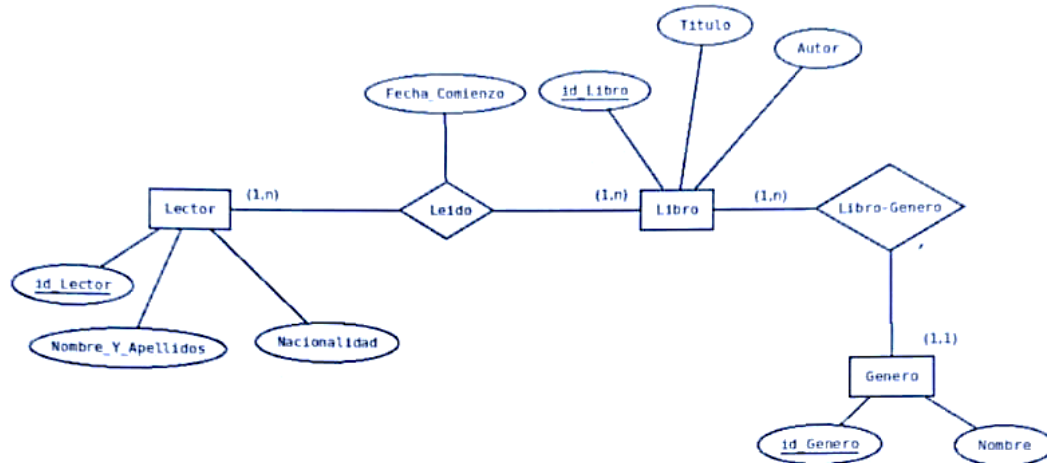




ENUNCIADO EJERCICIO PRÁCTICO OPOSICIONES
PES INFORMÁTICA ANDALUCÍA 2025

EJERCICIO 1: BASES DE DATOS (Total: 2.5 puntos).

Observe el siguiente diagrama entidad/relación:



Tenga en cuenta las siguientes definiciones y restricciones:

- ✚ id_Lector: Entero.
- ✚ Nombre_Y_Apellidos: Cadena de 50 caracteres. No puede estar vacío.
- ✚ Nacionalidad: Cadena de 20 caracteres. No puede estar vacío.
- ✚ Fecha_Comienzo: Fecha. No puede estar vacío. Representa la fecha en que un lector comienza a leer un determinado libro.
- ✚ id_Libro: Entero.
- ✚ Título: Cadena de 40 caracteres. No puede estar vacío.
- ✚ Autor: Cadena de 40 caracteres. No puede estar vacío.
- ✚ id_Genero: Entero.
- ✚ Nombre: Cadena de 40 caracteres. No puede estar vacío.

Paso a tablas:

Lector(id_Lector, Nombre_Y_Apellidos, Nacionalidad)

Leído(id_Lector, id_Libro, Fecha_Comienzo)

Libro(id_Libro, Título, Autor, id_Genero)

Genero(id_Genero, Nombre)

EJERCICIO 1.1. CREACIÓN DE TABLAS (1.25 puntos).

Detalle las sentencias SQL necesarias para crear las tablas de la base de datos que implemente el modelo anterior en tercera forma normal.

CREATE TABLE Lector(



```
id_Lector INTEGER PRIMARY KEY,  
Nombre_Y_Apellidos VARCHAR(50) NOT NULL,  
Nacionalidad VARCHAR(20) NOT NULL);
```

```
CREATE TABLE Genero(  
id_Genero INTEGER PRIMARY KEY,  
Nombre VARCHAR(40) NOT NULL);
```

```
CREATE TABLE Libro(  
id_Libro INTEGER PRIMARY KEY,  
Titulo VARCHAR(40) NOT NULL,  
Autor VARCHAR(40) NOT NULL,  
id_Genero INTEGER REFERENCES Genero(id_Genero) ON DELETE CASCADE ON  
UPDATE CASCADE);
```

```
CREATE TABLE Leido(  
id_Lector INTEGER REFERENCES Lector(id_Lector) ON DELETE CASCADE ON  
UPDATE CASCADE,  
id_Libro INTEGER REFERENCES Libro(id_Libro) ON DELETE CASCADE ON  
UPDATE CASCADE,  
Fecha_Comienzo DATE NOT NULL,  
PRIMARY KEY(id_Lector,id_Libro));
```

EJERCICIO 1.2. CONSULTA (1.25 puntos).

