MATERIALES DOCENTES DE LA ASIGNATURA INTRODUCCIÓN A LA CIBERSEGURIDAD

BLOQUE 4: PRÁCTICAS Y CASOS

CURSO ACADÉMICO 2022-2023

GRADO EN INGENIERÍA DE LA CIBERSEGURIDAD

Universidad Rey Juan Carlos

Marta Beltrán Pardo
Miguel Calvo Matalobos

4. PRÁCTICAS Y CASOS

Introducción a la Ciberseguridad

Práctica 1: Ensamblador

Marta Beltrán Pardo

Miguel Calvo Matalobos

Agradecimientos (versiones anteriores): Isaac Martín de Diego y Alberto Fernández Isabel

Contenidos

1. Introducción

Ya hemos estudiado en las primeras unidades de la asignatura que los computadores ejecutan programas (cuyo objetivo es resolver un problema determinado). Estos programas, están compuestos por secuencias de instrucciones y se escriben utilizando lenguajes de programación. Las secuencias de instrucciones pueden ser escritas o bien por personas (programadores), o generados de forma automática mediante las herramientas adecuadas.

Los programas se encuentran cargados en memoria principal. En la Figura 1 puede observarse el proceso de traducción desde un programa en un lenguaje de alto nivel (por ejemplo, C), hasta que es cargado en la memoria para su ejecución. A modo resumen, podemos decir que **un lenguaje de programación es una notación formal para describir algoritmos o funciones que serán ejecutados por un computador**.

A su vez, **los lenguajes de programación pueden dividirse en lenguajes de alto nivel** (los cuales facilitan la tarea de los programadores, ya que se encuentran más próximos a la forma de pensar y de expresarse de los humanos) y en **lenguajes de bajo nivel** (cercanos a la arquitectura de la máquina y, normalmente, más difíciles de comprender para los programadores). Dentro de los lenguajes de bajo nivel encontramos el lenguaje máquina, el cual es el único que la circuitería de la máquina puede interpretar; sus instrucciones se encuentran codificadas en binario (0s y 1s).

El **lenguaje ensamblador** es un lenguaje de bajo nivel diseñado para sacar el máximo partido a las características físicas de un computador. Sus principales peculiaridades son:

- Dependencia absoluta de la arquitectura del computador.
- Imposibilidad de transportar programas entre distintas máquinas, salvo que sean de la misma familia o compatibles entre sí.
- Programas muy largos (con gran número de instrucciones en comparación con los programas escritos en lenguajes de alto nivel).
- Códigos de operación, datos y referencias expresados en binario.

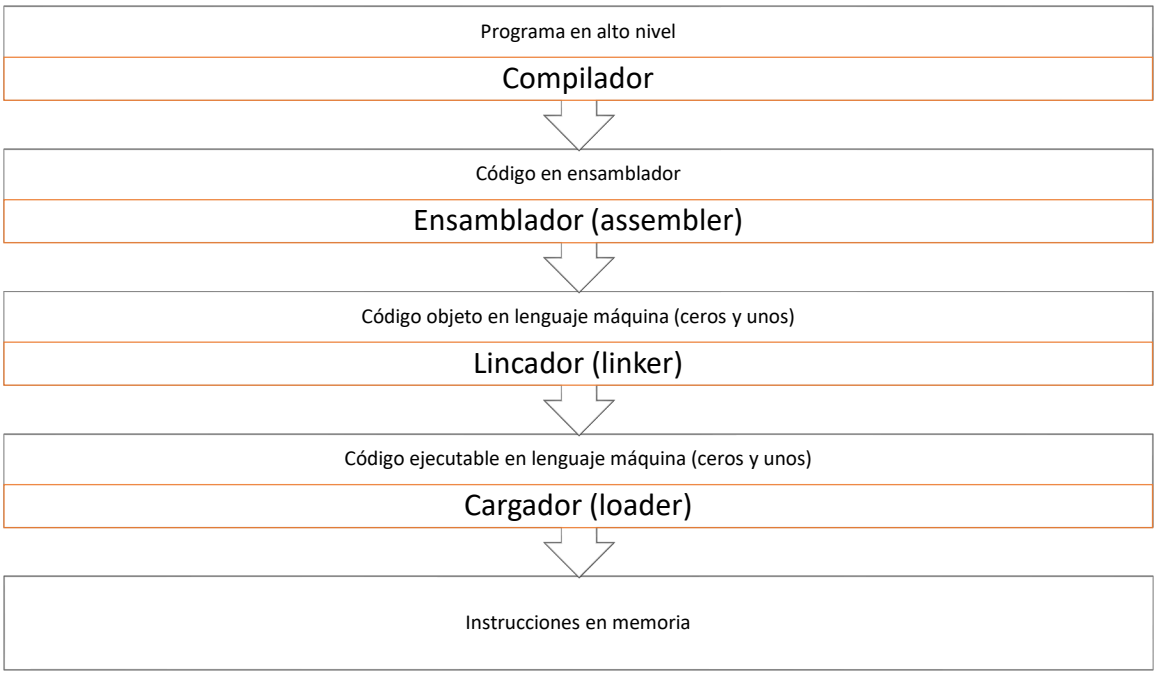


Figura 1. Proceso de traducción (programa en lenguaje alto nivel a memoria).

Por otro lado, el MIPS (Microprocessor without Interlocked Piepelines Stages) es un tipo específico de microprocesador desarrollado por MIPS Technologies. La arquitectura del MIPS es RISC (Reduced Instruction Set Computer), cuyas características principales son:

- Sus instrucciones son de tamaño fijo y se presentan en un reducido número de formatos.
- Sólo las instrucciones de carga y almacenamiento acceden a la memoria.
- Suelen disponer de muchos registros de propósito general.

Además, el MIPS puede encontrarse con un tamaño de registro de 32 bits (MIPS32) o con un tamaño de registro de 64 bits (MIPS64). Los procesadores MIPS se han utilizado en gran cantidad de productos como sistemas empotrados, dispositivos para Windows CE, routers CISCO, videoconsolas, etc., y, además, suelen utilizarse en el ámbito universitario para el aprendizaje de los

lenguajes ensamblador. El nanoMIPS, que hemos estudiado en clase, es sólo una versión pedagógica de estos procesadores con un repertorio de instrucciones muy reducido.

Para facilitar el desarrollo y el aprendizaje sobre MIPS32, la Universidad de Missouri State ha desarrollado el simulador MARS.

2. Material de la práctica

A continuación, se exponen los archivos necesarios para la realización de esta práctica, que pueden descargarse desde el Aula Virtual:

- **Practica1_Guion.pdf**: este documento. Se corresponde con el guion de la práctica y en él están contenidas todas las explicaciones necesarias para su realización. Además, puede consultarse el apartado "Consejos para hacer una memoria" para

- **Practica1_CodigoEjemplos.zip**: programas sencillos (con extensión .asm y .c) sobre los cuáles se deberá trabajar en esta práctica. En este mismo guion, en cada uno de los ejercicios propuestos del apartado "Enunciado de la práctica", se indica el archivo o archivos necesarios para su realización.

    - Practica1_example1.asm, Practica1_example1.c, Practica1_example1.7_resuelto.c. Estos archivos deben utilizarse en la realización del ejercicio 1 de esta práctica.

    - Practica2_example2.asm, Practica2_example2.c y Practica1_example2.4_resuelto.c. Estos archivos deben utilizarse en la realización del ejercicio 2 de esta práctica.

    - Practica3_example3.1.asm y Practica3_example3.2.asm. Estos archivos deben utilizarse en la realización del ejercicio 2 de esta práctica.

- **Practica1_MIPS32.pdf**: hoja de instrucciones del MIPS32 (con su repertorio completo).

Además, será necesario descargar el programa MARS en su última versión. Este programa, servirá para ejecutar los códigos de ejemplo proporcionados junto al enunciado de esta práctica (los que tienen extensión ".asm") y para realizar las diferentes tareas solicitadas. Puede descargarse en su última versión desde el siguiente enlace. Para ejecutarlo, será necesario tener instalado el entorno de ejecución de Java en el ordenador (puedes descargarse aquí).

3. Normativa y evaluación

En este apartado se detalla el formato de entrega de la práctica y la forma en la que se evaluará la misma:

- El porcentaje de la nota final de la asignatura al que corresponde esta práctica puede consultarse en la Guía docente de la propia asignatura.
- La práctica deberá realizarse, de forma obligatoria, en grupos de dos personas. Para la asignación de los grupos se deberán seguir las indicaciones del profesor.
- Cada grupo deberá:
    - Realizar una única memoria (puede utilizarse esta misma guía como plantilla) en la que responda las preguntas planteadas y/o en la que se exponga, de forma argumentada, las decisiones tomadas para la realización de la práctica.
    - Desarrollar y/o modificar tantos archivos de código como se soliciten en los diferentes ejercicios que forman la práctica.
- El resultado de la realización de la práctica consistirá en un fichero .zip llamado **Practica1.zip** que deberá entregarse a través del Aula Virtual en el espacio habilitado para ello y en la fecha límite allí indicada. Este fichero debe contener:
    - La memoria en formato PDF. En ella debe indicarse, en la primera página y de forma clara, el nombre y apellidos de los integrantes del grupo.
    - Los archivos de código resultantes de la realización de los ejercicios solicitados:
        - Practica1_example1.7_resuelto.asm
        - Practica1_example2.3_resuelto.asm
        - Practica1_example2.4_resuelto.asm
        - Practica1_example3.1_resuelto.asm
        - Practica1_example3.2_resuelto.asm
        - Practica1_example4.2_resuelto.asm

4. Consejos para hacer una memoria

A continuación, se exponen algunos consejos que ayudarán a mejorar el resultado final de un documento de memoria de un trabajo y/o práctica:

- **PORTADA**. En ella debe detallarse, de forma clara, el título de la práctica o el trabajo, el nombre y apellidos de la persona o personas que realizan el trabajo/práctica, los estudios que cursa y la asignatura, el curso académico, etc.

- **TÍTULOS**. Los títulos y subtítulos de las diferentes secciones o capítulos deben ser claros, concisos y descriptivos. Idealmente, los títulos comenzarán en nueva página, no siendo esto necesario para los subtítulos. Microsoft Word ofrece diferentes estilos de título ("Inicio" → "Estilos").

- **ÍNDICE**. Un documento de memoria debe llevar un índice o tabla de contenido. Esta, suele situarse justo después de la portada y en ella se indican las diferentes secciones o capítulos (títulos y subtítulos) y las páginas en las que se encuentran. Microsoft Word ofrece la generación automática de índices ("Referencias" → "Tabla de contenido").

- **PAGINACIÓN**. El documento debe tener sus páginas numeradas (salvo la portada). Para ello, puede utilizarse, en Microsoft Word, la opción "Número de página" ("Insertar" → "Número de página").

- **TEXTO**. El texto de la memoria debe estar justificado, con un buen espaciado entre párrafos y, para mejorar la lectura, una tipografía, un tamaño de letra y un interlineado adecuados (por ejemplo, Arial 11 e interlineado a 1,2). También debe tenerse en cuenta que, si el texto está escrito en español, cualquier palabra en otro idioma diferente deberá escribirse en cursiva. Además, no deberían existir errores ortográficos y/o gramaticales a lo largo del documento.

- **FIGURAS Y TABLAS**. Las figuras/imágenes y las tablas deben numerarse y rotularse (para ello, Microsoft Word ofrece la opción "Referencias" → "Insertar título"). Además, idealmente, tanto las figuras/imágenes como las tablas se referenciarán a lo largo del texto de la memoria (por

ejemplo: “En la figura X puede observarse el resultado de este experimento”), dándole de esta forma un sentido y una explicación textual a la propia figura/imagen/tabla.

- **CITAS y REFERENCIAS**. Si se consultan recursos (artículos, revistas, libros, páginas web, apuntes, etc.) y/o se debe indicar en el texto del documento de dónde se ha extraído cierta información (por ejemplo, una figura, un párrafo, etc.), estos deben citarse utilizando un estilo de cita normalizado (por ejemplo, mediante la herramienta “Insertar Cita” de Microsoft Word con estilo IEEE o APA). En caso de que en el documento se incluyan párrafos literales extraídos de alguna de las referencias, estos deberán entrecomillarse (por ejemplo: “Todo debería ser hecho lo más simple posible, pero no más simple.”).

- **BIBLIOGRAFÍA**. En relación con el punto anterior, todas las citas y referencias deben aparecer en un apartado de bibliografía (que suele situarse al final del documento). En él se mostrarán los detalles de cada una de esas citas/referencias que han ido insertándose a lo largo de la memoria. Microsoft Word permite insertar una bibliografía automáticamente (siempre y cuando las citas y referencias se hayan insertado tal y como se ha indicado en el punto anterior); para ello: “Referencias” → “Citas y bibliografía” → “Bibliografía”.

- **DOCUMENTO FINAL**. Para facilitar la lectura en cualquier sistema operativo y/o software sin pérdida de elementos de formato y/o desconfiguraciones/modificaciones inesperadas, la memoria debe presentarse en PDF (salvo que se indique lo contrario).

5. Enunciado de la práctica

En este apartado se describirán las distintas actividades, programas y ejercicios que deberá realizar cada grupo de prácticas, así como la información a completar y/o rellenar en la memoria.

Instrucciones

Antes de comenzar a realizar la práctica, es necesario ejecutar los códigos de ejemplo proporcionados junto al enunciado desde el simulador MARS. Se trata de que tes familiarices con el simulador y de que comprendas cómo se ejecutan las diferentes instrucciones, que etapas atraviesan, qué resultados producen, etc.

Para ello, una vez abierto el simulador, en el menú “File”, con la opción “Open”, debe seleccionarse el archivo .asm que se quiere abrir. El código en ensamblador se mostrará en la pestaña “Edit” y los registros del procesador pueden encontrarse a la derecha de la pantalla. Véase la Figura 2.

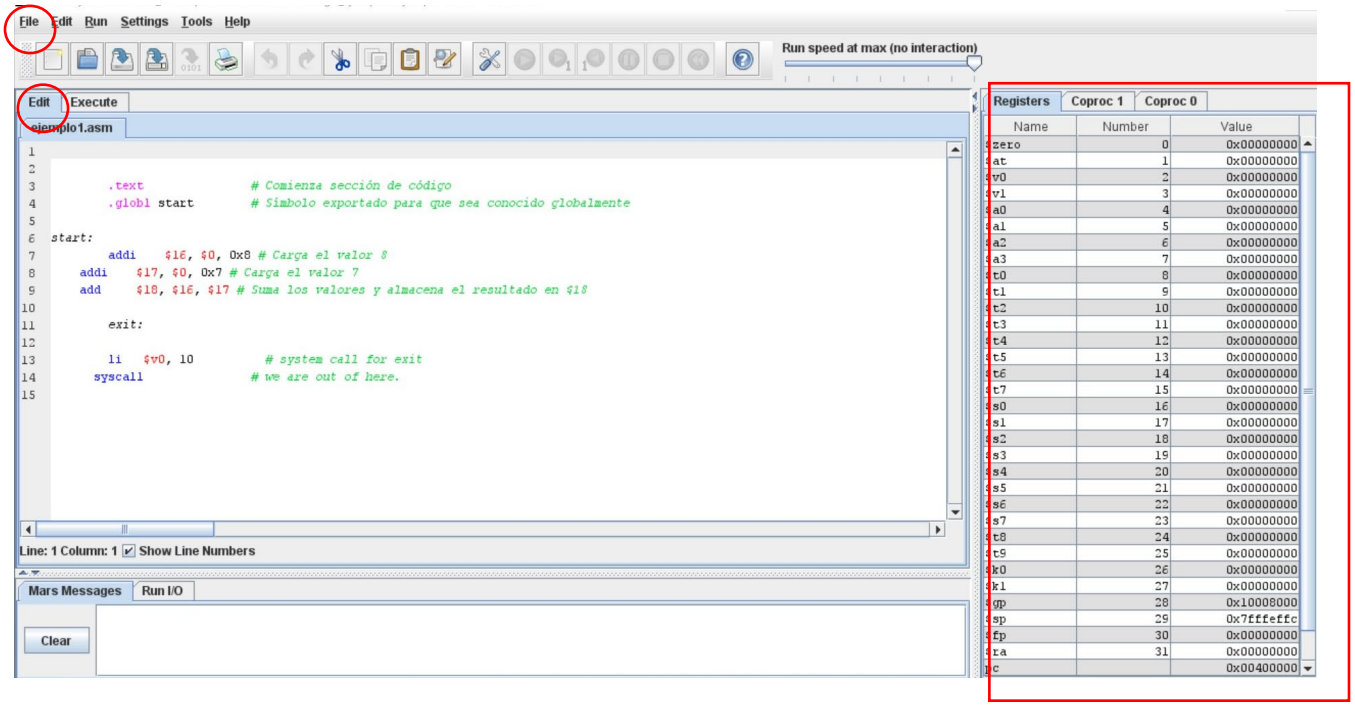


Figura 2. Abrir un archivo .asm desde el simulador MARS.

En el menú "Run", mediante la opción "Assemble", puede ensamblarse el código. Si existe algún error, este aparecerá detallado en la ventana de mensajes "Mars Messages", en la parte inferior del simulador.

Para ejecutar el código, una vez ensamblado, debe elegirse la opción "Go", también dentro del menú "Run" (si quieren ejecutarse todas las instrucciones) o "Step" (si quiere ejecutarse instrucción por instrucción para poder observar qué sucede con la ejecución de cada una de ellas). Véase la Figura 3.

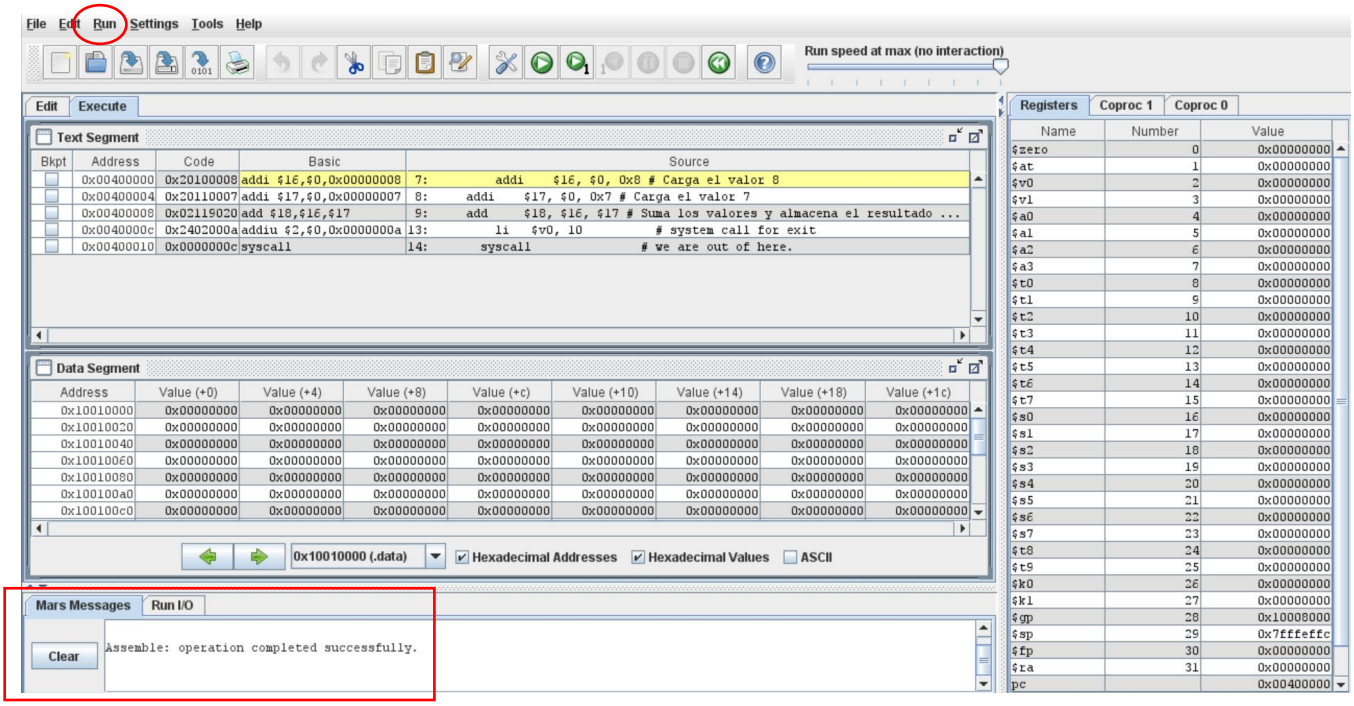


Figura 3. Ensamblado y ejecución en el simulador MARS.

Al ensamblar el código, se mostrará una nueva pestaña (en lugar de la pestaña "Edit"), "Execute", en la que puede verse el contenido de la memoria de instrucciones y la memoria de datos (recuerda que en esta arquitectura están separadas). Pueden verse las direcciones de memoria y sus contenidos en diferentes formatos estándar, según sea más cómodo.

Además, si el código ejecutado tiene algo de entrada/salida (E/S en español, I/O en inglés) como uso de teclado o de ratón, por ejemplo, esta información se mostrará en la pestaña "Run I/O", en la parte inferior del simulador (al lado de "Mars Messages").

Ejercicio 1

Para la realización de este ejercicio será necesario el código de ejemplo número 1 (Practica1_example1.asm y Practica1_example1.7_resuelto.c), proporcionado junto al enunciado de esta práctica. Este código calcula el área de un rectángulo y muestra el resultado.

Para ayudar a la comprensión de este código, además, puede observarse el fichero Practica1_example1.c, que se corresponde con el código proporcionado pero escrito en lenguaje C.

1. **¿Qué diferencia hay entre las instrucciones "add", "addi" y "li"?**

2. **¿Qué significan en el simulador las notaciones "$" y "0x"?**

3. **¿Se guarda alguna información en memoria? Justifica tu respuesta tanto si crees que no se guarda información en memoria como si consideras que sí lo hace. Además, si la respuesta es sí, ¿qué información se almacena en memoria y en qué zona y dirección? ¿por qué?**

4. **Ejecuta el código instrucción por instrucción e intenta comprender cómo van cambiando los valores de los diferentes registros en cada paso. Si se sustituyen en el código "$s0", "$s1", "$s2" por "$16", "$17" y "$18" respectivamente, ¿Existe algún cambio en los valores que se almacenan en los registros y/o en el resultado final? ¿Por qué ocurre esto?**

5. **¿Qué es una llamada al sistema o *syscall*? ¿Cómo se ejecuta en ensamblador? ¿Cuál es la llamada al sistema con código 1 en el MIPS32? ¿Qué código de llamada al sistema debería utilizarse para leer un entero?**

6. **¿Qué es el registro pc y qué almacena? ¿Puedes modificarlo? ¿Qué es el registro zero y qué almacena? ¿Puedes modificarlo?**

7. **Modifica el código proporcionado para que, en lugar de calcular el área de un rectángulo, calcule el área de un triángulo: (base x altura) / 2, tal y como lo hace el ejemplo proporcionado en C (Practica1_example1.7_resuelto.c). El fichero que debes entregar se llamará Practica1_example1.7_resuelto.asm.**

Ejercicio 2

Para la realización de este ejercicio será necesario el código de ejemplo número 2 (Practica1_example2.asm), proporcionado junto al enunciado de esta práctica.

