabajo, logré implementar los conocimientos adquiridos sobre los certificados y las firmas digitales, para realizar uno propio, de tal manera que se nos permitió identificar y elaborar correctamente las funciones de los tres componentes importantes durante el proceso de certificación, iniciando por la CA o certificadora raíz, la CI o certificadora intermedia y finalmente el sitio web, que está solicitando la firma para tener una validación, logrando observar que las tres partes son de igual importancia durante el proceso de certificación, y que sin una falla o no completa su participación, es imposible realizar los certificados finales, ya que entre ellas se comunican para lograr completar el proceso satisfactoriamente.

Anexo C: Cronograma para actualización del manual

Universidad Nacional Autónoma de México
Facultad de Ingeniería
Cronograma de actividades para la elaboración de material didáctico «Actividades Prácticas: Criptografía»
Cronograma 2 - Anexo 3
Elaboró: Patricia Nallely Gómez Flores
Asesora: M. en C. María Jaquelina López Barrientos

Actividad	Fecha estimada		Tiempo estimado
	Inicio	Fin	
Reunión con asesora para retomar actividades	06/01/2023	06/01/2023	1 hora
Reunión con asesora para retomar actividades	12/01/2023	12/01/2023	1 hora
Modificación y elaboración de prácticas.			
Práctica I: Introducción a Linux			
Modificación de la práctica	23/01/2023	01/02/2023	30 horas
Pruebas a la práctica	02/02/2023	06/02/2023	15 horas
Práctica II: Funciones hash			
Modificación de la práctica	07/02/2023	16/02/2023	30 horas
Pruebas a la práctica	17/02/2023	21/02/2023	15 horas
Reunión con asesora para revisar avances	24/02/2023	24/02/2023	2 horas
Práctica III: Data Encryption Standard			
Modificación de la práctica	22/02/2023	05/03/2023	40 horas
Pruebas a la práctica	06/03/2023	10/03/2023	15 horas
Práctica IV: Advanced Encryption Standard			
Modificación de la práctica	11/03/2023	20/03/2023	40 horas
Pruebas a la práctica	21/03/2023	25/03/2023	15 horas
Reunión con asesora para revisar avances	31/03/2023	31/03/2023	2 horas
Práctica V: Diffie Hellman			
Modificación de la práctica	26/03/2023	04/04/2023	30 horas
Pruebas a la práctica	05/04/2023	09/04/2023	15 horas
Práctica VI: RSA			
Modificación de la práctica	10/04/2023	19/04/2023	30 horas
Pruebas a la práctica	20/04/2023	24/04/2023	15 horas
Reunión con asesora para revisar avances	28/04/2023	28/04/2023	2 horas
Práctica VII: ElGamal			
Modificación de la práctica	25/04/2023	04/05/2023	30 horas
Pruebas a la práctica	05/05/2023	09/05/2023	15 horas
Práctica VIII: Firmas Digitales			
Modificación de la práctica	10/05/2023	19/05/2023	30 horas
Pruebas a la práctica	21/05/2023	25/05/2023	15 horas
Reunión con asesora para revisar avances	26/05/2023	26/05/2023	2 horas
Práctica IX: Esteganografía y PGP			
Modificación de la práctica	28/05/2023	06/06/2023	30 horas
Pruebas a la práctica	07/06/2023	11/06/2023	15 horas
Práctica X: Blockchain			
Elaboración de la práctica	12/06/2023	23/06/2023	40 horas
Pruebas a la práctica	24/06/2023	28/06/2023	15 horas
Reunión con asesora para revisar avances	30/06/2023	30/06/2023	2 horas
Práctica XI: Conexiones remotas			
Modificación de la práctica	29/06/2023	08/07/2023	30 horas
Pruebas a la práctica	09/07/2023	13/07/2023	15 horas
Proyecto Final			
Modificación de la práctica	14/07/2023	25/07/2023	30 horas
Pruebas a la práctica	26/07/2023	30/07/2023	15 horas
Reunión con asesora para revisar avances	04/08/2023	04/08/2023	2 horas
Mánual instalación de máquinas virtuales			
Elaboración de manual	05/08/2023	08/08/2023	8 horas
Acoplamiento del manual en un solo documento			
Ajuste de introducción	09/08/023	09/08/023	5 horas
Acoplamiento de prácticas	09/08/023	13/08/2023	20 horas
Realización de conclusiones	14/08/2023	14/08/2023	2 horas
Organización de Fuentes de Información	14/08/2023	14/08/2023	5 horas
Realización y organización de Anexos	14/08/2023	16/18/2023	10 horas
Reunión con asesora para revisión del manual	18/08/2023	18/08/2023	2 horas
Máquinas virtuales			
Revisión de montaje de máquinas	17/08/2023	17/08/2023	2 horas
Revisión de funcionamiento de máquinas	17/08/2023	20/08/2023	10 horas
Revisión general del material			
Reunión con asesora para revisión de todo el material	21/08/2023	21/08/2023	2 horas

Anexo D: Llenado de práctica

Cada una de las prácticas es entregada en formato PDF, esto para evitar el consumo de papel, por lo cual las prácticas tienen un formato de *formulario* en donde de acuerdo con la actividad se les solicita a los cursantes que coloquen capturas de pantalla de las actividades realizadas, o bien llenen con texto dichos recuadros.

Para el correcto llenado de la práctica puede hacer uso del lector de PDF Adobe Reader cuyo enlace de descarga se proporciona a continuación dando clic a la imagen.

Cuando realice la práctica verá algunos recuadros como el mostrado en la Figura N° A.1:

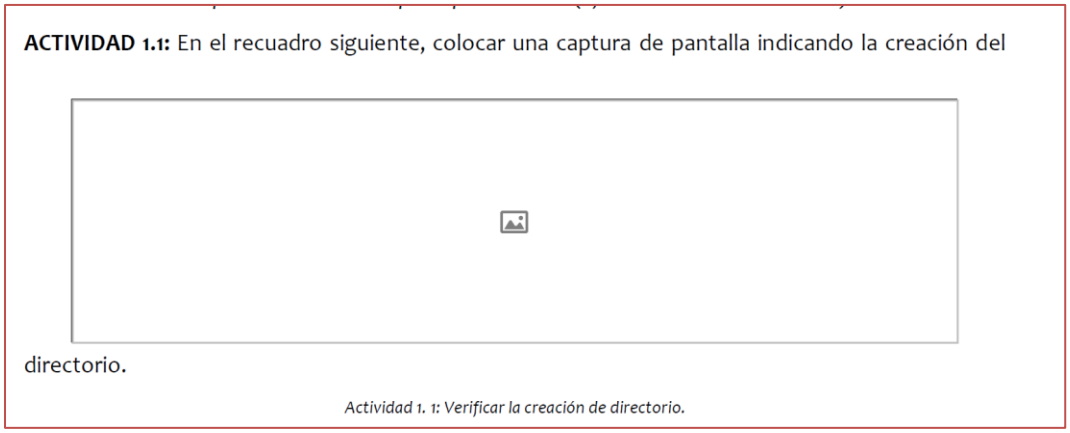


Figura N° A. 1: Colocar imagen en las prácticas.

Para eso, solo basta con que dé clic en el recuadro y este abrirá una nueva ventana en donde le solicitará hacer búsqueda de la imagen a incrustar, dicha ventana se muestra en la Figura N° A.2:

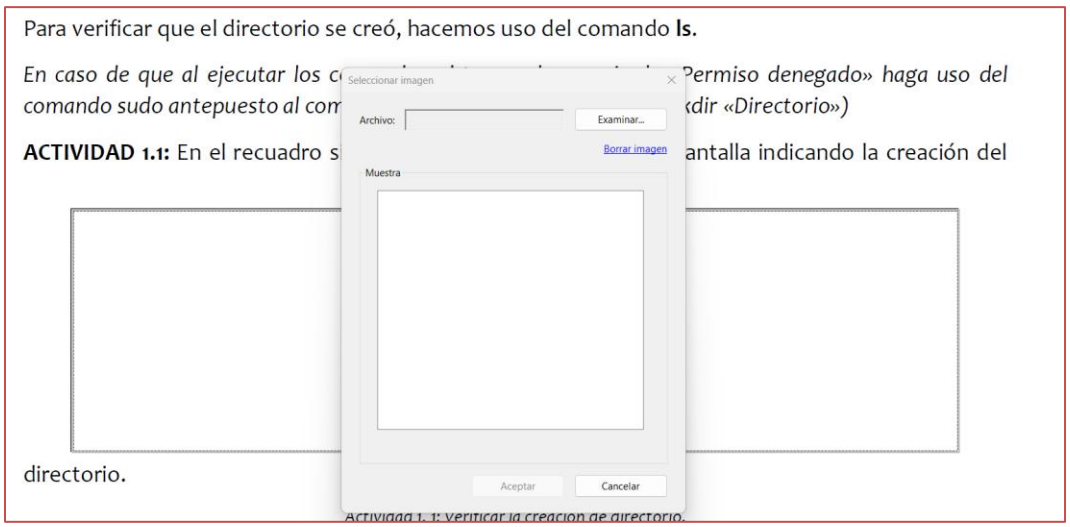


Figura N° A. 2: Agregar imagen en las prácticas.

Dé clic en examinar y posteriormente busque la imagen a agregar para completar su actividad.

En algunos otros recuadros, lo que se le pedirá que coloque es texto, tal como se muestra en la Figura N° A.3:

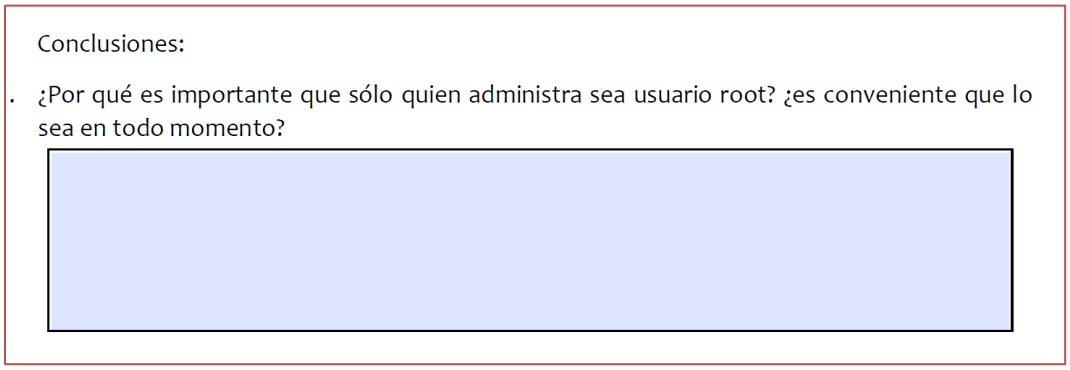


Figura N° A. 3: Agregar texto.

En estos casos, el agregar texto se realiza colocándose sobre el recuadro azul, y comenzar a escribir.

Se les pide que hagan uso de algún lector, ya que de cualquier otro método su práctica se vería dañada en cuanto a formato.

Anexo E: Editores de texto: Vim y Nano

A continuación, se proporcionan unas tablas con los comandos básicos para los editores de texto comunes en Linux.

Tabla N° A. 1: Comandos básicos para uso de vim.

vim(Sponsor warp, s.f.)

Comando	Función
i	Insertar texto.
I	Insertar al principio de la línea.
yy	Copiar línea de texto en la que se encuentra posicionado el cursor.
dd	Eliminar línea de texto en la que se encuentra posicionado el cursor.
p	Pegar la línea después del cursor.
x	Eliminar carácter.
:wq!	Guardar el archivo y salir.
:q!	Salir sin guardar.

Tabla N° A. 2: Comando básicos para el uso de nano.

nano (nano-editor, 2023)

Comando	Función
Ctrl + G	Ayuda.
Ctrl + X	Salir.
Ctrl + O	Guardar.
Ctrl + W	Buscar.
**Ctrl + **	Remplazar.
Ctrl + K	Cortar.
Ctrl + U	Pegar.
Ctrl + J	Justificar.

Anexo F: Archivo instrucciones: Pasos para entender AES

En máquina virtual

- **CIFRADO:** (teoria.com,s.f.)

Para la ejemplificación del cifrado aes, se hará uso del archivo cifrado_aes, el cual encontrará en el directorio **/twp/.AES_algoritmo**

AES trabaja el método de cifrado siguiendo la estructura mostrada en la Figura N° 4.3.1:

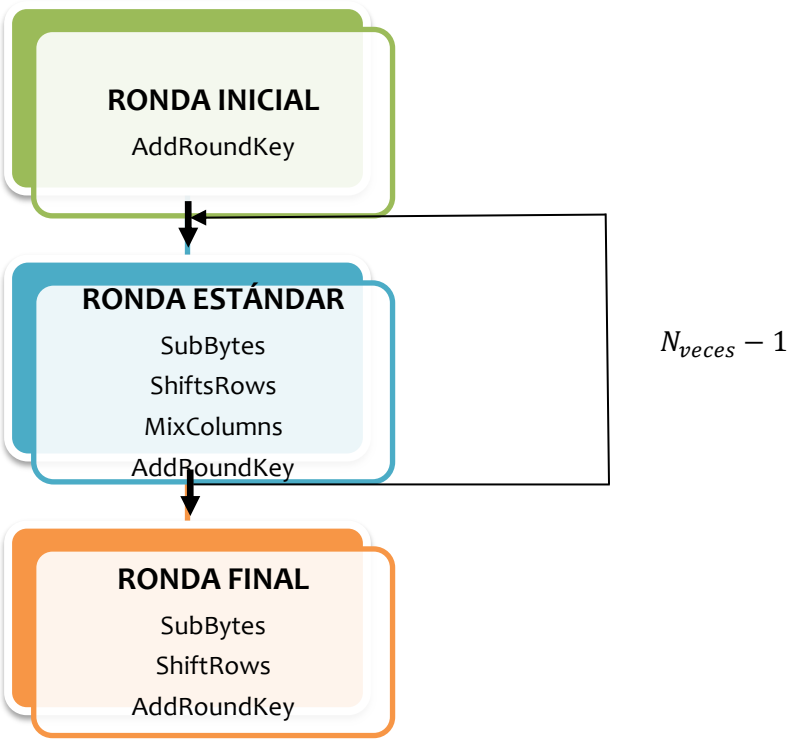


Figura N° 4.3. 1: Cifrado AES.

RONDA INICIAL: Se realiza la operación XOR ($\oplus$) entre el mensaje y la clave, elementos que constan de 16 bytes hexadecimales, el resultado, deben de ser acomodados en una matriz de 4x4 tomando en cuenta los índices de cada byte como se muestra en la Figura N° 4.3.2

00	04	08	12
01	05	09	13
02	06	10	14
03	07	11	15

Figura N° 4.3. 2: Acomodo matricial de elementos.

Copie el archivo cifrado_aes a su directorio AES y ejecute lo siguiente:

« python cifrado_aes --datos»

Coloque una captura de pantalla de lo mostrado en el recuadro *«Actividad 4.3: Datos»* que se encuentra en la práctica.

Teniendo los datos, los elementos, quedan de la siguiente forma:

Start of Round				**After SubBytes**				**After ShitfRows**				**After MixColumns**				**Round Key Value**			
32	88	31	e0													2b	28	ab	09
43	5a	31	37													7e	ae	f7	cf
f6	30	98	07													15	d2	15	4f
18	8d	a2	34													16	a6	88	3c

Figura N° 4.3. 3: Ronda Inicial.

En la Figura N° 4.3.3 se muestra el acomodo de los elementos para la ronda inicial, también conocida como ronda cero.

AddRoundKey: Haciendo uso de estos dos elementos, se realiza la operación $\oplus$ entre cada uno de los bytes del mensaje y la clave. Ejecute:

«python cifrado_aes --sround»

Coloque una captura de pantalla de lo mostrado en el recuadro *«Actividad 4.3: SRound»* que se encuentra en la práctica.

El resultado final, son los nuevos valores de Start of Round para la primera ronda.

RONDA ESTÁNDAR:

En este caso, este proceso se realizará 9 veces, dado que la clave consta de 128 bits.

SubBytes: Para realizar este paso, se toman los valores obtenidos en *AddRoundKey*. En este paso, se hace uso de una matriz que costa de 16 filas y 16 columnas. Como ejemplo, se toma el primer byte del resultado obtenido anteriormente, es decir «*19*», el número «*1*» se busca en la fila y el «*9*» en columna, la intersección de estas nos da un nuevo valor hexadecimal, que será el nuevo valor para dicha celda en la matriz, es decir «*d4*»

Ejecute:

« *python cifrado_aes --subbytes*»

Coloque una captura de pantalla de lo mostrado en el recuadro «*Actividad 4.3: SubBytes*» que se encuentra en la práctica. La matriz que se utiliza se muestra en la Figura N° 4.3.4.

		y															
		0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
x	0	63	7c	77	7b	f2	6b	6f	c5	30	01	67	2b	fe	d7	ab	76
	1	ca	82	c9	7d	fa	59	47	f0	ad	d4	a2	af	9c	a4	72	c0
	2	b7	fd	93	26	36	3f	f7	cc	34	a5	e5	f1	71	d8	31	15
	3	04	c7	23	c3	18	96	05	9a	07	12	80	e2	eb	27	b2	75
	4	09	83	2c	1a	1b	6e	5a	a0	52	3b	d6	b3	29	e3	2f	84
	5	53	d1	00	ed	20	fc	b1	5b	6a	cb	be	39	4a	4c	58	cf
	6	d0	ef	aa	fb	43	4d	33	85	45	f9	02	7f	50	3c	9f	a8
	7	51	a3	40	8f	92	9d	38	f5	bc	b6	da	21	10	ff	f3	d2
	8	cd	0c	13	ec	5f	97	44	17	c4	a7	7e	3d	64	5d	19	73
	9	60	81	4f	dc	22	2a	90	88	46	ee	b8	14	de	5e	0b	db
	a	e0	32	3a	0a	49	06	24	5c	c2	d3	ac	62	91	95	e4	79
	b	e7	c8	37	6d	8d	d5	4e	a9	6c	56	f4	ea	65	7a	ae	08
	c	ba	78	25	2e	1c	a6	b4	c6	e8	dd	74	1f	4b	bd	8b	8a
	d	70	3e	b5	66	48	03	f6	0e	61	35	57	b9	86	c1	1d	9e
	e	e1	f8	98	11	69	d9	8e	94	9b	1e	87	e9	ce	55	28	df
	f	8c	a1	89	0d	bf	e6	42	68	41	99	2d	0f	b0	54	bb	16

Figura N° 4.3. 4: Matriz SubBytes.