Dadas las tablas que ha realizado en el apartado anterior, detalle las sentencias SQL necesarias para realizar la siguiente consulta:

Todos los libros de género 'Terror' que se leyeron en 2024 por lectores italianos (Nacionalidad = 'Italia') con las cantidades de lecturas acumuladas en dicho período. Se considerarán leídos en 2024 todos los libros que se comenzaron a leer entre el 1 de enero y el 31 de diciembre de 2024.

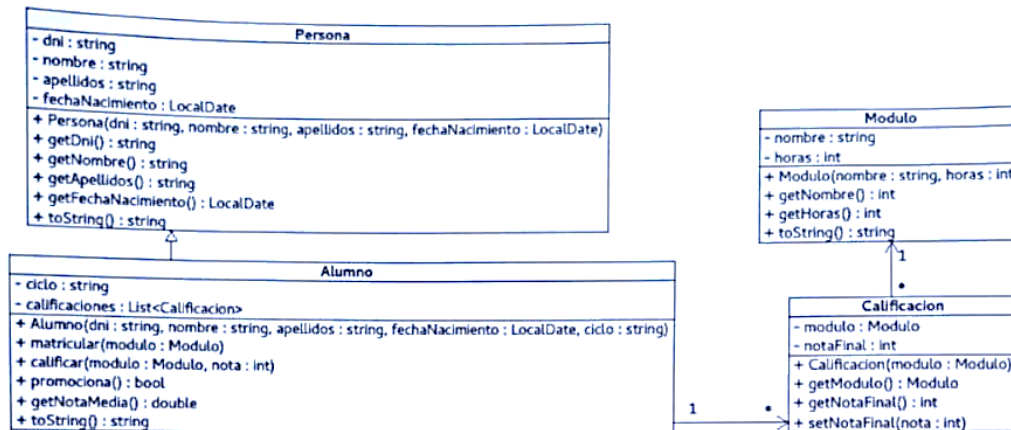
Debe aparecer una línea por cada libro que cumpla dicho requisito, sin repeticiones. Cada línea contendrá los siguientes campos: *Título*, *Autor* y *Total_Lecturas*.

```
SELECT LB.Titulo, LB.Autor, COUNT(*) AS "Total_Lecturas"  
FROM Lector LC INNER JOIN Leido LD ON LC.id_Lector = LD.id_Lector INNER  
JOIN Libro LB ON LD.id_Libro = LB.id_Libro INNER JOIN Genero G ON  
LB.id_Genero=G.id_Genero  
WHERE LC.Nacionalidad = "Italia" AND LD.Fecha_Comienzo BETWEEN "2024-01-  
01" AND "2024-12-31" AND G.Nombre = "Terror"  
GROUP BY LB.Titulo;
```



EJERCICIO 2: PROGRAMACIÓN (Total: 2.5 puntos).

Desarrollar en lenguaje Java la estructura representada en el siguiente Diagrama de Clases. En dicho diagrama se describe la estructura para representar los alumnos de un Ciclo Formativo con sus módulos y calificaciones.



Se deben desarrollar las clases *Alumno* y *Calificacion*. Suponemos el resto de clases ya desarrolladas previamente. A continuación, se describe la funcionalidad de los métodos más importantes:

- ✚ En la clase *Alumno*, el método *matricular()* añadirá un módulo y su calificación al alumno. La calificación inicial será 0, indicando que aún está sin calificar. El método *calificar()* servirá para asignarle una calificación numérica (1 – 10) al alumno en el módulo especificado. Si el alumno ya estaba matriculado de ese módulo, el método no hará nada.
- ✚ Los métodos que toman un módulo como argumento, si tienen que buscar dicho módulo entre las calificaciones, usarán el nombre del módulo para determinar la coincidencia.
- ✚ El método *promociona()* devolverá *true* si el alumno está en condiciones de promocionar, es decir, si la suma de las horas de los módulos aprobados es mayor o igual al 50% del total de horas de los módulos matriculados.
- ✚ El método *toString()* de *Alumno*, devolverá lo mismo que el de *Persona* (datos personales), añadiendo detrás, "Nota Media = XX.XX", donde XX.XX será la nota media del alumno, con dos cifras decimales como máximo.