Para ayudar a la comprensión de este código, además, puede observarse el fichero Practica1_example2.c, que se corresponde con el código proporcionado pero escrito en lenguaje C.

1. **¿Qué secuencia de control implementa el programa Practica1_example2.asm?, ¿Qué hace el código proporcionado (Practica1_example2.asm)? ¿Cuál es el resultado devuelto por pantalla? ¿Y si se modifica el contenido de la variable "max" ($s1) con el valor 101?**

2. **¿Se guarda alguna información en memoria? Justifica tu respuesta tanto si crees que no se guarda información en memoria como si consideras que sí lo hace. Además, si la respuesta es sí, ¿qué información se almacena en memoria y en qué zona y dirección? ¿por qué?**

3. **Intenta entender cómo van cambiando los valores de los diferentes registros al ir ejecutando el código instrucción por instrucción. Una vez comprendido el funcionamiento del código, modifícalo para que, en lugar de sumar los números pares, sume los números impares. El fichero que debes entregar con los resultados de este ejercicio se llamará Practica1_example2.3_resuelto.asm.**

4. **Trata de modificar el código para que, en lugar de sumar los valores de los números pares, el código devuelva la cantidad de números impares y de números pares que hay (en dos sumas diferentes), tal y como lo hace el ejemplo proporcionado en C (Practica1_example2.4_resuelto.c). El fichero que debes entregar con los resultados de este ejercicio se llamará Practica1_example2.4_resuelto.asm.**

Ejercicio 3

Para la realización de este ejercicio será necesario el código de ejemplo número 3 (Practica1_example3.1.asm y Practica1_example3.2.asm), proporcionados junto al enunciado de esta práctica. Con este programa, aprenderás el manejo de arrays en lenguaje ensamblador.

1. **En primer lugar, y tras analizar el código del programa facilitado para este primer apartado del ejercicio (Practica1_example3.1.asm), trata de explicar, de forma detallada y apoyándote en el pseudocódigo, lo que hace dicho programa. Añade comentarios también al código original (el fichero que debes entregar se llamará Practica1_example3.1_resuelto.asm).**

   Nota: Un seudocódigo expresa los pasos que se siguen para resolver un problema sin hacerlo en un lenguaje de programación concreto. Un ejemplo de pseudocódigo puede verse en la Figura 4.

```
inicio
    leer(n)
    pares = 0
    impares = 0
    i = 1
    hacer
        inicio
            si i%2 == 0
                pares = pares + 1
            sino
                impares = impares + 1
            i = i + 1
        fin
    mientras i <= n
    imprimir(pares, impares)
fin
```

Figura 4. Ejemplo de pseudocódigo.

2. **El siguiente apartado consiste en analizar y comprender el segundo programa facilitado para la realización de este ejercicio (Practica1_example3.2.asm), que cuenta el número de caracteres de una frase. A este programa, para poder funcionar, le faltan únicamente dos líneas de código (indicadas mediante un comentario en el fichero .asm facilitado).**

Debes completarlas y explicar el motivo de tu elección. El fichero que debes entregar, correspondiente al programa que cuenta el número de caracteres de una frase totalmente funcional, se llamará Practica1_example3.2_resuelto.asm.

Ejercicio 4

Para la realización de este ejercicio no se proporciona ningún código de ejemplo. Este ejercicio consiste en realizar, desde cero, un programa que:

- Pida, una a una, tres notas numéricas de un alumno (tres números enteros: n1, n2 y n3). Estos números serán solicitados por el programa al usuario, que los introducirá por teclado en cada ejecución.
- Calcule la media de esas tres notas: ((n1 + n2 + n3) / 3).
- Muestre por pantalla, mediante un texto, si el alumno está aprobado (cuando la media sea mayor o igual a 5) o suspenso (cuando la media sea menor que 5).

1. **Escribe el pseudocódigo del programa propuesto.**
2. **Traduce el pseudocódigo creado para el punto anterior a código ensamblador. No olvides comentar el código. El fichero que debes entregar se llamará Practica1_example4.2_resuelto.asm.**
3. **Prueba el funcionamiento del programa (introduciendo diferentes notas, notas no numéricas, decimales, etc.). No olvides adjuntar pantallazos de la demostración.**

5. Anexo I

En este apartado se incluyen los códigos, scripts y útiles necesarios para la realización de los ejercicios de esta práctica.

Practica1_example1.c

```
#include <stdio.h>

int main()
{
    int base=8;                     // Carga el valor 8 - base
    int altura=5;                   // Carga el valor 5 - altura
    int areaRect=base*altura;       // Multiplica los valores y almacena el resultado en $s2 - área rectángulo

    printf("%d",areaRect);
}
```

Practica1_example1.asm

```
    .text                       # Comienza sección de código
    .globl start                # Símbolo exportado para que sea conocido globalmente

start:
        li $s0, 0x8             # Carga el valor 8 - base
        li $s1, 0x5             # Carga el valor 5 - altura
        mul $s2, $s0, $s1       # Multiplica y almacena el resultado en $s2 - área rectángulo

        move $a0, $s2           # Se carga el area para mostrarlo
        li $v0, 1               # Se carga el número de llamada al sistema "print integer"
        syscall                 # Llamada al sistema
exit:

        li   $v0, 10            # Se carga el número de llamada al sistema para salir
        syscall                 # Llamada al sistema
```

Practica1_example1.7_resuelto.c

```
#include <stdio.h>

int main()
{
    int base=8;                     // Carga el valor 8 - base
    int altura=5;                   // Carga el valor 5 - altura
    int areaRect=base*altura;       // Multiplica los valores y almacena el resultado en $s2 - área rectángulo.

    int areaTrian=areaRect/2;       // Divide el área del rectángulo entre 2 - área triángulo.

    printf("%d",areaTrian);
}
```

Practica1_example2.c

```
#include <stdio.h>

int main(){
    int i = 0;
        int suma = 0;
        int resto = 0;

        for(i=0; i < 51; i = i+1){
                resto = i % 2;
                if (resto == 0){
                    suma = suma+i;
                }
        }
        printf("%d ",suma);
        return 0;
}
```

Practica1_example2.asm

```
    .text                           # Comienza sección de código
    .globl start                    # Símbolo exportado para que sea conocido globalmente

start:
        li      $s0, 0              # i = 0
        li      $s1, 51             # max = 51
        lw      $t0, sumatorio      # suma = 0

loop:
        addi $s0, $s0, 1            # i = i + 1
        beq $s0, $s1, end           # cuando i > max, termina

        div $s3, $s0, 2             # i / 2 (para ver si es par).
        mfhi $t1                    # Se guarda el resto de la división.
        beq $t1, $zero, suma        # Si el resto es 0 (par), lo suma.

        j       loop                # Vuelta al bucle.

suma:
        add     $t0, $t0, $s0       # sum = sum + i
        sw $t0, sumatorio           # Se carga el nuevo valor del sumatorio.
        j       loop                # Vuelta al bucle.

end:
        lw $a0, sumatorio           # Se carga el sumatorio para mostrarlo
        li $v0, 1                   # Se carga el número de llamada al sistema "print integer"
        syscall                     # Llamada al sistema

exit:
        li   $v0, 10                # Se carga el número de llamada al sistema para salir
        syscall                     # Llamada al sistema

.data
        sumatorio: .word 0
```

Practica1_example2.4_resuelto.c

```
#include <stdio.h>

int main(){
    int i = 0;
        int contP = 0;
        int contI = 0;
        int resto = 0;

        for(i=1; i < 51; i = i+1){
                resto = i % 2;
                if (resto == 0){
                        contP = contP+1;
                } else {
                        contI = contI+1;
                }
        }
        printf("%d ",contP);
        printf("%d ",contI);
        return 0;
}
```

Practica1_example3.1.asm

```
        .text
        .globl start

start:
        # Inicialización de registros.
        la $s0, size                    # $s1 = size
        lw $s1, 0($s0)
        ori $t0, $0, 0                  # $t0 = cont
        ori $s2, $0, 0                  # $s2 = sum
        ori $s3, $0, 0                  # $s3 = pos
        ori $s4, $0, 0                  # $s4 = neg

        # <init>
        ori $s5, $0, 0                  # $s5 = i
        la $s6, array                   # $s6 = &array

for:
        # if (<cond>)
        bge $s5, $s1, done

        # <for-body>
        lw $s7, 0($s6)
        add $s2, $s2, $s7
        add $t0, $t0, 1
        blez $s7, negativos
        add $s3, $s3, $s7
        j update

negativos:
        bgez $s7, update
```

```
        add $s4, $s4, $s7

update:
        # <update>
        addi $s5, $s5, 1
        addi $s6, $s6, 4                # Mover puntero del array
        j for

done:
        move   $a0, $t0
        li   $v0, 1
        syscall

        li $v0, 4
        la $a0, newline
        syscall

        move   $a0, $s2
        li   $v0, 1
        syscall

        li $v0, 4
        la $a0, newline
        syscall

        move   $a0, $s3
        li   $v0, 1
        syscall

        li $v0, 4
        la $a0, newline
        syscall

        move   $a0, $s4
        li   $v0, 1
        syscall

exit:
        li   $v0, 10
        syscall

# Inicialización de los datos
.data
        size:   .word 15
        array:  .word 7, 3, 1, -25, 99, 54, 7, 9, -25, -123, 98, 56, 0, 34, -39
        newline: .asciiz "\n"
```

Practica1_example3.2.asm

```
        .text
        .globl start

start:
        # Inicialización de registros.
        la $t0, array                   # $t0: array
        li $t1, 0                       # $t1: cont = 0

for:
        # if (<cond>)
        lb   $a0, 0($t0)                # Carga el elemento actual del array.
    beqz $a0, done                      # Comprueba si el recorrido del array ha terminado.

        li $v0, 11                      # "Print" del caracter actual (que está cargado en $a0).
        syscall

        li $v0, 4                       # "Print" del Salto de línea
        la $a0, newline
        syscall

        # FALTA                         # Mover puntero del array
    # FALTA                             # $t1: cont += 1

        j     for

done:
        li $v0, 4                       # "Print" del texto.
        la $a0, txt
        syscall

        li $v0,1                        # "Print" de "cont".
    add $a0, $0,$t1
    syscall

exit:
        li $v0, 10                      # Carga el número de llamada al sistema para salir
        syscall                         # Llamada al sistema


# Inicialización de los datos
.data
        array:  .asciiz "Hola don Pepito, hola don José"
        newline: .asciiz "\n"
        txt: .asciiz "El número de caracteres de la frase es: "
```

Introducción a la Ciberseguridad

Práctica 2: Sistemas Operativos

Marta Beltrán Pardo
Miguel Calvo Matalobos
Agradecimientos (versiones anteriores): Isaac Martín de Diego y Alberto Fernández Isabel

Contenidos

1. Introducción

En esta práctica vamos a trabajar con **GNU/Linux**, que es un tipo de sistema operativo (SO) UNIX, normalmente de código abierto, multiplataforma, multiusuario y multitarea. Todo su código fuente puede ser utilizado, modificado y redistribuido libremente por cualquier persona, empresa o institución bajo los términos de la Licencia Pública General de GNU. Se considera un sistema operativo robusto, portable, versátil, escalable, rápido. La mayor parte de su código está escrito en lenguaje de programación C y puede encontrarse en gran cantidad de dispositivos como teléfonos inteligentes, automóviles, supercomputadoras, electrodomésticos, ordenadores domésticos, servidores, etc.

Los sistemas operativos GNU/Linux se hacen llegar a los usuarios en forma de distribuciones (Ubuntu, Debian, CentOS y un largo etcétera), que no son más que el núcleo del sistema operativo y conjuntos de interfaces, aplicaciones y programas con las cuales el sistema cuenta directamente al ser instalado. Si hace falta ampliar este conjunto, se pueden descargar más desde diferentes repositorios. En esta práctica vamos a trabajar con la distribución **Kali Linux**, basada en Debian GNU/Linux y diseñada principalmente la seguridad informática (sobre todo ofensiva).

Figura 1. Algunas distribuciones de GNU/Linux
(https://commons.wikimedia.org/wiki/File:Linux_distro_foto_no_exif_(1).jpg)

Figura 2. Logotipo de Kali Linux
(https://commons.wikimedia.org/wiki/File:Kali_Linux_2.0_wordmark.svg)

En parte, gracias a que son sistemas de código abierto, su diseño facilita que sus usuarios adquieran, si así lo quieren, conocimientos más profundos sobre el SO, su configuración funcionamiento, etc. Para alcanzar esta meta, es imprescindible leer documentación, consultar los manuales de los propios comandos y, por supuesto, practicar.

En esta práctica, mediante pequeñas explicaciones y ejercicios prácticos, se tratarán diversos temas como el árbol de directorios en Linux, algunos comandos básicos, la administración en Linux, los procesos, la instalación de programas y actualizaciones, los logs del sistema, el control de acceso y contraseñas, etc.

Recurriremos para ello a la virtualización, de manera que tengáis una distribución GNU/Linux disponible para trabajar en vuestros equipos. **VirtualBox** es un software de código abierto, perteneciente a Oracle Corporation, que actúa como hipervisor y, por ende, permite virtualizar diferentes sistemas informáticos creando máquinas virtuales donde el usuario puede ejecutar otros SO. Este software es compatible con Windows, Linux o macOS. Al configurar una nueva máquina virtual, el usuario puede especificar cuántos núcleos de CPU desea adjudicarle a su máquina virtual, cuánta memoria RAM, espacio en disco duro, etc.; en este punto, debe tenerse en cuenta que los valores del hardware virtualizado que vamos a adjudicar a la máquina virtual no deben superar, en ningún caso, los del host físico (además, se debe dejar un margen en los recursos para que este host sea capaz de ejecutar su propio SO y el software que corre en él, como puede ser el propio VirtualBox). Esta herramienta puede descargarse aquí.

2. Material de la práctica

A continuación, se exponen los archivos necesarios para la realización de esta práctica, que pueden descargarse desde el Aula Virtual:

- **Practica2_Guion.pdf**: este documento. Se corresponde con el guion de la práctica y en él están contenidas todas las explicaciones necesarias para su realización.
- **Practica2_CommandReference.pdf**: hoja con los principales comandos para UNIX/Linux y su explicación.
- **Practica2_Material.zip**: archivo comprimido que contiene el material necesario (los ficheros “shadow” y “passwd” para la realización de los ejercicios del apartado “Control de acceso y contraseñas”. Para utilizarlo, se debe descomprimir.

Además, será necesario descargar el programa **VirtualBox** en su última versión. Este programa, servirá para crear la máquina virtual que se utilizará a lo largo del desarrollo de esta práctica. Puede descargarse desde el siguiente enlace. También será necesario el pack de extensiones “Oracle VM VirtualBox”, que permitirá la configuración y el uso de ciertos parámetros y características en las máquinas virtuales (puede descargarse aquí).

También hay que descargar una distribución del sistema operativo “**Kali Linux**”. Este SO se ofrece en diferentes versiones, entre ellas, una preparada para su ejecución como MV para VirtualBox. Este sistema operativo será el utilizado a lo largo del desarrollo de esta práctica y puede descargarse, en su última versión, de este enlace.

3. Normativa y evaluación

En este apartado se detalla el formato de entrega de la práctica y la forma en la que se evaluará la misma:

- El porcentaje de la nota final de la asignatura al que corresponde esta práctica puede consultarse en la Guía docente de la propia asignatura.
- La práctica deberá realizarse de manera individual.
- Cada estudiante deberá:
    - Leer, comprender y realizar los ejercicios propuestos en esta guía.
    - Realizar una prueba tipo test que se compartirá en el Aula Virtual al final de la segunda sesión de la práctica.
- La calificación de la práctica dependerá de la realización de esta prueba tipo test a través del Aula Virtual. NO será necesario entregar ninguna memoria ni documento de prácticas, pero sí deberán haberse realizado los ejercicios propuestos en esta guía para comprender las preguntas de la prueba tipo test y poder superarla con éxito.

4. Enunciado de la práctica

En este apartado se describirán las distintas actividades, programas y ejercicios que deberá realizar cada grupo de prácticas.

Instrucciones

Antes de comenzar a realizar la práctica, es necesario descargar tanto la herramienta VirtualBox (es importante que selecciones la opción asociada al sistema operativo que tengas instalado en tu ordenador y a su arquitectura del sistema), como una distribución del sistema operativo "Kali Linux" (este SO se ofrece en diferentes versiones, entre ellas y recomendada para la realización de esta práctica, una preparada para su ejecución como MV para VirtualBox) y el pack de "extensiones de Oracle VM VirtualBox".

Una vez descargados, se procederá a instalar la aplicación Oracle VM VirtualBox. Para ello, seguiremos la instalación guiada, tal y como se muestra en la Figura 1.

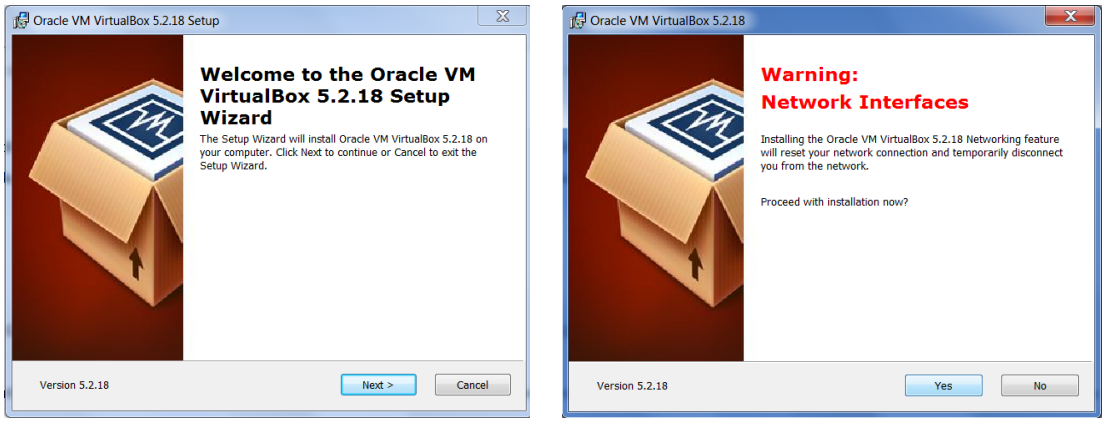


Figura 1. Instalación de Oracle VM VirtualBox.

Una vez instalado VirtualBox, deben instalarse el pack de extensiones "Oracle VM VirtualBox Extension Pack" descargado previamente. Para ello, bastará con hacer doble clic sobre el archivo

descargado y pulsar el botón “Instalar” de la pantalla que esta acción habrá abierto. Tras aceptar el acuerdo de licencia y aceptar los permisos de instalación, el paquete habrá quedado instalado.

El siguiente paso, será la importación de la máquina virtual de “Kali Linux” descargada previamente. Para ello, en primer lugar, se debe abrir el software VirtualBox (véase la Figura 2) e ir al menú “Archivo” → “Importar servicio virtualizado” (Figura 3); esta acción abrirá un desplegable como el que puede observarse en la Figura 4, donde debe seleccionarse, pulsando en el icono del directorio, el archivo “.ova” correspondiente a la máquina virtual de “Kali Linux” que se ha descargado. En caso de haberse descargado una iso de “Kali Linux” en lugar de la máquina virtual ya creada, debe pincharse sobre “Nueva” y configurar todas las opciones que el software nos solicite (nombre, tipo, versión, cantidad de RAM, tamaño de disco duro, etc.).

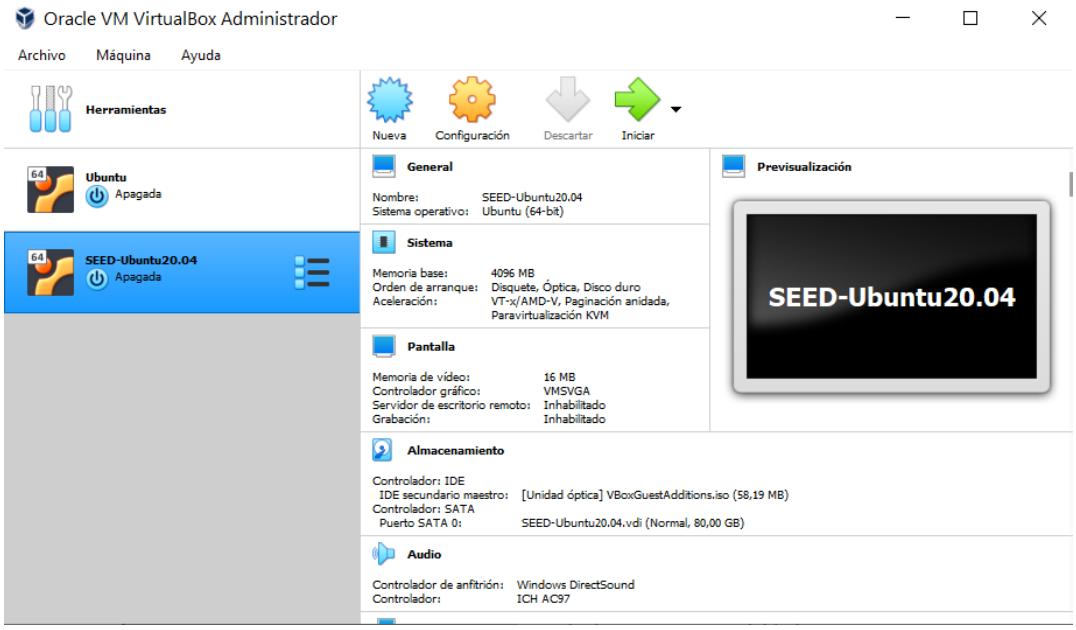


Figura 2. Pantalla principal de VirtualBox.