Los valores obtenidos al realizar este paso son los valores colocados en el campo de After SubBytes para la primera ronda.

Shift Rows: Para realizar este paso, se toma en cuenta el resultado obtenido en *SubBytes*. En este paso se hace un corrimiento circular de *n* número de bytes como se indican en la Figura N° 4.3.5:

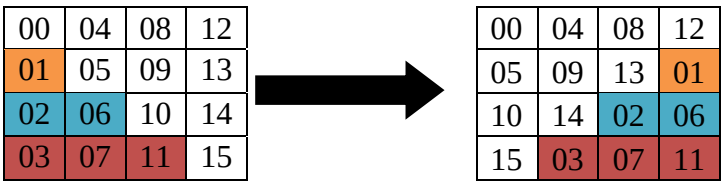


Figura N° 4.3. 5: Corrimiento de bytes.

Ejecute:

« python cifrado_aes --srows»

Coloque una captura de pantalla de lo mostrado en el recuadro «*Actividad 4.3: Shift Rows*» que se encuentra en la práctica.

Mix Columns: Para realizar este paso, se toma en cuenta el resultado obtenido en *Shift Rows*. Mix Columns hace uso del teorema de Galios, y su uso en AES, consta de distintos pasos:

1. Se hace una multiplicación de matrices, por lo cual se hace uso de la matriz obtenida en Shift Rows y una matriz llamada fija, esta es:
2.

02	03	01	01
01	02	03	01
01	01	02	03
03	01	01	02

Figura N° 4.3. 6: Matriz fija.

Tomando el primer byte de cada una de las matrices se realiza: $(02 \bullet d4)$[21], por lo cual, tomando como ejemplo para la Figura N° 4.3.7:

[21] El operador • es la multiplicación de un anillo de polinomios con coeficientes en $GF(2^8)$. Para realizar dicha operación en el cifrado y descifrado AES, se hace uso de las tablas L y E.

02	03	01	01
01	02	03	01
01	01	02	03
03	01	01	02

d4	e0	b8	1e
bf	b4	41	27
5d	52	11	98
30	ae	f1	e5

Mc_{00}	Mc_{04}	Mc_{08}	Mc_{12}
Mc_{01}	Mc_{05}	Mc_{09}	Mc_{13}
Mc_{02}	Mc_{06}	Mc_{10}	Mc_{14}
Mc_{03}	Mc_{07}	Mc_{11}	Mc_{15}

Figura N° 4.3. 7: Multiplicación de matrices.

El resultado para obtener el índice Mc_{00} es:

$$Mc_{00} = (02 \bullet \text{d4}) \oplus (03 \bullet \text{bf}) \oplus (01 \bullet \text{5d}) \oplus (01 \bullet 30)$$

Del mismo modo se realizan para los demás índices, para Mc_{04} quedaría:

$$Mc_{04} = (02 \bullet \text{e0}) \oplus (03 \bullet \text{b4}) \oplus (01 \bullet 52) \oplus (01 \bullet \text{ae})$$

Para obtener todos los valores, se hace uso de las tablas L y E.

3. La tabla L se utiliza para obtener su valor equivalente, como lo realizado en SubBytes de cada uno de los elementos que se van a multiplicar.

   Como ejemplo, se toman los valores (02 • d4) de estos dos, se debe tomar su equivalente en la tabla L, se busca «*d*» en la fila y «*4*» en columna, lo mismo con «*0*» se busca en fila y el «*2*» en las columnas, los resultados son «*41*» y «*19*» respectivamente.

   De los dos valores obtenidos se realiza una suma de los hexadecimales, el resultado es «*5a*».

4. Ahora con este resultado se obtiene su equivalente en la tabla E, del mismo modo, buscando la intersección de «*5*» en fila y «*a*» en columna. El resultado es «*b3*»

   Se realizan el mismo proceso para (03• bf), (01 • 5d), (01 • 30)

5. Se realiza la operación $\oplus$ con los 4 elementos hexadecimales obtenidos, es decir:

$$Mc_{00} = \text{b3} \oplus \text{da} \oplus \text{5d} \oplus 30 = 04$$

La tabla L se muestra en la Figura N° 4.3.8

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0		00	19	01	32	02	1A	C6	4B	C7	1B	68	33	EE	DF	03
1	64	04	E0	0E	34	8D	81	EF	4C	71	08	C8	F8	69	1C	C1
2	7D	C2	1D	B5	F9	B9	27	6A	4D	E4	A6	72	9A	C9	09	78
3	65	2F	8A	05	21	0F	E1	24	12	F0	82	45	35	93	DA	8E
4	96	8F	DB	BD	36	D0	CE	94	13	5C	D2	F1	40	46	83	38
5	66	DD	FD	30	BF	06	8B	62	B3	25	E2	98	22	88	91	10
6	7E	6E	48	C3	A3	B6	1E	42	3A	6B	28	54	FA	85	3D	BA
7	2B	79	0A	15	9B	9F	5E	CA	4E	D4	AC	E5	F3	73	A7	57
8	AF	58	A8	50	F4	EA	D6	74	4F	AE	E9	D5	E7	E6	AD	E8
9	2C	D7	75	7A	EB	16	0B	F5	59	CB	5F	B0	9C	A9	51	A0
A	7F	0C	F6	6F	17	C4	49	EC	D8	43	1F	2D	A4	76	7B	B7
B	CC	BB	3E	5A	FB	60	B1	86	3B	52	A1	6C	AA	55	29	9D
C	97	B2	87	90	61	BE	DC	FC	BC	95	CF	CD	37	3F	5B	D1
D	53	39	84	3C	41	A2	6D	47	14	2A	9E	5D	56	F2	D3	AB
E	44	11	92	D9	23	20	2E	89	B4	7C	B8	26	77	99	E3	A5
F	67	4A	ED	DE	C5	31	FE	18	0D	63	8C	80	C0	F7	70	07

Figura N° 4.3. 8: Tabla L.

La tabla E se muestra en la Figura N° 4.3.9

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	01	03	05	0F	11	33	55	FF	1A	2E	72	96	A1	F8	13	35
1	5F	E1	38	48	D8	73	95	A4	F7	02	06	0A	1E	22	66	AA
2	E5	34	5C	E4	37	59	EB	26	6A	BE	D9	70	90	AB	E6	31
3	53	F5	04	0C	14	3C	44	CC	4F	D1	68	B8	D3	6E	B2	CD
4	4C	D4	67	A9	E0	3B	4D	D7	62	A6	F1	08	18	28	78	88
5	83	9E	B9	D0	6B	BD	DC	7F	81	98	B3	CE	49	DB	76	9A
6	B5	C4	57	F9	10	30	50	F0	0B	1D	27	69	BB	D6	61	A3
7	FE	19	2B	7D	87	92	AD	EC	2F	71	93	AE	E9	20	60	A0
8	FB	16	3A	4E	D2	6D	B7	C2	5D	E7	32	56	FA	15	3F	41
9	C3	5E	E2	3D	47	C9	40	C0	5B	ED	2C	74	9C	BF	DA	75
A	9F	BA	D5	64	AC	EF	2A	7E	82	9D	BC	DF	7A	8E	89	80
B	9B	B6	C1	58	E8	23	65	AF	EA	25	6F	B1	C8	43	C5	54
C	FC	1F	21	63	A5	F4	07	09	1B	2D	77	99	B0	CB	46	CA
D	45	CF	4A	DE	79	8B	86	91	A8	E3	3E	42	C6	51	F3	0E
E	12	36	5A	EE	29	7B	8D	8C	8F	8A	85	94	A7	F2	0D	17
F	39	4B	DD	7C	84	97	A2	FD	1C	24	6C	B4	C7	52	F6	01

Figura N° 4.3. 9: Tabla E.

Ejecute:

« python cifrado_aes --mcolumns»

Coloque una captura de pantalla de lo mostrado en el recuadro *«Actividad 4.3: Mix Columns»* que se encuentra en la práctica.

Finalmente se realiza nuevamente AddRoundKey, que se recuerda que es la operación ⊕, en este caso, entre la matriz correspondiente a Mix Columns y la subclave para esta ronda, el resultado será Start of Round de la siguiente ronda.

Obtención de las claves:

Teniendo la clave, se extrae la última columna, sombreada en color amarillo, de esta, el primer elemento pasa a ser el último, y de esta última columna se obtiene sus valores equivalentes, haciendo uso de la misma matriz usada en SubBytes. El proceso se muestra en la Figura N ° 4.3.10

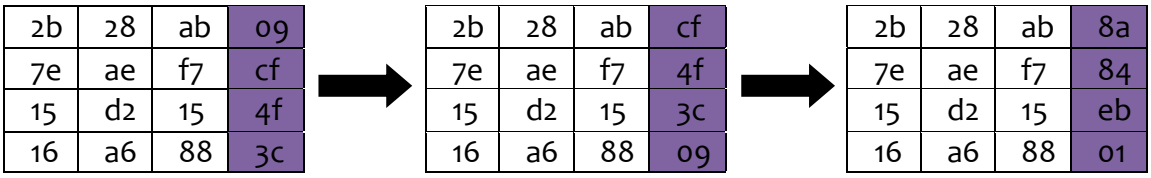


Figura N° 4.3. 10: Primer paso para obtener claves.

Posteriormente, se hace uso de la matriz RCON, la cual se muestra en la Figura N° 4.3.11:

10	10	10	10	10	10	10	10	10	10	10	10
00	00	00	00	00	00	00	00	00	00	00	00
00	00	00	00	00	00	00	00	00	00	00	00
00	00	00	00	00	00	00	00	00	00	00	00

Figura N° 4.3. 11: Matriz RCON.

En este caso, esta consta de 10 columnas dado que son 10 rondas a realizar.

Se realiza la operación ⊕ entre la primera columna de la clave, el resultado mostrado en la Figura N° 4.3.9 y la primera columna de la matriz RCON, esto por ser la primera clave, para la segunda ronda se hará uso de la segunda columna y así sucesivamente, el resultado de dicha operación será la primera columna para la nueva clave, como se muestra en la Figura N° 4.3.12.

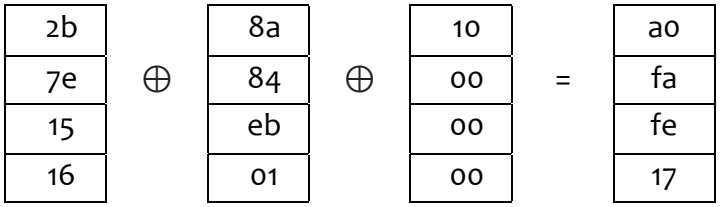


Figura N° 4.3. 12: Obtención de la primera columna para la primera subclave.

Finalmente, para obtener las siguientes columnas, se realiza un ⊕ de la columna correspondiente de la clave original y la columna del resultado anterior inmediato, es decir, para la columna dos, se realiza la operación ⊕ con el resultado obtenido en la Figura N° 4.3.12. Para la columna tres, se realiza la operación ⊕ con el resultado obtenido para la columna dos y así sucesivamente.

Ejecute:

« python cifrado_aes --keys»

Coloque una captura de pantalla de lo mostrado en el recuadro *«Actividad 4.3: SubClaves»* que se encuentra en la práctica.

Para obtener la siguiente Subclave, se realiza el mismo proceso, solo que ahora la subclave a utilizar será la obtenida en la ronda anterior, no la original.

RECORDANDO: En el apartado de **RONDA ESTÁNDAR** se menciona al final la realización de la AddRoundKey entre MixColumns y la Subclave, cuyo resultado será los bytes de Start of Round para la siguiente ronda. Para visualizar el resultado ejecute:

« python cifrado_aes --sroundmc»

Coloque una captura de pantalla de lo mostrado en el recuadro *«Actividad 4.3: SRound MColumns - Subclave»* que se encuentra en la práctica.

Recuerde que el proceso anterior se realiza 9 veces. Ejecute:

« python cifrado_aes --restandar»

Lo anterior le permitirá observar los datos obtenidos en las 9 rondas.

Llene la tabla *«Actividad 4.3: Ronda 1 – Ronda 9»* que se encuentra en su práctica de acuerdo a los datos obtenidos al ejecutar lo anterior.

RONDA FINAL: La ronda final realiza las operaciones *SubBytes, ShiftRows y AddRoundKey*, además de obtener la *Subclave* correspondiente para dicha ronda.

El orden de dicha ronda es:

- *SRound*: Que es el resultado de ⊕ entre el resultado MixColumns y la Subclave correspondientes a la ronda nueve, es decir *AddRoundKey*
- *SubBytes:* Resultado de realizar dicha operación utilizando SRound.
- *SRows:* Resultado de realizar dicha operación utilizando el resultado SubBytes.
- *Key:* La clave correspondiente a la ronda diez.

Finalmente, para obtener la cadena correspondiente al cifrado, se realiza la operación ⊕ entre el resultado arrojado por *SRows* y *Key*. Ejecute:

« python cifrado_aes --rfinal»

Coloque una captura de pantalla de lo mostrado en el recuadro «*Actividad 4.3: Ronda Final*» que se encuentra en la práctica.

- **DESCIFRADO:**
  Para la ejemplificación del descifrado aes, se hará uso del archivo descifrado_aes, el cual encontrará en el directorio **/twp/.AES_algoritmo**

  AES trabaja el método de cifrado siguiendo la estructura mostrada en la Figura N° 4.3.13:

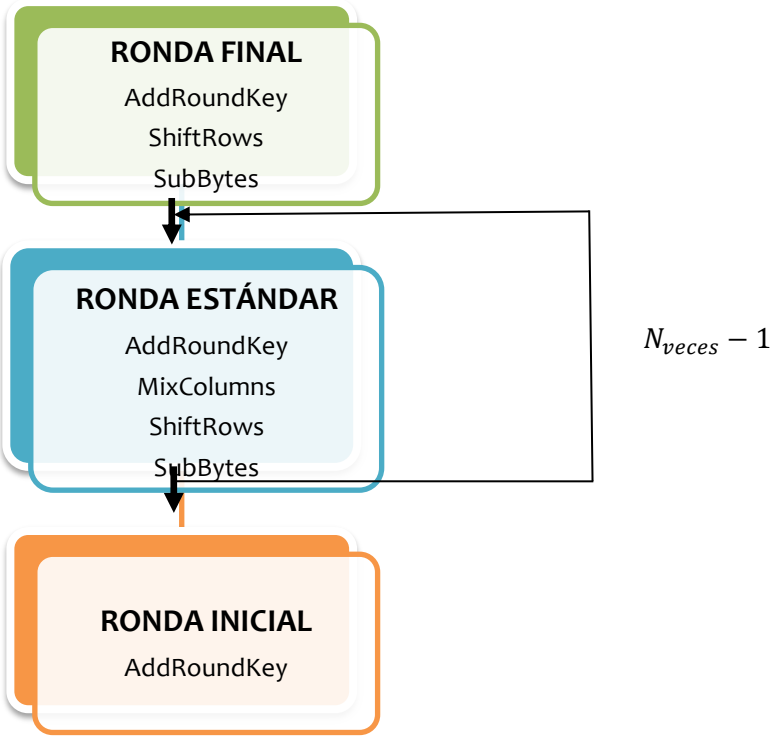


Figura N° 4.3. 13: Descifrado AES.

El descifrado AES es el proceso inverso del cifrado, por lo cual para que sea entendible, se comenzará por la explicación de la ronda final, no sin antes especificar que la obtención de las subclaves se realiza exactamente igual que en el cifrado, por lo cual, estas no sufren cambio alguno. Antes, ejecute:

python descifrado_aes --datos»

Coloque una captura de pantalla de lo mostrado en el recuadro *«Actividad 4.3: Datos descifrado»* que se encuentra en la práctica.

RONDA FINAL: Al ser el proceso inverso, recuerde, ¿Cuál fue el último paso realizado en le cifrado? Fue la realización de la operación XOR(⊕) entre el resultado de *Shift Rows* y la *Subclave[10]*, es decir *AddRoundKey*. Ahora, lo que tenemos es el mensaje cifrado, como se menciono, las subclaves no sufren cambios por lo cual son las mismas a utilizar, entonces se realiza la operación ⊕ entre el *mensaje cifrado* y la *Subclave[10]*.

ACLARACIÓN: Para que no exista confusión, como entradas, el descifrado recibe una clave y un mensaje cifrado, para poder realizar satisfactoriamente el descifrado, **la clave debe ser la misma** que se utilizó en el cifrado, por lo cual las subclaves puedes obtenerse a partir de la clave original y alojarse en alguna lista, para después hacer uso de ellas. Ejecute:

« python descifrado_aes --subclaves»

Lo que se muestra son las diez subclaves alojadas en una lista llamada subclaves. Cada una de estas será utilizada cuando se requiera, para este caso, se necesita la número diez, después se hará uso de la nueve y así sucesivamente.

Coloque una captura de pantalla de lo mostrado en el recuadro *«Actividad 4.3: Subclaves descifrado»* que se encuentra en la práctica.