A continuación, se muestra un ejemplo de uso de las clases descritas junto con la salida generada:

Ejemplo de Uso:



MARTÍNEZ & DE HARO

PREPARADORES DE OPOSICIONES

PREPARADORES MARTÍNEZ & DE HARO
Preparación Oposiciones PES Informática y PES SAI

```
Alumno alumno1 = new Alumno("1111111H", "Alejandro", "Fernández", LocalDate.of(2000, 3, 15),  
"DAW");  
Alumno alumno2 = new Alumno("2222222M", "Daniel", "Travieso", LocalDate.of(1992, 7, 5), "DAW");  
Modulo progr = new Modulo("Programacion", 8);  
Modulo bd = new Modulo("Bases de Datos", 7);  
Modulo sis = new Modulo("Sistemas", 4);  
Modulo endes = new Modulo("Entornos de Desarrollo", 3);  
alumno1.maticular(progr);  
alumno1.maticular(bd);  
alumno1.maticular(sis);  
alumno1.maticular(endes);  
  
alumno2.maticular(progr);  
alumno2.maticular(bd);  
alumno2.maticular(sis);  
alumno2.maticular(endes);  
  
alumno1.calificar(progr, 8);  
alumno1.calificar(bd, 6);  
alumno1.calificar(sis, 6);  
alumno1.calificar(endes, 10);  
  
alumno2.calificar(progr, 4);  
alumno2.calificar(bd, 4);  
alumno2.calificar(sis, 6);  
alumno2.calificar(endes, 10);  
  
System.out.println(alumno1 + ", promociona: " + (alumno1.promociona()?"Si":"No"));  
System.out.println(alumno2 + ", promociona: " + (alumno2.promociona()?"Si":"No"));
```

Resultado:

```
Alejandro Fernández Nota Media = 7.5, promociona: Si  
Daniel Travieso Nota Media = 6.0, promociona: No
```

Observaciones:

- ✚ Se valorarán positivamente la limpieza y claridad en el código, así como la simplificación (hacer el código lo más sencillo posibles).
- ✚ También se valorará el uso de comentarios allá donde sea conveniente alguna aclaración.
- ✚ Se escribirá sólo el código de las clases. No es necesario especificar los "imports" ni los paquetes.
- ✚ Se separarán con claridad las distintas clases (con un espacio amplio o una línea horizontal), para facilitar su lectura.
- ✚ Se respetarán los nombres que aparecen en el diagrama.
- ✚ No se debe añadir ninguna funcionalidad no solicitada.

Propuesta de solución:

Clase Alumno



```
import java.time.LocalDate;
import java.util.ArrayList;
import java.util.List;

public class Alumno extends Persona{
    private String ciclo;
    private List<Calificacion> calificaciones;

    public Alumno(String dni, String nombre, String apellidos, LocalDate fechaNacimiento, String ciclo){
        super(dni, nombre, apellidos, fechaNacimiento);
        this.ciclo=ciclo;
        calificaciones = new ArrayList<Calificacion>();
    }

    public void matricular(Modulo modulo){
        for(int i=0; i<calificaciones.size();i++){
            if(calificaciones.get(i).getModulo().getNombre().equals(modulo.getNombre()))
                return;
        }
        Calificacion c = new Calificacion(modulo);
        calificaciones.add(c);
    }

    //Creo que el enunciado está mal y quiere decir que si el módulo NO está en el listado
    //de calificaciones el método no hará nada...
    public void calificar(Modulo modulo, int nota){
        for(int i=0; i<calificaciones.size();i++){
            if(calificaciones.get(i).getModulo().getNombre().equals(modulo.getNombre())){
                calificaciones.get(i).setNotaFinal(nota);
                return;
            }
        }
    }

    public boolean promociona(){
        int Total_Horas=0, Horas_Aprobados=0;
        for(int i=0; i<calificaciones.size();i++){
            if(calificaciones.get(i).getNotaFinal()>=5){
                Horas_Aprobados += calificaciones.get(i).getModulo().getHoras();
            }
            Total_Horas += calificaciones.get(i).getModulo().getHoras();
        }
        if(Horas_Aprobados >= (Total_Horas/2))
            return true;
        else
            return false;
    }
}
```



```
public double getNotaMedia(){
    double media = 0;
    for(int i=0; i<calificaciones.size();i++)
        media += calificaciones.get(i).getNotaFinal();
    media = media / calificaciones.size();
    return media;
}

public String toString(){
    return String.format("%s Nota Media = %.2f", super.toString(),this.getNotaMedia());
}
}
```