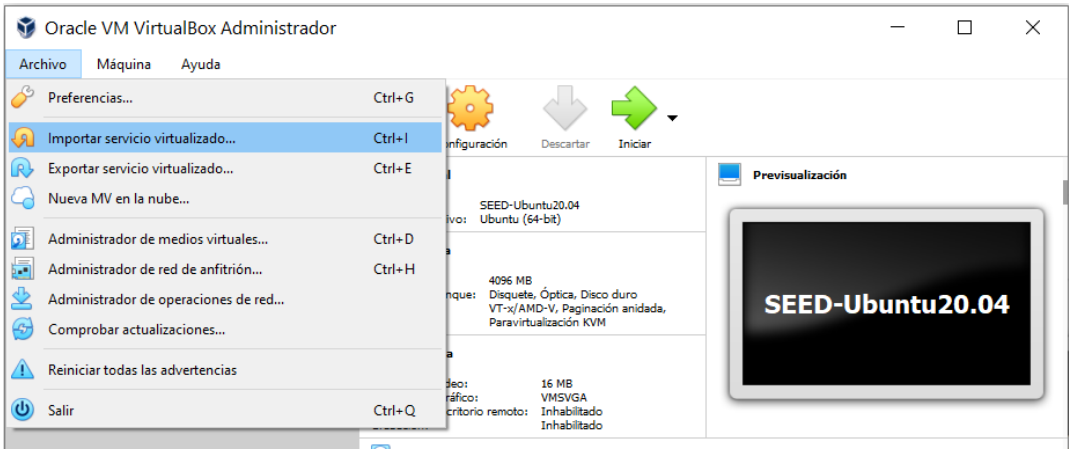


Figura 3. Menú de importación de servicio virtualizado en VirtualBox.

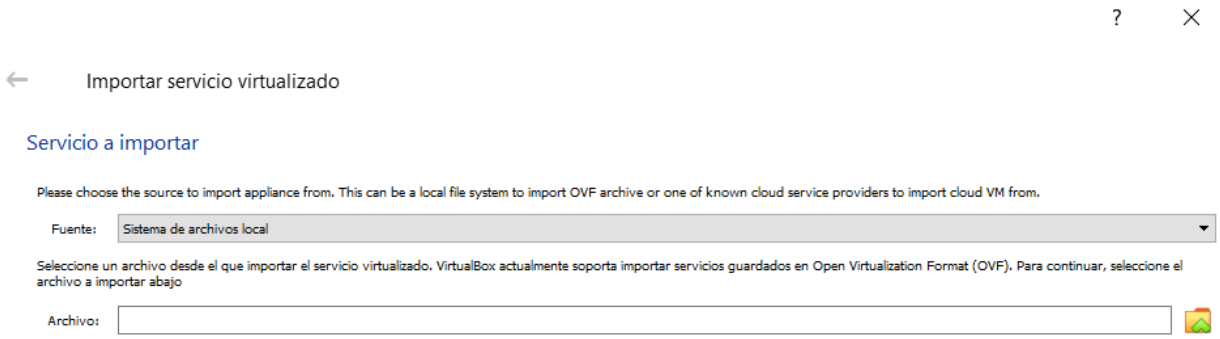


Figura 4. Desplegable para la importación de un servicio virtualizado en VirtualBox.

Una vez revisados todos los parámetros de la máquina virtual que se va a importar (es importante que se tenga en cuenta tanto el número de CPUs como el tamaño de la RAM para, en ningún caso, superar la CPU/RAM del host anfitrión). Véase la Figura 5.

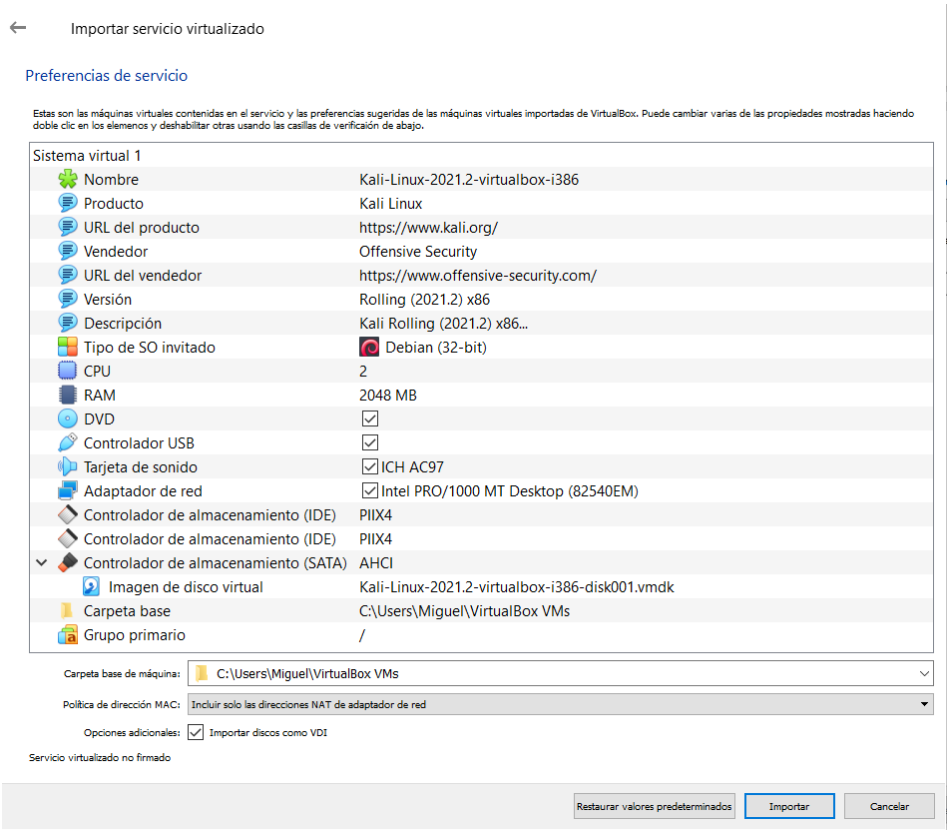


Figura 5. Revisión de la configuración del servicio virtualizado a importar en VirtualBox.

Tras pulsar el botón “Importar”, saltará un mensaje de acuerdo de licencia de software que, después de leerse en la URL indicada en el propio mensaje, debe aceptarse (con el botón “Acepto”); tal y como se muestra en la Figura 6. La aceptación de este acuerdo comenzará la importación de la MV, que puede tardar varios minutos (véase la Figura 7).

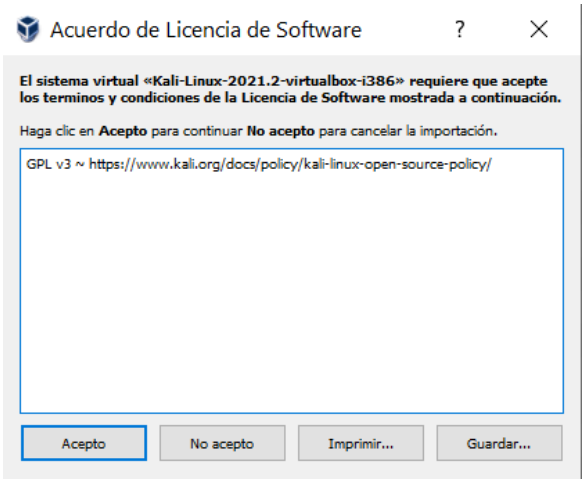


Figura 6. Acuerdo de licencia de software en la importación de la MV "Kali Linux" desde VirtualBox.

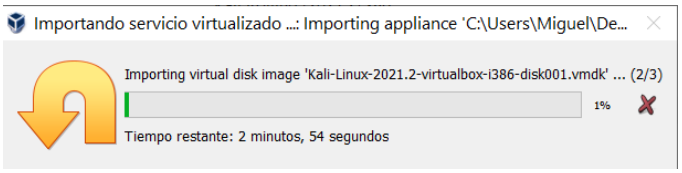


Figura 7. Importación del servicio virtualizado desde VirtualBox.

Al finalizar este proceso, se mostrará de nuevo la pantalla principal del software VirtualBox y podrá observarse en ella la MV que se acaba de importar. Para arrancarla, debe hacerse doble clic sobre ella (véase la Figura 8). Si se desea configurar algún parámetro tras la importación, bastará con hacer clic derecho sobre la máquina a configurar y seleccionar la opción "Configuración" (véase la Figura 9).

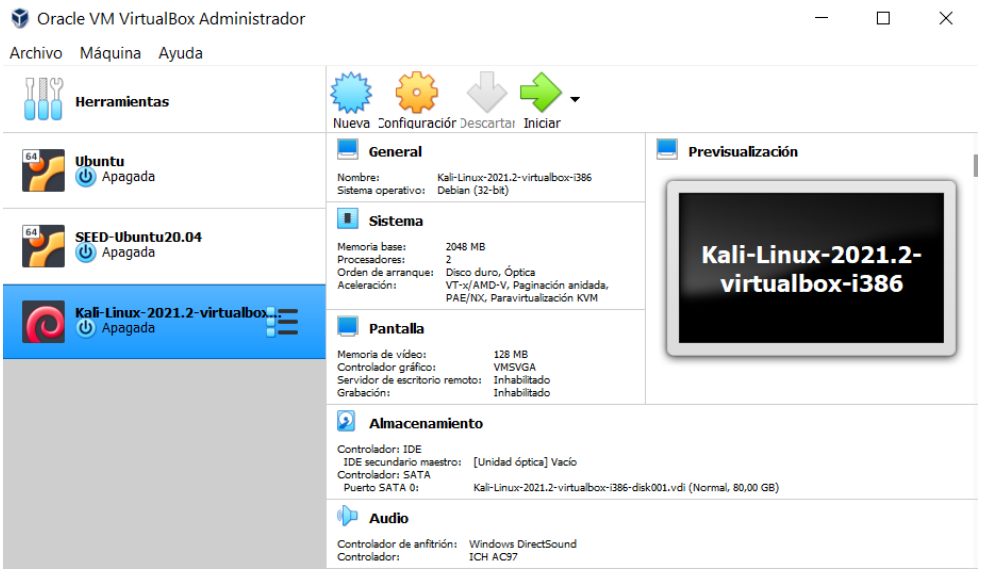


Figura 8. Pantalla principal de VirtualBox con la MV "Kali Linux" importada.

NOTA: Si al tratar de iniciar la máquina, VirtualBox muestra un mensaje de error como el que se observa en la Figura 10, es necesario dirigirse a la máquina virtual en cuestión en la pantalla principal de VirtualBox y hacer clic derecho sobre ella, seleccionando "Configuración" en el menú desplegable. Después, en el menú de la izquierda, seleccionar "USB" y cambiar la opción seleccionada "Controlador USB 2.0 (OHCI + EHCI)" por la opción "Controlador USB 1.1 (OHCI)". Véase la Figura 11.

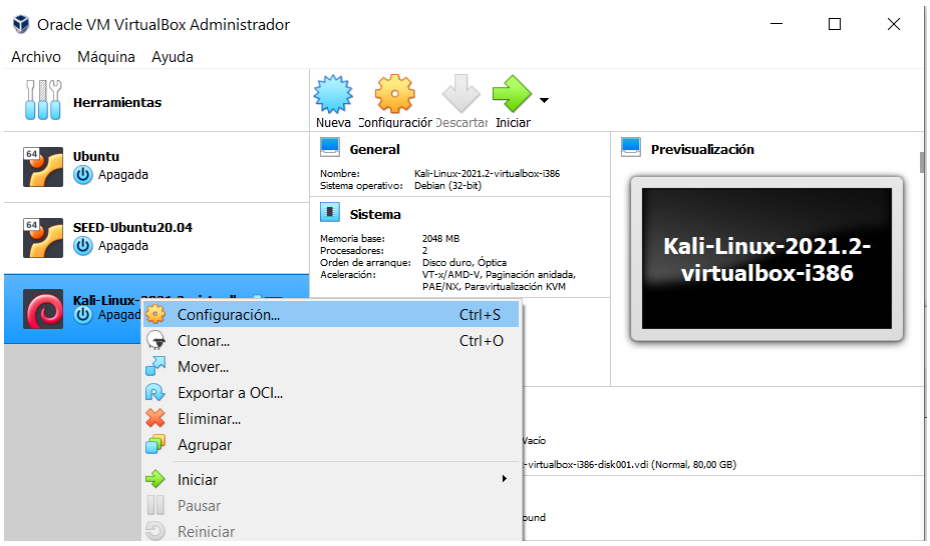


Figura 9. Configuración de una MV ya importada en VirtualBox.

Figura 10. Mensaje de error al iniciar la MV "Kali Linux" en VirtualBox.

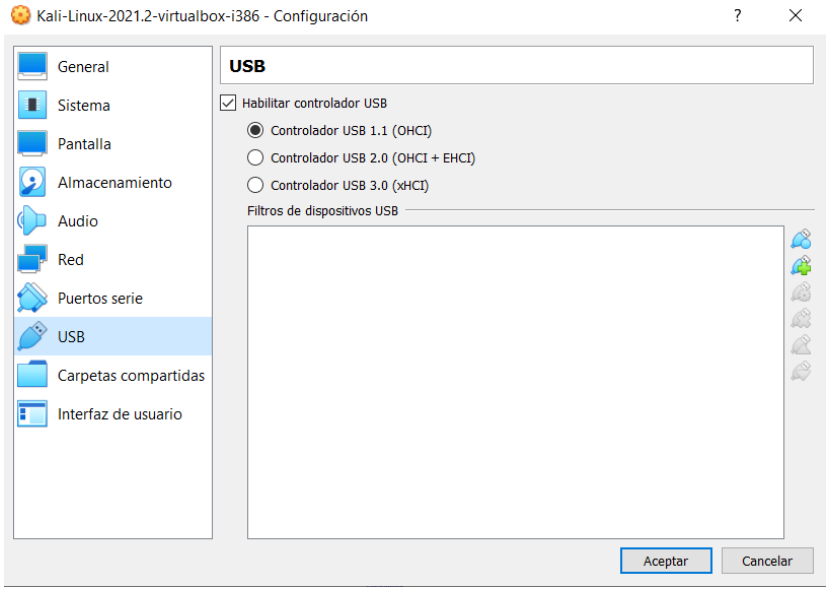


Figura 11. Configuración del controlador USB de una MV en VirtualBox.

Para apagar la máquina, podrá hacerse como con cualquier sistema operativo ("Inicio" → "Apagar") o utilizando el menú "Archivo" → "Cerrar" (dentro de la pantalla de la propia máquina), tal y como puede observarse en la Figura 12.

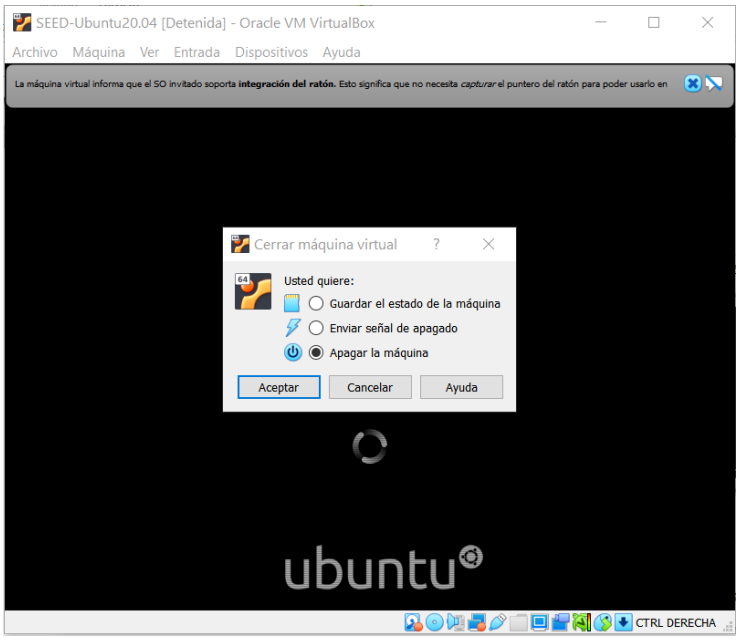


Figura 12. Apagado de una MV en VirtualBox.

Para iniciar sesión dentro de la MV de “Kali Linux” que acaba de importarse, deben utilizarse las credenciales predeterminadas de la misma. Estas son:

- Usuario: kali
- Contraseña: Kali

Tras iniciar sesión, para cambiar la distribución de teclado (por defecto está en inglés), debe hacerse clic sobre la terminal (véase la Figura 13), escribir “`sudo dpkg-reconfigure keyboard-configuration`” (sin las comillas) y pulsar Enter. Este comando nos abrirá el configurador del teclado, a través del cual debemos elegir el lenguaje de este y guardar los cambios.

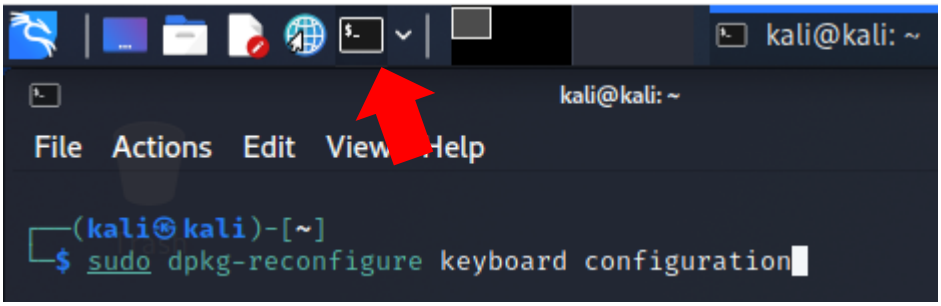


Figura 13. Cambio de distribución de teclado en Kali Linux.

NOTA: Si se utiliza Windows como host anfitrión, para que la pantalla de la MV escale automáticamente y pueda verse a pantalla completa, se recomienda la instalación de “VirtualBox Guest Additions”. Para ello:

- Tras ejecutar la MV, en el menú superior, debe seleccionarse la opción “Dispositivos” → “Insertar imagen de CD de las <<Guest Additions>>”.
- En la máquina, tras iniciar sesión, debe abrirse una terminal (véase la Figura 13), escribirse “`sudo bash /media/cdrom0/VBoxLinuxAdditions.run`” (sin las comillas) y pulsar Enter; esta acción solicitará la contraseña del usuario de la máquina y, tras introducirla, nos pedirá que aceptemos la instalación (debe escribirse “yes” y pulsar Enter).
- Reiniciar la máquina.

- Más información al respecto aquí.

Tras la finalización de esta práctica, el alumno deberá ser capaz de manejarse en un sistema operativo Linux, así como saber utilizarlo mediante la terminal, conocer sus principales comandos, la utilidad de estos y saber aprovecharlos. Se recomienda el uso del comando “man” (comando que muestra el manual de otros comandos; su uso es man *comando* dentro de una terminal). El alumno puede apoyarse también en la hoja de comandos facilitada junto a esta guía (Practica2_CommandReference.pdf) y/o mediante búsquedas en Internet.

1. Linux: Árbol de directorios

En Linux, sin importar el número de discos duros y/o particiones que se tengan, los directorios y los ficheros se organizan en forma de árbol o árbol invertido (véase la Figura 14), siendo la raíz el directorio "/" (barra). Además, este SO, está pensado para poder ser dividido en partes más pequeñas, que, a su vez, pueden estar contenidas en su propio disco duro o partición. Las principales partes divisibles son el sistema de archivos raíz o "/", /usr, /var y /home, teniendo cada una de ellas un propósito diferente (todos los comandos están en un mismo lugar, los archivos de datos en otro, la documentación en otro, etc.).

Tal y como ya se ha mencionado, en Linux, todos los archivos y dispositivos de almacenamiento se muestran en un único árbol de directorios, mediante una estructura jerárquica. La dirección o ruta de un fichero o directorio dentro de esta estructura, también llamada *path*, puede ser absoluta (cuando se indica la ruta completa; por ejemplo, "/home/luis/trabajo/informe.pdf") o relativa (cuando se indica la ruta a partir del directorio en el que nos encontramos; por ejemplo, si estamos situados en /home, y queremos indicar la ruta del informe, podríamos especificar "./luis/trabajo/informe.pdf"). Cabe destacar que, en una ruta, el símbolo "." Indica el directorio actual y ".." el directorio padre. Veamos algunos ejemplos:

- Observando la Figura 14, si nos situamos en el directorio /home/marcos y queremos indicar la ruta relativa de "direcciones.txt", podemos decir que esta es "../luis/trabajo/direcciones.txt":

../	**luis/**	**trabajo/**	**direcciones.txt**
Se vuelve al directorio "/home".	Se avanza al directorio "luis" (/home/luis)	Se avanza al directorio "trabajo" (/home/luis/trabajo)	Se indica la ruta final del fichero "direcciones.txt" (/home/luis/trabajo/direcciones.txt)

- Observando la Figura 14, si nos situamos en el directorio /usr/bin y queremos indicar la ruta relativa de "/", podemos decir que esta es "../..:

../	**..**
Se vuelve al directorio "/usr".	Se vuelve al directorio "/".

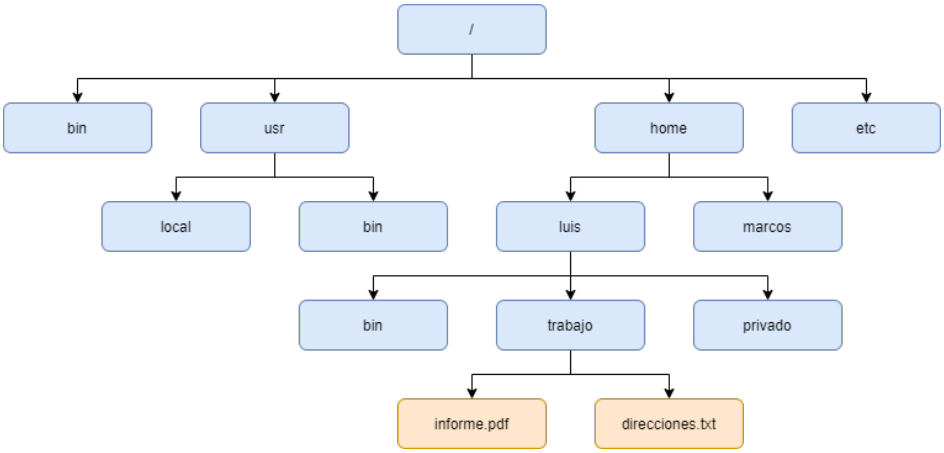


Figura 14. Ejemplo de estructura de árbol en los directorios de Linux.

A continuación, se detallan algunos de los directorios (d) y ficheros (f) más relevantes dentro de un SO Linux:

- **/bin**. Este directorio, contiene los archivos ejecutables que se corresponden con los diferentes comandos que todos los usuarios del sistema pueden utilizar.
- **/boot**. Este otro directorio es el encargado de almacenar todos los archivos que se necesitan para arrancar el sistema (incluido el propio kernel).
- **/dev**. Es el directorio encargado de almacenar los identificadores de los diferentes dispositivos. Entre ellos, se encuentran los de almacenamiento (por ejemplo, /dev/sda1, /dev/sda2, /dev/sda3, etc.), los de entrada/salida (por ejemplo, /dev/tty, /dev/tty0, /dev/tty1, etc.).
- **/home**. Se trata del directorio en el que pueden encontrarse los directorios personales de cada uno de los usuarios del sistema.
- **/root**. Al contrario que el resto de los usuarios, el usuario root tiene su directorio personal en /root (no en /home). Por lo tanto, es el directorio local para el usuario root.