Ahora, ejecute:

« python descifrado_aes --daddroundkey»

Coloque una captura de pantalla de lo mostrado en el recuadro *«Actividad 4.3: AddRoundKey descifrado»* que se encuentra en la práctica.

Comparé, ¿El resultado mostrado es el mismo al último paso realizado en el cifrado?

Siguiendo la lógica de ser el proceso inverso, los siguientes pasos a realizar son:

ShiftRows: En el caso de *ShiftRows,* la rotación de bytes ahora es hacia el lado izquierdo, como se muestra en la Figura N° 4.3.14, a este método se le conoce como *InvShiftRows*.

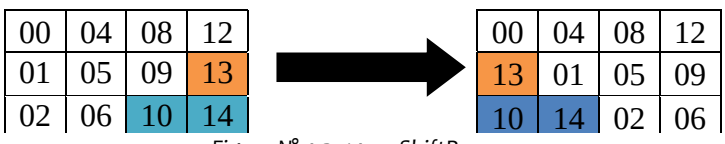


Figura N° 4.3. 14: nvShiftRows.

Recuerde que se hace uso de los valores obtenidos en AddRoundKey. Ejecute:

« python descifrado_aes --invsrows»

Coloque una captura de pantalla de lo mostrado en el recuadro *«Actividad 4.3: Inv Shift Rows»* que se encuentra en la práctica.

SubBytes: Conocido como *InvSubBytes*, realiza el mismo procedimiento que *SubBytes* en el cifrado, solo que ahora, hace uso de una matriz diferente. En la Figura N° 4.3.15 se muestra la matriz a utilizar.

		y															
		0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
x	0	52	09	6a	d5	30	36	a5	38	bf	40	a3	9e	81	f3	d7	fb
	1	7c	e3	39	82	9b	2f	ff	87	34	8e	43	44	c4	de	e9	cb
	2	54	7b	94	32	a6	c2	23	3d	ee	4c	95	0b	42	fa	c3	4e
	3	08	2e	a1	66	28	d9	24	b2	76	5b	a2	49	6d	8b	d1	25
	4	72	f8	f6	64	86	68	98	16	d4	a4	5c	cc	5d	65	b6	92
	5	6c	70	48	50	fd	ed	b9	da	5e	15	46	57	a7	8d	9d	84
	6	90	d8	ab	00	8c	bc	d3	0a	f7	e4	58	05	b8	b3	45	06
	7	d0	2c	1e	8f	ca	3f	0f	02	c1	af	bd	03	01	13	8a	6b
	8	3a	91	11	41	4f	67	dc	ea	97	f2	cf	ce	f0	b4	e6	73
	9	96	ac	74	22	e7	ad	35	85	e2	f9	37	e8	1c	75	df	6e
	a	47	f1	1a	71	1d	29	c5	89	6f	b7	62	0e	aa	18	be	1b
	b	fc	56	3e	4b	c6	d2	79	20	9a	db	c0	fe	78	cd	5a	f4
	c	1f	dd	a8	33	88	07	c7	31	b1	12	10	59	27	80	ec	5f
	d	60	51	7f	a9	19	b5	4a	0d	2d	e5	7a	9f	93	c9	9c	ef
	e	a0	e0	3b	4d	ae	2a	f5	b0	c8	eb	bb	3c	83	53	99	61
	f	17	2b	04	7e	ba	77	d6	26	e1	69	14	63	55	21	0c	7d

Figura N° 4.3. 15: Matriz inversa SubBytes.

Recuerde que siempre hará uso de los datos obtenidos en el paso anterior. Ejecute:

« python descifrado_aes --invsubbytes»

Coloque una captura de pantalla de lo mostrado en el recuadro *«Actividad 4.3: Inv Shift Rows»* que se encuentra en la práctica.

Para visualizar los datos ejecute: *« python descifrado_aes --rfinal»*

RONDA ESTANDAR: La ronda estándar comenzará con *AddRoundKey*, entre la subclave correspondiente, es decir la subclave nueve, y el último resultado obtenido, es decir, el *InvSubBytes* de la ronda anterior.

Mix Columns: Se realiza el mismo procedimiento que en el cifrado, solo que la matriz fija a utilizar es la siguiente:

0e	0b	0d	09
09	0e	0b	0d
0d	09	0e	0b
0b	0d	09	0e

Figura N° 4.3. 16: Matriz fija Inv MixColumns.

Recuerde que se utilizan los valores obtenidos en el paso anterior inmediato. Ejecute:

« python descifrado_aes --invmcolumns»

Coloque una captura de pantalla de lo mostrado en el recuadro «*Actividad 4.3: Inv Mix Columns*» que se encuentra en la práctica.

Los siguientes dos pasos ya los conoce, se realiza ShiftRows y SubBytes. El proceso se realiza nueve veces. Ejecute:

« python descifrado_aes --restandar»

Llene la tabla «*Ronda 9 – Ronda 1 Inversa*» que se encuentra en su práctica de acuerdo a los datos obtenidos al ejecutar lo anterior.

Visualice esta tabla con la tabla «*Actividad 4.3: Ronda 1 – Ronda 9*». ¿Hay similitudes?

RONDA INICIAL: Finalmente se realiza la operación *AddRoundKey* entre la clave original y el último conjunto de bytes arrojado de la ronda uno, es decir, los valores correspondientes a *SubBytes*, finalmente se obtiene el mensaje original. Ejecute:

« python cifrado_aes --rinicial»

Coloque una captura de pantalla de lo mostrado en el recuadro «*Actividad 4.3: rinicial*» que se encuentra en la práctica.

Como observo, el descifrado es exactamente el proceso inverso al cifrado, solo tenga cuidado en hacer uso correcto de matrices para cada uno de los casos.

Anexo G: Archivo instrucciones: Firma ElGamal

En máquina virtual

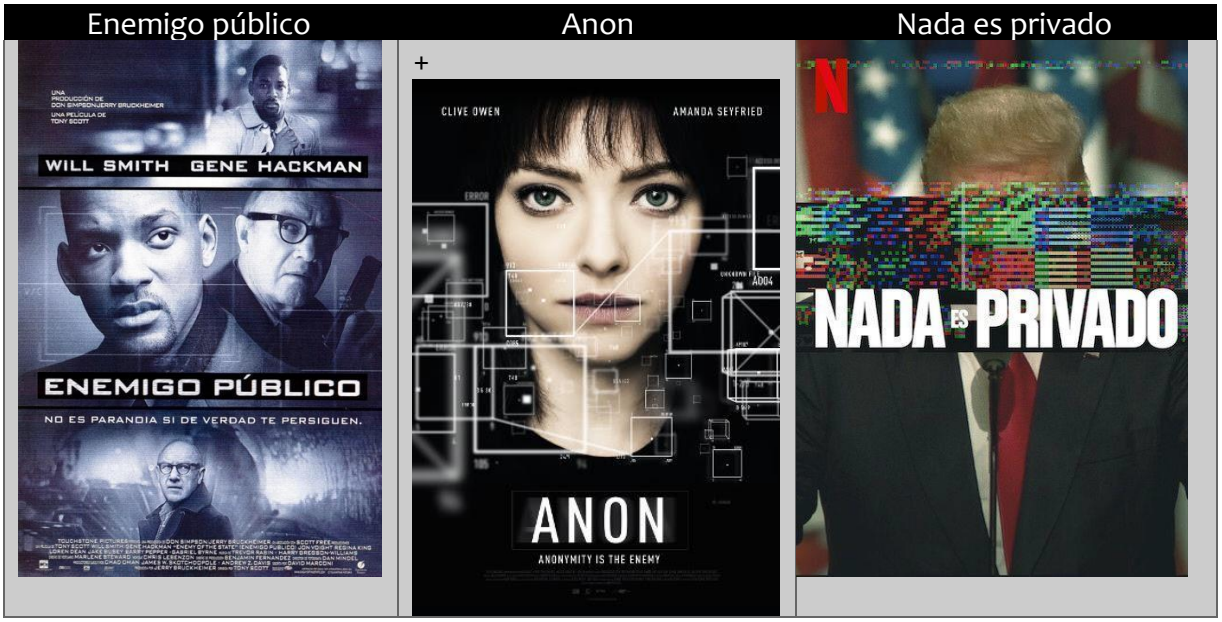


Figura N° 8.2. 1: Películas y documentales.

Vea una película o documental referente a temas de ciberseguridad, se recomienda cualquiera de las mostradas en la Figura 8.2.1, pero usted puede escoger cualquier otra que prefiera.

En un nuevo archivo, cuyo nombre debe seguir la siguiente nomenclatura:

Comentario_INICIALES

Donde INICIALES son las iniciales de su nombre comenzado por apellido. Escriba un comentario de al menos 10 reglones de la película o documental seleccionado. Puede realizar su comentario en un archivo .txt, pdf, docx, el que sea de su preferencia.

Firme el archivo usando su script realizado en la Actividad 8.1, cifre el archivo y obtenga base64 del nombre de su archivo. Coloque el resultado de base64 en un nuevo archivo llamado:

FirmaElGamal_INICIALES

Donde nuevamente INICIALES son las iniciales de su nombre comenzando por apellido. En otra cadena debe colocar los datos necesarios para que el receptor pueda comprobar la firma, y descifrar el archivo, de esta cadena obtenga también Base64. ***Use datos diferentes a los mostrados en los ejemplos.***

Debe entregar los datos, tal como le fueron entregados para realizar esta actividad, es decir como se muestra en la Figura N° 8.12 de la práctica.

Envié los archivos necesarios a su docente.

Anexo H: Archivo instrucciones: Firma RSA

En máquina virtual

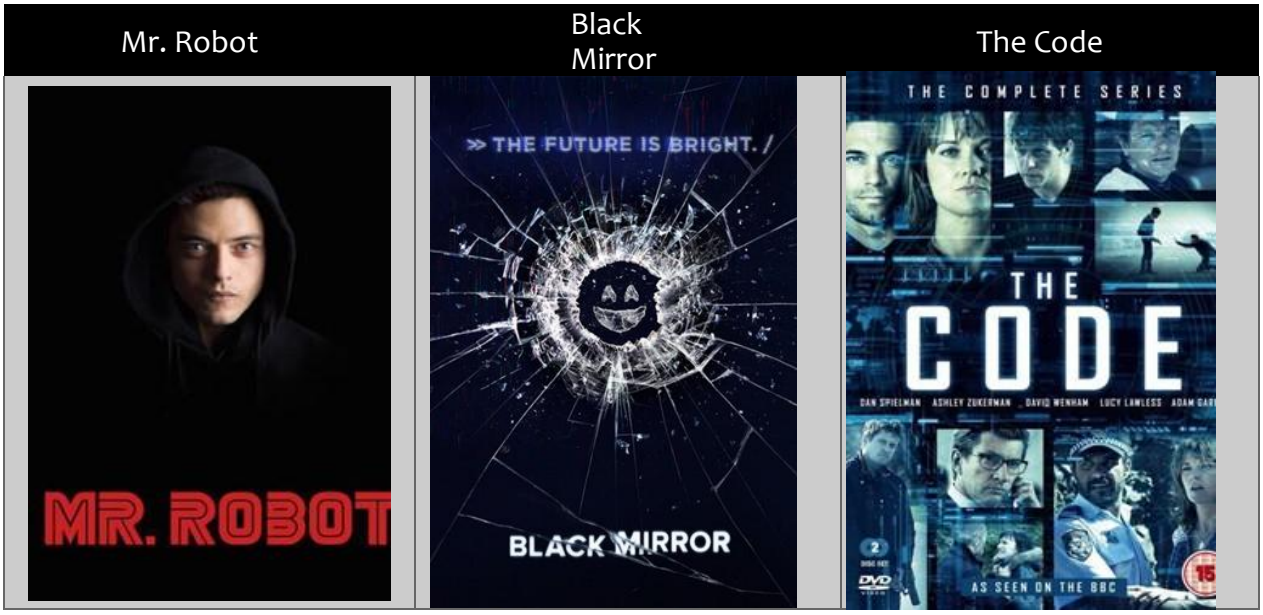


Figura N° 8.4. 1: Series.

Vea un capítulo de alguna serie referente a temas de ciberseguridad, se recomienda cualquiera de las mostradas en la Figura 8.4.1, pero usted puede escoger cualquier otra que prefiera.

En caso de escoger alguna de la Figura N° 8.4.1, de las series Mr.Robot o The Code se recomienda ver el primer capítulo, ya que las series son secuenciales. De Black Mirror puede escoger el que guste.

En un nuevo archivo, cuyo nombre debe seguir la siguiente nomenclatura:

RSAComentario_INICIALES

Donde INICIALES son las iniciales de su nombre comenzado por apellido. Escriba un comentario de al menos 10 reglones de la serie seleccionada. Puede realizar su comentario en un archivo .txt, pdf, docx, el que sea de su preferencia.

Firme el archivo usando su script realizado en la Actividad 8.3, cifre el archivo y obtenga base64 del nombre de su archivo. Coloque el resultado de base64 en un nuevo archivo llamado:

FirmaRSA_INICIALES

Donde nuevamente INICIALES son las iniciales de su nombre comenzando por apellido. En otra cadena debe colocar los datos necesarios para que el receptor pueda comprobar la firma, y descifrar el archivo, de esta cadena obtenga también Base64. ***Use datos diferentes a los mostrados en los ejemplos.***

Debe entregar los datos, tal como le fueron entregados para realizar esta actividad. Envié los archivos necesarios a su docente.

Anexo I: Archivo instrucciones – Actividad I: Proof of Work

En máquina virtual

La criptomoneda que va a trabajar se llama **cripty**, es una criptomoneda que trabaja mediante el método de PoW.

Un punto a destacar es que los bloques no pueden ser vulnerados ya que cada bloque contiene el hash del bloque anterior, pero ¿Qué pasa con el primer bloque? Para el primer bloque, que se denomina como el «*bloque génesis*» cada una de las criptomonedas, constituye su propio bloque génesis.

Bloque cero o bloque génesis: Es el primer bloque de cualquier blockchain, y cada blockchain indica cuál es su primer bloque, recuerde que blockchain tiene dentro de cada bloque el hash del bloque anterior, es así como se evita romper con la integridad de los bloques.

El valor hash del bloque génesis para la criptomoneda cripty es:

07f3269921b0ef86f8d837abbcbf4d3d09ccad9580c399c613ac5a0671d8b6b4

Este es el valor hash del bloque anterior para el primer bloque.

En la criptomoneda cripty, cada bloque se comprende de:

a. 10 transacciones
b. Hash del bloque anterior
c. Nonce
d. Hash del bloque correspondiente

Cada una de las transacciones sigue la siguiente nomenclatura:

Emisor – receptor: cantidad de cripty enviada – fecha hora*

Cada línea termina con un asterisco (*****) al final, éste funciona como delimitador.

Pero la incógnita a resolver por los mineros es ¿Cuál es el valor nonce que cumpla con las condiciones del hash del bloque? Es por eso que los mineros necesitan tener un poder computación alto, ya que va a depender de la criptomoneda, cual sea *el reto por resolver*.

Para **cripty**, el reto a resolver es buscar el valor nonce que cumpla con las condiciones del hash.

Para ejemplificar, se presenta en la Figura N° 10.1.1 el primer bloque de datos:

Aj - Kt: 10 Criptys – 23/06/2023 - 12:53 hrs*	**Fe - Po: 13 Criptys – 23/06/2023 - 16:09 hrs***
Bi - Ls: 2 Criptys – 23/06/2023 - 13:49 hrs*	**Gd - Qn: 7 Criptys – 23/06/2023 - 17:39 hrs***
Ch - Mr: 3 Criptys – 23/06/2023 - 14:03 hrs*	**Hc - Rm: 2 Criptys – 23/06/2023 - 18:07 hrs***
Dg - Nq: 0.5 Criptys – 23/06/2023 - 14:34 hrs*	**Ib - Sl: 9 Criptys – 23/06/2023 - 20:57 hrs***
Ef - Op: 5 Criptys – 23/06/2023 - 15:25 hrs*	**Ja - Tk: 4 Criptys – 23/06/2023 - 23:09 hrs***
07f3269921b0ef86f8d837abbcbf4d3d09ccad9580c399c613ac5a0671d8b6b4*	
NONCE*	
071846fb0cd6a93924634fba9fe5e1eecc92f42e05de88b3834dd851844e31**b5**	

Figura N° 10.1. 1: Primer bloque de datos.

Como se observa en la Figura N° 10.1.1, el bloque tiene 2 columnas en donde la primera *(columna de color naranja)* contiene las primeras cinco transacciones, y la segunda *(columna de color verde)* las siguientes cinco transacciones, así corresponden a las 10 transacciones, posteriormente tiene el hash del bloque anterior, que en este caso, al ser el primer bloque, el hash que tiene es el hash del bloque génesis, seguido del valor NONCE y finalmente el hash completo del bloque.

De todos los bloques, el único valor que para el ejemplo no se conoce es el valor de NONCE, por lo cual el reto es buscar ese valor nonce, que cumpla con las condiciones del hash del bloque.