Clase Clasificacion

```
public class Calificacion {
    private Modulo modulo;
    private int notaFinal;

    public Calificacion(Modulo modulo){
        this.modulo=modulo;
        notaFinal = 0;
    }

    public Modulo getModulo() {
        return modulo;
    }

    public int getNotaFinal() {
        return notaFinal;
    }

    public void setNotaFinal(int nota) {
        this.notaFinal = nota;
    }
}
```

EJERCICIO 3: REDES. (2.5 puntos).

La empresa Goodluck, S.L. cuenta en su sede principal con un sistema informático compuesto por varios servidores y 50 equipos de usuario. Su red se organiza en una DMZ, una LAN PROTEGIDA y una LAN SALIDA. El sistema está conectado a Internet por un router de fibra óptica de una compañía de telecomunicaciones.

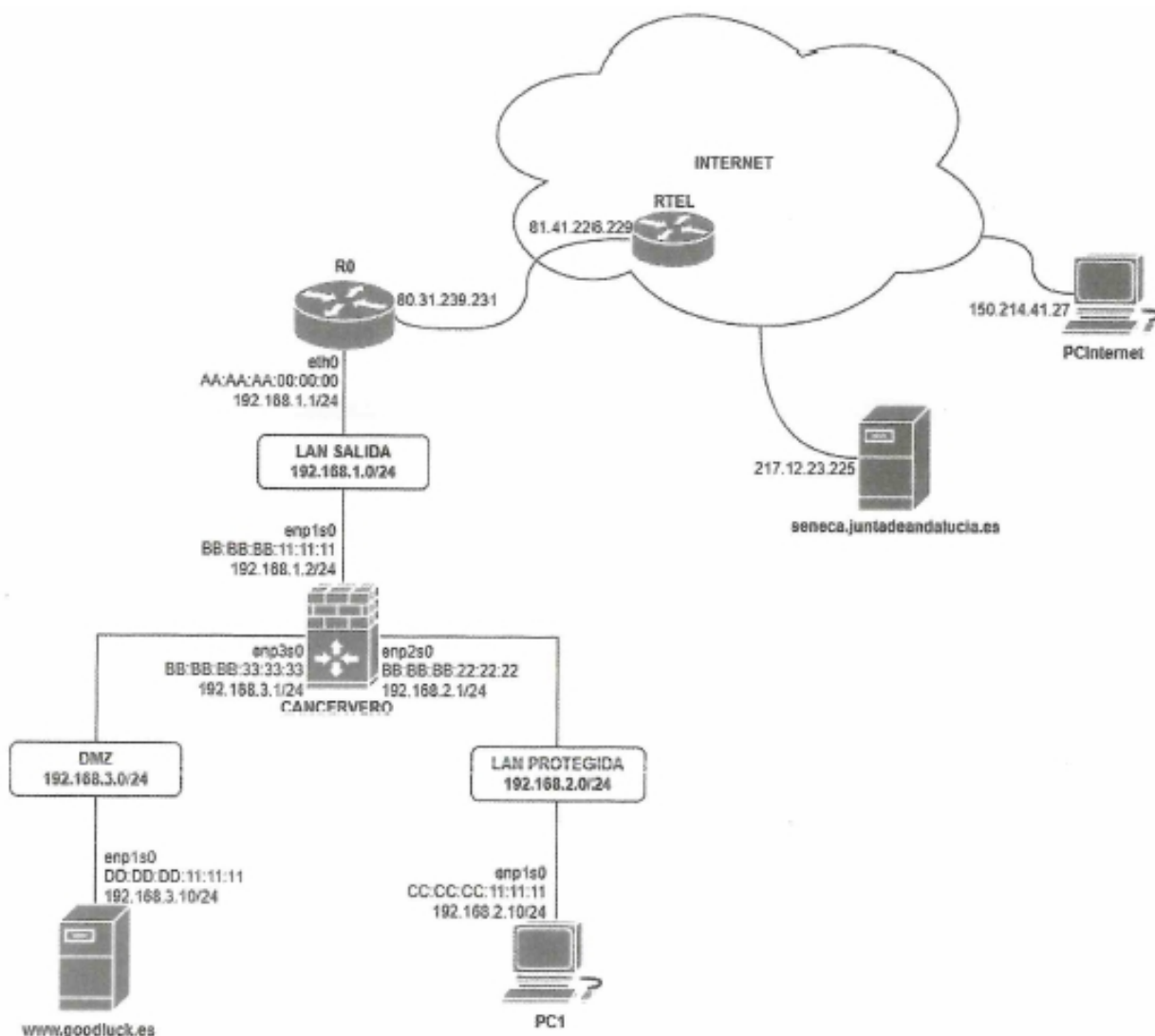
A continuación, se muestra un esquema lógico que incluye Internet y la red indicada. En este esquema se muestran únicamente los dispositivos relevantes para el desarrollo del ejercicio.