- **/sbin**. Este directorio contiene archivos ejecutables tanto de las utilidades para el arranque del sistema, como para la solución de problemas, mantener el sistema de archivos, etc.

- **/usr**. Principalmente, es un directorio que incluye archivos de comandos (en modo de solo lectura) que son accesibles por todos los usuarios del sistema.

- **/var**. En este directorio se almacenan diferentes subdirectorios y archivos variables (de ahí su nombre) o cambiantes como, por ejemplo, los logs del sistema.

- **/lib**. Es el contenedor de las diferentes librerías que comparten los programas en el sistema de archivos raíz.

- **/proc**. En este directorio se almacena un sistema de archivos virtual que no existe físicamente en el disco duro, está creado únicamente en memoria y se utiliza para ofrecer información relacionada con el propio sistema.

- **/etc**. Este directorio contiene, principalmente, los archivos de configuración (tanto del sistema como de parte del software instalado en él). Entre estos archivos cabe destacar:

    - **/etc/passwd**. En este fichero puede encontrarse una pequeña base de datos de los usuarios del sistema con nombre de usuario, nombre real, contraseña (cifrada (o “hasheada”), etc.

    - **/etc/group**. Se trata de un fichero similar a /etc/passwd, pero con la información de los diferentes grupos de usuarios.

    - **/etc/fstab**. En este fichero están contenidos los diferentes sistemas de archivos que deben montarse en el arranque del sistema.

    - **/etc/shadow**. Se trata de un fichero que contiene contraseñas cifradas (en sistemas donde se encuentre instalado y/o activo el sistema de contraseñas ocultas). A diferencia de /etc/passwd, en este fichero también se almacena la fecha de caducidad de una contraseña, el tiempo mínimo entre cambios de contraseña, etc.

    - **/etc/security**. En este caso, se trata de un directorio que contiene ficheros en los que se establecen una serie de condiciones de seguridad por defecto para el equipo.

- **/etc/shells**. Es un fichero en el que se lista las diferentes rutas de los intérpretes de línea de comandos admitidos en nuestro sistema.

El comando "cd", desde la terminal de Linux, se utiliza para poder movernos entre directorios. Por otro lado, el comando "ls", también desde la terminal de Linux, muestra o lista los ficheros o subdirectorios que hay contenidos en un directorio.

Recuerda que, si necesitas ayuda con el funcionamiento o el uso de alguno de los comandos, puedes utilizar el comando "man", seguido del nombre del comando del que tengas dudas sobre su uso (por ejemplo: man cd); esto te mostrará una pequeña explicación del comando y las diferentes opciones para utilizarlo. También puedes utilizar la hoja de comandos facilitada junto a esta guía (Practica2_CommandReference.pdf). Además, como irás aprendiendo a la vez que adquieras experiencia… ¡los buscadores son nuestros amigos!

Ejercicio 1.1

En una terminal dentro de la MV de Kali Linux dirígete hasta la ruta del sistema de archivos raíz ("/") y lista los subdirectorios que allí están contenidos:

- cd /
- ls

¿Observas alguno de los directorios y/o archivos que se han mencionado en párrafos anteriores? ¿Cuáles?

Ejercicio 1.2

Utilizando el comando "cd", dirígete hasta el directorio "/tmp" (puedes utilizar la ruta absoluta o la ruta relativa), ¿Qué contiene ese directorio?

2. Linux: Comandos básicos

Un intérprete de comandos es un programa informático que permite traducir órdenes provenientes de los usuarios. Estas órdenes se interpretan mediante un conjunto de instrucciones que se envían directamente al conjunto de herramientas que forman el sistema operativo. Es decir, un intérprete de comandos (también llamado *Shell*) es un programa que hace de interfaz de texto entre el usuario y el sistema operativo. La sintaxis típica en un intérprete de comandos es:

- (prompt*[1]) comando [parámetro 1] … [parámetro n]

> *[1] El ***prompt*** es el carácter (o conjunto de caracteres) que, en una línea de comandos, se muestra al principio. Esto indica que la línea de comandos está a la espera de órdenes por parte del usuario. El carácter o caracteres puede ser diferente dependiendo del intérprete de comandos.

Algunos comandos útiles en Linux son:

- **cat**. Muestra el contenido de un fichero.
- **cd**. Cambia de directorio. Si no lleva ningún argumento, cambia al directorio home.
- **cp**. Copia un fichero o directorio.
- **echo**. Imprime un mensaje pasado como parámetro.
- **find**. Busca ficheros y directorios.
- **ls**. Muestra el contenido de un directorio.
- **mv**. Renombra o mueve un fichero o directorio.
- **pwd**. Indica cual es el directorio actual.
- **mkdir**. Crea un nuevo directorio.

- **rm**. Borra un fichero o directorio.
- **chmod**. Proporciona permisos de escritura, lectura o ejecución a un fichero.

Recuerda que, si necesitas ayuda con el funcionamiento o el uso de alguno de los comandos, puedes utilizar el comando "man", seguido del nombre del comando del que tengas dudas sobre su uso (por ejemplo: man cd); esto te mostrará una pequeña explicación del comando y las diferentes opciones para utilizarlo. También puedes utilizar la hoja de comandos facilitada junto a esta guía (Practica2_CommandReference.pdf). Además, como irás aprendiendo a la vez que adquieras experiencia… ¡los buscadores son nuestros amigos!

Ejercicio 2.1

Investiga que comando debes utilizar para crear un directorio y cómo debes hacerlo. Una vez sepas hacerlo, crea tres directorios en el directorio "home" del usuario "kali" de tu MV. Llámalos "practica1", "practica2" y "otros". ¿Qué comando has utilizado? Para crearlos, ¿has usado una ruta absoluta o una ruta relativa?

Ejercicio 2.2

Mediante el comando "df" y apoyándote en el manual del comando ("man df"), ¿Qué comando debes utilizar para mostrar el espacio en disco con una presentación "más amigable"? ¿Y si quieres mostrar además el tipo de cada uno de los sistemas de ficheros?

Ejercicio 2.3

El comando "ls" muestra el contenido de un directorio. Gracias a alguno de sus parámetros, pueden mostrarse también archivos ocultos, ¿Cuántos archivos ocultos hay en el directorio /home/Kali? ¿Con qué comando los has listado?

Ejercicio 2.4

Dirígete al directorio "practica2" que has creado en ejercicios anteriores. Una vez en él, crea un fichero llamado "README.txt" y edítalo (por ejemplo, con "nano", "gedit" o "vim") para escribir en él el nombre de los integrantes del grupo. ¿Qué comando has utilizado para crear el fichero? ¿Y para editarlo?

Ejercicio 2.5

Muestra el contenido del fichero que acabas de crear en la propia consola, sin utilizar ningún editor de texto. ¿Qué comando te permite hacer esto?

Ejercicio 2.6

Apoyándote en el manual del comando "cp", copia tres veces, en el mismo directorio, el fichero "README.txt". El nombre de los ficheros resultantes será: "info.txt", "datos.txt" y "grupo.txt". ¿Cómo lo has hecho?

Ejercicio 2.7

Borra el fichero antiguo "README.txt" y quédate con el resto de los ficheros. ¿Qué comando te ha permitido borrar el fichero?

Ejercicio 2.8

Borra el directorio "practica1" que creaste antes. ¿Qué comando te ha permitido borrar el directorio? ¿En qué se diferencia con el comando que has usado para borrar el fichero?

3. Linux: Comprimir y descomprimir archivos

El objetivo principal de comprimir o empaquetar archivos es reducir su tamaño para que, entre otras, estos ocupen menos y así poderlo transferir, guardar, etc. De una forma más cómoda. En Linux, además de poder hacerse mediante software gráfico, existe la posibilidad de comprimir y descomprimir archivos desde la terminal. Los archivos comprimidos, cada uno con diferentes propiedades, pueden ser muy variados; entre estos destacan los ficheros “tar”, los “gz”, los “bz2”, los “tar.gz” y los ficheros “zip”.

- **tar**.
    - Comprimir: `tar -cf archivo-comprimido.tar /directorio/a/comprimir`
        - -c indica que se debe crear un archivo.
        - -f indica que el siguiente argumento es el nombre del fichero final.
    - Descomprimir: `tar -xf archivo.comprimido.tar /directorio/a/descomprimir`
        - -x indica que se debe descomprimir un archivo.
        - -f indica que el siguiente argumento es el nombre del fichero a descomprimir.
- **gz**.
    - Comprimir: `gzip -9 archivo-comprimido.gz`
        - -9 indica el factor de compresión (9 es el mayor).
    - Descomprimir: `gzip -d archivo-comprimido.gz`
        - -d indica que se debe descomprimir un archivo.
- **bz2**.
    - Comprimir: `bzip2 archivo-comprimido.bz2`
    - Descomprimir: `bzip2 -d archivo-comprimido.bz2`

  - -d indica que se debe descomprimir un archivo.

- **tar.gz**.
  - Comprimir: tar -czf archivo-comprimido.tar.gz /directorio/a/comprimir
    - -c indica que se debe crear un archivo.
    - -z indica que debe utilizarse el compresor gzip.
    - -f indica que el siguiente argumento es el nombre del fichero final.
  - Descomprimir: tar -xzf archivo.comprimido.tar.gz /directorio/a/descomprimir
    - -x indica que se debe descomprimir un archivo.
    - -z indica que está comprimido con gzip.
    - -f indica que el siguiente argumento es el nombre del fichero a descomprimir.
- **zip**.
  - Comprimir: zip archivo-comprimido.zip /directorio/a/comprimir
  - Descomprimir: unzip archivo.comprimido.zip /directorio/a/descomprimir

Recuerda que, si necesitas ayuda con el funcionamiento o el uso de alguno de los comandos, puedes utilizar el comando “man”, seguido del nombre del comando del que tengas dudas sobre su uso (por ejemplo: man cd); esto te mostrará una pequeña explicación del comando y las diferentes opciones para utilizarlo. También puedes utilizar la hoja de comandos facilitada junto a esta guía (Practica2_CommandReference.pdf). Además, como irás aprendiendo a la vez que adquieras experiencia… ¡los buscadores son nuestros amigos!

Ejercicio 3.1

Comprime el directorio “practica2” que creaste en ejercicios anteriores y guárdalo en “/tmp”. El nombre del fichero comprimido debe ser “entrega.tar”. ¿Qué comando has utilizado?

Ejercicio 3.2

Dirígete al directorio “/tmp” y descomprime allí mismo el fichero que has creado en el ejercicio anterior (“entrega.tar”), ¿Con qué comando lo has conseguido hacer?

4. Linux: Administración

En todos los sistemas operativos de tipo UNIX, por defecto, existe un usuario básico llamado “root”. Este usuario, se corresponde con el administrador del sistema o superusuario. Su directorio de trabajo, tal y como se menciona en el apartado “Linux: Árbol de directorios”, es “/root”. Este usuario no tiene limitados sus privilegios (puede borrar, leer, escribir, instalar, etc. Cualquier cosa), por lo tanto, trabajar con este usuario supone un riesgo para la seguridad del sistema.

El resto de los usuarios (ya sean administradores o no), tienen un directorio del cuál son propietarios que está alojado en “/home”. Estos usuarios estarán identificados con un UID (*Unique identifier*), al que va asociado el nombre de usuario en minúsculas del propio usuario (por ejemplo, el usuario root tiene UID 0). Así mismo, todo usuario pertenece al menos a un grupo, que, en combinación, permiten determinar los permisos de acceso a ficheros.

Los tres tipos de permisos que un usuario o un grupo puede tener sobre los ficheros y directorios en Linux son:

- **r**. Permiso de lectura. También se representa con el número 4.
    - En los archivos, permite visualizar el contenido.
    - En directorios, permite saber que archivos y directorios contiene.
- **w**. Permiso de escritura. También se representa con el número 2.
    - En los archivos, permite modificar el contenido del archivo.
    - En directorios, permite crear archivos o subdirectorios dentro del directorio. Además, con este permiso se pueden borrar los directorios, copiar archivos en el directorio, mover, cambiar el nombre, etc.
- **x**. Permiso de ejecución. También se representa con el número 1.
    - En los archivos, permiten, valga la redundancia, ejecutarse (si fuera un programa).
    - En directorios, permite utilizar el nombre del directorio cuando se accede a archivos dentro del directorio (por ejemplo, para buscar con el comando find).

A su vez, estos permisos se asignan en tres categorías y solo el propietario (o el superusuario) puede cambiar estos permisos. Las categorías son:

- **u**. Permisos del propietario.
- **g**. Permisos del grupo.
- **o**. Permisos para otros.

Además, los directorios y los ficheros suelen estar marcados, al inicio de la cadena de caracteres que define sus permisos, con una “d” o un “-“ respectivamente.

Si se observa la Figura 15, después del carácter que define el tipo del elemento de la lista (los archivos con “-“ y los directorios con “d”), pueden encontrarse los permisos, agrupados de tres en tres (por categorías). En primer lugar, están los permisos del propietario, seguidos de las del grupo y, por último, los de otros. Por ejemplo, si se observa el directorio “midirectorio” de la Figura 15, se indica lo siguiente: “drwxr-xr-x”, donde:

- d. Indica que se trata de un directorio.
- rwx. Indica que el propietario tiene permisos de lectura, escritura y ejecución sobre el directorio.
- r-x. Indica que el grupo tiene permisos de lectura y ejecución sobre el directorio.
- r-x. Indica que “otros” tienen permisos de lectura y ejecución sobre el directorio.

También puede encontrarse (de nuevo en la Figura 15) la palabra “kali” o “root”. Esto se corresponde con el usuario y grupo respectivamente, que son los propietarios del archivo o directorio en sí.

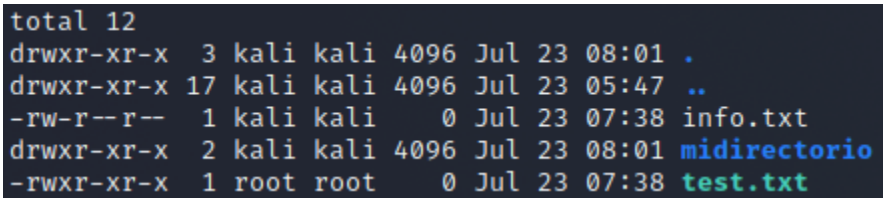


Figura 15. Muestra de los permisos de diferentes archivos en Linux.

Para el **manejo de grupos y usuarios** de un fichero/directorio, suele utilizarse el comando "chown". Por otro lado, para el **manejo de permisos** de ficheros y/o directorios suele utilizarse el comando "chmod". Este último comando, puede usarse de diferentes maneras. Algunas de ellas son:

- Por ejemplo, chmod u=rw,go=r info.txt, que otorga permisos de lectura ("r) y escritura ("w") para el propietario ("u") y de lectura ("r") para el grupo y otros ("g" y "o") sobre el fichero "info.txt".

- Otro ejemplo es chmod o+x info.txt. En este caso, se añade permiso de ejecución ("x") para otros ("o") en el fichero "info.txt". Si se utilizara el símbolo "-" (chmod o-x info.txt), se eliminaría el permiso de ejecución para otros.

- Un tercer ejemplo es, chmod 753 info.txt, en este caso, cada uno de los dígitos se corresponde con uno de los grupos de permisos (el primero, 7, con el del propietario; el segundo, 5, con el del grupo; y el tercero, 3, con el de otros).

  Si nos remontamos a la explicación de los permisos, el de lectura es un 4, el de escritura un 2 y el de ejecución un 1; para calcular el dígito correspondiente a cada grupo, simplemente hay que sumarlos. Por ejemplo, el 7 se correspondería con todos los permisos (4+2+1), el 5 con lectura + ejecución (4+1) y el 3 con escritura + ejecución (2+1).

  De esta forma, en el caso del ejemplo, se le estarían otorgando permisos de lectura, escritura y ejecución al propietario; lectura y ejecución al grupo; y escritura y ejecución a otros.

A veces, por los permisos del usuario que utilizamos, no podemos realizar ciertas acciones o ejecutar ciertos comandos. En los sistemas operativos de tipo Unix, existe una palabra reservada (asociada a una utilidad) llamada "sudo" (*Super User Do*), que permite **ejecutar programas con los privilegios de seguridad de otros usuarios** (normalmente el usuario root). Por ejemplo, si quisiéramos crear con el usuario "kali", dentro de la MV de la práctica, el fichero "/etc/mi-fichero.txt", deberíamos hacerlo mediante "sudo touch /etc/mi-fichero.txt", ya que el usuario "kali" por sí solo no tiene permisos para escribir en ese directorio.

Recuerda que, si necesitas ayuda con el funcionamiento o el uso de alguno de los comandos, puedes utilizar el comando “man”, seguido del nombre del comando del que tengas dudas sobre su uso (por ejemplo: man cd); esto te mostrará una pequeña explicación del comando y las diferentes opciones para utilizarlo. También puedes utilizar la hoja de comandos facilitada junto a esta guía (Practica2_CommandReference.pdf). Además, como irás aprendiendo a la vez que adquieras experiencia… ¡los buscadores son nuestros amigos!

Ejercicio 4.1

Cambia los permisos sobre el fichero “info.txt” que creaste en ejercicios anteriores para que tanto propietario, como grupo, como otros tengan permisos de lectura, escritura y ejecución. ¿Qué comando has utilizado?

Ejercicio 4.2

Utilizando una técnica diferente a la que has utilizado en el ejercicio anterior, cambia los permisos del fichero “datos.txt” para que tenga permisos de lectura y escritura para el propietario y de solo lectura para grupo y otros. ¿Cómo lo has hecho?

Ejercicio 4.3

Ayudándote de Internet… ¿Cuál es el UID más alto permitido en los sistemas Linux? Aparte del 0, que se reserva para el usuario root, ¿Hay algún otro UID que suela reservarse para usuarios especiales?

Ejercicio 4.4

Cambia el propietario y el grupo del fichero "grupo.txt" por "root" y "root" respectivamente. ¿Cómo has conseguido hacerlo?

Ejercicio 4.5

Elimina el permiso de ejecución al fichero "otros". ¿Qué comando has utilizado?

Ejercicio 4.6

Utilizando el comando “ls” y con la ayuda de “man”, encuentra la combinación de parámetros necesaria para mostrar los permisos, el propietario y el grupo de todos los ficheros del directorio “/home/kali”. ¿Qué comando has utilizado?

5. Linux: Procesos

Por cada software, comando, aplicación, herramienta, etc. Que se ejecuta en un sistema Linux, se genera, al menos, un proceso. Este proceso suele corresponderse con una tarea, que trabaja de forma independiente, con su propia misión, sus permisos, etc. El tiempo que el proceso se mantenga activo, utilizará ciertos recursos del sistema (como CPU, RAM, etc.).

Cada uno de los procesos posee un PID (*Process ID* o número de identificación de proceso), un número de 5 dígitos único para cada proceso. Este PID no se liberará hasta que el proceso al que se ha destinado termine (muera).

Los procesos pueden ejecutarse en **primer plano o en *background***:

- De forma predeterminada, los procesos iniciados por un usuario se ejecutan en primer plano; se toma la entrada del símbolo del sistema y se muestra la salida en la pantalla del ordenador. Si se está ejecutando un proceso en primer plano, la terminal evita que se inicie un nuevo proceso hasta que el existente termine.

- Los procesos de fondo no requieren entrada de teclado y puede iniciarse otro proceso desde el terminal mientras que el anterior se ejecuta en segundo plano. Para iniciar un proceso en segundo plano, al escribir el comando, un usuario puede añadir un ampersand (&) al final (por ejemplo, `rm -fr /var/www/data &`). Para cambiar entre procesos en primer y segundo plano pueden utilizarse los comandos “fg” y “bg”.

Los procesos, a su vez, pueden ser de **diferentes tipos**:

- **Init**. Se trata del primer proceso que se crea cuando se inicia una máquina Linux o Unix. Todos los procesos de un sistema están asociados a un proceso padre, por lo tanto, todos los demás procesos del sistema son hijos del proceso “init”. Este proceso no se puede eliminar (únicamente cuando se apaga el sistema) y siempre se le asigna el PID número 1.

- **Proceso padre y proceso hijo**. Los procesos, están asociados a un proceso padre o proceso principal.

- **Proceso zombi y proceso huérfano**. Cuando completan su ejecución, los procesos hijo se terminan o mueren, actualizándose entonces el proceso padre para continuar la tarea que se le asignó. Sin embargo, en ocasiones, si el proceso padre muere antes que el proceso hijo, los procesos hijo se convierten en huérfanos, y el proceso padre de todos los procesos "init" se convierte en su nuevo PID. Por otro lado, un proceso que se mata, pero aún muestra su entrada en el estado de los procesos o en la tabla de procesos se denomina proceso zombi (están muertos y no se utilizan).

- **Proceso Daemon**. Se trata de procesos en segundo plano, relacionados con el sistema, que a menudo se ejecutan con permisos de root y solicitudes de servicios de otros procesos. La mayoría de las veces se ejecutan en segundo plano.

Recuerda que, si necesitas ayuda con el funcionamiento o el uso de alguno de los comandos, puedes utilizar el comando “man”, seguido del nombre del comando del que tengas dudas sobre su uso (por ejemplo: man cd); esto te mostrará una pequeña explicación del comando y las diferentes opciones para utilizarlo. También puedes utilizar la hoja de comandos facilitada junto a esta guía (Practica2_CommandReference.pdf). Además, como irás aprendiendo a la vez que adquieras experiencia… ¡los buscadores son nuestros amigos!

Ejercicio 5.1

¿Qué parámetros muestra el comando “ps -ef”? ¿Qué significa la “e” en el comando? ¿Y la “f”?

Ejercicio 5.2

El comando "sleep" temporiza un intervalo de tiempo en Linux. Ejecuta el comando para que haga un "*sleep*" de 20 minutos y envía el proceso a *background*. ¿Qué comando has utilizado?

Ejercicio 5.3

¿Cuál es el PID del proceso que se creó en el anterior ejercicio? ¿Y el PID de su proceso padre? ¿En qué comando te has apoyado para averiguarlos?

Ejercicio 5.4

Mata el proceso que se creó al ejecutar el comando "*sleep*" de 20 minutos, ¿Qué comando has utilizado?

Ejercicio 5.5

Lanza un comando para que se haga un "*sleep*" de 5 minutos y envíalo a *background*. Después, localiza su PID y trae el proceso de nuevo a primer plano. ¿Con qué comando has conseguido recuperar el proceso en primer plano?

Ejercicio 5.6

Ayudándote de Internet… ¿Existe la posibilidad de que un proceso se configura para lanzarse al iniciar el sistema operativo? ¿Cómo?