Por lo cual, se le pide realice lo siguiente: de cada bloque genere una cadena tomando en cuenta el orden de las transacciones, recuerde que las primeras cinco corresponden a la primera columna, las siguientes cinco corresponden a la segunda columna, el hash del bloque anterior y se añade **un valor** que es el valor NONCE que se está buscando, cada línea termina con un asterisco (*), y a toda esta cadena se le obtiene el hash, ya conoce que el hash utilizado es SHA256, de manera que se obtiene y se compara si cumple con las condiciones del hash del bloque, si es así el valor corresponde al valor NONCE, si no, se continúa buscando este valor. Lo descrito se muestra en la Figura N° 10.1.2:

Aj - Kt: 10 Criptys – 23/06/2023 - 12:53 hrs*Bi - Ls: 2 Criptys – 23/06/2023 - 13:49 hrs*Ch - Mr: 3 Criptys – 23/06/2023 - 14:03 hrs*Dg - Nq: 0.5 Criptys – 23/06/2023 - 14:34 hrs*Ef - Op: 5 Criptys – 23/06/2023 - 15:25 hrs*Fe - Po: 13 Criptys – 23/06/2023 - 16:09 hrs*Gd - Qn: 7 Criptys – 23/06/2023 - 17:39 hrs*Hc - Rm: 2 Criptys – 23/06/2023 - 18:07 hrs*Ib - Sl: 9 Criptys – 23/06/2023 - 20:57 hrs*Ja - Tk: 4 Criptys – 23/06/2023 - 23:09 hrs*07f3269921b0ef86f8d837abbcbf4d3d09ccad9580c399c613ac5a0671d8b6b4*1*

Figura N° 10.1. 2: Datos del primer bloque acomodados en cadena.

En la Figura N° 10.1.2 se muestra el bloque acomodado en cadena y agregando un valor, que corresponde al valor de NONCE, se obtiene el hash y éste tiene como resultado:

$$45e1a70dd4410e1bb9c611a65e2dadb29b9716f8d4854fe155eb42e6d0893a0d$$
$$\neq$$
$$071846fb0cd6a93924634fba9fe5e1eecc92f42e05de88b3834dd851844e31b5$$

Dado que el resultado es diferente a las condiciones del hash proporcionado, el valor no corresponde y debe ser cambiado hasta obtener el hash que cumpla con las condiciones proporcionadas.

Tenga cuidado en **no** agregar caracteres o espacios adicionales, recuerde que el hash cambia si la entrada cambia por mínima que sea.

Finalmente, el **primer bloque** queda de la siguiente manera (véase Figura N° 10.1.3):

Aj - Kt: 10 Criptys – 23/06/2023 - 12:53 hrs*	Fe - Po: 13 Criptys – 23/06/2023 - 16:09 hrs*
Bi - Ls: 2 Criptys – 23/06/2023 - 13:49 hrs*	Gd - Qn: 7 Criptys – 23/06/2023 - 17:39 hrs*
Ch - Mr: 3 Criptys – 23/06/2023 - 14:03 hrs*	Hc - Rm: 2 Criptys – 23/06/2023 - 18:07 hrs*
Dg - Nq: 0.5 Criptys – 23/06/2023 - 14:34 hrs*	Ib - Sl: 9 Criptys – 23/06/2023 - 20:57 hrs*
Ef - Op: 5 Criptys – 23/06/2023 - 15:25 hrs*	Ja - Tk: 4 Criptys – 23/06/2023 - 23:09 hrs*
07f3269921b0ef86f8d837abbcbf4d3d09ccad9580c399c613ac5a0671d8b6b4*	
386380*	
071846fb0cd6a93924634fba9fe5e1eecc92f42e05de88b3834dd851844e31b5	

Figura N° 10.1. 3: Datos completos de primer bloque.

En donde el valor NONCE correspondiente al hash es: **386380**

A continuación, se muestran los valores para el segundo y tercer bloque:

Segundo bloque (véase Figura N° 10.1.4):

Hc - Rm: 12 Criptys – 24/06/2023 - 00:07 hrs*	Fe - Po: 7 Criptys – 24/06/2023 - 07:09 hrs*
Ib - Sl: 3 Criptys – 24/06/2023 - 00:57 hrs*	Gd - Qn: 4 Criptys – 24/06/2023 - 07:39 hrs*
Ja - Tk: 9 Criptys – 24/06/2023 - 01:09 hrs*	Ch - Mr: 2 Criptys – 24/06/2023 - 08:03 hrs*
Dg - Nq: 2 Criptys – 24/06/2023 - 02:34 hrs*	Aj - Kt: 15 Criptys – 24/06/2023 - 10:53 hrs*
Ef - Op: 5 Criptys – 24/06/2023 - 05:25 hrs*	Bi - Ls: 4 Criptys – 24/06/2023 - 12:49 hrs*
071846fb0cd6a93924634fba9fe5e1eecc92f42e05de88b3834dd851844e31b5*	
497778*	
0720a45911c4ab832cb89d948c20923a4e53fb669aff632e1c3e12c587ffb4b6	

Figura N° 10.1. 4: Datos completos de segundo bloque.

Tercer bloque (véase Figura N° 10.1.5):

Ch - Mr: 1 Criptys – 25/06/2023 - 15:03 hrs*	Dg - Nq: 0.5 Criptys – 25/06/2023 - 19:34 hrs*

Aj - Kt: 2 Criptys – 25/06/2023 - 15:53 hrs*	Ef - Op: 3 Criptys – 25/06/2023 - 20:25 hrs*
Bi - Ls: 4 Criptys – 25/06/2023 - 16:49 hrs*	Hc - Rm: 9 Criptys – 25/06/2023 - 21:07 hrs*
Fe - Po: 6 Criptys – 25/06/2023 - 17:09 hrs*	Ib - Sl: 10 Criptys – 25/06/2023 - 21:57 hrs*
Gd - Qn: 8 Criptys – 25/06/2023 - 17:39 hrs*	Ja - Tk: 0.5 Criptys – 25/06/2023 - 23:09 hrs*
0720a45911c4ab832cb89d948c20923a4e53fb669aff632e1c3e12c587ffb4b6*	
1338052*	
073db5354ba992ed6d1015a5d505eebf2d2390304c82ed18c22951d402ec51b7	

Figura N° 10.1. 5: Datos completos de tercer bloque.

Para verificar si se cumple con las condiciones del hash del bloque es importante que tener presente que en los tres primeros bloques, se mostró el hash completo del bloque, sin embargo, cada hash tiene una particularidad, ¿Puede identificarla? El bloque génesis inicia con **07** e incrementa el siguiente número de acuerdo con el número de bloque, es por eso que el primer bloque tiene como inicio **071**, el segundo bloque **072** y así sucesivamente.

Del mismo modo al final de cada hash, dado que el bloque génesis termina en ***b*4**, el primer bloque termina en ***b*5**, el segundo bloque en ***b*6** y así de manera consecutiva.

Esta es la particularidad de la criptomoneda Cripty, y debe tomarla en cuenta para la búsqueda del NONCE y la obtención del hash del bloque que corresponda.

A partir del quinto bloque, debe encontrar el valor NONCE correspondiente y el hash con las particularidades ya mostradas. Verá que conforme se avance en los bloques el tiempo de procesamiento será mayor.

Realice un script que le ayude a buscar ese valor NONCE, el hash correspondiente del bloque y además muestre el tiempo que tardó en encontrar dichos valores. Su script debe preguntar desde qué valor NONCE comienza a buscar, el valor no puede ser menor a **999**, si el usuario no ingresa ningún valor, debe comenzar a buscar desde este valor, es decir 999.

Recuerde comentar todo su script, tome en cuenta los posibles errores y hágalo llegar a su docente como se le indique.

Usando el cuarto bloque como ejemplo, se muestra una manera en la cual puede obtener el hash, este es un script que pide ya en cadena el contenido del bloque, hasta el hash del bloque anterior, ya que el script es el que irá iterando con los números, y como ya se conoce cuál es la estructura del hash, se sabe que para el cuarto bloque su hash debe iniciar con **074** y terminar con **b8** por lo cual el mismo script lo determina.

Los valores del **cuarto bloque** son (véase Figura N° 10.1.6)

Hc - Rm: 9 Criptys – 26/06/2023 – 03:37 hrs*	Bi - Ls: 4 Criptys – 26/06/2023 - 08:49 hrs*
Ib - Sl: 10 Criptys – 26/06/2023 - 04:57 hrs*	Dg - Nq: 0.5 Criptys – 26/06/2023 - 09:34 hrs*
Ja - Tk: 0.5 Criptys – 26/06/2023 - 06:09 hrs*	Fe - Po: 6 Criptys – 26/06/2023 - 11:09 hrs*
Ch - Mr: 1 Criptys – 26/06/2023 - 07:03 hrs*	Gd - Qn: 8 Criptys – 26/06/2023 - 12:39 hrs*
Aj - Kt: 2 Criptys – 26/06/2023 - 07:53 hrs*	Ef - Op: 3 Criptys – 26/06/2023 - 14:25 hrs*
073db5354ba992ed6d1015a5d505eebf2d2390304c82ed18c22951d402ec51b7*	
NONCE*	
074.. ……. b8	

Figura N° 10.1. 6: Datos de cuarto bloque.

Así, al ingresar los datos en el script, el resultado es el mostrado en la Figura N° 10.1.7

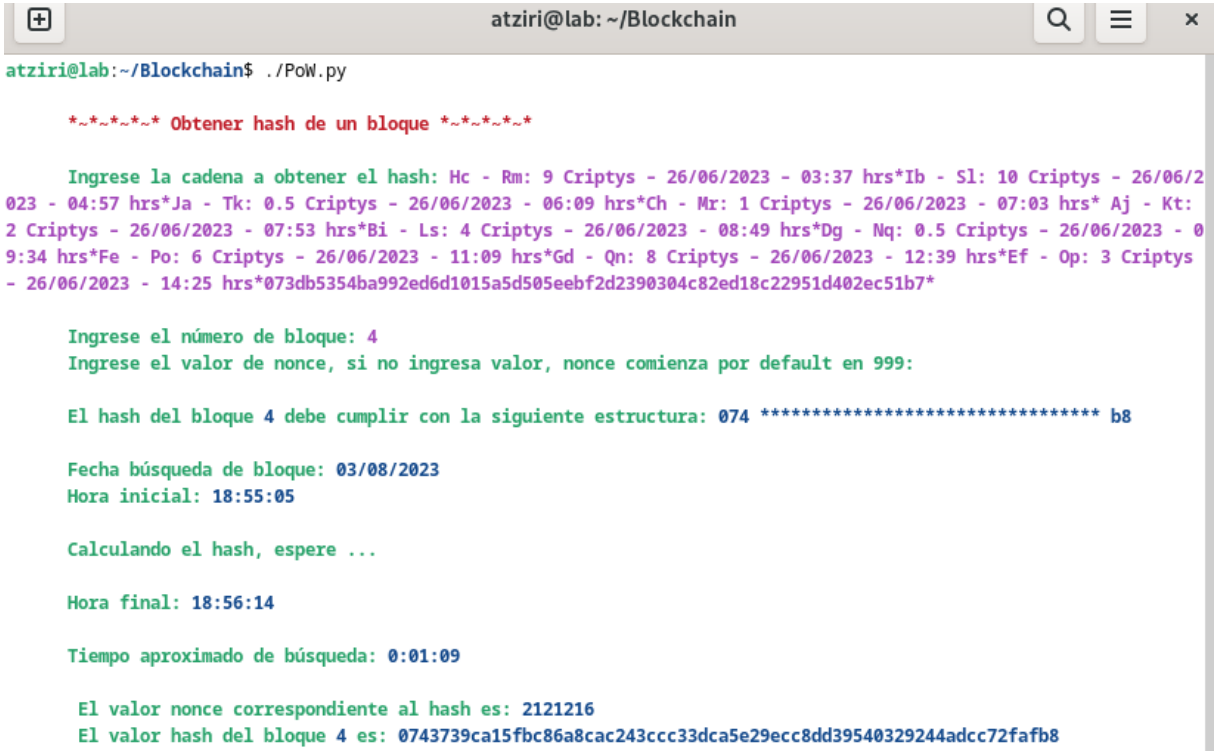


Figura N° 10.1. 7: Uso de script para cuarto bloque.

El hash obtenido cumple con las condiciones, por lo cual se procede al siguiente bloque, recuerde que debe ir incorporando el hash del bloque anterior al nuevo bloque.

En la Figura N° 10.1.8 se muestra el fragmento del script donde se observa el tiempo que llevó buscar el hash, el valor nonce y el hash del bloque.

```
Tiempo aproximado de búsqueda: 0:00:11

 El valor nonce correspondiente al hash es: 386380
 El valor hash del bloque 1 es: 071846fb0cd6a93924634fba9fe5e1eecc92f42e05de88b3834dd851844e31b5

Tiempo aproximado de búsqueda: 0:00:15

 El valor nonce correspondiente al hash es: 497778
 El valor hash del bloque 2 es: 0720a45911c4ab832cb89d948c20923a4e53fb669aff632e1c3e12c587ffb4b6

Tiempo aproximado de búsqueda: 0:00:38

 El valor nonce correspondiente al hash es: 1338052
 El valor hash del bloque 3 es: 073db5354ba992ed6d1015a5d505eebf2d2390304c82ed18c22951d402ec51b7
```

Figura N° 10.1. 8: Datos de primer a tercer bloque.

En la práctica coloque una captura de pantalla donde corresponda, dicha captura debe mostrar la ejecución de su script para cada uno de los bloques.

En el apartado correspondiente, coloque el hash para el bloque y el valor nonce encontrado.

Los bloques se muestran a continuación:

Quinto bloque (véase Figura N° 10.1.9):

Ja - Tk: 0.5 Criptys – 27/06/2023 - 06:09 hrs*	Bi - Ls: 4 Criptys – 27/06/2023 - 13:49 hrs*
Ch - Mr: 1 Criptys – 27/06/2023 - 07:03 hrs*	Dg - Nq: 0.5 Criptys – 27/06/2023 - 14:34 hrs*
Aj - Kt: 2 Criptys – 27/06/2023 - 07:53 hrs*	Hc - Rm: 9 Criptys – 27/06/2023 – 15:37 hrs*
Fe - Po: 6 Criptys – 27/06/2023 - 11:09 hrs*	Ib - Sl: 10 Criptys – 27/06/2023 - 15:57 hrs*
Gd - Qn: 8 Criptys – 27/06/2023 - 12:39 hrs*	Ef - Op: 3 Criptys – 27/06/2023 - 20:25 hrs*
0743739ca15fbc86a8cac243ccc33dca5e29ecc8dd39540329244adcc72fafb8*	
¿?*	
075.. ……..b9	

Figura N° 10.1. 9: Datos quinto bloque.

Sexto bloque (véase Figura N° 10.1.10):

Bi - Ls: 0.1 Criptys – 28/06/2023 - 08:49 hrs*	Fe - Po: 0.5 Criptys – 28/06/2023 - 16:09 hrs*
Gd - Qn: 0.4 Criptys – 28/06/2023 - 11:39 hrs*	Dg - Nq: 0.3 Criptys – 28/06/2023 - 18:34 hrs*
Aj - Kt: 0.7 Criptys – 28/06/2023 - 12:53 hrs*	Ef - Op: 0.9 Criptys – 28/06/2023 - 20:25 hrs*
Hc - Rm: 0.9 Criptys – 28/06/2023 – 14:37 hrs*	Ja - Tk: 2 Criptys – 28/06/2023 - 22:09 hrs*
Ib - Sl: 0.8 Criptys – 28/06/2023 - 15:57 hrs*	Ch - Mr: 0.4 Criptys – 28/06/2023 - 23:03 hrs*
075.. ……..**b9***	
¿?*	
076.. ……..**b10**	

Figura N° 10.1. 10: Datos sexto bloque.

Séptimo bloque (véase Figura N° 10.1.11):

Hc - Rm: 3 Criptys – 29/06/2023 – 03:37 hrs*	Gd - Qn: 1 Criptys – 29/06/2023 - 15:39 hrs*
Bi - Ls: 5 Criptys – 29/06/2023 - 08:49 hrs*	Ef - Op: 5 Criptys – 29/06/2023 - 16:25 hrs*
Dg - Nq: 0.5 Criptys – 29/06/2023 - 09:34 hrs*	Ib - Sl: 15 Criptys – 29/06/2023 - 17:57 hrs*
Fe - Po: 8 Criptys – 29/06/2023 - 11:09 hrs*	Ja - Tk: 3 Criptys – 29/06/2023 - 20:09 hrs*
Aj - Kt: 2 Criptys – 29/06/2023 - 14:53 hrs*	Ch - Mr: 9 Criptys – 29/06/2023 - 22:03 hrs*
076.. ……..**b10***	
¿?*	
077.. ……..**b11**	

Figura N° 10.1. 11: Datos séptimo bloque.

Octavo bloque (véase Figura N° 10.1.12):

Hc - Rm: 2 Criptys – 30/06/2023 – 03:37 hrs*	Ch - Mr: 6 Criptys – 30/06/2023 - 15:03 hrs*
Bi - Ls: 5 Criptys – 30/06/2023 - 08:49 hrs*	Aj - Kt: 4 Criptys – 30/06/2023 - 15:53 hrs*
Dg - Nq: 7 Criptys – 30/06/2023 - 09:34 hrs*	Fe - Po: 3 Criptys – 30/06/2023 - 17:09 hrs*
Gd - Qn: 9 Criptys – 30/06/2023 - 12:39 hrs*	Ib - Sl: 1 Criptys – 30/06/2023 - 18:57 hrs*
Ef - Op: 8 Criptys – 30/06/2023 - 14:25 hrs*	Ja - Tk: 3 Criptys – 30/06/2023 -20:09 hrs*
077.. ……..**b11***	
¿?*	
078.. ……..***b12***	

Figura N° 10.1. 12: Datos séptimo bloque.

Noveno bloque (véase Figura N° 10.1.13):

Ef - Op: 1 Criptys – 01/07/2023 - 05:25 hrs*	Gd - Qn: 10 Criptys – 01/07/2023 - 12:39 hrs*
Aj - Kt: 2 Criptys – 01/07/2023 - 07:53 hrs*	Hc - Rm: 9 Criptys – 01/07/2023 – 13:37 hrs*
Ja - Tk: 3 Criptys – 01/07/2023 - 09:09 hrs*	Ib - Sl: 8 Criptys – 01/07/2023 - 14:57 hrs*
Fe - Po: 4 Criptys – 01/07/2023 - 11:09 hrs*	Bi - Ls: 7 Criptys – 01/07/2023 - 18:49 hrs*
Ch - Mr: 5 Criptys – 01/07/2023 - 12:03 hrs*	Dg - Nq: 6 Criptys – 01/07/2023 - 19:34 hrs*
078.. ……..**b12***	
¿?*	
079.. ……..**b13**	

Figura N° 10.1. 13: Datos noveno bloque.