MARTÍNEZ & DE HARO

PREPARADORES DE OPOSICIONES

PREPARADORES MARTÍNEZ & DE HARO
Preparación Oposiciones PES Informática y PES SAI



En el esquema, cada equipo de la empresa tiene identificadas sus tarjetas de red Gigabit Ethernet con su nombre, dirección MAC¹ y dirección IP.

Tenga en cuenta las siguientes consideraciones:

- ✚ **R0** es un router proporcionado por la compañía de telecomunicaciones, que ofrece conectividad a Internet mediante fibra óptica. Está configurado de la siguiente manera:
 - ✓ Realiza NAPT² (masquerading) de LAN SALIDA hacia Internet.
 - ✓ Port Forwarding de 80.31.239.231:443 a 192.168.1.2:443.
- ✚ **RTEL** es un router de la compañía de telecomunicaciones, ubicado en Internet, que actúa como puerta de enlace de R0 para todo el tráfico con destino a Internet.

¹ Las MAC indicadas son ficticias, no se corresponden con la realidad, sólo buscan que usted las identifique cómodamente.

² NAPT: Network Address and Port Translation (RFC 2663), o NAT dinámico con sobrecarga de puertos. En contextos didácticos o informales, también se le puede llamar PAT.



- + www.goodluck.es es un servidor web alojado en la DMZ de la empresa.
- + **PC1** es un equipo de usuario situado en la LAN PROTEGIDA de la empresa.
- + **seneca.juntadeandalucia.es** es un servidor web público ubicado en Internet.
- + **PCInternet** es un PC conectado directamente a Internet, con IP pública.
- + Se supone que el servidor DNS en Internet del dominio *goodluck.es* funciona perfectamente, traduciendo www.goodluck.es por 80.31.239.231.
- + Se supone que el servidor DNS interno de la organización funciona perfectamente, traduciendo www.goodluck.es por 192.168.3.10.
- + **Cancervero** es una máquina con Debian 12 (sin entorno gráfico) con el paquete iptables correctamente instalado y operativo, que cumple todas las funciones que se especifican a continuación:
 - ✓ El enrutamiento IP está activado.
 - ✓ La ruta por defecto es 192.168.1.1 a través de la interfaz enp1s0.
 - ✓ Realiza NAPT (masquerading) para la red DMZ hacia la red LAN SALIDA.
 - ✓ Realiza NAPT (masquerading) para la red LAN PROTEGIDA hacia la red LAN SALIDA.
 - ✓ Realiza Port Forwarding de 192.168.1.2:443 a 192.168.3.10:443.
 - ✓ Implementa una serie de reglas de filtrado que no vienen al caso para el presente ejercicio.