Ejercicio 5.7

Investiga qué comando te muestra, en tiempo real, las estadísticas y el estado de todos los procesos en ejecución. ¿Qué comando es? ¿Qué información muestra?

Ejercicio 5.8

El "pipe" (|), en Linux, permite encadenar la ejecución de diferentes comandos, pasando el *output* de uno como el *input* de otro. Esta utilidad, suele ayudar mucho en las búsquedas sobre los resultados de la ejecución de un comando. Teniendo en cuenta que el comando "grep" sirve para hacer búsquedas, ¿con qué comando y qué parámetros buscarías todos los procesos que contengan la palabra "net"?

6. Linux: Instalación de programas, scripts y actualizaciones

Instalar programas y aplicaciones en Linux suele ser realmente sencillo, al igual que lo es actualizar el propio sistema operativo. Para ello, existen gran cantidad de gestores de paquetes, se puede instalar software a partir del código fuente, mediante paquetes ".deb", ejecutar scripts, etc.

- En primer lugar, cabe mencionar los **gestores de paquetes**. Como su propio nombre indica, ayudan a la gestión de paquetes (conjuntos de ficheros, normalmente dependientes de la distribución de Linux para la que han sido creados, utilizados para comprimir aplicaciones en distintos formatos y/o medios de instalación). Los gestores de paquetes permiten mantener un registro del software que está instalado en un ordenador y, además, facilitan la instalación, actualización, desinstalación, etc. De nuevo software. Entre los gestores de paquetes para Linux más destacados se encuentran apt (*Advanced Packaging Tool;* el gestor de paquetes por defecto en GNU/Linux), pacman, yum, entropy, rpm, upkg, ZYpp, etc.

  Cada gestor de paquetes se utiliza de una forma diferente. Por ejemplo, para usar apt se sigue la siguiente combinación:

  - (prompt) apt-get [configuración 1] … [configuración n] opción

  Si por ejemplo se quisiera re-sincronizar el índice de paquetes de apt, se utilizaría el comando "apt-get update", para actualizar todos los paquetes podría hacerse mediante "apt-get upgrade", si hubiera que instalar un paquete concreto bastaría con ejecutar "apt-get install nombre-paquete", etc.

- La instalación de programas desde **código fuente**, en Linux, es algo muy habitual. Al ser un sistema operativo Open Source, muchos de los desarrolladores de software para este SO cuelgan el código directamente en GitHub u otras plataformas similares, y es el propio usuario el que debe compilar e instalar el software (o, al menos, tiene la posibilidad de hacerlo); de esta forma, entre otras, se puede modificar el código a su antojo o revisarlo antes de compilarlo e instalarlo. Para esta acción, una vez descargado el código fuente, descomprimido y situados desde la terminal en el directorio en el que se encuentra, se utilizan principalmente tres comandos:

  - ./configure

        - Es importante tener en cuenta que el fichero “configure” debe tener permisos de ejecución.
        - Con este comando se configura el modo de compilación.
    - make
        - Se construye el programa.
    - sudo make install
        - Se instala el programa.

- Los paquetes **.deb** son paquetes listos para ser ejecutados e instalados (como ocurre con los .exe en Windows). Si se ejecutan en un entorno gráfico, bastará con hacer doble clic sobre el fichero .deb a instalar, lo cual lanzará la aplicación que se encargará de instalarlo de una forma sencilla. Mediante una terminal, gracias al comando “dpkg”, también pueden instalarse (utilizando “sudo dpkg -i fichero.deb”) o desinstalarse (con “sudo dpkg -r fichero.deb”) estos paquetes.

- En Linux se pueden encontrar diferentes **scripts ejecutables**. Algunos de estos scripts (los más comunes), tienen extensiones como .sh o .py. Para instalar y/o ejecutar estos scripts, es necesario dirigirse al directorio en el que se encuentra el script en sí (utilizando el comando “cd”). Para que pueda ejecutarse, es imprescindible que el fichero tenga permisos de ejecución (véase el apartado “Linux: Administración” de esta misma guía) y, desde la terminal, utilizar la siguiente combinación:
    - (prompt) intérprete nombre_script [parámetro 1] … [parámetro n]

  Donde “intérprete” (del lenguaje de programación al que corresponda el script), será el encargado de ejecutar “nombre_script”, que, a su vez, recibirá tantos parámetros como sean necesarios. Así, por ejemplo, si se quisiera ejecutar un script escrito en lenguaje bash, podría utilizarse el siguiente comando: “sh nombre_script.sh”. De la misma forma, si se fuera a ejecutar un script escrito en Python, bastaría con utilizar “python nombre_script.py”.

  Otra forma de ejecutar scripts es mediante:

    - ./nombre_script [parámetro 1] … [parámetro n]

En este caso, para que el script pueda ejecutarse, es imprescindible que, en su primera línea de código, se indique la *Shell* que va a interpretar el fichero. Por ejemplo, si se trata de bash deberá contener "#!/bin/bash" o si se está tratando de crear un script escrito en lenguaje Python, "#!/usr/bin/env python".

Recuerda que, si necesitas ayuda con el funcionamiento o el uso de alguno de los comandos, puedes utilizar el comando "man", seguido del nombre del comando del que tengas dudas sobre su uso (por ejemplo: man cd); esto te mostrará una pequeña explicación del comando y las diferentes opciones para utilizarlo. También puedes utilizar la hoja de comandos facilitada junto a esta guía (Practica2_CommandReference.pdf). Además, como irás aprendiendo a la vez que adquieras experiencia… ¡los buscadores son nuestros amigos!

Ejercicio 6.1

Utilizando el gestor de paquetes de Linux ("apt-get"), instala el paquete "chromium". ¿Qué comando has utilizado?

Ejercicio 6.2

De nuevo utilizando el gestor de paquetes de Linux ("apt-get"), desinstala (borrando también todos los ficheros de configuración) el paquete "chromium". Apóyate en el comando "man" para saber todas las opciones de "apt-get". ¿Qué comando has utilizado para la desinstalación?

Ejercicio 6.3

Crea (y ejecuta) un pequeño script en bash que actualice el índice de paquetes de apt, actualice todos los paquetes a su última versión y elimine los paquetes que se han instalado automáticamente para satisfacer las dependencias de otros paquetes, pero que ya no son necesarios. Por último, el script debe mostrar un mensaje por pantalla indicando que ya se ha terminado de ejecutar. Pega el código de tu script:

Ejercicio 6.4

Siguiendo las instrucciones del creador *1 (puedes encontrarlas en el README.md del proyecto en GitHub), descarga, compila e instala NotepadQQ en tu máquina virtual. ¿Has encontrado algún problema en la instalación? ¿Cuál?

*1 Antes de proceder con la descarga/compilación/instalación de NotepadQQ en tu máquina virtual, debes ejecutar el siguiente comando, que instalará todos los paquetes necesarios: sudo apt-get install qtbase5-dev qtchooser qt5-qmake qtbase5-dev-tools qtwebengine5-dev qttools5-dev-tools libqt5websockets5-dev libqt5svg5 libqt5svg5-dev libuchardet-dev pkg-config

7. Linux: Logs del sistema

Los logs o registros del sistema son ficheros de texto en los que se registran, de forma cronológica, la mayoría de las actividades, sucesos e incidencias relevantes que ocurren en un Sistema Operativo. Estos registros son muy útiles para conocer qué ocurre en un ordenador y, por supuesto, poder solucionar problemas y evitar que estos se repitan en el futuro. Entre estos registros se pueden encontrar cosas como:

- Los paquetes que se han instalado o desinstalado.
- Información sobre los diferentes accesos remotos.
- Autenticaciones fallidas.
- Errores de programas o servicios.
- Bloqueos del firewall.
- Etc.

Entre los logs, que en Linux suelen estar almacenados en el directorio "/var/log", pueden encontrarse los **logs del sistema** (registran información relacionada con el SO, como, por ejemplo, de los servicios, accesos, etc.) y los **logs de programas** (almacenan información y eventos relevantes del software o programa que los crea). Entre los logs del sistema cabe destacar "auth.log" (en él se encuentran todas las acciones que involucran la autenticación), "syslog" (es el cajón de sastre donde se registran gran cantidad de logs, tanto del sistema como de algunos programas), "faillog" (contiene intentos fallidos de autenticación de los usuarios en el sistema, "lastlog" (en él se registran la fecha/hora del último inicio de sesión en el sistema de cada usuario), "boot.log" (registra información relacionada con el arranque del sistema), "daemon.log" (en él pueden encontrarse información relacionada con los diferentes procesos *Daemon*), etcétera.

Recuerda que, si necesitas ayuda con el funcionamiento o el uso de alguno de los comandos, puedes utilizar el comando "man", seguido del nombre del comando del que tengas dudas sobre su uso (por ejemplo: man cd); esto te mostrará una pequeña explicación del comando y las diferentes opciones

para utilizarlo. También puedes utilizar la hoja de comandos facilitada junto a esta guía (Practica2_CommandReference.pdf). Además, como irás aprendiendo a la vez que adquieras experiencia… ¡los buscadores son nuestros amigos!

Ejercicio 7.1

¿En qué fichero podrías encontrar los inicios de sesión que se han producido en la máquina? ¿Y los que se han hecho en remoto con SSH?

Ejercicio 7.2

Apoyándote en los comandos "tail" y "head", ¿Cuál es la última línea que se ha registrado en el fichero de log "user.log"? ¿Y la primera?

Ejercicio 7.3

Dirígete al directorio "/var/log" y lista su contenido. Ayudándote de Internet… ¿Qué significan los números que aparecen al final de los ficheros de log (por ejemplo, "debug.1", "auth.log.1", etc.)?

Ejercicio 7.4

¿Qué comando podrías utilizar para ver las primeras 15 líneas del fichero /var/log/syslog? ¿Y las últimas 20 líneas?

8. Control de acceso y contraseñas

Cabe destacar que, a pesar del falso mito que existe de que Linux o MacOS son SO más seguros que Windows, esto para nada es así. Si bien es cierto que los usuarios de Linux suelen estar más concienciados con la seguridad y que existe un menor número de equipos con este SO (lo cual lo convierte en un objetivo minoritario para los delincuentes), no se debe descuidar la seguridad cuando utilizamos sistemas operativos de este tipo.

Uno de los principales aspectos de seguridad, además de mantener el SO actualizado, es la **contraseña**. Esta debe ser robusta (difícil de descubrir para un programa y/o una persona). Para que una contraseña pueda considerarse robusta, esta debe tener, al menos, una longitud mínima de 8 caracteres; no contener nombres, apellidos, fechas de cumpleaños, nombres de usuario, empresa, etc.; no utilizar palabras del diccionario de ningún idioma; no estar formada con números y/o letras adyacentes en el teclado; contener mayúsculas, minúsculas, números y caracteres especiales; etc.

En Linux, para cambiar la contraseña de acceso para el usuario “root”, desde la terminal, deben seguirse los siguientes pasos:

- Teclear `sudo su`.
- Introducir la contraseña actual.
- Teclear `passwd root` y pulsar enter.
- Escribir la nueva clave y pulsar enter.

De igual forma, para cambiar la contraseña de cualquier otro usuario, por ejemplo, el usuario “kali”, se seguirán los siguientes pasos:

- Teclear `su kali` (de esta forma, se cambiará de usuario al usuario “kali”).
- Introducir la contraseña actual.
- Teclear `passwd kali` y pulsar enter.
- Escribir la nueva clave y pulsar enter.

La herramienta **John the Ripper** sirve para recuperar y auditar la contraseña. Este software es de código abierto y se encuentra disponible para muchos sistemas operativos, entre ellos, Linux. En la distribución “Kali Linux” se encuentra instalada por defecto. Esta herramienta, admite la recuperación y auditoría de cientos de tipos de cifrado y *hash* (entre las que pueden encontrarse contraseñas de usuario, aplicaciones web como WordPress, software colaborativo, bases de datos, claves privadas cifradas, archivos, etc.).

Para comprobar que John the Ripper está correctamente instalado en el equipo y que es capaz de realizar un test de rendimiento del hardware del sistema que disponemos (para saber la capacidad que tendrá la herramienta en su labor de descifrar una contraseña), puede utilizarse el comando `john --test`. Si no estuviera instalada, basta con ejecutar el comando `sudo apt install john` o instalarla desde su código fuente siguiendo los siguientes pasos:

- `sudo apt update && sudo apt full-upgrade -y`
- `reboot`
- `sudo apt install build-essential libssl-dev yasm libgmp-dev libpcap-dev libnss3-dev libkrb5-dev pkg-config`
- `cd /home/kali`
- `wget https://github.com/openwall/john/archive/bleeding-jumbo.zip`
- `unzip bleeding-jumbo.zip`
- `rm bleeding-jumbo.zip`
- `cd john-bleeding-jumbo/src/`
- `./configure && make`
- `cd ../run`
- `./john --test`

Recuerda que, si necesitas ayuda con el funcionamiento o el uso de alguno de los comandos, puedes utilizar el comando “man”, seguido del nombre del comando del que tengas dudas sobre su uso (por ejemplo: man cd); esto te mostrará una pequeña explicación del comando y las diferentes opciones para utilizarlo. También puedes utilizar la hoja de comandos facilitada junto a esta guía (Practica2_CommandReference.pdf). Además, como irás aprendiendo a la vez que adquieras experiencia… ¡los buscadores son nuestros amigos!

Ejercicio 8.1

Con la ayuda de Internet, crea un usuario en tu MV llamado “user1” y ponle la contraseña “user1”. ¿Qué comando o comandos has utilizado?

Ejercicio 8.2

Cambia la contraseña del usuario creado en el anterior ejercicio por una más robusta, ¿Con qué comando has modificado la contraseña?

Ejercicio 8.3

Busca el comando adecuado y borra el usuario “user1”, ¿Cómo lo has hecho?

Ejercicio 8.4

Ahora, supongamos que ha llegado a nuestras manos el fichero "/etc/passwd" y el fichero "/etc/shadow" de una máquina y vamos a analizarlo (Practica2_Material.zip).

a) **En primer lugar, debemos crear una carpeta de trabajo dentro de nuestra MV, en "/home/kali", a la que llamaremos "crack". Después, accederemos a ella desde la terminal. En esta carpeta copiaremos los ficheros "shadow" y "passwd" adjuntos a esta guía (Practica2_Material.zip).**

b) **El siguiente paso será dirigirnos al fichero "passwd" que ha llegado a nuestras manos e investigar su contenido. En este fichero, se puede encontrar información sobre quién puede acceder al sistema y qué puede hacer dentro de él. ¿Puedes encontrar algún usuario "llamativo"? Copia en el siguiente recuadro la línea correspondiente a ese o esos usuarios:**

Una línea de ejemplo del fichero "/etc/passwd" es la siguiente, donde:

1	2	3	4	5	6	7
user1:	*x:*	*1001:*	*1001:*	*:*	*/home/user1:*	*/bin/sh*
Nombre de usuario.	Contraseña (el carácter "x" indica que la contraseña cifrada de almacena en /etc/shadow).	Identificador del usuario.	Identificador del grupo al que pertenece el usuario.	Información adicional sobre los usuarios.	Directorio del usuario.	Shell del usuario.

c) Dirígete ahora al fichero "shadow" y localiza el nombre o nombres de los usuarios que te han parecido más llamativos en el ejercicio anterior. Copia en el siguiente recuadro la línea o líneas correspondientes a dicho usuario junto a su contraseña cifrada/hasheada:

Una línea de ejemplo del fichero "/etc/shadow" es la siguiente, donde:

1	2	3	4	5	6	7	8	9	10
user1:	*$6*	*$4wgbTgh2*	*$vrupCDRPVxAsmV07ptc1h5Eh7.Pcx7oRvBZ/OqJQiG69l9jOISutMG1guIJy7RrA1iOjZJy37E.d.c1HYwcap1:*	*17820:*	*0:*	*99999:*	*7:*	:	:
Nombre de usuario.	Código asociado al algoritmo de cifrado/hash.	Salt*2.	Contraseña cifrada/ hasheada.	Fecha del último cambio de contraseña (desde el 01/01/1970).	Mínimo nº de días requerido entre cambios de contraseña.	Máximo nº de días requerido entre cambios de contraseña	Nº de días antes que se avisará cuando expire la contraseña.	Cantidad de días después de que el usuario esté inactivo para deshabilitar la contraseña	Caducidad de la cuenta (desde el 01/01/1970).

2 El **salt es una combinación de bits aleatorios que se usan, junto a la propia contraseña, como las entradas en una función de derivación de claves.*

d) ¿Qué algoritmo de cifrado (elemento nº 2) se ha utilizado para cifrar la contraseña del usuario "seed"? ¿Y para el usuario "test"?

e) **Con la ayuda de Internet, ¿Qué opciones de algoritmos de cifrado existen y a qué códigos se corresponden?**

Ejercicio 8.5

Para utilizar John the Ripper, al igual que ocurre con otras herramientas, es necesario introducir los valores necesarios en el formato que la herramienta espera recibirlos.

- **Ve al directorio "/home/kali/crack", donde deberían estar contenidos los ficheros "passwd" y "shadow".**
- **Mediante la herramienta "unshadow" (preinstalada en "Kali Linux"), combina estos dos ficheros en el formato que espera recibirlos John the Ripper (sudo unshadow passwd shadow > passwords.txt).**

¿Qué contiene ahora el fichero "passwords.txt"?

Ejercicio 8.6

Una de las opciones de John the Ripper es tratar de recuperar una contraseña utilizando un diccionario de contraseñas. Puedes encontrar uno dentro del SO "Kali Linux", en la ruta "/usr/share/john/password.lst", aunque, si buscas un poco, en Internet podrás encontrar infinidad de ellos (en diferentes idiomas, con distintas combinaciones, etc.

Mediante esta opción, se pueden comparar las contraseñas contenidas en un fichero (cuyo formato se corresponde con la combinación del fichero "/etc/passwd" y "/etc/shadow" (gracias

a la herramienta "unshadow"), frente a las generadas por el software John the Ripper utilizando el diccionario que se le pasa como argumento. Es decir, para cada elemento en el diccionario, se genera un password cifrado/hasheado que se compara con el password contenido en el fichero generado por la herramienta "unshadow". Si existe una coincidencia, se sabe que el password del diccionario que se ha utilizado para generar el password en el fichero resultante de ejecutar "unshadow", es el correspondiente a ese usuario.

a) **Ayudándote de la opción --help (john --help), ¿Con qué opción podríamos indicarle a John the Ripper que lea las diferentes palabras que están contenidas en el diccionario "/usr/share/john/password.lst"?**

b) **Ejecuta el comando sudo john wordlist=/usr/share/john/password.lst /home/kali/crack/passwords.txt, ¿Se ha conseguido descifrar alguna contraseña? ¿Cuál? Puedes observar las contraseñas descifradas con la opción --show de John the Ripper, seguida del fichero que contenía dichas contraseñas.**

c) **Sobre el mismo fichero, explora la opción de utilizar John the Ripper mediante fuerza bruta, sin usar un diccionario (probando las diferentes combinaciones de caracteres). ¿Cuál es el comando que has utilizado? ¿Has conseguido extraer alguna otra contraseña?**

5. Anexo I

En este apartado se incluyen los códigos, scripts y útiles necesarios para la realización de los ejercicios de esta práctica.

passwd

```
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:/var/run/ircd:/usr/sbin/nologin
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
systemd-network:x:100:102:systemd Network Management,,,:/run/systemd:/usr/sbin/nologin
systemd-resolve:x:101:103:systemd Resolver,,,:/run/systemd:/usr/sbin/nologin
systemd-timesync:x:102:104:systemd Time Synchronization,,,:/run/systemd:/usr/sbin/nologin
messagebus:x:103:106::/nonexistent:/usr/sbin/nologin
syslog:x:104:110::/home/syslog:/usr/sbin/nologin
_apt:x:105:65534::/nonexistent:/usr/sbin/nologin
tss:x:106:111:TPM software stack,,,:/var/lib/tpm:/bin/false
uuidd:x:107:114::/run/uuidd:/usr/sbin/nologin
tcpdump:x:108:115::/nonexistent:/usr/sbin/nologin
avahi-autoipd:x:109:116:Avahi autoip daemon,,,:/var/lib/avahi-autoipd:/usr/sbin/nologin
usbmux:x:110:46:usbmux daemon,,,:/var/lib/usbmux:/usr/sbin/nologin
rtkit:x:111:117:RealtimeKit,,,:/proc:/usr/sbin/nologin
dnsmasq:x:112:65534:dnsmasq,,,:/var/lib/misc:/usr/sbin/nologin
cups-pk-helper:x:113:120:user for cups-pk-helper service,,,:/home/cups-pk-helper:/usr/sbin/nologin
speech-dispatcher:x:114:29:Speech Dispatcher,,,:/run/speech-dispatcher:/bin/false
avahi:x:115:121:Avahi mDNS daemon,,,:/var/run/avahi-daemon:/usr/sbin/nologin
kernoops:x:116:65534:Kernel Oops Tracking Daemon,,,:/:/usr/sbin/nologin
saned:x:117:123::/var/lib/saned:/usr/sbin/nologin
nm-openvpn:x:118:124:NetworkManager OpenVPN,,,:/var/lib/openvpn/chroot:/usr/sbin/nologin
hplip:x:119:7:HPLIP system user,,,:/run/hplip:/bin/false
whoopsie:x:120:125::/nonexistent:/bin/false
colord:x:121:126:colord colour management daemon,,,:/var/lib/colord:/usr/sbin/nologin
geoclue:x:122:127::/var/lib/geoclue:/usr/sbin/nologin
pulse:x:123:128:PulseAudio daemon,,,:/var/run/pulse:/usr/sbin/nologin
gnome-initial-setup:x:124:65534::/run/gnome-initial-setup/:/bin/false
gdm:x:125:130:Gnome Display Manager:/var/lib/gdm3:/bin/false
seed:x:1000:1000:SEED,,,:/home/seed:/bin/bash
```

```
systemd-coredump:x:999:999:systemd Core Dumper:/:/usr/sbin/nologin
telnetd:x:126:134::/nonexistent:/usr/sbin/nologin
ftp:x:127:135:ftp daemon,,,:/srv/ftp:/usr/sbin/nologin
sshd:x:128:65534::/run/sshd:/usr/sbin/nologin
vboxadd:x:998:1::/var/run/vboxadd:/bin/false
test:x:1002:1002:,,,:/home/test:/bin/bash
```

shadow

```
root:!:18590:0:99999:7:::
daemon:*:18474:0:99999:7:::
bin:*:18474:0:99999:7:::
sys:*:18474:0:99999:7:::
sync:*:18474:0:99999:7:::
games:*:18474:0:99999:7:::
man:*:18474:0:99999:7:::
lp:*:18474:0:99999:7:::
mail:*:18474:0:99999:7:::
news:*:18474:0:99999:7:::
uucp:*:18474:0:99999:7:::
proxy:*:18474:0:99999:7:::
www-data:*:18474:0:99999:7:::
backup:*:18474:0:99999:7:::
list:*:18474:0:99999:7:::
irc:*:18474:0:99999:7:::
gnats:*:18474:0:99999:7:::
nobody:*:18474:0:99999:7:::
systemd-network:*:18474:0:99999:7:::
systemd-resolve:*:18474:0:99999:7:::
systemd-timesync:*:18474:0:99999:7:::
messagebus:*:18474:0:99999:7:::
syslog:*:18474:0:99999:7:::
_apt:*:18474:0:99999:7:::
tss:*:18474:0:99999:7:::
uuidd:*:18474:0:99999:7:::
tcpdump:*:18474:0:99999:7:::
avahi-autoipd:*:18474:0:99999:7:::
usbmux:*:18474:0:99999:7:::
rtkit:*:18474:0:99999:7:::
dnsmasq:*:18474:0:99999:7:::
cups-pk-helper:*:18474:0:99999:7:::
speech-dispatcher:!:18474:0:99999:7:::
avahi:*:18474:0:99999:7:::
kernoops:*:18474:0:99999:7:::
saned:*:18474:0:99999:7:::
nm-openvpn:*:18474:0:99999:7:::
hplip:*:18474:0:99999:7:::
whoopsie:*:18474:0:99999:7:::
colord:*:18474:0:99999:7:::
geoclue:*:18474:0:99999:7:::
pulse:*:18474:0:99999:7:::
gnome-initial-setup:*:18474:0:99999:7:::
gdm:*:18474:0:99999:7:::
seed:$6$n8DimvsbIgU0OxbD$YZ0h1EAS4bGKeUIMQvRhhYFvkrmMQZdr/hB.Ofe3KFZQTgFTcRgoIoKZdO0rhDRxxaITL4b/scpdbTfk/nwFd
0:18590:0:99999:7:::
systemd-coredump:!!:18590::::::
telnetd:*:18590:0:99999:7:::
```

```
ftp:*:18590:0:99999:7:::
sshd:*:18590:0:99999:7:::
vboxadd:!:18786::::::
test:$6$IMtT.qDUmzrXfde6$dbHyvoT9GCVJEiST4AwSIHdoLsPMtbiPdX6j.spkDoAF8rXQhpO5q1f5uFVmTdsQiGtvag2VkKn9DTVHs4jMu
1:18834:0:99999:7:::
```

CASO PRÁCTICO 1 – REDACCIÓN DE UNA POLÍTICA DE SEGURIDAD

CONTEXTO

Imagina que redactas estas políticas desde el equipo de seguridad de la universidad, así que por lo menos tienes que distinguir tres roles: estudiantes, personal docente e investigador (PDI) y personal de administración y servicios (PAS).