Décimo bloque (véase Figura N° 10.1.14):

Fe - Po: 5 Criptys – 02/07/2023 - 11:09 hrs*	Aj - Kt: 7 Criptys – 02/07/2023 - 15:53 hrs*
Gd - Qn: 3 Criptys – 02/07/2023 - 12:39 hrs*	Hc - Rm: 8 Criptys – 02/07/2023 – 16:37 hrs*
Ja - Tk: 4 Criptys – 02/07/2023 - 13:09 hrs*	Ib - Sl: 10 Criptys – 02/07/2023 -17:57 hrs*
Ch - Mr: 1 Criptys – 02/07/2023 - 14:03 hrs*	Bi - Ls: 9 Criptys – 02/07/2023 - 20:49 hrs*
Ef - Op: 2 Criptys – 02/07/2023 - 14:25 hrs*	Dg - Nq: 5 Criptys – 02/07/2023 - 23:34 hrs*
079. ……..**b13***	
¿?*	
0710. ……..**b14**	

Figura N° 10.1. 14: Datos décimo bloque.

Anexo J: Archivo instrucciones – Actividad II: Proof of Work

En máquina virtual

En la Actividad 10.2, continuará trabajando con la criptomoneda **cripty**, los bloques permanecen con la misma estructura, solo, se añade el campo de Recompensa, tal como se desglosa a continuación.

El valor hash del bloque génesis para la criptomoneda cripty se conserva:

07f3269921b0ef86f8d837abbcbf4d3d09ccad9580c399c613ac5a0671d8b6b4

Es este el valor hash del bloque anterior para el primer bloque

En la criptomoneda cripty, cada bloque se comprende de:

a. 10 transacciones
b. Hash del bloque anterior
c. Valor de la recompensa
d. Nonce
e. Hash del bloque correspondiente

Cada una de las transacciones sigue la siguiente nomenclatura:

Emisor – receptor: cantidad de cripty enviada – fecha hora*

Cada línea termina con un asterisco (*) al final, este funciona como delimitador.

Cabe destacar que el orden de las transacciones se sigue tomando de la misma manera que en la Actividad 10.1, las cinco columnas de la izquierda corresponden a las primeras cinco transacciones, las cinco columnas de la derecha corresponden a las siguientes cinco transacciones.

Del mismo modo que en la Actividad 10.1, se presentan a continuación los primeros tres bloques.

En la Figura N° 10.2.1 el primer bloque de datos:

Aj - Kt: 10 Criptys – 23/06/2023 - 12:53 hrs*	Fe - Po: 13 Criptys – 23/06/2023 - 16:09 hrs*
Bi - Ls: 2 Criptys – 23/06/2023 - 13:49 hrs*	Gd - Qn: 7 Criptys – 23/06/2023 - 17:39 hrs*
Ch - Mr: 3 Criptys – 23/06/2023 - 14:03 hrs*	Hc - Rm: 2 Criptys – 23/06/2023 - 18:07 hrs*
Dg - Nq: 0.5 Criptys – 23/06/2023 - 14:34 hrs*	Ib - Sl: 9 Criptys – 23/06/2023 - 20:57 hrs*
Ef - Op: 5 Criptys – 23/06/2023 - 15:25 hrs*	Ja - Tk: 4 Criptys – 23/06/2023 - 23:09 hrs*
07f3269921b0ef86f8d837abbcbf4d3d09ccad9580c399c613ac5a0671d8b6b4*	
Recompensa: 5 Criptys*	
NONCE*	
071.. …… **b5**	

Figura N° 10.2. 1: Datos primer bloque.

Dado que es un bloque con un dato más, el valor de NONCE ya no será el mismo, es por eso que los valores con respecto a la actividad anterior serán distintos.

Usando el script creado en la actividad anterior, el valor nonce y el hash correspondiente al primer bloque se muestra en la Figura N°10.2.2:

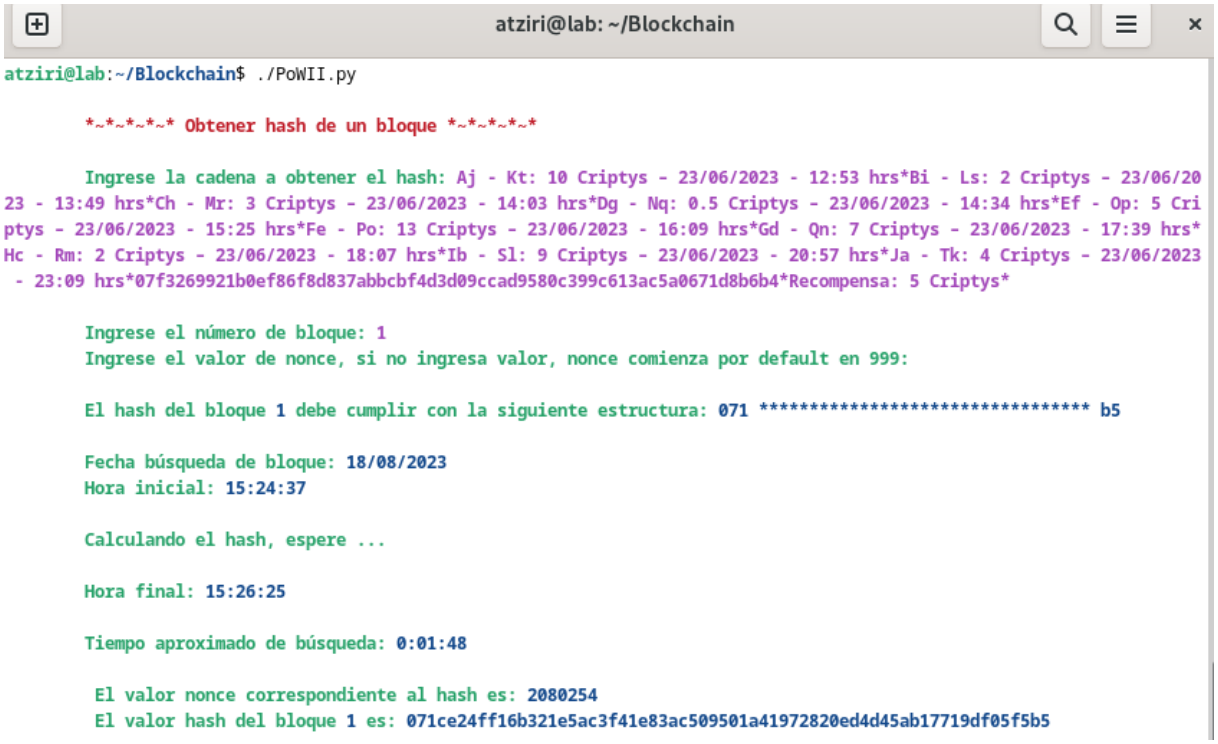


Figura N° 10.2. 2: Obtención de valor nonce y hash de primer bloque.

Por lo cual, el primer bloque completo se muestra en la Figura N° 10.2.3:

Aj - Kt: 10 Criptys – 23/06/2023 - 12:53 hrs*	Fe - Po: 13 Criptys – 23/06/2023 - 16:09 hrs*
Bi - Ls: 2 Criptys – 23/06/2023 - 13:49 hrs*	Gd - Qn: 7 Criptys – 23/06/2023 - 17:39 hrs*
Ch - Mr: 3 Criptys – 23/06/2023 - 14:03 hrs*	Hc - Rm: 2 Criptys – 23/06/2023 - 18:07 hrs*
Dg - Nq: 0.5 Criptys – 23/06/2023 - 14:34 hrs*	Ib - Sl: 9 Criptys – 23/06/2023 - 20:57 hrs*
Ef - Op: 5 Criptys – 23/06/2023 - 15:25 hrs*	Ja - Tk: 4 Criptys – 23/06/2023 - 23:09 hrs*
07f3269921b0ef86f8d837abbcbf4d3d09ccad9580c399c613ac5a0671d8b6b4*	
Recompensa: 5 Criptys*	
2080254*	
071ce24ff16b321e5ac3f41e83ac509501a41972820ed4d45ab17719df05f5b5	

Figura N° 10.2. 3: Datos completos de primer bloque.

Los valores correspondientes para el segundo y tercer bloque se muestran a continuación

Segundo bloque (véase Figura N° 10.2.4):

Hc - Rm: 12 Criptys – 24/06/2023 - 00:07 hrs*	Fe - Po: 7 Criptys – 24/06/2023 - 07:09 hrs*
Ib - Sl: 3 Criptys – 24/06/2023 - 00:57 hrs*	Gd - Qn: 4 Criptys – 24/06/2023 - 07:39 hrs*
Ja - Tk: 9 Criptys – 24/06/2023 - 01:09 hrs*	Ch - Mr: 2 Criptys – 24/06/2023 - 08:03 hrs*
Dg - Nq: 2 Criptys – 24/06/2023 - 02:34 hrs*	Aj - Kt: 15 Criptys – 24/06/2023 - 10:53 hrs*
Ef - Op: 5 Criptys – 24/06/2023 - 05:25 hrs*	Bi - Ls: 4 Criptys – 24/06/2023 - 12:49 hrs*
071ce24ff16b321e5ac3f41e83ac509501a41972820ed4d45ab17719df05f5b5*	
Recompensa: 8 Criptys*	
1622279*	
07226f243a5d5f1e3b6aec06de32bdd5a00504ccc9af2cb8624a8a6fe6c177b6	

Figura N° 10.2. 4: Datos completos de segundo bloque.

Tercer bloque (véase Figura N° 10.2.5):

Ch - Mr: 1 Criptys – 25/06/2023 - 15:03 hrs*	Dg - Nq: 0.5 Criptys – 25/06/2023 - 19:34 hrs*
Aj - Kt: 2 Criptys – 25/06/2023 - 15:53 hrs*	Ef - Op: 3 Criptys – 25/06/2023 - 20:25 hrs*
Bi - Ls: 4 Criptys – 25/06/2023 - 16:49 hrs*	Hc - Rm: 9 Criptys – 25/06/2023 - 21:07 hrs*
Fe - Po: 6 Criptys – 25/06/2023 - 17:09 hrs*	Ib - Sl: 10 Criptys – 25/06/2023 - 21:57 hrs*
Gd - Qn: 8 Criptys – 25/06/2023 - 17:39 hrs*	Ja - Tk: 0.5 Criptys – 25/06/2023 - 23:09 hrs*
07226f243a5d5f1e3b6aec06de32bdd5a00504ccc9af2cb8624a8a6fe6c177b6*	
Recompensa: 6 Criptys*	
34664*	
073bef7e6b9390c1e3853856a88f60b55f2bd452c7d76946be29b9c703e142b7	

Figura N° 10.2. 5: Datos completos de tercer bloque.

Los valores del **cuarto bloque** son (véase Figura N° 10.2.6)

Hc - Rm: 9 Criptys – 26/06/2023 – 03:37 hrs*	Bi - Ls: 4 Criptys – 26/06/2023 - 08:49 hrs*
Ib - Sl: 10 Criptys – 26/06/2023 - 04:57 hrs*	Dg - Nq: 0.5 Criptys – 26/06/2023 - 09:34 hrs*
Ja - Tk: 0.5 Criptys – 26/06/2023 - 06:09 hrs*	Fe - Po: 6 Criptys – 26/06/2023 - 11:09 hrs*
Ch - Mr: 1 Criptys – 26/06/2023 - 07:03 hrs*	Gd - Qn: 8 Criptys – 26/06/2023 - 12:39 hrs*
Aj - Kt: 2 Criptys – 26/06/2023 - 07:53 hrs*	Ef - Op: 3 Criptys – 26/06/2023 - 14:25 hrs*
073bef7e6b9390c1e3853856a88f60b55f2bd452c7d76946be29b9c703e142b7*	
Recompensa: 5 Criptys*	
NONCE*	
074.. ……. b8	

Figura N° 10.2. 6: Datos de cuarto bloque.

Por lo cual, al ingresar los datos en el script, el resultado es el mostrado en la Figura N° 10.2.7:

```
atziri@lab: ~/Blockchain
atziri@lab:~/Blockchain$ ./PoWII.py

        *~*~*~*~* Obtener hash de un bloque *~*~*~*~*

        Ingrese la cadena a obtener el hash: Hc - Rm: 9 Criptys - 26/06/2023 - 03:37 hrs*Ib - Sl: 10 Criptys - 26/06/20
23 - 04:57 hrs*Ja - Tk: 0.5 Criptys - 26/06/2023 - 06:09 hrs*Ch - Mr: 1 Criptys - 26/06/2023 - 07:03 hrs*Aj - Kt: 2 Cri
ptys - 26/06/2023 - 07:53 hrs*Bi - Ls: 4 Criptys - 26/06/2023 - 08:49 hrs*Dg - Nq: 0.5 Criptys - 26/06/2023 - 09:34 hrs
*Fe - Po: 6 Criptys - 26/06/2023 - 11:09 hrs*Gd - Qn: 8 Criptys - 26/06/2023 - 12:39 hrs*Ef - Op: 3 Criptys - 26/06/202
3 - 14:25 hrs*073bef7e6b9390c1e3853856a88f60b55f2bd452c7d76946be29b9c703e142b7*Recompensa: 5 Criptys*

        Ingrese el número de bloque: 4
        Ingrese el valor de nonce, si no ingresa valor, nonce comienza por default en 999:

        El hash del bloque 4 debe cumplir con la siguiente estructura: 074 ********************************** b8

        Fecha búsqueda de bloque: 18/08/2023
        Hora inicial: 15:50:48

        Calculando el hash, espere ...

        Hora final: 15:51:03

        Tiempo aproximado de búsqueda: 0:00:15

         El valor nonce correspondiente al hash es: 512383
         El valor hash del bloque 4 es: 0745aa629780aab19a9806f93e62ef46eca4e5ac229cf366bd560e5f1c7eb6b8
```

Figura N° 10.2. 7: Uso de script para cuarto bloque.

En la práctica coloque una captura de pantalla donde corresponda, dicha captura debe mostrar la obtención del hash utilizando su script para cada uno de los bloques.

En el apartado correspondiente coloque el hash para el bloque y el valor nonce encontrado.

Los bloques se muestran a continuación:

Quinto bloque (véase Figura N° 10.2.8):

Ja - Tk: 0.5 Criptys – 27/06/2023 - 06:09 hrs*	Bi - Ls: 4 Criptys – 27/06/2023 - 13:49 hrs*
Ch - Mr: 1 Criptys – 27/06/2023 - 07:03 hrs*	Dg - Nq: 0.5 Criptys – 27/06/2023 - 14:34 hrs*
Aj - Kt: 2 Criptys – 27/06/2023 - 07:53 hrs*	Hc - Rm: 9 Criptys – 27/06/2023 – 15:37 hrs*
Fe - Po: 6 Criptys – 27/06/2023 - 11:09 hrs*	Ib - Sl: 10 Criptys – 27/06/2023 - 15:57 hrs*
Gd - Qn: 8 Criptys – 27/06/2023 - 12:39 hrs*	Ef - Op: 3 Criptys – 27/06/2023 - 20:25 hrs*
0745aa629780aab19a9806f93e62ef46eca4e5ac229cf366bd560e5f1c7eb6b8*	
Recompensa: 6 Criptys*	
¿?*	
075.. ……..**b9**	

Figura N° 10.2. 8: Datos quinto bloque.

Sexto bloque (véase Figura N° 10.2.9):

Bi - Ls: 0.1 Criptys – 28/06/2023 - 08:49 hrs*	Fe - Po: 0.5 Criptys – 28/06/2023 - 16:09 hrs*
Gd - Qn: 0.4 Criptys – 28/06/2023 - 11:39 hrs*	Dg - Nq: 0.3 Criptys – 28/06/2023 - 18:34 hrs*
Aj - Kt: 0.7 Criptys – 28/06/2023 - 12:53 hrs*	Ef - Op: 0.9 Criptys – 28/06/2023 - 20:25 hrs*
Hc - Rm: 0.9 Criptys – 28/06/2023 – 14:37 hrs*	Ja - Tk: 2 Criptys – 28/06/2023 - 22:09 hrs*
Ib - Sl: 0.8 Criptys – 28/06/2023 - 15:57 hrs*	Ch - Mr: 0.4 Criptys – 28/06/2023 - 23:03 hrs*
075.. ……..**b9***	
Recompensa: 3 Criptys*	
¿?*	
076.. ……..**b10**	

Figura N° 10.2. 9: Datos sexto bloque.

Séptimo bloque (véase Figura N° 10.2.10):

Hc - Rm: 3 Criptys – 29/06/2023 – 03:37 hrs*	Gd - Qn: 1 Criptys – 29/06/2023 - 15:39 hrs*
Bi - Ls: 5 Criptys – 29/06/2023 - 08:49 hrs*	Ef - Op: 5 Criptys – 29/06/2023 - 16:25 hrs*
Dg - Nq: 0.5 Criptys – 29/06/2023 - 09:34 hrs*	Ib - Sl: 15 Criptys – 29/06/2023 - 17:57 hrs*
Fe - Po: 8 Criptys – 29/06/2023 - 11:09 hrs*	Ja - Tk: 3 Criptys – 29/06/2023 - 20:09 hrs*
Aj - Kt: 2 Criptys – 29/06/2023 - 14:53 hrs*	Ch - Mr: 9 Criptys – 29/06/2023 - 22:03 hrs*
076.. ……..**b10***	
Recompensa: 12 Criptys*	
¿?*	
077.. ……..**b11**	

Figura N° 10.2. 10: Datos séptimo bloque.