RESPONDA A LAS SIGUIENTES CUESTIONES:

Ejercicio 3.1. PC1 se desea conectar a la web *seneca.juntadeandalucia.es* por https. Para ello, PC1 envía un segmento TCP con el flag SYN=1 que viaja desde PC1 a *seneca.juntadeandalucia.es*. Responda a las siguientes cuestiones relativas a esa comunicación:

3.1.a. En el tramo de red **entre PC1 y Cancervero**, indique los siguientes campos de la trama Ethernet, del paquete IP y del segmento TCP que se transmiten en ese momento: MAC origen, MAC destino, IP origen, IP destino, puerto origen y puerto destino.

MAC origen = CC:CC:CC:11:11:11, MAC destino = BB:BB:BB:22:22:22

IP origen = 192.168.2.10, IP destino = 217.12.23.225

Puerto origen = 40500 (aleatorio), Puerto destino = 443

3.1.b. En el tramo de red **entre Cancervero y R0**, indique los siguientes campos de la trama Ethernet, del paquete IP y del segmento TCP que se transmiten en ese momento: MAC origen, MAC destino, IP origen, IP destino, puerto origen y puerto destino.

MAC origen = BB:BB:BB:11:11:11, MAC destino = AA:AA:AA:00:00:00

IP origen = 192.168.1.2, IP destino = 217.12.23.225

Puerto origen = 50000 (aleatorio), Puerto destino = 443



3.1.c. En el tramo de red **entre R0 y seneca.juntadeandalucia.es**, indique los siguientes campos del paquete IP y del segmento TCP que se transmiten en ese momento: IP origen, IP destino, puerto origen y puerto destino.

IP origen = 80.31.239.231, IP destino = 217.12.23.225

Puerto origen = 50000, Puerto destino = 443

Ejercicio 3.2. *seneca.juntadeandalucia.es* responde al mensaje anterior (ejercicio 3.1), enviándole a PC1 un segmento TCP con el flag SYN=1 y el flag ACK=1. Responda a las siguientes cuestiones relativas a esa comunicación:

3.2.a. En el tramo de red **entre seneca.juntadeandalucia.es y R0**, indique los siguientes campos del paquete IP y del segmento TCP que se transmiten en ese momento: IP origen, IP destino, puerto origen y puerto destino.

IP origen = 217.12.23.225, IP destino = 80.31.239.231

Puerto origen = 443, Puerto destino = 50000

3.2.b. En el tramo de red **entre R0 y Cancervero**, indique los siguientes campos de la trama Ethernet, del paquete IP y del segmento TCP que se transmiten en ese momento: MAC origen, MAC destino, IP origen, IP destino, puerto origen y puerto destino.

MAC origen = AA:AA:AA:00:00:00, MAC destino = BB:BB:BB:11:11:11

IP origen = 217.12.23.225, IP destino = 192.168.1.2

Puerto origen = 443, Puerto destino = 50000

3.2.c. En el tramo de red **entre Cancervero y PC1**, indique los siguientes campos de la trama Ethernet, del paquete IP y del segmento TCP que se transmiten en ese momento: MAC origen, MAC destino, IP origen, IP destino, puerto origen y puerto destino.

MAC origen = BB:BB:BB:22:22:22, MAC destino = CC:CC:CC:11:11:11

IP origen = 217.12.23.225, IP destino = 192.168.2.10

Puerto origen = 443, Puerto destino = 405000

Ejercicio 3.3. PCInternet se desea conectar a la web *www.goodluck.es* por https. Para ello, PCInternet envía un segmento TCP con el flag SYN=1 que viaja desde PCInternet a *www.goodluck.es*. Responda a las siguientes cuestiones relativas a esa comunicación:

3.3.a. En el tramo de red **entre PCInternet y R0**, indique los siguientes campos del paquete IP y del segmento TCP que se transmiten en ese momento: IP origen, IP destino, puerto origen y puerto destino.