Supón que parte de las infraestructuras y sistemas de información están en un centro de datos local (la plataforma de Aula Virtual, MyApps y el directorio de usuarios) y que lo demás se consume a proveedores en la nube (a través de O365: correo electrónico, Teams u otros como sistemas para matriculación, nóminas, etc.). Además, la universidad proporciona portátiles al PAS y permite al profesorado y al PAS trabajar con sus propios teléfonos móviles ya que no se les proporciona ninguno corporativo.

FORMA DE TRABAJAR

En equipos de cuatro personas (dos parejas de prácticas, por ejemplo), yo os asigno la política que tenéis que redactar y entregarme.

Alternativas para realizar el caso

P.1.1 Formación y concienciación

Esta política necesita establecer, como mínimo:

Estándares

a. Objetivos del programa de formación y concienciación por rol/responsabilidad y áreas de contenido mínimas por rol/responsabilidad que deben tratarse (matriz de formación y concienciación).
b. Necesidades adicionales de formación y concienciación para ciertos roles específicos.
c. Involucración esperada del personal en las actividades de formación y concienciación (son obligatorias, optativas, hay diferentes oportunidades para realizarlas, etc.); recompensas y política de reconocimiento o sanciones.
d. Involucración esperada de los empleados subcontratados y de otros socios y terceras partes en actividades de formación y concienciación específicas.
e. Temporización recomendada para ciertas actividades (por ejemplo, cuando se contrata a un nuevo empleado, cuando se le promociona, cuando se adquiere un nuevo recurso, cuando se modifica una política importante del modelo de control, etc.).
f. Instrucciones detallas sobre cómo proceder para obtener consejo en materia de ciberseguridad o para notificar necesidades de formación y concienciación específicas que se detecten.

P.1.2 Gestión de recursos humanos y seguridad del personal

Esta política necesita establecer, como mínimo:

Estándares

a. Condiciones de no divulgación para cada nivel de riesgo del personal.
b. Acceptable Use Policy (AUP) o reglas de comportamiento para cada nivel de riesgo del personal (reglas que describen las responsabilidades de los individuos y el comportamiento que se espera de ellos con respecto a los activos y datos de la organización).
c. Lista de conflictos de interés identificados.
d. Si los procesos no están automatizados, pasos necesarios para notificar a las instancias interesadas la terminación de contratos, trasferencia o re-asignación del personal y para devolver llaves, tokens, medios de almacenamiento, terminales móviles, etc.

P.1.3 Protección del entorno físico y ambiental

Esta política debe establecer, como mínimo:

Estándares

a. Lista de roles y responsabilidades que deben ocuparse para poder acceder a los diferentes activos físicos (recursos)
b. Pasos necesarios para solicitar, de manera individual, acceso a activos físicos de manera temporal o permanente.
c. Instrucciones detalladas acerca de cómo y cuándo traspasar los perímetros físicos.
d. Pasos necesarios para entregar, mover y retirar activos físicos de la infraestructura cuando estos procesos requieren traspasar perímetros de seguridad. Será necesario un conjunto específico de instrucciones para los terminales móviles.
e. Pasos necesarios para transportar, manejar y borrar medios de almacenamiento.
f. Instrucciones detalladas acerca de cómo obtener, guardar y gestionar (para renovar o cambiar periódicamente si es necesario) llaves, contraseñas, combinaciones o cualquier otro tipo de credencial física.
g. Pasos necesarios para notificar la pérdida/compromiso/daño de llaves, contraseñas, combinaciones o cualquier otro tipo de credencial física.
h. Instrucciones detalladas acerca de cómo comportarse cuando se activan alarmas relacionadas con la seguridad física.
i. Instrucciones detalladas acerca de cuándo y cómo personas externas y visitantes pueden obtener acceso físico a los diferentes recursos y cuándo y cómo deben ser escoltados para realizar sus labores.
j. Instrucciones detalladas acerca de cómo los diferentes recursos deben estar protegidos ambientalmente (temperatura, humedad, vibraciones, polvo, etc.).

P.1.4 IAAA y control de accesos

Esta política debería establecer, como mínimo:

Estándares

a. Instrucciones detalladas acerca de cuándo y cómo bloquear y cerrar sesión en diferentes activos.
b. Pasos necesarios para establecer y mantener una contraseña fuerte.
c. Instrucciones detalladas acerca de cuándo es necesario establecer y mantener contraseñas diferentes para entornos seguros e inseguros.
d. Pasos necesarios para obtener, mantener, refrescar, etc. otros tipos de autenticadores.
e. Prácticas específicas para acceder a activos utilizando terminales móviles (criptografía, conexión de red, etc.).
f. Pasos necesarios para notificar contraseñas y/o autenticadores perdidos, comprometidos o dañados.
g. Pasos necesarios para solicitar modificaciones en cuentas, usuarios, roles, privilegios, permisos y capacidades.

P.1.5 Protección de datos y seguridad de la información

Esta política debe establecer, como mínimo:

Estándares

a. Categorías de datos.
b. Lista de roles y responsabilidades que pueden acceder a las diferentes categorías de datos
c. Pasos necesarios para solicitar, individualmente, permisos adicionales tanto temporales como permanentes.
d. Instrucciones detalladas acerca de cuándo, dónde y cómo deben cifrarse/descifrarse los datos.
e. Cuando se requiere intervención del usuario (idealmente, esto debería ser automático), instrucciones detalladas sobre los mecanismos criptográficos que se aceptan dependiendo de la categoría de los datos: función hash, algoritmo, longitud de clave etc.
f. Prácticas específicas de “mesa limpia”.
g. Prácticas específicas para evitar que se espíe a los empleados por encima del hombro (recomendaciones para la ubicación de los monitores y teclados, para utilizar terminales móviles, etc.).
h. Prácticas específicas para destruir documentos impresos.
i. Medios y servicios de almacenamiento autorizados/prohibidos y prácticas específicas para almacenar datos en ellos, para transportarlos y para utilizarlos.
j. Cuando se requiera la intervención del usuario (idealmente, esto debería esta automatizado), instrucciones detalladas para evitar o eliminar metadatos de distintos tipos de documentos, imágenes y ficheros.

P.1.6 Respuesta ante incidentes

Esta política necesita establecer:

Estándares

a. Actividades de respuesta a incidentes por rol/responsabilidad y por tipo de incidente.
b. Formación en respuesta a incidentes necesaria para cada rol/responsabilidad.
c. Temporización recomendada (o plazos) u obligatoria para ciertas actividades específicas.
d. Procedimientos para operar ciertos activos específicos en modelo manual, con todas sus conexiones con el exterior interrumpidas, hasta que se puedan recuperar condiciones de operación seguras en todos los aspectos.

e. Instrucciones detalladas acerca de cómo proceder para obtener consejo acerca de la respuesta a incidentes o para notificar aspectos relacionados con incidentes (que estén ocurriendo en el presente o ya pasados).
f. Procedimiento de comunicación y listado de personal con el que contactar en el caso de un incidente incluyendo fabricantes, administradores, personal de soporte, etc.

P.1.7 Bring Your Own Device (BYOD)

Esta política necesita establecer:

Estándares

a. Formas de solicitar el uso de BYOD, de ponerlo en marcha y de informar cuándo un dispositivo se ha perdido, robado o ha sido comprometido.
b. Lista de escenarios en los que el dispositivo del usuario podría ser borrado de manera remota.
c. Soporte por parte del departamento TI.
d. Lista de aplicaciones obligatorias: navegador, gestor de email, mensajería instantánea, anti-malware, firewall personal, etc.
e. Lista de aplicaciones y prácticas prohibidas (por ejemplo, jailbreak o root).
f. Lista de app stores permitidas.
g. Instrucciones para configurar las aplicaciones de manera segura.
h. Instrucciones para limitar el acceso al dispositivo (PIN, biometría, etc.).
i. Instrucciones para bloquear el dispositivo tras un tiempo de inactividad.
j. Instrucciones de conectividad (configuración de WiFi, Bluetooth).
k. Instrucciones sobre actualización de SO y aplicaciones.
l. Lista de formas de acceso permitidas a los datos de la organización (cifrado, almacenamiento).

Introducción a la Ciberseguridad

Práctica 3.1: Redes e Internet

Marta Beltrán Pardo
Miguel Calvo Matalobos
Agradecimientos (versiones anteriores): Isaac Martín de Diego y Alberto Fernández Isabel

Contenidos

1. Introducción

En la tercera práctica de la asignatura vamos a realizar varias tareas relacionadas con las Redes e Internet. Esta práctica, se divide en dos partes (Practica3.1_Guion.pdf y Practica3.2_Guion.pdf):

- En la primera parte de la práctica (Practica3.1_Guion.pdf), programaremos un escáner de puertos en C. Posteriormente montaremos una máquina virtual con LAMP (Linux + Apache + MySQL + PHP) para probar nuestro escáner de puertos contra un servidor controlado.

- En la segunda parte (Practica3.2_Guion.pdf), montaremos nuestra primera página web con HTML y PHP para añadir una zona privada empleando una base de datos MySQL. El objetivo de esta segunda parte será entender la diferencia entre código estático y dinámico. Además, probaremos nuestra primera inyección SQL para poder acceder a la zona privada sin tener cuenta de usuario.

2. Material de la práctica

A continuación, se exponen los archivos necesarios para la realización de esta práctica, que pueden descargarse desde el Aula Virtual:

- **Practica3.1_Guion.pdf**: este documento. Se corresponde con el guion de la primera parte de la práctica y en él están contenidas todas las explicaciones necesarias para su realización.
- **Practica3.1_Myportscanner.c**: pequeño esquema o estructura del escáner de puertos para que no haya que partir de cero en su programación durante la práctica.

Además, será necesario descargar el programa **VirtualBox** en su última versión. Este programa, servirá para crear la máquina virtual que se utilizará a lo largo del desarrollo de esta práctica. Puede descargarse desde el siguiente enlace. También será necesario el pack de extensiones “Oracle VM VirtualBox”, que permitirá la configuración y el uso de ciertos parámetros y características en las máquinas virtuales (puede descargarse aquí).

Para el desarrollo de esta práctica, hará falta también la descarga de una distribución de un sistema operativo Linux. Puede utilizarse la máquina virtual “Kali Linux” utilizada en prácticas anteriores o una máquina virtual creada a partir de una ISO de “Kali Linux”, “Ubuntu” o cualquier otra distribución de Linux que conozcas. Este sistema operativo será el utilizado a lo largo de esta práctica.

3. Normativa y evaluación

En este apartado se detalla el formato de entrega de la práctica y la forma en la que se evaluará la misma:

- El porcentaje de la nota final de la asignatura al que corresponde esta práctica puede consultarse en la Guía docente de la propia asignatura.
- La práctica deberá realizarse, de forma obligatoria, en grupos de dos personas. Para la asignación de los grupos se deberán seguir las indicaciones del profesor.
- Cada grupo deberá:
    - Realizar una única memoria (puede utilizarse esta misma guía como plantilla) en la que responda las preguntas planteadas y/o en la que se exponga, de forma argumentada, las decisiones tomadas para la realización de la práctica.
    - Desarrollar y/o modificar tantos archivos de código como se soliciten en los diferentes ejercicios que forman la práctica.
- El resultado de la realización de la práctica consistirá en un fichero .zip llamado **Practica3.zip** que deberá entregarse a través del Aula Virtual en el espacio habilitado para ello y en la fecha límite allí expuesta. Este fichero debe contener:
    - La memoria completa de la práctica 3 en formato PDF. En ella debe indicarse, en la primera página y de forma clara, el nombre y apellidos de los integrantes del grupo.
    - Los archivos de código resultantes de la realización de los ejercicios solicitados, de momento (habrá más asociados a la segunda parte de la práctica, se indicará en el siguiente guion):
        - Practica3_myportscanner_resuelto.c

    Recuerda lo que has aprendido con la práctica 1 acerca de la redacción de memorias de prácticas.

4. Enunciado de la práctica

En este apartado se describirán las distintas actividades, programas y ejercicios que deberá realizar cada grupo de prácticas, así como la información a completar y/o rellenar en la memoria.

4.1 Programar un escáner de puertos

En la primera parte de la práctica, vamos a programar un sencillo escáner de puertos en el lenguaje de programación C.

Ejercicio 1

Un **puerto** identifica a una aplicación que recurre a la capa de transporte para conectarse con la red. Por ejemplo, los programas de correo electrónico POP3, como Outlook, envían y reciben correos electrónicos a través de puertos específicos, 110 y 995 para recibir correos electrónicos, 25, 2525 y 443 para enviar correos electrónicos, y puertos 143 y 993 para conectarse a servidores IMAP. El modelo OSI (*Open System Interconnection*) es el encargado de definir estos puertos y su uso. Los números de puerto se indican mediante 16 bits (por tanto, existen 2^{16} = 65536 puertos). La entidad IANA (*Internet Assigned Numbers Authority*) es la encargada de asignar a cada puerto su número específico. En la Tabla 1 puede encontrarse una pequeña muestra de puertos utilizados habitualmente.

<u>Pregunta 1.1</u>. Empleando como referencia al organismo IANA, averigua el nombre del puerto 993, su protocolo de transporte, su descripción y la fecha de la última modificación en su definición.

Nombre	Puerto	Comentario
ftp	20	*FTP - data*
frp	21	*FTP – control*
ssh	22	*SSH Remote Login Protocol*
telnet	23	*TELNET*
smtp	25	*Simple Mail Transfer Protocol*
domain	53	*Domain Name Server*
bootps	67	*Bootstrap Protocol – server*
bootpc	68	*Bootstrap Protocol – client*
tftp	69	*Trivial File Transfer*
http	80	*World Wide Web*
pop3	110	*Post Office Protocol – version 3*
sunprc	111	*SUN Remote Procedure Call*
netbios-ssn	139	*NETBIOS Session Service (SMB)*
imap	143	*Internet Message Access Protocol*
snmp	161	*Simple Network Management Protocol*
bgp	179	*Border Gateway Protocol*
irc	194	*Internet Relay Chat Protocol*
ldap	389	*Lightweight Directory Access Protocol*
https	443	*Http secure*
ipp	631	*Internet Printing Protocol*
wins	1512	*Windows Internet Name Service*
nfsd	2049	*NFS server*
squid	3128	*Squid Web Proxy*
mysql	3306	*MySql*

Tabla 1. Puertos utilizados habitualmente.

Ejercicio 2

Por otro lado, un **escáner de puertos** es un programa que sirve para examinar los puertos de un equipo o servidor red en busca de uno de los tres estados posibles: abierto, cerrado o filtrado. Estas herramientas son muy útiles para los administradores en relación con el diagnóstico de problemas de red y la conectividad, sin embargo, también suelen utilizarse por los atacantes para, por ejemplo, detectar posibles puntos de acceso a una máquina y/o identificar que dispositivos se están conectados y a la escucha en una red (como firewalls, proxis, servidores VPN, etc.).

Como ejemplo de escáner de puertos, podemos encontrar el software "**nmap**". Esta herramienta, disponible tanto para Linux como para Windows, es una de las más conocidas y utilizadas tanto por administradores de sistemas como por atacantes. Será necesaria su instalación para la realización de este ejercicio.

- Windows: Descargar desde este enlace.
- Linux: Instalar mediante el comando sudo apt-get install nmap.

Un ejemplo de uso de esta herramienta es mostrar los puertos abiertos en local (mediante el comando nmap localhost). En la *Figura 1* puede observarse que el único puerto abierto en local (o lo que es lo mismo, que no necesariamente sale a Internet o es visible desde Internet), es el 631 (correspondiente al IIP).

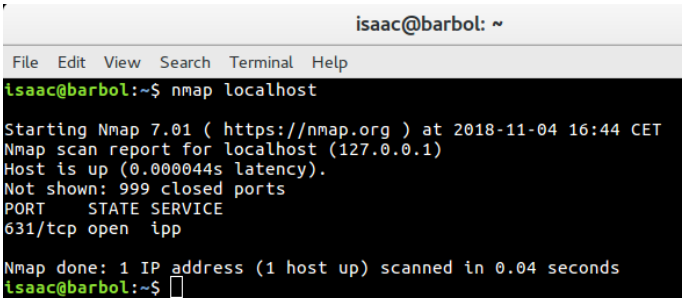


Figura 1. Mostrando los puertos locales con nmap.

Pregunta 2.1. ¿Qué es el localhost? ¿A qué dirección IP corresponde este nombre? ¿Qué es un *loopback*?

Ejercicio 3

Además de ver los puertos locales de nuestra máquina, también pueden observarse los que están abiertos desde Internet. Para ello, también puede utilizarse la herramienta “nmap”, pero esta vez con la dirección IP asignada a nuestro equipo.

Existen diferentes formas de averiguar la dirección IP, por ejemplo, en Linux, mediante “ifconfig” (esta herramienta no está instalada por defecto en todas las distribuciones Linux. Para instalarla, puede utilizarse el comando sudo apt-get install net-tools.

Lo más probable es que tengas asignada una dirección privada dentro de tu red doméstica o del aula (compruébalo con el comando ifconfig). Tras averiguar la dirección IP asignada a tu máquina, prueba a realizar un escaneo con “nmap” (nmap IP-de-tu-maquina). Puedes escanear también la dirección privada de tu router y la pública, a ver qué diferencias observas.

Cuidado, no lances escaneos de puertos indiscriminadamente contra direcciones IP reales públicas. Ese proceso puede tardar bastante tiempo en completarse y generará un tráfico considerable en la red. Además, según la legislación del país, el escaneo de puertos puede ser **ilegal**, ya que es posible que degrade el rendimiento de la máquina escaneada.

Pregunta 3.1. Menciona alguna página web que puedas consultar para averiguar y geolocalizar la dirección IP pública de tu router doméstico. Saca conclusiones acerca de los resultados de tus escaneos, tanto del ordenador como del router.

Ejercicio 4

El escaneo de puertos puede darte mucha información (puertos abiertos, protegidos tras un firewall, sistemas operativos instalados, servicios corriendo, etc.), “nmap” es una herramienta muy potente y con muchas opciones.

SSH es un acrónimo de *Secure Shell*. Este protocolo de red permite que dos ordenadores se comuniquen de forma segura y puedan compartir datos, información o incluso realizar operaciones desde una máquina en otra. Una característica inherente de SSH es que la comunicación está cifrada. Para realizar estas acciones, se necesita un cliente (cliente SSH; por ejemplo, instalado en nuestro equipo), que será el encargado de conectarse al servicio (servidor SSH) y, de esta forma transferir los datos hacia/desde un equipo a otro utilizando una interfaz gráfica o la línea de comandos (dependiendo del cliente SSH elegido).

En tu máquina virtual con SO Linux, instala, desde la línea de comandos, un servidor SSH. Para ello, desde la terminal, ejecuta los siguientes comandos:

- sudo apt-get install openssh-server
- sudo systemctl enable ssh
- sudo systemctl start ssh

Pregunta 4.1. Con ayuda del manual de nmap (man nmap), investiga que comando o comandos necesitarías para extraer la versión utilizada en el servicio SSH de tu máquina virtual. ¿Qué comando te ha servido para realizar esta tarea? ¿Qué información, aparte de la versión del servicio SSH, has conseguido extraer?

Ejercicio 5

Antes de comenzar a realizar este ejercicio, es necesario revisar el código proporcionado junto al enunciado (“Practica3_myportscanner.c”). Se trata de que te familiarices con el código y comprendas como se ejecutan las diferentes instrucciones, que finalidad tiene el programa o su estructura, etc.

Para programar el escáner de puertos, se proporciona, junto a este documento, un esquema del código en C que debe completarse (“**Practica3_myportscanner.c”**). Este código NO COMPILA; en él, existen 10 huecos que deben rellenarse (indicados por XXXi, donde “i” es un indicador de 1 hasta 10). No es necesario utilizar el esquema proporcionado si no se quiere, puedes programar tu propio escáner desde cero, siempre y cuando lo justifiques y expliques convenientemente.