Octavo bloque (véase Figura N° 10.2.11):

Hc - Rm: 2 Criptys – 30/06/2023 – 03:37 hrs*	Ch - Mr: 6 Criptys – 30/06/2023 - 15:03 hrs*
Bi - Ls: 5 Criptys – 30/06/2023 - 08:49 hrs*	Aj - Kt: 4 Criptys – 30/06/2023 - 15:53 hrs*
Dg - Nq: 7 Criptys – 30/06/2023 - 09:34 hrs*	Fe - Po: 3 Criptys – 30/06/2023 - 17:09 hrs*
Gd - Qn: 9 Criptys – 30/06/2023 - 12:39 hrs*	Ib - Sl: 1 Criptys – 30/06/2023 - 18:57 hrs*
Ef - Op: 8 Criptys – 30/06/2023 - 14:25 hrs*	Ja - Tk: 3 Criptys – 30/06/2023 -20:09 hrs*
077.. ……..**b11***	
Recompensa: 3 Criptys*	
¿?*	
078.. ……..***b12***	

Figura N° 10.2. 11: Datos séptimo bloque.

Noveno bloque (véase Figura N° 10.2.12):

Ef - Op: 1 Criptys – 01/07/2023 - 05:25 hrs*	Gd - Qn: 10 Criptys – 01/07/2023 - 12:39 hrs*
Aj - Kt: 2 Criptys – 01/07/2023 - 07:53 hrs*	Hc - Rm: 9 Criptys – 01/07/2023 – 13:37 hrs*
Ja - Tk: 3 Criptys – 01/07/2023 - 09:09 hrs*	Ib - Sl: 8 Criptys – 01/07/2023 - 14:57 hrs*
Fe - Po: 4 Criptys – 01/07/2023 - 11:09 hrs*	Bi - Ls: 7 Criptys – 01/07/2023 - 18:49 hrs*
Ch - Mr: 5 Criptys – 01/07/2023 - 12:03 hrs*	Dg - Nq: 6 Criptys – 01/07/2023 - 19:34 hrs*
078.. ……..**b12***	
Recompensa: 7 Criptys*	
¿?*	
079.. ……..**b13**	

Figura N° 10.2. 12: Datos noveno bloque.

Décimo bloque (véase Figura N° 10.2.13):

Fe - Po: 5 Criptys – 02/07/2023 - 11:09 hrs*	Aj - Kt: 7 Criptys – 02/07/2023 - 15:53 hrs*
Gd - Qn: 3 Criptys – 02/07/2023 - 12:39 hrs*	Hc - Rm: 8 Criptys – 02/07/2023 – 16:37 hrs*
Ja - Tk: 4 Criptys – 02/07/2023 - 13:09 hrs*	Ib - Sl: 10 Criptys – 02/07/2023 -17:57 hrs*
Ch - Mr: 1 Criptys – 02/07/2023 - 14:03 hrs*	Bi - Ls: 9 Criptys – 02/07/2023 - 20:49 hrs*
Ef - Op: 2 Criptys – 02/07/2023 - 14:25 hrs*	Dg - Nq: 5 Criptys – 02/07/2023 - 23:34 hrs*
079. ……..**b13***	
Recompensa: 9 Criptys*	
¿?*	
0710. ……..**b14**	

Figura N° 10.2. 13: Datos décimo bloque

Conclusiones

Este material didáctico, denominado *Actividades prácticas: Criptografía*, cumple con el objetivo de ser un complemento a la materia teórica, donde se muestran distintas herramientas y aplicaciones que la criptografía tiene en el ámbito profesional.

Las actividades buscan ser lo más detalladas posible, de manera que el estudiante no tenga inconvenientes en su realización. Es por eso que, para las actividades solicitadas, se incluye un ejemplo del resultado esperado.

El material puede ser utilizado en cualquier equipo que cuente con la máquina virtual y los archivos de práctica. Se solicita asegurarse de disponer de un equipo adecuado que soporte el manejo de máquinas virtuales, para garantizar un mejor desarrollo de las prácticas.

Se considera que las prácticas tendrán un resultado favorable entre el estudiantado, ya que no solo ponen en práctica los conocimientos adquiridos en la materia de Criptografía, sino que también introducen a los alumnos al uso del sistema operativo Linux y refuerzan sus conocimientos en programación.

Las prácticas fueron puestas a prueba en un taller denominado "Taller Virtual de Criptografía Práctica", al que asistieron aproximadamente 20 alumnos. En dicho taller, se presentaron las prácticas incluidas en el manual. En el Anexo B se incluye el cartel promocional, así como algunos comentarios de los participantes sobre las prácticas, las cuales fueron tomadas en cuenta para la mejora del material.

Por diversos motivos, incluidos los personales, no se pudo continuar con el trámite de titulación. Sin embargo, el material fue retomado y se actualizaron, especialmente, las herramientas utilizadas y el sistema operativo. La planificación de la actualización de las prácticas, que se presenta en este trabajo, está disponible en el Anexo C.

A nivel personal, la realización de este trabajo me permitió profundizar mis conocimientos en la materia y, como reto, adentrarme en el mundo de la programación. La creación de los scripts me ayudó a mejorar en ese aspecto y a enriquecer una de las habilidades más importantes para un profesional en computación. Del mismo modo, el haber actualizado el material me hizo darme cuenta de que la tecnología avanza muy rápido, aunque no siempre sea notorio.

Las prácticas no son exclusivas de los estudiantes de la materia de Criptografía; cualquier estudiante de la Facultad de Ingeniería que necesite el material podrá utilizarlo, ya que está disponible en el sitio del laboratorio.

El material queda disponible para que, cuando se considere oportuno, sea puesto a disposición del alumnado y profesorado de la facultad, y sirva para complementar la asignatura.

Referencias

[Acceder a un servidor mediante ssh y un certificado]. (2016, 20 de marzo). WildUnix. https://wildunix.es/posts/acceder-a-un-servidor-mediante-ssh-y-un-certificado/

Adobe. (2023, 27 de junio). Editar objetos o imágenes en un archivo PD https://helpx.adobe.com/mx/acrobat/using/edit-images-or-objects-pdf.html

Adobe. (2022, 21 de octubre). Ajuste de botones de acción en los formularios PDF. https://helpx.adobe.com/mx/acrobat/using/setting-action-buttons-pdf-forms.html

Adobe. (2022, 21 de octubre). Permitir o bloquear los vínculos del sitio web en los PDF. https://helpx.adobe.com/mx/acrobat/using/allow-or-block-links-internet.html

[Advanced Encryption Standard (AES) - Proceso de cifrado (encrypt)]. (s.f.). teoría.com. https://www.teoria.com/jra/aes/encrypt.html

[Agregar elemento a lista o arreglo en Python]. (2018, 17 de octubre). Parzibyte. https://parzibyte.me/blog/2018/10/17/agregar-elemento-lista-arreglo-python/

Alejandro, G. (2014, 24 de enero). Cómo ofuscar u ocultar código de nuestros scripts Bash. DesdeLinux. https://blog.desdelinux.net/como-ofuscar-u-ocultar-codigo-de-nuestros-scripts-bash/

Alex. (2017, 20 de noviembre). Cambiar el editor por defecto de la línea de comandos. https://cambiatealinux.com/cambiar-el-editor-por-defecto-de-la-linea-de-comandos

[Algoritmo de Euclides Extendido]. (S.f.). Planetcalc. https://es.planetcalc.com/3311/

[Algoritmos de Programación con Python]. (S.f.). Uniwebsidad. https://uniwebsidad.com/libros/algoritmos-python/capitulo-7/listas

Amazon Web Services. (s.f.). ¿Qué es la criptografía? Amazon Web Services. https://aws.amazon.com/es/what-is/cryptography/

Amos, D. (s.f.). Python GUI Programming With Tkinter. Real Python. https://realpython.com/python-gui-tkinter/

Andy. (2013, 11 de noviembre). Advanced Encryption Standard (AES) with Python. https://old.nixaid.com/advanced-encryption-standard-aes-with-python/

Antón, E. (2008, 27 de agosto). Listas en python. https://pythonr2.wordpress.com/2008/08/27/listas-en-python/

Aparicio, A. y Magnitopic, A. (2018, 4 de agosto). Generar números aleatorios sin repetición y ordenarlos en Python. Alto código. https://altocodigo.blogspot.com/2018/08/generar-numeros-aleatorios-sin.html

[Argon2]. (2023, 13 de julio). En Wikipedia. https://en.wikipedia.org/wiki/Argon2#Variable-length_hash_function

[Argon2 Hash Generator & Verifier]. (2022). ESSE.TOOLS. https://argon2.online/

[argon2 on Debian 11 (Bullseye)]. (2023, 17 de agosto). Linux Packages. https://linux-packages.com/debian/package/argon2

Banda, F. (2022, 21 de marzo). Comprobar si la variable está vacía en Bash. DelftStack. https://www.delftstack.com/es/howto/linux/check-if-variable-is-empty-in-bash/

Bernstein, C. y Cobb, M. (S.f.). Advanced Encryption Standard (AES). TechTarget. https://www.techtarget.com/searchsecurity/definition/Advanced-Encryption-Standard

Biryukov, A. Dinu, D. y Khovrayovich, D. (2017, 24 de marzo). Argon2: the memory-hard function for password hashing and other applications. University of Luxembourg. https://www.cryptolux.org/images/0/0d/Argon2.pdf

Bit2Me Academy. (s.f.). *¿Qué es Proof of Work (PoW)?* Bit2Me Academy. https://academy.bit2me.com/que-es-proof-of-work-pow/

[Bitwise Left Shift]. (S.f.). RIP Tutorial. https://riptutorial.com/python/example/2470/bitwise-left-shift

Brown, K. (2022, 4 de marzo). Bash Scripting: Conditionals. LINUXCONFIG. https://linuxconfig.org/bash-scripting-conditionals

[Bucles while y for en Python]. (S.f.). Tutorial Python. https://tutorialpython.com/bucles-while-y-for-en-python/

[can import msvcrt and click on windows but not on Linux]. (2018, 5 de agosto). Forum Python. https://python-forum.io/thread-12011.html

Campos, J. (2011, 22 de julio). El algoritmo de Diffie-Hellman. https://javiercampos.es/blog/2011/07/22/el-algoritmo-de-diffie-hellman/

[Capítulo 5. Problemas que debe tener en cuenta para bullseye]. (S.f.). debian. https://www.debian.org/releases/bullseye/amd64/release-notes/ch-information.es.html#pam-default-password

Carles, J. (2019, 24 de mayo). Reparar paquetes rotos y dependencias incumplidas en Debian y Ubuntu. Geekland. https://geekland.eu/reparar-paquetes-rotos-linux/

Carles, J. (2022, 3 de julio). Uso del comando grep en Linux y UNIX con ejemplos. keekland. https://geekland.eu/uso-del-comando-grep-en-linux-y-unix-con-ejemplos/

Carmona, M. (2023, 1 de enero). Ruta absoluta vs relativa en Linux: ¿Cuál es la diferencia? It´s Foss. https://itsfoss.com/es/ruta-absoluta-relativa-linux/

Caro, P. (2013, 4 de septiembre). pcaro90/Python-AES. GitHub. https://github.com/pcaro90/Python-AES/blob/master/AES.py

[58 - tkinter : controles Button y Label]. (s.f.). Tutorialesprogramación. https://www.tutorialesprogramacionya.com/pythonya/detalleconcepto.php?punto=58&codigo=58&inicio=45

[colorama - Texto y fondo coloreados en la consola]. (2016, 5 de abril). Recursos Python. https://recursospython.com/guias-y-manuales/colorama-texto-fondo-coloreados-la-consola/

[Como autenticar en ssh con certificado]. (2019, 16 de noviembre). sebelk. https://sergiobelkin.com/posts/como-usar-certificados-ssh-para-autenticar/

[Cómo conocer dirección IP de una página web]. (2019, 10 de septiembre). Solvetic Sistemas. https://www.solvetic.com/tutoriales/article/4397-como-conocer-direccion-ip-pagina-web/

[Como generar un hash MD5 en Linux (y tipo Unix)]. (2017, 2 de mayo). Archivo Geek. https://archivogeek.com/3245/como-generar-verificar-un-hash-md5-en-linux/

[Como instalar pip en Linux y Windows]. (2016, 31 de octubre). PythonDiario. https://pythondiario.com/2016/10/como-instalar-pip-en-linux-y-windows.html

[Cómo puedo conseguir Linux]. (s.f.). Estréllate y Arde.ORG. https://www.estrellateyarde.org/que-es-linux/como-puedo-conseguir-linux

[Cómo usar comando Apt en Linux. (2017, 3 de octubre)]. Solvetic. https://www.solvetic.com/tutoriales/article/4364-como-usar-comando-apt-linux/

ConceptoDefinición. (2021). *¿Qué es Linux?* ConceptoDefinición. https://conceptodefinicion.de/linux/

[Concurrencia de procesos]. (2008, 01 de enero). Web Programación. https://webprogramacion.com/concurrencia-de-procesos/

[Convert hex to binary]. (2021). Stackoverflow. https://stackoverflow.com/questions/1425493/convert-hex-to-binary

Cortesi, D. (2022, 17 de abril), PyInstaller Manual. https://pyinstaller.org/en/stable/

[Crear certificados digitales con OpenSSL]. (2011, 30 de noviembre). imaginanet. https://www.imaginanet.com/blog/crear-certificados-digitales-con-openssl.html

[Create Executable of Python Script using PyInstaller]. (2023, 29 de abril). Data to Fish. https://datatofish.com/executable-pyinstaller/

[Criptografia a fondo y herramientas gratis de cifrado]. (2016, 18 de diciembre). Solvetic Seguridad. https://www.solvetic.com/tutoriales/article/2638-criptografia-y-algunas-herramientas-de-cifrado-que-usar/

[Criptografía en Python con PyCrypto]. (2014, 19 de abril). Ellaberintodefalken. https://www.ellaberintodefalken.com/2014/04/criptografia-python-pycrypto.html

Cuervo, V. (2015, 24 de noviembre). Añadir un elemento a una lista en Python. Línea de Código. https://lineadecodigo.com/python/anadir-un-elemento-a-una-lista-en-python/

Darkcrizt. (2023, 24 de junio). BLAKE3 una función hash criptográfica segura, rápida y paralelizable. LinuxAdictos. https://www.linuxadictos.com/blake3-una-funcion-hash-criptografica-segura-rapida-y-paralelizable.html

De Anda, N. (2018, 8 de agosto). ENLACES/LIGAS. factor evolución. https://www.factor.mx/portal/base-de-conocimiento/enlaces-ligas/

Debian. (s.f.). *¿Por qué Debian? Debian*. Recuperado de https://www.debian.org/intro/why_debian.es.html

debian. (2022, 24 de julio). C.2. El Árbol de Directorios. Apéndice C. Particionado en Debian. https://www.debian.org/releases/stable/s390x/apcs02.es.html

debian. (S.f.). Package: python3-pycryptodome (3.18.0+dfsg1-1 and others). https://packages.debian.org/sid/python3-pycryptodome

De Luz, S. (2023, 28 de junio). Configuración para máxima seguridad. Redes zone. https://www.redeszone.net/tutoriales/servidores/servidor-openssh-linux-configuracion-maxima-seguridad/

Departamento de Ciencias e Ingeniería de la Computación, Universidad Nacional del Sur [SCIC-UNS]. (s.f.). RSA. Creación de claves en el sistema RSA [Diapositivas]. https://cs.uns.edu.ar/~ldm/mypage/data/ss/info/ejemplo-rsa.pdf

Donohue, B. (2014, 10 de abril). ¿Qué Es Un Hash Y Cómo Funciona? kaspersky daily. https://latam.kaspersky.com/blog/que-es-un-hash-y-como-funciona/2806/

Echarri, C. (2017, 18 de marzo). tkinter Grid. Tutor de Programación. Blogspot. http://acodigo.blogspot.com/2017/03/tkinter-grid.html

EITCA. (s.f.). ¿Cuáles son los pasos involucrados en el protocolo de intercambio de claves ECDH de Diffie-Hellman de curva elíptica? EITCA. https://es.eitca.org/la-seguridad-cibern%C3%A9tica/eitc-es-criptograf%C3%ADa-cl%C3%A1sica-avanzada-acc/criptograf%C3%ADa-de-curva-el%C3%ADptica/criptograf%C3%ADa-de-curva-el%C3%ADptica-ecc/examen-revisi%C3%B3n-curva-el%C3%ADptica-criptograf%C3%ADa-ecc/%C2%BFCu%C3%A1les-son-los-pasos-involucrados-en-el-protocolo-de-intercambio-de-claves-ecdh-de-Diffie-Hellman-de-curva-el%C3%ADptica%3F/

[Encrypt data with AES]. (2022, 27 de noviembre). PyCryptodome. https://pycryptodome.readthedocs.io/en/latest/src/examples.html#encrypt-data-with-aes

[Encrypt & Decrypt Files With Password Using OpenSSL]. (2016, 19 de diciembre). ShellHacks. https://www.shellhacks.com/encrypt-decrypt-file-password-openssl/

Esplanada, R. (2023, 30 de enero). Texto de color impreso en Python. Delft Stack. https://www.delftstack.com/es/howto/python/python-print-colored-text/

[ESTEGANOGRAFÍA CON STEGHIDE: GUÍA COMPLETA DE USO]. (s.f.). ESGEEKS. https://esgeeks.com/steghide-guia-de-uso/

facturadigital. (2009, 19 de octubre). Firma Digital [vídeo]. YouTube. https://youtu.be/ogch2r2l3JE?si=4_bLTuUQy11q3_OG

[File Encryption And Decryption Using Python]. (2018, 8 de noviembre). Eudonix Learning Solutions. https://blog.eduonix.com/software-development/file-encryption-decryption-using-python/