IP origen = 150.214.41.27, IP destino = 80.31.239.231

Puerto origen = 39000, Puerto destino = 443

3.3.b. En el tramo de red **entre R0 y Cancervero**, indique los siguientes campos de la trama Ethernet, del paquete IP y del segmento TCP que se transmiten en ese momento: MAC origen, MAC destino, IP origen, IP destino, puerto origen y puerto destino.



MAC origen = AA:AA:AA:00:00:00, MAC destino = BB:BB:BB:11:11:11

IP origen = 150.214.41.27, IP destino = 192.168.1.2

Puerto origen = 39000, Puerto destino = 443

3.3.c. En el tramo de red **entre Cancervero y www.goodluck.es**, indique los siguientes campos de la trama Ethernet, del paquete IP y del segmento TCP que se transmiten en ese momento: MAC origen, MAC destino, IP origen, IP destino, puerto origen y puerto destino.

MAC origen = BB:BB:BB:33:33:33, MAC destino = DD:DD:DD:11:11:11

IP origen = 150.214.41.27, IP destino = 192.168.3.10

Puerto origen = 39000, Puerto destino = 443

Ejercicio 3.4. www.goodluck.es responde al mensaje anterior (ejercicio 3.3), enviándole a PCInternet un segmento TCP con el flag SYN=1 y el flag ACK=1. Responda a las siguientes cuestiones relativas a esa comunicación:

3.4.a. En el tramo de red **entre www.goodluck.es y Cancervero**, indique los siguientes campos de la trama Ethernet, del paquete IP y del segmento TCP que se transmiten en ese momento: MAC origen, MAC destino, IP origen, IP destino, puerto origen y puerto destino.

MAC origen = DD:DD:DD:11:11:11, MAC destino = BB:BB:BB:33:33:33

IP origen = 192.168.3.10, IP destino = 150.214.41.27

Puerto origen = 443, Puerto destino = 39000

3.4.b. En el tramo de red **entre y Cancervero y R0**, indique los siguientes campos de la trama Ethernet, del paquete IP y del segmento TCP que se transmiten en ese momento: MAC origen, MAC destino, IP origen, IP destino, puerto origen y puerto destino.

MAC origen = BB:BB:BB:11:11:11, MAC destino = AA:AA:AA:00:00:00

IP origen = 192.168.1.2, IP destino = 150.214.41.27

Puerto origen = 443, Puerto destino = 39000

3.4.c. En el tramo de red **entre R0 y PCInternet**, indique los siguientes campos del paquete IP y del segmento TCP que se transmiten en ese momento: IP origen, IP destino, puerto origen y puerto destino.

IP origen = 80.31.239.231, IP destino = 150.214.41.27

Puerto origen = 443, Puerto destino = 39000

Observaciones importantes:

- ✚ Para responder a este ejercicio, identifique en su respuesta el número de la misma, por ejemplo:

3.1.a:

MAC origen = XX:XX:XX:XX:XX:XX, MAC destino = XX:XX:XX:XX:XX:XX

IP origen=X.X.X.X, IP destino= X.X.X.X

Puerto origen=XXXXX, Puerto destino=XXXXX



- ✚ En el caso de puertos que sean aleatorios, elija el que quiera dentro del rango adecuado, y vigile atentamente que sea coherente con el resto de respuestas del ejercicio.

EJERCICIO 4: SEGURIDAD CON IPTABLES Y SCRIPTING EN BASH. (TOTAL: 2.5 puntos).

Para este ejercicio se tomará como base el esquema de red del Ejercicio 3, donde dispone de todas las IPs y nombres de interfaces de red. Se trabajará sobre la máquina **Cancervero**, un sistema Debian 12 sin entorno gráfico, con iptables instalado y plenamente operativo.