Pregunta 5.1. Podemos decir, de modo muy simple, que un socket es una manera de hablar con otro computador usando descriptores de ficheros estándar en Linux. Para inicializar un socket se emplea el siguiente formato:

```
int sockfd = socket(int familia, int tipo, int protocolo)
```

¿Qué significado tienen los valores de entrada de la función socket “familia”, “tipo” y “protocolo”? Identifica esta llamada en el código proporcionado. Explica los valores concretos que se emplean en el código. ¿Qué significado tiene cuando la función devuelve un -1? ¿Qué significa si devuelve un valor distinto de -1?

Pregunta 5.2. Completa el código para hacer funcional tu escáner de puertos. Una vez completado el código en C, compílalo (gcc Practica3_myportscanner.c -o myportscanner), pruébalo (./myportscanner) y comenta los resultados. Compara con los que obtuviste con nmap (obviamente nuestro escáner es mucho más sencillo, pero por lo menos compara que detecta los mismos estados para los puertos o que si encuentras diferencias las puedes explicar).

4.2 Montar una máquina virtual con LAMP

LAMP (Linux, Apache, MySQL, PHP) es un conjunto de software de código abierto que se instala normalmente en conjunto para habilitar un servidor para alojar sitios y aplicaciones web dinámicas. Esta plataforma, utiliza el sistema operativo Linux como base. En él, se despliega Apache como servidor web, MySQL como gestor de base de datos relacionales y PHP como lenguaje de programación. Existen otras pilas similares como WAMP (Windows, Apache, MySQL, PHP), MAMP (Mac OS, Apache, MySQL, PHP) o XAMPP (Windows/Linux/Solaris/MacOS, Apache, MySQL, PHP, PERL).

Como ya se montó una máquina virtual con "Kali Linux" en prácticas anteriores, puede utilizarse esta máquina para el despliegue de LAMP. Otra opción es instalar una nueva máquina virtual con sistema operativo Linux (creada a partir de una ISO de "Kali Linux", "Ubuntu" o cualquier otra distribución Linux que conozcas), siendo esta nueva máquina uno de nuestros servidores vulnerables desde ahora (independiente de la máquina "Kali Linux"). Una vez instalado y configurado LAMP, esta parte de la práctica consiste en probar el escáner puertos, previamente programado ("Programar un escáner de puertos"), desde la máquina virtual con "Kali Linux".

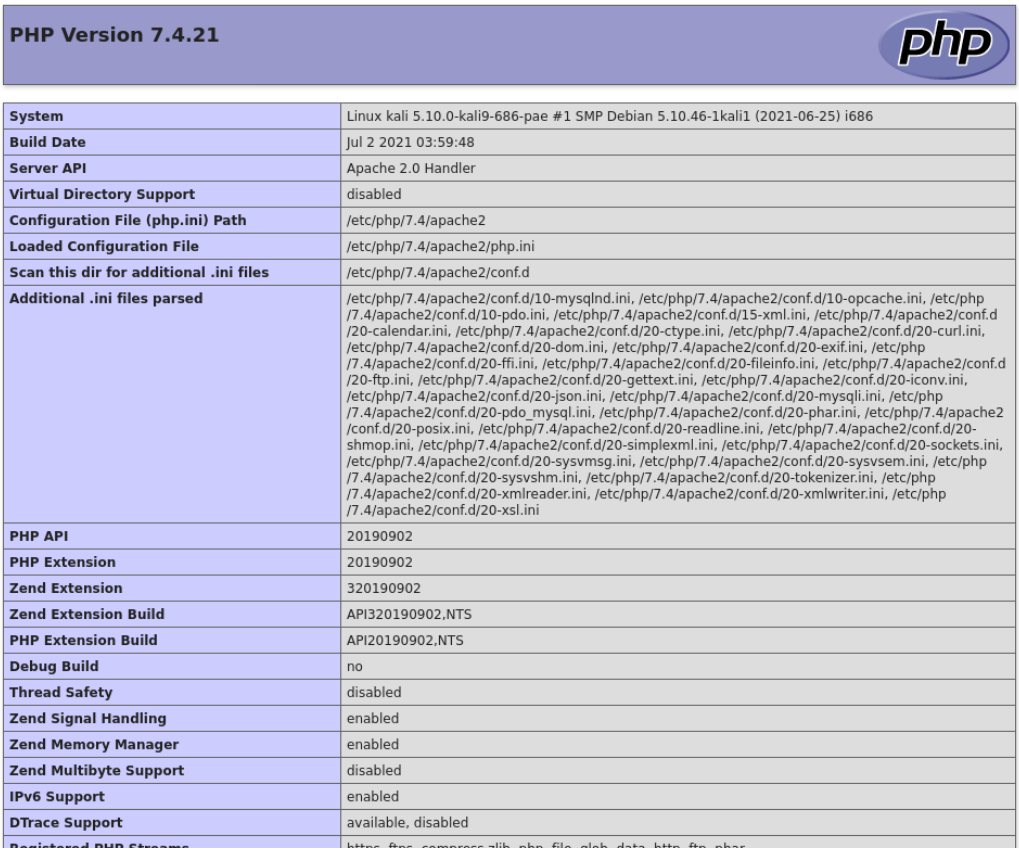
NOTA: Para habilitar y configurar Internet en "Kali Linux", una vez instada la máquina virtual, debemos dirigirnos a la pestaña de Configuración en VirtualBox. Una vez allí, elegimos el apartado "Red", donde se seleccionará la opción "Conectad a adaptador puente" y se elegirá el adaptador de red que haya conectado a Internet (por ejemplo, el adaptador Wifi). Desplegando la opción "Avanzadas", se debe elegir "permitir todo en modo promiscuo". Tras aceptar, se debe iniciar "Kali Linux" y probar la nueva configuración (por ejemplo, comprobando que puede navegarse por la web).

Ejercicio 6

En este ejercicio, debes crear un servidor LAMP en una máquina virtual con Linux. Para ello, sigue los siguientes pasos:

- Actualiza los paquetes y librerías instalados en tu máquina Linux:
    - sudo apt-get update
    - sudo apt-get upgrade
- Instala Apache2 e inícialo:
    - sudo apt-get install apache2
    - sudo /etc/init.d/apache2 start ó sudo service apache2 start (para pararlo se puede utilizar la opción “stop” y para reiniciarlo, “restart”).
    - sudo /etc/init.d/apache2 status ó sudo service apache2 status (para comprobar el estado del servicio).
- Instala MySQL o MariaDB e inícialo:
    - **NOTA**: Dependiendo del SO en el que estés haciendo la instalación del motor de base de datos, puede que no sea MySQL sino MariaDB. Todo es muy similar, busca un manual específico si te surgen dudas con la sintaxis. Por lo tanto, si no funciona la opción de MySQL, prueba con MariaDB.
    - sudo apt-get install mysql-server ó sudo apt-get install mariadb-server
    - sudo /etc/init.d/mysql start ó sudo service mysql start (para pararlo se puede utilizar la opción “stop” y para reiniciarlo, “restart”).
    - mysql_secure_installation (Enter + n + y + Contraseña que queramos).

- Instala PHP, algunas utilidades e inícialo:
    - sudo apt-get install php libapache2-mod-php php-mysql php-cgi php-curl
    - sudo /etc/init.d/apache2 restart ó sudo service apache2 restart (se reinicia el servicio “Apache2” para que la configuración incluya la instalación de PHP).
- Comprueba si está funcionando:
    - cd /var/www/html
    - nano info.php
        - Escribir en el fichero: *<?php phpinfo(); ?>*
        - Guardar: Ctrl+o (+ Enter).
        - Salir: Ctrl+x (+ Enter).
    - Abre el navegador y visita la página http://localhost/info.php. Si ves algo similar a lo que se muestra en la Figura 2, has tenido éxito en la instalación de LAMP.



PHP Version 7.4.21

System	Linux kali 5.10.0-kali9-686-pae #1 SMP Debian 5.10.46-1kali1 (2021-06-25) i686
Build Date	Jul 2 2021 03:59:48
Server API	Apache 2.0 Handler
Virtual Directory Support	disabled
Configuration File (php.ini) Path	/etc/php/7.4/apache2
Loaded Configuration File	/etc/php/7.4/apache2/php.ini
Scan this dir for additional .ini files	/etc/php/7.4/apache2/conf.d
Additional .ini files parsed	/etc/php/7.4/apache2/conf.d/10-mysqlnd.ini, /etc/php/7.4/apache2/conf.d/10-opcache.ini, /etc/php/7.4/apache2/conf.d/10-pdo.ini, /etc/php/7.4/apache2/conf.d/15-xml.ini, /etc/php/7.4/apache2/conf.d/20-calendar.ini, /etc/php/7.4/apache2/conf.d/20-ctype.ini, /etc/php/7.4/apache2/conf.d/20-curl.ini, /etc/php/7.4/apache2/conf.d/20-dom.ini, /etc/php/7.4/apache2/conf.d/20-exif.ini, /etc/php/7.4/apache2/conf.d/20-ffi.ini, /etc/php/7.4/apache2/conf.d/20-fileinfo.ini, /etc/php/7.4/apache2/conf.d/20-ftp.ini, /etc/php/7.4/apache2/conf.d/20-gettext.ini, /etc/php/7.4/apache2/conf.d/20-iconv.ini, /etc/php/7.4/apache2/conf.d/20-json.ini, /etc/php/7.4/apache2/conf.d/20-mysqli.ini, /etc/php/7.4/apache2/conf.d/20-pdo_mysql.ini, /etc/php/7.4/apache2/conf.d/20-phar.ini, /etc/php/7.4/apache2/conf.d/20-posix.ini, /etc/php/7.4/apache2/conf.d/20-readline.ini, /etc/php/7.4/apache2/conf.d/20-shmop.ini, /etc/php/7.4/apache2/conf.d/20-simplexml.ini, /etc/php/7.4/apache2/conf.d/20-sockets.ini, /etc/php/7.4/apache2/conf.d/20-sysvmsg.ini, /etc/php/7.4/apache2/conf.d/20-sysvsem.ini, /etc/php/7.4/apache2/conf.d/20-sysvshm.ini, /etc/php/7.4/apache2/conf.d/20-tokenizer.ini, /etc/php/7.4/apache2/conf.d/20-xmlreader.ini, /etc/php/7.4/apache2/conf.d/20-xmlwriter.ini, /etc/php/7.4/apache2/conf.d/20-xsl.ini
PHP API	20190902
PHP Extension	20190902
Zend Extension	320190902
Zend Extension Build	API320190902,NTS
PHP Extension Build	API20190902,NTS
Debug Build	no
Thread Safety	disabled
Zend Signal Handling	enabled
Zend Memory Manager	enabled
Zend Multibyte Support	disabled
IPv6 Support	enabled
DTrace Support	available, disabled
Registered PHP Streams	https, ftps, compress.zlib, php, file, glob, data, http, ftp, phar

Figura 2. Comprobación de la instalación de LAMP.

RECUERDA ARRANCAR LOS SERVICIOS (Apache2 y MySQL) CADA VEZ QUE INICIES/REINICIES LA MÁQUINA.

Pregunta 6.1. Una vez instalado y configurado LAMP, muestra, mediante un pantallazo, el resultado de acceder a la página http://localhost/info.php.

Ejercicio 7

Tras finalizar con la instalación y configuración de LAMP, debes crear una base de datos (BBDD) sencilla y un usuario de MySQL con las características que se muestran en la Tabla 2.

Nombre de la BBDD	*accesos*
Nombre de la tabla	*user_pass*
Campos de la tabla	*user; password*
Usuario	*app*
Permisos del usuario	*ALL PRIVILEGES (sobre la tabla "user_pass" de la BBDD "accesos")*

Tabla 2. Características solicitadas para la base de datos.

AYUDA:

- Para acceder a MySql: `mysql -u root -p`
- Para crear una base de datos: `CREATE DATABASE nombre_de_la_bbdd;`
- Para usar una base de datos: `USE nombre_de_la_bbdd;`
- Para crear una tabla en la base de datos: `CREATE TABLE nombre_de_la_tabla (campo1 tipo_campo1, campo2 tipo_campo2, campo3 tipo_campo3, …);`
- Para crear un nuevo usuario: `CREATE USER 'nombre-usuario'@'localhost' IDENTIFIED BY 'password-usuario';`
- Para dar permisos a un usuario: `GRANT TIPO-DE-PERMISO ON nombreBBDD.nombreTabla TO 'nombre-usuario'@'localhost';`
- Para aplicar los cambios de permisos de un usuario: `FLUSH PRIVILEGES;`
- Para salir de MySQL: exit

<u>Pregunta 7.1</u>. Una vez creada la base de datos y la tabla con los campos requeridos (véase la Tabla 2), inserta mediante comandos (con "INSERT INTO"), dos usuarios con sus correspondientes contraseñas. ¿Qué comando o comandos has utilizado?

5. Referencias de interés

How to Install Ubuntu 20.04 on VirtualBox - https://linuxhint.com/install_ubuntu_virtualbox_2004/

Cómo instalar la pila Linux, Apache, MySQL y PHP (LAMP) en Ubuntu 20.04 - https://www.digitalocean.com/community/tutorials/how-to-install-linux-apache-mysql-php-lamp-stack-on-ubuntu-20-04-es

Instalar LAMPP *Stack* en Ubuntu 20.04 - https://www.librebyte.net/series/instalar-lampp-stack-en-ubuntu-20-04/

6. Anexo I

En este apartado se incluyen los códigos, scripts y útiles necesarios para la realización de los ejercicios de esta práctica.

Practica3.1_myportscanner.c

```
#include "stdio.h"
#include "sys/socket.h"
#include "errno.h"
#include "netdb.h"
#include "string.h"
#include "stdlib.h"

 // Cuidado, está diseñado para Linux. Si se quiere compilar para ejecutar en sistemas Windows, las librerías
y funciones de sockets cambian un poco.
int main(int argc , char **argv)
{
    struct hostent *host;
    int err, port , connection ,start , end;
    char hostname[100];
    struct sockaddr_in sa;

    //Host que se desea escanear
    printf("Introduce hostname o dirección IP : ");
    scanf("%s", &XXX1);

    //Puerto de inicio para el escaneo
    printf("\nPuerto de inicio: ");
    scanf("%d" , XXX2);

    //Puerto final para el escaneo
    XXX3
    XXX4

    //Inicializar la estructura sockaddr_in
    strncpy((char*)&sa , "" , sizeof sa);
    sa.sin_family = XXX5;

    //Dirección IP
    if(isdigit(hostname[0]))
    {
        printf("Doing inet_addr...");
        sa.sin_addr.s_addr = inet_addr(hostname);
        printf("Hecho\n");
    }
    //Resolución de nombre a IP
    else if( (host = gethostbyname(hostname)) != 0)
    {
        printf("Doing gethostbyname...");
        strncpy((char*)&sa.sin_addr , (char*)host -> h_addr , sizeof sa.sin_addr);
        printf("Hecho\n");
    }
```

```
    else
    {
        herror(hostname);
        exit(2);
    }

    //Bucle de escaneo de puertos
    printf("Comienza el escaneo de puertos... \n");
    for(XXX6)
    {
        //Convertir el identificador de puerto
        sa.sin_port = htons(port);
        //Crear el socket
        connection = XXX7(AF_INET , SOCK_STREAM , 0);

        //Comprobar si el socket se ha creado correctamente
        if(connection < 0)
        {
            perror("\nSocket");
            exit(1);
        }
        //Conectarse al socket creado
        err = connect(XXX8 , (struct sockaddr*)&sa , sizeof sa);

        //Si no se consigue realizar la conexión
        if( err < XXX9 )
        {
            //printf("%s %-5d %s\r" , hostname , i, strerror(errno));
            fflush(stdout);
        }
        //Si sí se consigue realizar la conexión
        else
        {
            printf("%-5d open\n",  port);
        }
                //Cerramos el socket
        close(XXX10);
    }

    printf("\r");
    fflush(stdout);
    return(0);
}
```

Introducción a la Ciberseguridad

Práctica 3.2: Redes e Internet

Marta Beltrán Pardo
Miguel Calvo Matalobos
Agradecimientos (versiones anteriores): Isaac Martín de Diego y Alberto Fernández Isabel

Contenidos

1. Introducción

En la tercera práctica de la asignatura vamos a realizar varias tareas relacionadas con las Redes e Internet. Esta práctica, se divide en dos partes (Practica3.1_Guion.pdf y Practica3.2_Guion.pdf):

- En la primera parte de la práctica (Practica3.1_Guion.pdf), programaremos un escáner de puertos en C. Posteriormente montaremos una máquina virtual con LAMP (Linux + Apache + MySQL + PHP) para probar nuestro escáner de puertos.

- En la segunda parte (Practica3.2_Guion.pdf), montaremos nuestra primera página web con HTML y PHP para añadir una zona privada empleando una base de datos MySQL. El objetivo de esta segunda parte será entender la diferencia entre código estático y dinámico. Además, probaremos nuestra primera inyección SQL para poder acceder a la zona privada sin tener cuenta de usuario.

2. Material de la práctica

A continuación, se exponen los archivos necesarios para la realización de esta práctica, que pueden descargarse desde el Aula Virtual:

- **Practica3.2_Guion.pdf**: este documento. Se corresponde con el guion de la primera parte de la práctica y en él están contenidas todas las explicaciones necesarias para su realización.

Además, será necesario descargar el programa **VirtualBox** en su última versión. Este programa, servirá para crear la máquina virtual que se utilizará a lo largo del desarrollo de esta práctica. Puede descargarse desde el siguiente enlace. También será necesario el pack de extensiones "Oracle VM VirtualBox", que permitirá la configuración y el uso de ciertos parámetros y características en las máquinas virtuales (puede descargarse aquí).

Para el desarrollo de esta práctica, hará falta también la descarga de una distribución de un sistema operativo Linux. En este sentido, puede utilizarse la máquina virtual "Kali Linux" utilizada en prácticas anteriores o una máquina virtual creada a partir de una ISO de "Kali Linux", "Ubuntu" o cualquier otra distribución Linux que conozcas. Este sistema operativo será el utilizado a lo largo del desarrollo de esta práctica.

Por último, se deberá tener instalada y configurada, en la máquina virtual mencionada en el anterior párrafo, la pila LAMP (véase la primera parte de la práctica, "Practica3.1_Guion.pdf").

3. Normativa y evaluación

En este apartado se detalla el formato de entrega de la práctica y la forma en la que se evaluará la misma:

- El porcentaje de la nota final de la asignatura al que corresponde esta práctica puede consultarse en la Guía docente de la propia asignatura.
- La práctica deberá realizarse, de forma obligatoria, en grupos de dos personas. Para la asignación de los grupos se deberán seguir las indicaciones del profesor.
- Cada grupo deberá:
    - Realizar una única memoria (puede utilizarse esta misma guía como plantilla) en la que responda las preguntas planteadas y/o en la que se exponga, de forma argumentada, las decisiones tomadas para la realización de la práctica.
    - Desarrollar y/o modificar tantos archivos de código como se soliciten en los diferentes ejercicios que forman la práctica.
- El resultado de la realización de la práctica consistirá en un fichero .zip llamado **Practica3.zip** que deberá entregarse a través del Aula Virtual en el espacio habilitado para ello y en la fecha límite allí expuesta. Este fichero debe contener:
    - La memoria completa en formato PDF. En ella debe indicarse, en la primera página y de forma clara, el nombre y apellidos de los integrantes del grupo.
    - Los archivos de código resultantes de la realización de los ejercicios solicitados:
        - Practica3.2_holamundo.html
        - Practica3.2_styles.css
        - Practica3.2_control_de_accesos.php

4. Enunciado de la práctica

En este apartado se describirán las distintas actividades, programas y ejercicios que deberá realizar cada grupo de prácticas, así como la información a completar y/o rellenar en la memoria. Recuerda entregar también los ficheros HTML y PHP resultantes de la realización de esta práctica.

4.1 Página Web con HTML

Ejercicio 8

En la segunda parte de la práctica, en primer lugar, vamos a montar nuestra primera página web con HTML. Podemos crear una página web sencilla, por ejemplo: Practica3.2_holamundo.html (véase la Tabla 1). Esta página, deberá insertarse en nuestro servidor web (concretamente en el directorio "/var/www/html" de Apache2).

```
<!DOCTYPE html>
<html>
  <head>
    <meta http-equiv="Content-Type" content="text/html;charset=utf-8" />
    <title>Hola Mundo</title>
  </head>
  <body>
    <p>¡Hola Mundo! </p>
  </body>
</html>
```

Tabla 1. Características solicitadas para la base de datos.

A continuación, se pide añadir una serie de elementos a la página web. No se pretende diseñar una web funcional (para ser colgada en un servidor y que todo el mundo tenga acceso a ella), el objetivo de esta práctica es utilizar algunas de las herramientas y opciones HTML más comunes.

RECUERDA ARRANCAR LOS SERVICIOS (Apache2 y MySQL) CADA VEZ QUE REINICIES LA MÁQUINA.

Pregunta 8.1. Crea la página web sencilla indicada en la Tabla 1 y verifica que funciona. Muestra un pantallazo al acceder a ella desde un navegador web.

Pregunta 8.2. Añade una tabla a la página web dentro de la etiqueta <body> (véase el ejemplo de la Tabla 2). En ella, deben incluirse tres filas y tres columnas. Las columnas corresponden a los campos "Profesor", "Despacho" y "Edificio". Las filas corresponden a la siguiente información:

- **Marta Beltrán | 122 | Departamental II**
- **Isaac Martín | 167 | Departamental II**
- **Miguel Calvo | S/N | Departamental II**

```
<table style="width:100%">
  <tr>
    <th>Firstname</th>
    <th>Lastname</th>
    <th>Age</th>
  </tr>
  <tr>
    <td>Jill</td>
    <td>Smith</td>
    <td>50</td>
  </tr>
```

```
  <tr>
    <td>Eve</td>
    <td>Jackson</td>
    <td>94</td>
  </tr>
</table>
```

Tabla 2. Ejemplo de tabla en HTML.

Pregunta 8.3. Investiga sobre las etiquetas "h" de HTML y añade una cabecera con el título "Listado de profesores" a la página web. ¿Qué línea HTML has tenido que añadir?