[Firma Digital: Conceptos]. (s.f.). Tecnología + Informática. https://www.tecnologia-informatica.com/firma-digital-conceptos/

[Firmar archivos en Linux Mint]. (s.f.). HAPPSEERVICIOS. https://happservicios.wordpress.com/2017/04/06/firmar-archivos-en-linux-mint/comment-page-1/

[Formato del certificado y de la clave para la importación]. (s.f.). aws. https://docs.aws.amazon.com/es_es/acm/latest/userguide/import-certificate-format.html

García, D. (2023, 1 de junio). ¿Qué es y para que sirve el código ASCII? https://es.godaddy.com/blog/que-es-y-para-que-sirve-el-codigo-ascii/

García, J. (2017, 23 de junio). How does RSA work? Hackernoon. https://hackernoon.com/how-does-rsa-work-f44918df914b

Gargallo, J. (s.f.). DES. http://spi1.nisu.org/recop/al02/jgargallo/index.html

GeeksforGeeks. (s.f.). getent command in Linux with examples. GeeksforGeeks. https://www.geeksforgeeks.org/getent-command-in-linux-with-examples/

GeeksforGeeks. (s.f.). Image Steganography using StegoSuite in Linux. GeeksforGeeks. https://www.geeksforgeeks.org/image-steganography-using-stegosuite-in-linux/

Gite, V. (2021, 7 de enero). Where is My Linux GNU C or GCC Compilers Are Installed. https://www.cyberciti.biz/faq/locate-linux-gnu-c-or-gcc-compiler-location/

Gite, V. (2023, 9 de mayo). Bash get exit code of command on a Linux / Unix. https://www.cyberciti.biz/faq/bash-get-exit-code-of-command/

Gite, V. (2023, 19 de agosto). Linux ip Command Examples. nixCraft. https://www.cyberciti.biz/faq/linux-ip-command-examples-usage-syntax/

Glosario Terminología Informática [GTI]. (2015, 9 de octubre). ElGamal digital signature. http://www.tugurium.com/gti/termino.php?Tr=ElGamal%20digital%20signature

González, S. (s.f.). 24/7 Expert Support. LINUXTOTAL. https://www.linuxtotal.com.mx/index.php?cont=info__tips_022

González, S. (s.f.). Encuentra cualquier cosa en Linux con find. LINUXTOTAL. https://www.linuxtotal.com.mx/index.php?cont=info_admon_022

González, S. (s.f.). Permisos de archivos y directorios. LINUXTOTAL. https://www.linuxtotal.com.mx/index.php?cont=info_admon_011

González Velarde, E. J. (2022). *Actividad 2. FORO Criptografía. La base fundamental del cifrado* [Documento de curso]. Course Hero. https://www.coursehero.com/es/file/201857992/KSG2-A2-EJGVdocx/

Gordon, S. (s.f.). Diffie Hellman Secret Key Exchange using OpenSSL. https://sandilands.info/sgordon/diffie-hellman-secret-key-exchange-with-openssl

Grabbe, J. O. (s.f.). The DES Algorithm Illustrated. https://page.math.tu-berlin.de/~kant/teaching/hess/krypto-ws2006/des.htm

[Guía de expresiones regulares en Python]. (2015). Soporte-Platzi. https://platzi.com/blog/expresiones-regulares-python/

Guix, E. (2007, enero). Cifrado de ficheros mediante DES-CBC. https://openaccess.uoc.edu/bitstream/10609/726/1/40048tfc.pdf

Gus. (2017, 25 de septiembre). ELEGIR LA VERSIÓN DE PYTHON POR DEFECTO. PARCHEINFORMATICO. https://parcheinformatico.wordpress.com/2017/09/25/elegir-la-version-de-python-por-defecto/

Gustavo, B. (2023, 11 de agosto). Cómo usar los comandos find y locate en Linux. Hostinger Tutoriales. https://www.hostinger.mx/tutoriales/como-usar-comando-find-locate-en-linux/

Gutiérrez, C. (2013, 18 de enero). Funcionamiento del algoritmo RSA. welivesecurity. https://www.welivesecurity.com/la-es/2013/01/18/funcionamiento-del-algoritmo-rsa/

Harán, J. M. (2021, 1 de diciembre). Las contraseñas más comunes del 2021 son también las más inseguras. https://www.welivesecurity.com/la-es/2021/12/01/contrasenas-mas-comunes-2021-son-mas-inseguras/

Hermoso, R. (2011, 11 de febrero). [Tip] Copiar ficheros usando nc (netcat). TechnoBot. https://robleshermoso.wordpress.com/2011/02/11/tip-copiar-ficheros-usando-nc-netcat/

[How do I implement a DES algorithm in Python?]. (s.f.). Quora. https://www.quora.com/How-do-I-implement-a-DES-algorithm-in-Python

[how to convert a string to hex]. (2017, 22 de enero). MyBB. https://python-forum.io/thread-1715.html

[How to Install PyInstaller]. (2022, 2 de diciembre). PyInstallerhttps://pyinstaller.org/en/stable/installation.html

[How To Install rhash on Debian 11]. (S.f.). installati.one. https://installati.one/install-rhash-debian-11/?expand_article=1

[How to Install VirtualBox Guest Additions on Ubuntu 18.04]. (2019, 20 de julio). Linuxize. https://linuxize.com/post/how-to-install-virtualbox-guest-additions-in-ubuntu/

[HTML - Impedir copiar texto]. (2020, 18 de agosto). Interactive Programmers Community. https://www.lawebdelprogramador.com/foros/HTML/1217390-impedir-copiar-texto.html

IBM. (s.f.). ¿Qué es la seguridad informática? IBM. https://www.ibm.com/mx-es/topics/it-security

Ingenio Linux. (2018, 21 de mayo). Servicio SSH - parte 3: Autenticación con certificado sin contraseña [vídeo]. YouTube. https://www.youtube.com/watch?v=tSyc1htBeMQ&t=357s

Interactive Programmers Community. (2013, 19 de noviembre). Python - Quitar salto de línea. Iwp. https://www.lawebdelprogramador.com/foros/Python/1406257-Quitar-salto-de-linea.html

[Instalación y utilización de pip en Windows, Linux y OS X]. (2014, 19 de abril). Recursos Python. https://recursospython.com/guias-y-manuales/instalacion-y-utilizacion-de-pip-en-windows-linux-y-os-x/

[Installation. On The Cargo Book]. (s.f.). GitHub. https://doc.rust-lang.org/cargo/getting-started/installation.html

[Install Rust]. (s.f.). Rust. https://www.rust-lang.org/tools/install

Iteramos. (s.f.). Función inversa multiplicativa modular en Python. https://www.iteramos.com/pregunta/82300/modular-inverso-multiplicativo-de-la-funcion-en-python

Kaspersky. (s.f.). 15 consejos y normas de seguridad para usar Internet. Kaspersky. https://latam.kaspersky.com/resource-center/preemptive-safety/top-10-preemptive-safety-rules-and-what-not-to-do-online

Kaushik, N. (2012, 9 de abril). Difference Between PGP and GPG. http://www.differencebetween.net/technology/software-technology/difference-between-pgp-and-gpg/

Kekayan. (2018, 7 de julio). Encrypt files using AES with OPENSSL. Medium. https://kekayan.medium.com/encrypt-files-using-aes-with-openssl-dabb86d5b748

Keopx. (2010, 12 de abril). Cambiar nombre de máquina en Debian/Ubuntu. https://www.keopx.net/blog/cambiar-nombre-de-maquina-en-debianubuntu

Kharechko, M. (2006, 18 de agosto). CONVERT STRING TO HEX (PYTHON RECIPE). ActivateState. https://code.activestate.com/recipes/496969-convert-string-to-hex/

Kili, A. (2023, 10 de agosto). How To Install PIP to Manage Python Packages in Linux. TecMint. https://www.tecmint.com/install-pip-in-linux/

Kirstein, E. (2018, 16 de mayo). Glank/Galois. GitHub. https://github.com/Glank/Galois

Kusanagi, M. (2012, 5 de diciembre). Algoritmo de Euclides extendido[Python]. http://gordosfrikis.blogspot.com/2012/12/algoritmo-de-euclides-extendidopython.html

Landman, N. Williams, C. Ross, E. y Khim, J. (s.f.). Secure Hash Algorithms. Brilliant. https://brilliant.org/wiki/secure-hashing-algorithms/

Laro, A. (2017, 6 enero). Cómo instalar diferentes versiones de GCC y alternar entre ellas. https://www.alvarolara.com/2017/01/06/como-instalar-diferentes-versiones-de-gcc-y-alternar-entre-ellas/

[La sentencia case-esac]. (s.f.). Programación para la bioinformática. https://blogs.upm.es/estudiaciencia/la-sentencia-case-esac/

Lastra, S. (2021, 22 de septiembre). Como insertar imágenes en un archivo PDF. LightPDF. https://lightpdf.com/es/insertar-imagenes-en-un-pdf.html

Lech-Hostalot, D. (2006, noviembre). Ataque de Factorización a RSA. https://www.researchgate.net/publication/259475258_Ataque_de_Factorizacion_a_RSA

[Linux passwd command]. (2021, 13 de marzo). Computer Hope. https://www.computerhope.com/unix/upasswor.htm

Lonston, B. (2023, 14 de julio). A Complete Guide to Usage of 'usermod' command – 15 Practical Examples with Screenshots. TecMint. https://www.tecmint.com/usermod-command-examples/

López, A. (2023, 12 de abril). Criptografía: Qué son los algoritmos hash y para qué se utilizan. Redes zone. https://www.redeszone.net/tutoriales/seguridad/criptografia-algoritmos-hash/

López Barrientos, M. J. (2016). Criptografía (2ª ed.). Editorial Universidad Nacional Autónoma de México.

[Los cuadros de diálogo (messagebox)]. (s.f.). Programación Fácil. https://programacionfacil.org/cursos/tkinter/capitulo_10_los_cuadros_de_dialogo_messagebox.html

[Los primeros cincuenta millones números primos]. (S.f.). NUM3ROS PR7MOS. https://numerosprimos.org/lista-cincuenta-millones-numeros-primos/

[man page (Español)]. (2023, 6 de mayo). archlinux. https://wiki.archlinux.org/title/Man_page_(Espa%C3%B1ol)

Math for All. (2016, 24 de septiembre). Números Primos de Fermat - ¿CUÁLES SÓN? [Vídeo]. YouTube. https://youtu.be/hHK7e7yDlmE?si=Ky-9v-dDWI0YjKgE

Mauro_88. (2013, 16 de mayo). Autenticación y firma digital - Apuntes - Seguridad Informática - Parte3, Apuntes de Ingeniería Infórmatica. docsity. https://www.docsity.com/es/autenticacion-y-firma-digital-apuntes-seguridad-informatica-parte3/332918/

Medina, L. A. (2013, 7 de marzo). Comandos Linux cd. https://www.comoinstalarlinux.com/comandos-linux-cd/

Menjívar, M. (2021, 28 de febrero). Explicación de los grupos de usuarios en Linux: cómo agregar un nuevo grupo, agregar un nuevo miembro, y cambiar de grupo. freeCodeCamp. https://www.freecodecamp.org/espanol/news/explicacion-de-los-grupos-de-usuarios-en-linux/#:~:text=El%20grupo%20primario%20de%20un,que%20un%20usuario%20sea%20parte

Microsoft. (2023, 9 de mayo). Lenguaje de expresiones regulares - Referencia rápida. https://learn.microsoft.com/es-es/dotnet/standard/base-types/regular-expression-language-quick-reference

Microsoft. (S.f.). Configurar encabezados y pies de página para diferentes secciones de un documento. Microsoft Support. https://support.microsoft.com/es-es/office/configurar-encabezados-y-pies-de-p%C3%A1gina-para-diferentes-secciones-de-un-documento-94332643-a6e9-46aa-ab29-064f1d356db6

Miller, S. (2020, 18 de julio). How to create a Linux user, IT SECURITY. Qualys Community. https://qualys.my.site.com/discussions/s/question/0D52L00004TntntSAB/how-to-create-a-linux-user

Mudumbai, V. (2018, 16 de diciembre). Using SSH certificates — (SSH Recipes in Go) — An interlude. Medium. https://blog.tarkalabs.com/ssh-recipes-in-go-an-interlude-6fa88a03d458

[Name]. (2017, 11 de octubre). debian. https://manpages.debian.org/unstable/libcrypt-dev/crypt.5.en.html

Navok. S. (2022). Argon2. En Practical Cryptography for Developers. https://cryptobook.nakov.com/mac-and-key-derivation/argon2

nc (netcat): Connection refused problem. (S.f.). Unix & Linux Forums Content. https://www.unix.com/ip-networking/178877-nc-netcat-connection-refused-problem.html

[Network configuration (Español)]. (2023, 27 de febrero). Archilinux. https://wiki.archlinux.org/title/Network_configuration_(Espa%C3%B1ol)

Nguyen, J. (2015, 9 de diciembre). Sign server and client certificates. OpenSSL Certificate Autority. https://jamielinux.com/docs/openssl-certificate-authority/sign-server-and-client-certificates.html

[9.2. Utilizando diccionarios en Python]. (S.f.). Uniwebsidad. https://uniwebsidad.com/libros/algoritmos-python/capitulo-9/utilizando-diccionarios-en-python

[Números primos del 1 al 1000000]. (S.f.). NUM3ROS PR7MOS. https://numerosprimos.org/numeros-primos-menores-de-1-a-1000000/

[Obtener información de la dirección IP con Python]. (2014, 27 de noviembre). Solvetic. https://www.solvetic.com/tutoriales/article/1341-obtener-informacion-de-la-direccion-ip-con-python/

[8 metodologías que todo profesor del siglo XXI debería conocer]. (S.f.). Realinfluencers. https://www.realinfluencers.es/2018/09/09/8-metodologias-profesor-siglo-xxi-deberia-conocer/

O'Connor, J. Neves, S. Philippe, J. y Zooko, W.. (2021, 2 de noviembre). BLAKE3. One function, fast everywhere . GitHub. https://github.com/BLAKE3-team/BLAKE3-specs/blob/master/blake3.pdf

O'Connor, J. Neves, S. Philippe, J. y Zooko, W. (2023, 6 de Julio). BLAKE3. GitHub. https://github.com/BLAKE3-team/BLAKE3

[openssl ciphers]. (S.f.). PTC MKS Toolkit. https://www.mkssoftware.com/docs/man1/openssl_ciphers.1.asp

OpenSSL Project. (n.d.). OpenSSL: The Open Source toolkit for SSL and TLS. OpenSSL Project. https://openssl-library.org/

Oracle. (s.f.). *¿Por qué usar VirtualBox?*. *VirtualBox*. Recuperado de https://www.virtualbox.org/manual/topics/Introduction.html#virt-why-useful

Oriol. (2015, 14 de junio). El Árbol de Directorios de Linux. Conoce las Principales Carpetas. ComputerNewAge. https://computernewage.com/2015/06/14/el-arbol-de-directorios-de-linux-al-detalle-que-contiene-cada-carpeta/

Orovengua, J. (2013, 23 de diciembre). Buscar ficheros grandes en Linux (mayores de un tamaño) con el comando find. LinuxParty. https://www.linuxparty.es/35-linux/9007-buscar-ficheros-grandes-en-linux-mayores-de-un-tamano-con-el-comando-find

Orovengua, J. (2017, 22 de junio). Encriptar y desencriptar ficheros en Linux, desde la consola. LinuxParty. https://www.linuxparty.es/18-encriptacion/9764-encriptar-y-desencriptar-ficheros-en-linux-desde-la-consola.html

Overview of nano's shorcuts. (s.f.) Nano-editor. https://www.nano-editor.org/dist/latest/cheatsheet.html

Pardo, C. J. (2017, 21 de octubre). Que paquetes tengo instalados en Python? VISIÓN POR COMPUTADOR. https://carlosjuliopardoblog.wordpress.com/2017/10/21/que-paquetes-tengo-instalados-en-python/

Paz, A. (2014. 13 de agosto). Herramienta para realizar técnicas de esteganografía y estegoanálisis. Gurú de la informática. https://gurudelainformatica.es/herramienta-para-realizar-tecnicas-de-esteganografia-y-estegoanalisis

Pedro I. Hernández G. (2017, 6 de abril). firmar documentos en linux mint [vídeo]. YouTube. https://www.youtube.com/watch?v=Hlz88i6yi8A

[PEM, DER, CRT y CER: codificaciones y conversiones X.509]. (2020, 7 de julio). SSL. https://www.ssl.com/es/gu%C3%ADa/codificaciones-y-conversiones-pem-der-crt-y-cer-x-509/

[Permission denied on authorized_key file]. (2019, 3 de octubre). superuser. https://superuser.com/questions/677996/permission-denied-on-authorized-key-file

Perseo. (2019, 28 de marzo). Permisos y derechos en Linux. DesdeLinux. https://blog.desdelinux.net/permisos-y-derechos-en-linux/

Pounder, L. (2022, 22 de mayo). How to Create Executable Applications in Python. tom'sHARDWARE. https://www.tomshardware.com/how-to/create-python-executable-applications

Programación Fácil. (2020, 29 de octubre). Los cuadros de diálogo (MESSAGEBOX) - Curso Tkinter de cero - Capítulo 10 [Vídeo]. YouTube. https://youtu.be/7YXcAFZjdHw

[Programando: Cifrado César en Python]. (2014, 5 de abril). Curioseando en Programación. https://tpec05.blogspot.com/2014/04/programando-cifrado-cesar-en-python.html

[Publicado Debian 12 bookworm]. (2023, 10 de junio). debian. https://www.debian.org/News/2023/20230610

Pykes, K. (2022). Two Simple Methods To Convert A Python File To An Exe File. datacamp. https://www.datacamp.com/tutorial/two-simple-methods-to-convert-a-python-file-to-an-exe-file

[Python – Capítulo 18: Creación de sublistas]. (s.f.). El Club del Autodidacta. http://elclubdelautodidacta.es/wp/2011/09/python-capitulo-18-creacion-de-sublistas/

[Python: Cómo copiar una lista]. (s.f.). El Club del Autodidacta. http://elclubdelautodidacta.es/wp/2012/09/python-como-copiar-una-lista/

[Python DES libraries [duplicate]]. (2013, 9 de diciembre). Stack overflow. https://stackoverflow.com/questions/20459456/python-des-libraries

[Python Encryption and Decryption with PyCryptodome]. (2019, 14 de mayo). Nitratine. https://nitratine.net/blog/post/python-encryption-and-decryption-with-pycryptodome/

[Python - listas (comparar entre dos listas)]. (S.f.). Interactive Programmers Community. https://www.lawebdelprogramador.com/foros/Python/1581596-listas-comparar-entre-dos-listas.html

[Python optparse.OptionError() Examples]. (S.f.). ProgramCreek. https://www.programcreek.com/python/example/55983/optparse.OptionError

Python Package Index [PyPI]. (2023, 19 de mayo). pycryptodome 3.18.0. https://pypi.org/project/pycryptodome/

Python Package Index [PyPI]. (2023, 19 de mayo). pyinstaller 5.13.1. https://pypi.org/project/pyinstaller/

Python Package Index [PyPI]. (2023, 15 de agosto). argon2-cffi 23.1.0. https://pypi.org/project/argon2-cffi/

Python Package Index [PyPI]. (2018, 23 de agosto). crc64iso 0.0.2. https://pypi.org/project/crc64iso/

Python Package Index [PyPI]. (2023, 24 de agosto). cx-Freeze 6.15.6. https://pypi.org/project/cx-Freeze/

Python Package Index [PyPI]. (2013, 17 de octubre). pycrypto 2.6.1. https://pypi.org/project/pycryptodome/

[Python - Se me cierra el termina una vez ejecutado un código]. (S.f.). Interactive Programmers Community. https://www.lawebdelprogramador.com/foros/Python/1362039-Se-me-cierra-el-termina-una-vez-ejecutado-un-codigo.html

Python Software Foundation. (2023, 16 de agosto). argparse — Analizador sintáctico (Parser) para las opciones, argumentos y sub-comandos de la línea de comandos. https://docs.python.org/es/3/library/argparse.html#module-argparse

Python Software Foundation. (2023, 16 de agosto). 8. Errores y excepciones. https://docs.python.org/es/3/tutorial/errors.html

Python Software Foundation. (2023, 16 de agosto). tkinter – Interface de Python para Tcl/Tk. tkinter — Interface de Python para Tcl/Tk

Python Software Foundation. (2023, 19 de agosto). hashlib — Secure hashes and message digests. https://docs.python.org/3/library/hashlib.html

Python Software Foundation. (2023, 25 de agosto). optparse — Parser for command line options. https://docs.python.org/3/library/optparse.html

PyTutorials. (2018, 14 de marzo). Convert PY to EXE Automatically [vídeo]. YouTube. https://youtu.be/OZSZHmWSOeM?si=Mg9d9G1c7A7ShKg7

Qing, S. (2019, 31 de marzo). SuQinghang / CoursesCode. GitHub. https://github.com/SuQinghang/CoursesCode/tree/master/Cryptology_Exp/code/DES

[¿Qué es una clave PGP?]. (s.f.). GNOME HELP. https://help.gnome.org/users/seahorse/stable/about-pgp.html.es#:~:text=Una%20clave%20PGP%20es%20una,p%C3%BAblica%20y%20la%20clave%20privada.

[¿Qué es una función de derivación de clave?]. (2022, 26 de julio). KeepCoding. https://keepcoding.io/blog/funcion-de-derivacion-de-clave/

Quin, S. (2019, 31 de mayo). CoursesCode/Cryptology_Exp/code/DES/DES_encrypt.py. GitHub. https://github.com/SuQinghang/CoursesCode/blob/master/Cryptology_Exp/code/DES/DES_encrypt.py

Robin, D. (2019, 20 de abril). Pydes. GitHub. https://github.com/RobinDavid/pydes/blob/master/pydes.py

Rocío, G. R. (2023, 13 de marzo). Cómo establecer una conexión SSH en Windows y Linux. adsl zone. https://www.adslzone.net/como-se-hace/internet/servidor-remoto-ssh/

Rock, D. (2014, 18 de julio). Esteganografía en Linux sin instalar software adicional. https://donnierock.com/2014/07/18/esteganografia-en-linux-sin-instalar-software-adicional/

Rouse, M. (2015, 22 de junio). What Does Secure Hash Algorithm Mean? https://www.techopedia.com/definition/10328/secure-hash-algorithm-sha

Ruíz, P. (2017, 13 de marzo). APRENDE A USAR EL COMANDO SHC PARA COMPILAR SCRIPTS. https://miotroblogsite.wordpress.com/2017/03/13/aprende-a-usar-el-comando-shc-para-compilar-scripts/

[rust-lang/cargo]. (2023, 17 de agosto). GitHub. https://github.com/rust-lang/cargo#compiling-from-source

[Rutas explicadas: Absoluta, relativa, UNC y URL. ArcMap]. (2016). Esri. https://desktop.arcgis.com/es/arcmap/10.3/tools/supplement/pathnames-explained-absolute-relative-unc-and-url.htm

[Ruta relativa y absoluta, diferencias y características]. (2018, 1 de julio). ZEOKAT. https://www.vozidea.com/diferencias-ruta-relativa-y-absoluta

[Saber si estamos conectados a internet [bash]]. (2010, 26 de mayo). Poesía Binaria. https://poesiabinaria.net/2010/05/saber-si-estamos-conectados-a-internet-bash/

Saive, R. (2023, 3 mayo). 7 Tools to Encrypt/Decrypt and Password Protect Files in Linux. TecMint. https://www.tecmint.com/linux-password-protect-files-with-encryption/

Sánchez, J. (2016, 11 de septiembre). Inverso de a módulo n: Python. https://justyusblog.wordpress.com/2016/09/11/inverso-de-a-modulo-n-python/

Samdare, B. (s.f.). groupmod command in Linux with examples. geeksforgeeks. https://www.geeksforgeeks.org/groupmod-command-in-linux-with-examples/

Sánchez, J. (1999, diciembre). Descripción del algoritmo DES. (Data Encription Standard). http://www.satorre.eu/descripcion_algoritmo_des.pdf

Sarya. (S.f.). How to fix "_tkinter.TclError: no display name and no $DISPLAY environment variable" error. Its Linux FOSS. https://itslinuxfoss.com/tkinter-no-display-name-no-display-environment-variable/#google_vignette

Selinger, P. (2011, 11 de octubre). MD5 Collision Demo. https://www.mathstat.dal.ca/~selinger/md5collision/

Simmons, G. (2023, 25 de julio). Data Encryption Standard, cryptology. Britannica. https://www.britannica.com/topic/Data-Encryption-Standard

SignPlus. (n.d.). Digital signature. https://www.sign.plus/es/electronic-signature/digital-signature

Sintes, B. (2019, 4 de abril). Valores aleatorios: la biblioteca random. https://www.mclibre.org/consultar/python/lecciones/python-biblioteca-random.html

Soto, D. (S.f.). Los bits que dejan huella: criptografía, cifrado de datos y huella digital. https://www.peritacionesinformaticas.es/miblog/49-huella-digital

[SSH certificate based authentication - a quick guide]. (2020, 5 de junio). All things cloud. https://allthingscloud.eu/2020/01/05/ssh-certificate-based-authentication-a-quick-guide/

Stack Exchange Network - Ask Ubuntu. (2019, 1 de mayo). How do I check the SHA1 hash of a file?. https://askubuntu.com/questions/61826/how-do-i-check-the-sha1-hash-of-a-file

Stack Exchange Network - Superuser. (2019). OpenSSL 3DES encrytion parameters. https://superuser.com/questions/769273/openssl-3des-encrytion-parameters

Stack Exchange Network - Unix & Linux. (2021, 2 de mayo). How to install g++ 4.9 on debian stretch. https://unix.stackexchange.com/questions/334888/how-to-install-g-4-9-on-debian-stretch

Stack overflow. (2017, 6 de abril). Crear objetos con nombres similares con un ciclo for en python. https://es.stackoverflow.com/questions/60867/crear-objetos-con-nombres-similares-con-un-ciclo-for-en-python

Stack overflow. (2017, 28 de julio). Eliminar espacio al leer un texto en python. https://es.stackoverflow.com/questions/91159/eliminar-espacio-al-leer-un-texto-en-python

Stack overflow. (2018). Posición de una dato en una lista (Python). https://es.stackoverflow.com/questions/63234/posici%C3%B3n-de-una-dato-en-una-lista-python

Stack overflow. (2021, 7 de noviembre). How can I convert a .py to .exe for Python? https://stackoverflow.com/questions/41570359/how-can-i-convert-a-py-to-exe-for-python

Stack overflow. (2023, 3 de abril). How to install Pycrypto for Python 3.7.2? https://stackoverflow.com/questions/54142430/how-to-install-pycrypto-for-python-3-7-2/75917200#75917200

Stack overflow. (2023, 6 de mayo). Encrypt and decrypt using PyCrypto AES-256. https://stackoverflow.com/questions/12524994/encrypt-and-decrypt-using-pycrypto-aes-256

Stack Overflow. (2023, 8 de mayo). ImportError: No module named 'Tkinter' [duplicate]. https://stackoverflow.com/questions/25905540/importerror-no-module-named-tkinter

Stack overflow. (2023, 18 de junio). Convert bytes to a string in Python 3. https://stackoverflow.com/questions/606191/convert-bytes-to-a-string-in-python-3

[StegoSuite | Como OCULTAR ARCHIVOS en Imagenes en Kali Linux]. (S.f.). Gabriel Coding. https://gabrielcoding.com/stegosuite-como-ocultar-archivos-en-imagenes-en-kali-linux/#:~:text=imagen%20usando%20Stegosuite-,%C2%BFQu%C3%A9%20es%20Stegosuite%3F,secretos%20en%20archivos%20de%20im%C3%A1genes.

[Steps in the AES Encryption Process]. (S.f.). eTutorials.org. http://etutorials.org/Networking/802.11+security.+wi-fi+protected+access+and+802.11i/Appendixes/Appendix+A.+Overview+of+the+AES+Block+Cipher/Steps+in+the+AES+Encryption+Process/

Sterne, B. (2007, 10 de junio). AES Tutorial / Python Implementation. https://brandon.sternefamily.net/2007/06/aes-tutorial-python-implementation/

Steven Gordon. (2012, 31 de enero). DES Encryption using OpenSSL [vídeo]. YouTube. https://youtu.be/VdE21ku7SMs

[Symmetric encryption using pycrypto]. (S.f.). RIP Tutorial. https://riptutorial.com/python/example/18926/symmetric-encryption-using-pycrypto

Szostak, M. (2022, 19 de abril). How to improve user password security with Argon2? Boldare https://www.boldare.com/blog/how-to-improve-user-password-security-with-argon2/

Tecnología Nolly. (s.f.). *Los 5 mejores sistemas operativos de código abierto*. Tecnología Nolly. https://tecnologianolly.com/los-5-mejores-sistemas-operativos-de-codigo-abierto/

[Tema 4: Firmas digitales]. (2004, mayo). Comunicación de datos. [Diapositivas]. http://personales.upv.es/~fjmartin/cdii_web/traspas/Firmas_sin_fondo_2x.pdf

[Tema 8. Linux. Sistema de Ficheros. 8.2.3. Directorios más importantes en Linux]. (S.f.). Muraluv. http://mural.uv.es/oshuso/823_directorios_ms_importantes_en_linux.html

[TERMINOLOGIA DE LA FIRMA DIGITAL]. (2014, 26 de septiembre). Program Creek. https://criptografiafirmasdigitales.wordpress.com/

Thalskarth, M. (2014, 15 de marzo). Contraseñas con Hash+Salt y Hashes lentos. https://thalskarthmaelstrom.wordpress.com/2014/03/15/hash-salt-y-hashes-lentos/

[Tkinter askyesno]. (s.f.). Python Tutorial. https://www.pythontutorial.net/tkinter/tkinter-askyesno/

[Tkinter Entry]. (s.f.). Python Tutorial. https://www.pythontutorial.net/tkinter/tkinter-entry/

[Tkinter Label]. (S.f.). Python Tutorial. https://www.pythontutorial.net/tkinter/tkinter-label/

[13 - Curso de Python - Leer y escribir ficheros en python]. (S.f.). ChuWiki. https://chuidiang.org/index.php?title=13_-_Curso_de_Python_-_Leer_y_escribir_ficheros_en_python

[3 Ways to Set Options for a Tk Themed Widget]. (S.f.). Python Tutorial. https://www.pythontutorial.net/tkinter/tkinter-options/

[Tutorial: Aprende a eliminar caracteres en una cadena de texto en Python]. (2021, 23 de noviembre). Facialix. https://blog.facialix.com/eliminar-caracteres-en-una-cadena-de-texto-en-python/

Universidad Autónoma de Ciudad Juárez. (s.f.). Unidad 6: Seguridad en las redes de computadoras. https://www.uacj.mx/CGTI/CDTE/JPM/Documents/IIT/infseguridad/U6-5.html

Universidad Complutense de Madrid [UCM]. (2016, 10 de noviembre). 2.1 Introducción a la criptografía [Diapositivas]. https://www.ucm.es/data/cont/docs/72-2016-11-10-2%20Comunicaciones%20seguras.pdf

Universidad de la República de Uruguay - Facultad de Ingeniería [FING-EDU]. (2013). Breve manual de programación en python. https://www.fing.edu.uy/~darosa/manualFinal.pdf+

UPM. (2015, 1 de septiembre). Píldora formativa 28: ¿Cómo funcionan los algoritmos DES y 3DES? [vídeo]. YouTube. https://youtu.be/XwUOwqSHzyo

UPM. (2015, 2 de noviembre). Píldora formativa 30: ¿Cómo se cifra con el algoritmo AES? [vídeo]. YouTube. https://youtu.be/tzj1RoqRnvo?si=ka4or8_HSIW94bLH

UPM. (2016, 1 de septiembre). Píldora formativa 38: ¿Qué es el intercambio de clave de Diffie y Hellman? [vídeo]. YouTube. https://youtu.be/TWhax2wQOrU?si=QyeIZZcIBT031J6u

UPM. (2017,13 de marzo). Píldora formativa 44: ¿Cómo funciona el hash MD5? [vídeo]. YouTube. https://www.youtube.com/watch?v=dxqxxwi9pwk

UPM. (2017, 30 de mayo). Píldora formativa 46: ¿Qué son SHA-2 y SHA-3? [vídeo]. YouTube. https://youtu.be/hEa_IC1-JQI

[Using AES for Encryption and Decryption in Python Pycrypto]. (S.f.). Novixys Software Dev Blog. https://www.novixys.com/blog/using-aes-encryption-decryption-python-pycrypto/

[Usuarios y Grupos en Linux]. (s.f.). Sempiterna Serendipia. https://santi-gf.github.io/usuarios-grupos/

Vargas, S. (2019, 18 de octubre). Bash Script con Salida en Colores. https://medium.com/linux-tips-101/bash-script-con-salida-en-colores-82bab9263998

Vazhayil, A. (2015, 25 de abril). AES – Advanced Encryption Standard. https://captanu.wordpress.com/2015/04/25/aes/

[22 - Listas: eliminación de elementos]. (s.f.). Tutoriales de programación. https://www.tutorialesprogramacionya.com/pythonya/detalleconcepto.php?punto=22&codigo=22&inicio=15

Verjel, F. (2020, 11 de octubre). ¿Cómo deshabilitar el click derecho de una web?. https://franyerverjel.com/blog/como-deshabilitar-el-click-derecho-de-una-web

Vichino. (2017, 23 de octubre). Scripting Linux – Sustituir salto de línea. https://vichino.wordpress.com/tag/quitar-saltos-de-linea-con-sed/

[Vim Cheat Sheet]. (s.f.) Sponsor warp. https://vim.rtorr.com/

Vinda, E. (2013, 12 de octubre). Sencilla explicación sobre AES. Slideshare. https://es.slideshare.net/elvisvinda/sencilla-explicacin-sobre-aes

Vollebregt. B. (2023). brentvollebregt / auto-py-to-exe. GitHub. https://github.com/brentvollebregt/auto-py-to-exe

VPN Unlimited. (s.f.). Elliptic Curve Digital Signature Algorithm (ECDSA). VPN Unlimited. https://www.vpnunlimited.com/es/help/cybersecurity/elliptic-curve-digital-signature-algorithm

Wikipedia. (s.f.). *Criptografía de curva elíptica*. Wikipedia. https://es.wikipedia.org/wiki/Criptograf%C3%ADa_de_curva_el%C3%ADptica

Wikipedia. (2024). RIPEMD-160. Wikipedia. https://es.wikipedia.org/wiki/RIPEMD-160

Yago, J. (2013, 25 de septiembre). CONSEJOS PRÁCTICOS A LA HORA DE INTEGRAR CRIPTOGRAFÍA. SBD. http://www.securitybydefault.com/2013/09/consejos-practicos-la-hora-de-integrar.html

Yesmin, F. (2019). Bash conditional statement. linuxhint. https://linuxhint.com/bash_conditional_statement/

Yesmin, F. (2023). Bash Error Handling. linuxhint. https://linuxhint.com/bash_error_handling/

zeroFruit. (2019, 12 de febrero). What is AES? — Step by Step. Medium. https://zerofruit-web3.medium.com/what-is-aes-step-by-step-fcb2ba41bb20