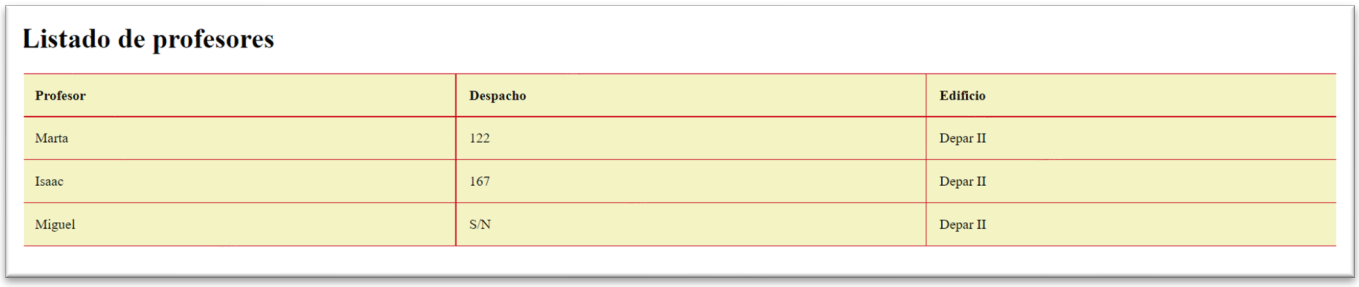
Pregunta 8.4. Cambia el estilo de la tabla utilizando el código CSS de la Tabla 3. ¿Qué has tenido que modificar o añadir en la tabla para que se muestre en color amarillo?

```
<style>
    table, th, td {
        border: 1px solid black;
        border-collapse: collapse;
    }
```

```
    th, td {
        padding: 15px;
        text-align: left;
    }
    table#t01 {
        width: 100%;
        background-color: #f1f1c1;
    }
</style>
```

Tabla 3. Ejemplo de código CSS para tabla en HTML.

<u>Pregunta 8.5</u>. Modifica el CSS que acabas de insertar para que el borde de la tabla (véase la Figura 1) se muestre del mismo color que tiene la corona del logotipo de la Universidad Rey Juan Carlos. Para descubrir el código de color puedes utilizar GIMP, la extensión para Google Chrome "ColorZilla" o alguna herramienta similar. ¿Qué línea has añadido y dónde para realizar este cambio? Haz un pantallazo del resultado.



Listado de profesores

Profesor	Despacho	Edificio
Marta	122	Depar II
Isaac	167	Depar II
Miguel	S/N	Depar II

Figura 1. Tabla HTML con CSS aplicado.

Pregunta 8.6. Añade un formulario a la web. Dicho formulario ha de solicitar un nombre y una contraseña (oculta cuando se escribe, sustituida por "*"). Si das al botón "Acceder" del formulario, la petición de acceso será enviada a una página llamada "Practica3.2_control_de_accesos.php" (esta página de momento no tendrá ninguna funcionalidad, se escribirá en próximos apartados de la práctica). Además, añade una cabecera al formulario que diga "Acceso alumnos", (de tamaño menor que la anterior "Listado de Profesores"). El resultado debe ser similar al de la Figura 2. ¿Qué código has utilizado?

Figura 2. Formulario de Acceso HTML.

Pregunta 8.7. En este ejercicio, se debe añadir una lista a la web. Esta lista debe incluir una cabecera (de tamaño aún menor que la cabecera del formulario de acceso) de color azul que diga "Listado de prácticas". Para ello, investiga sobre las etiquetas "ul" y "li" de HTML y lee la ayuda que te proporcionamos. El resultado debe ser similar al de la Figura 3. Pega el código utilizado para cambiar el color de la cabecera (tanto el CSS como la etiqueta HTML de la propia cabecera).

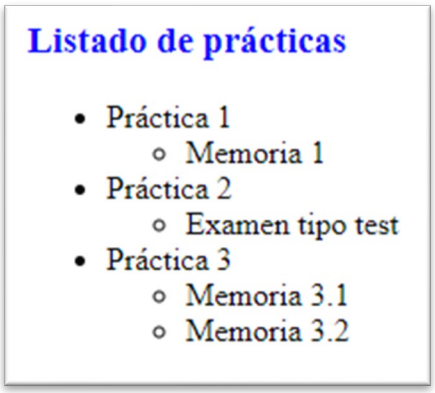


Figura 3. Formulario de Acceso HTML.

AYUDA:

- Una práctica habitual en CSS cuando se quiere cambiar el color de diferentes elementos en una web es añadir una nueva clase para ese. Por ejemplo, en el fichero CSS y/o en el apartado <style> del propio HTML se incluiría lo siguiente:

```
.nombre_del_color {
        color: color_deseado;
}
```

- Además, en el elemento sobre el que se quiere modificar el color, se debería indicar la clase, por ejemplo: `<p class="nombre_del_color"> Mi texto </p>`

Pregunta 8.8. Después de añadir el listado de elementos y buscando un resultado similar al de la Figura 4, añade un nuevo apartado en tu web con una cabecera que diga "Enlaces externos" y una sub-cabecera que diga "Enlaces de interés". En este apartado, debes incluir un enlace a la web de la URJC, otro al de la ETSI y otro al Aula Virtual. El color de los enlaces debe ser (investiga sobre "CSS Links"):

- **Verde cuando no haya sido visitado.**
- **Rosa cuando haya sido visitado.**
- **Subrayado y rojo cuando esté activo.**
- **Azul cuando se tenga el ratón encima.**

¿Qué código CSS has utilizado? Adjunta un pantallazo del resultado.

Figura 4. Enlaces en HTML.

Pregunta 8.9. Graba un audio con la herramienta de grabación de tu ordenador (por ejemplo, "Grabadora de voz" en Windows, "Audio Recorder" en Ubuntu, etc.). En este audio, todos los integrantes del grupo debéis presentaros (diciendo vuestro nombre, la carrera que estáis haciendo y el nombre de la asignatura). Después, incluye el audio dentro de la página web (con cabecera "Presentación"), apoyándote en la etiqueta "audio" de HTML5. El resultado debe ser similar al de la Figura 5. ¿Qué código has utilizado para incluir tu audio en el HTML?

Figura 5. Audio en HTML.

Pregunta 8.10. Cuando los estilos se van complicando, como ha ocurrido en esta web tan sencilla ¿cómo suelen gestionarse? ¿Se dejan en el HTML como hemos hecho en esta primera página o se separan de los contenidos de alguna manera? Hazlo en tu web y llama al nuevo fichero "Practica3.2_styles.css".

El resultado final de este apartado (“Página Web con HTML”), debería ser algo similar a lo que se muestra en la Figura 6.

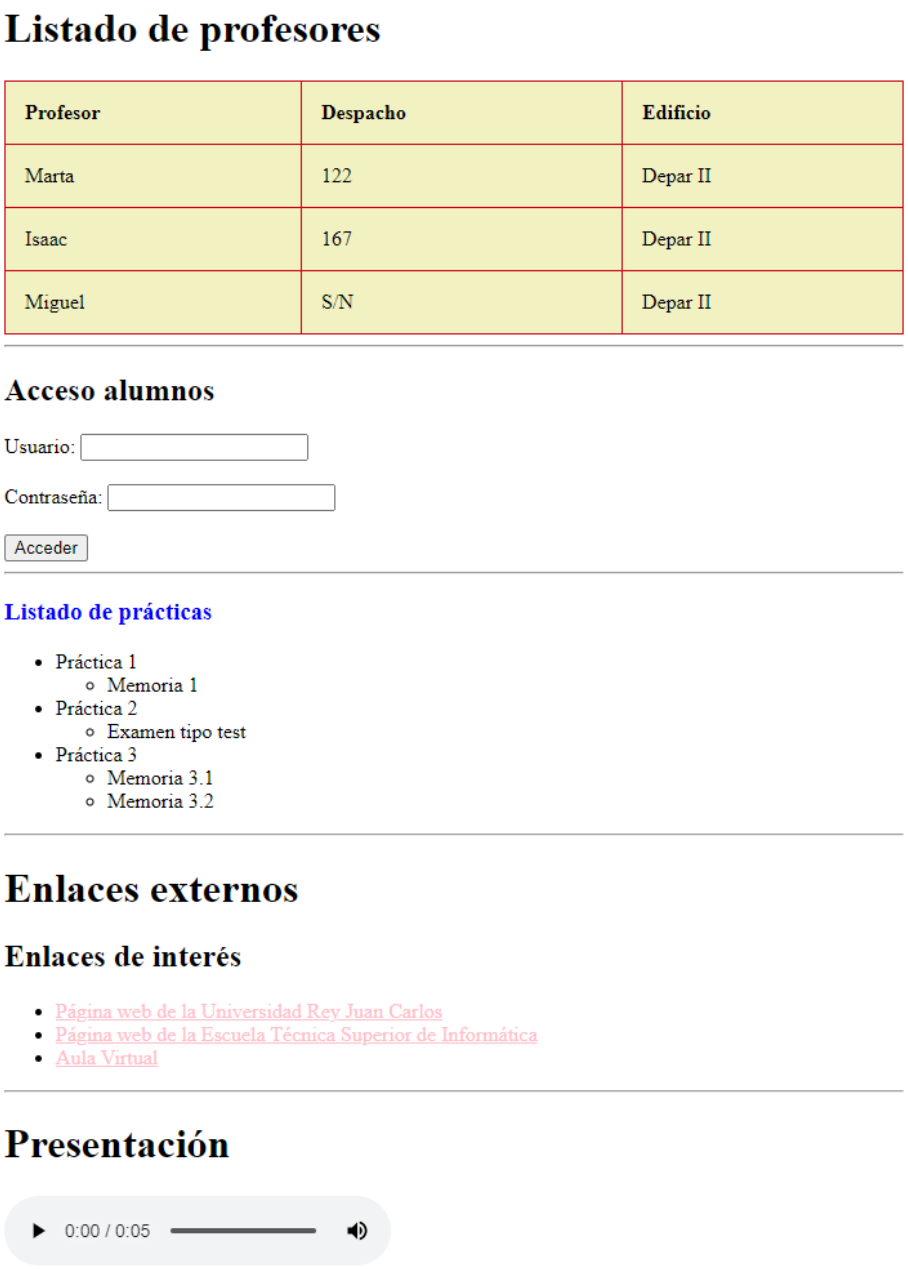
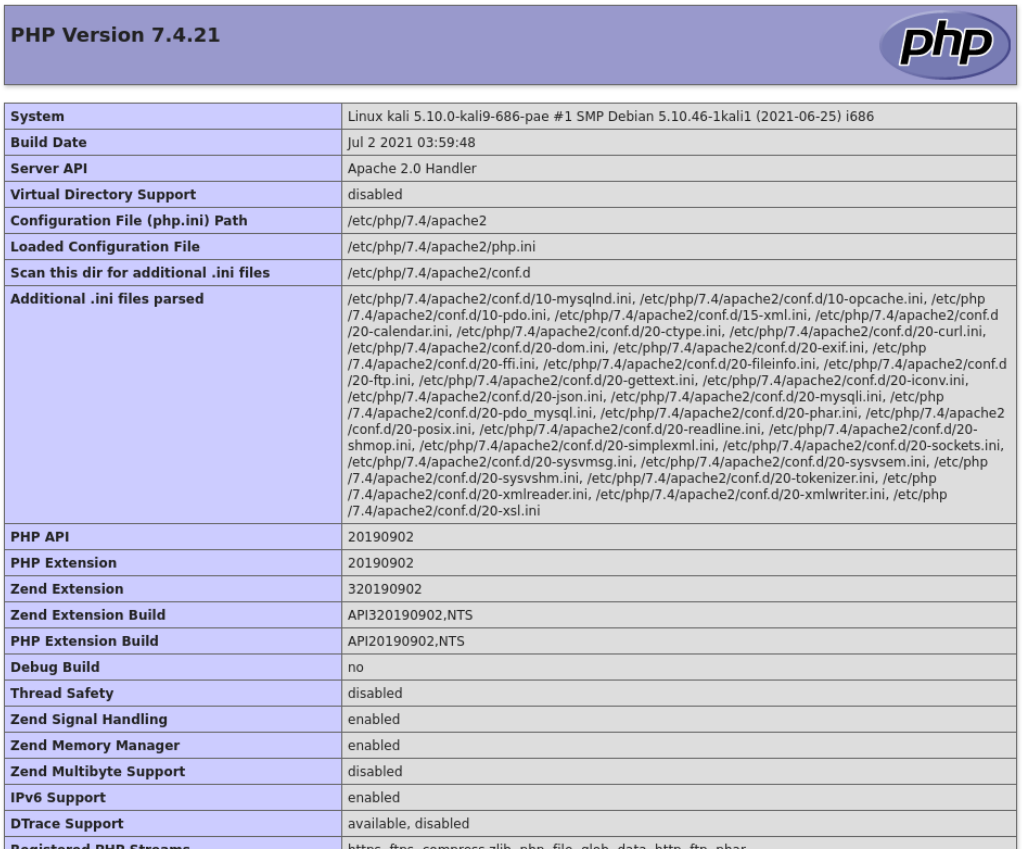


Figura 6. Resultado final del ejercicio “5.1 Página Web con HTML”.

4.2 Página Web con PHP

Ejercicio 9

En la primera parte de esta práctica (“Practica3.1_Guion.pdf”), se instaló toda la pila LAMP y, por ende, debería estar instalado PHP en la máquina virtual. Además, en el directorio “/var/www/html” debería existir un archivo llamado “info.php” (que también se creó en la primera parte de la práctica) y que, accediendo desde el navegador (http://localhost/info.php), debería mostrar algo similar a lo que puede observarse en la Figura 7.



PHP Version 7.4.21

System	Linux kali 5.10.0-kali9-686-pae #1 SMP Debian 5.10.46-1kali1 (2021-06-25) i686
Build Date	Jul 2 2021 03:59:48
Server API	Apache 2.0 Handler
Virtual Directory Support	disabled
Configuration File (php.ini) Path	/etc/php/7.4/apache2
Loaded Configuration File	/etc/php/7.4/apache2/php.ini
Scan this dir for additional .ini files	/etc/php/7.4/apache2/conf.d
Additional .ini files parsed	/etc/php/7.4/apache2/conf.d/10-mysqlnd.ini, /etc/php/7.4/apache2/conf.d/10-opcache.ini, /etc/php/7.4/apache2/conf.d/10-pdo.ini, /etc/php/7.4/apache2/conf.d/15-xml.ini, /etc/php/7.4/apache2/conf.d/20-calendar.ini, /etc/php/7.4/apache2/conf.d/20-ctype.ini, /etc/php/7.4/apache2/conf.d/20-curl.ini, /etc/php/7.4/apache2/conf.d/20-dom.ini, /etc/php/7.4/apache2/conf.d/20-exif.ini, /etc/php/7.4/apache2/conf.d/20-ffi.ini, /etc/php/7.4/apache2/conf.d/20-fileinfo.ini, /etc/php/7.4/apache2/conf.d/20-ftp.ini, /etc/php/7.4/apache2/conf.d/20-gettext.ini, /etc/php/7.4/apache2/conf.d/20-iconv.ini, /etc/php/7.4/apache2/conf.d/20-json.ini, /etc/php/7.4/apache2/conf.d/20-mysqli.ini, /etc/php/7.4/apache2/conf.d/20-pdo_mysql.ini, /etc/php/7.4/apache2/conf.d/20-phar.ini, /etc/php/7.4/apache2/conf.d/20-posix.ini, /etc/php/7.4/apache2/conf.d/20-readline.ini, /etc/php/7.4/apache2/conf.d/20-shmop.ini, /etc/php/7.4/apache2/conf.d/20-simplexml.ini, /etc/php/7.4/apache2/conf.d/20-sockets.ini, /etc/php/7.4/apache2/conf.d/20-sysvmsg.ini, /etc/php/7.4/apache2/conf.d/20-sysvsem.ini, /etc/php/7.4/apache2/conf.d/20-sysvshm.ini, /etc/php/7.4/apache2/conf.d/20-tokenizer.ini, /etc/php/7.4/apache2/conf.d/20-xmlreader.ini, /etc/php/7.4/apache2/conf.d/20-xmlwriter.ini, /etc/php/7.4/apache2/conf.d/20-xsl.ini
PHP API	20190902
PHP Extension	20190902
Zend Extension	320190902
Zend Extension Build	API320190902,NTS
PHP Extension Build	API20190902,NTS
Debug Build	no
Thread Safety	disabled
Zend Signal Handling	enabled
Zend Memory Manager	enabled
Zend Multibyte Support	disabled
IPv6 Support	enabled
DTrace Support	available, disabled

Figura 7. Comprobación de la instalación de LAMP.

A continuación, se pide dotar de funcionalidad al formulario que has incluido antes en la página web HTML. Si recuerdas:

- En la primera parte de la práctica, has creado una base de datos llamada “accesos” con una tabla “user_pass” con campos “user” y “password”. Además, has añadido dos filas como mínimo a esta tabla.

- En esta segunda parte de la práctica has creado en tu página web HTML (Practica3.2_holamundo.html) un formulario para dar acceso a una zona privada que pide un campo “usuario” y un campo “contraseña” y que muestra un botón “Acceder”.

PHP nos permite conectar este formulario con la base de datos de manera que un usuario que se encuentre en la tabla y que escribe correctamente su nombre y su contraseña, tendrá éxito al dar al botón “Acceder”. De la misma forma, si un usuario no se encuentra en la tabla o no escribe correctamente su nombre y/o contraseña, se le devolverá un error.

Pregunta 9.1. ¿Qué hay que modificar en el código HTML de la página “Practica3.2_holamundo.html” para que al dar al botón “Acceder” del formulario se procese el control de accesos (“Practica3.2_control_de_accesos.php”) contra la base de datos de usuarios y contraseñas que tenemos en MySQL?

Pregunta 9.2. ¿Cómo debe programarse la página “Practica3.2_control_de_accesos.php”? Escríbela completando el esqueleto de la Tabla 4 y comprueba que funciona correctamente en todos los casos posibles.

NOTA: No te limites a copiar el código y rellenar los huecos, intenta entenderlo, comentarlo, mejorar la gestión de los errores, etc. No hace falta programar la página web para la zona privada, con los mensajes que se muestra en los “echo” es suficiente.

```
<?php

if (isset($_POST['XXX1']) && isset($_POST['XXX2']) && !empty($_POST['XXX1']) && !empty($_POST['XXX2'])) {

        $user = $_POST['XXX1'];
        $pass = $_POST['XXX2'];

        $conn = new mysqli('localhost', 'app', 'XXX3', 'XXX4');

        if ($conn->connect_error) {
                die("Falló la conexión a BBDD: " . $conn->connect_error);
        }

        $resultado = $conn->query("SELECT * FROM XXX5 WHERE XXX6 AND XXX7;");

        if ($resultado->XXX8){
                echo 'Usuario encontrado: Puedes acceder a la zona privada';
        } else {
                echo 'Usuario no encontrado: Vuelve a intentarlo';
        }

        XXX9->close();
} else {
        echo 'Introduce un usuario y una contraseña.';
}

?>
```

Tabla 4. Estructura del código del fichero “Practica3.2_control_de_accesos.php”.

AYUDA:

- Sin necesitas mostrar los errores que tienes en PHP mientras desarrollas, puedes utilizar, colocándolo al inicio del código, las siguientes líneas:

  ini_set('display_errors', 1);

  ini_set('display_startup_errors', 1);

  error_reporting(E_ALL);

- Ten en cuenta que en una página web en producción, esos errores JAMÁS deberían mostrarse, ya que pueden ser utilizados por los atacantes para descubrir fallos en nuestras aplicaciones.

4.3 Inyección SQL

Ejercicio 10

Tal y como se ha desarrollado y escrito la página “Practica3.2_control_de_accesos.php”, se está confiando en la entrada que viene desde el lado del usuario. Directamente, los datos que el usuario introduce en el formulario son los que se utilizan para construir la consulta que se lanza a la base de datos (recuperados mediante “$_POST['XXXXX']”), lo cual hace nuestra aplicación completamente insegura.

Pregunta 10.1. Sabiendo que el adversario no tiene cuenta en nuestra aplicación y, por lo tanto, no está en la base de datos, ¿Qué podría escribir en el campo del formulario de la contraseña para ganar acceso a la zona privada? ¿Qué daría siempre un resultado TRUE al lanzar la consulta SQL contra la base de datos? Este patrón de ataque se denomina inyección SQL.

AYUDA:

- Piensa en operadores lógicos como el AND o el OR y en cómo afectarían al resultado de la consulta si se inyectaran en alguna posición concreta.
- Como eres el desarrollador de “Practica3.2_control_de_accesos.php” y estás empezando, puedes ayudarte con la inyección SQL haciendo un “echo” en PHP de la consulta que se lanzaría a la base de datos. De esta forma, podrás ver el resultado de introducir diferentes parámetros a través del formulario y comprobar si la consulta está bien formada o no.

Pregunta 10.2. ¿Cómo tendría que mejorarse la página "Practica3.2_control_de_accesos.php" para evitar este tipo de inyecciones? Propón al menos dos alternativas y explícalas (no hace falta que las implementes).

CASO PRÁCTICO 2 – ASPECTOS ÉTICOS DE LA CIBERSEGURIDAD

CONTEXTO

Es imprescindible que tengáis la capacidad de emitir juicios críticos sobre situaciones que pueden ser ambiguas o éticamente discutibles. Sobre todo, a partir del curso próximo cuando comencéis a adquirir competencias en seguridad ofensiva.

FORMA DE TRABAJAR

En equipos de cuatro personas (dos parejas), yo os asigno un tema. Cada pareja dentro del equipo tiene que argumentar a favor de una posición diferente (una a favor del SI, otra a favor del NO). Esto son dos bloques del entregable: a favor, en contra. Y hay que redactar un tercer bloque con unas conclusiones. Los argumentos tienen que basarse en una fase de documentación (lecturas, charlas, vídeos, noticias, etc.). Por lo tanto, es necesario que incluyáis una sección de Bibliografía completa y actualizada al final del documento.

TEMAS

1. Hack back (identificar el origen de un ciberataque y contra-atacar).
   https://worldcrunch.com/tech-science/hack-back-when-a-cyber-attack-victim-turns-digital-vigilante
2. Enseñanza de seguridad ofensiva desde el bachillerato (CTFs, bug bounties, ejercicios gamificados, etc.).
   https://www.business-live.co.uk/technology/students-recruited-ethical-hacking-boost-22061760
3. Inclusión del hacking ético como delito en la legislación.
   https://www.itproportal.com/features/should-the-law-treat-ethical-hackers-differently-to-regular-citizens/
   https://www.legaltoday.com/practica-juridica/derecho-penal/penal/delito-de-hacking-y-comentario-de-sentencia-2021-04-16/
4. Full disclosure de vulnerabilidades (publicación de nuevas vulnerabilidades en redes sociales e Internet de manera masiva: see something, say something).
   https://www.theregister.com/2021/09/24/apple_zeroday/
5. Pago de rescates para desmontar mafias de ransomware o que alquilan botnets.
   https://apnews.com/article/technology-joe-biden-europe-business-government-and-politics-cd21d84b5fd070421f871610b40e91d0
6. Exigencia de responsabilidades a los proveedores que alojan o almacenan materiales ilegales (material de hacking, pornografía infantil, etc.).
   https://portswigger.net/daily-swig/insider-phd-hacking-education-channel-suspended-from-youtube-for-severe-guideline-violations
   https://www.nytimes.com/2021/08/05/technology/apple-iphones-privacy.html