TAREA: Debe realizar un script en Bash llamado **completar_iptables.sh**, que se ejecutará en el directorio `/etc/iptables` del dispositivo *Cancervero*, como usuario *root*. El script **generará automáticamente un nuevo script llamado *firewall.sh***. El script **completar_iptables.sh** se debe basar en la siguiente plantilla, que usted debe completar. En su código, debe utilizar obligatoriamente las variables `DIR`, `BASE`, `FIREWALL` y `LISTA`.

```
#!/bin/bash
# completar_iptables.sh

# Directorio y ficheros
DIR="/etc/iptables"
BASE="$DIR/base_iptables.sh"
FIREWALL="$DIR/firewall.sh"
LISTA="$DIR/listablanca.txt"

#PARTE 1
... Código script de la parte 1

#PARTE 2
... Código script de la parte 2

#PARTE 3
... Código script de la parte 3

#PARTE 4
... Código script de la parte 4
```



A continuación, se describen las distintas partes que debe desarrollar:

PARTE 1:

- ✚ Creará un nuevo fichero */etc/iptables/firewall.sh*, le asignará la propiedad al usuario root y permisos de lectura, escritura y ejecución solo para root.
- ✚ Cargará el inicio del fichero *firewall.sh*, todo el contenido del fichero */etc/iptables/base_iptables.sh*.

PARTE 2:

- ✚ Añadirá el fichero *firewall.sh* las reglas iptables necesarias para permitir el tráfico de entrada y salida en la interfaz *lo* (loopback) en el propio *Cancervero*³.

PARTE 3:

- ✚ Añadirá al fichero *firewall.sh* las reglas iptables necesarias para permitir el acceso por SSH desde los equipos de usuario con IPs 192.168.2.4, 192.168.2.5, 192.168.2.6 y 192.168.2.7, hacia cualquier equipo en la DMZ. Se valorará el uso de la sentencia *for*.

PARTE 4:

- ✚ Añadirá al fichero *firewall.sh* las reglas iptables necesarias para permitir el acceso por SSH (puerto 22) al propio *Cancervero*, a través de la interfaz *enp2s0* (LAN PROTEGIDA), exclusivamente desde las IPs de los equipos con rol "técnico" en el fichero */etc/iptables/listablanca.txt*.

Tenga en cuenta las siguientes consideraciones:

- ✚ Recuerde que, como se ha dicho ya, este ejercicio usa el esquema y descripción de red del ejercicio 3.
- ✚ Suponga una política por defecto DROP.
- ✚ El fichero *listablanca.txt* contiene un número variables de líneas, cada una con información estructurada de un equipo. Cada campo está separado por un único espacio y se presenta con el siguiente formado exacto:

```
nombre_equipo:userpc14 192.168.2.14 rol:tecnico
nombre_equipo:userpc25 192.168.2.25 rol:tecnico
nombre_equipo:userpc36 192.168.2.36 rol:auditor
nombre_equipo:userpc47 192.168.2.47 rol:tecnico
```

Se asume que el fichero contiene entradas válidas y todas sus IPs pertenecen a la red LAN PROTEGIDA.

³ La interfaz *lo* debe estar explícitamente permitida cuando se trabaja con políticas por defecto DROP, ya que muchos procesos internos del sistema dependen del tráfico de loopback.



MARTÍNEZ & DE HARO

PREPARADORES DE OPOSICIONES

PREPARADORES MARTÍNEZ & DE HARO
Preparación Oposiciones PES Informática y PES SAI

- Observación técnica: Debian 12 admite el uso del módulo *state* por razones de retrocompatibilidad, aunque se recomienda utilizar el módulo moderno *conntrack* en su lugar. En este ejercicio usted puede emplear *conntrack* o *state*; la funcionalidad es la misma y la calificación no se verá afectada. La sintaxis del módulo antiguo:

-m state -state ESTABLISHED, RELATED

... es funcionalmente equivalente a la del módulo moderno:

-m conntrack -ctstate ESTABLISHED, RELATED



Propuesta de solución:

```
#!/bin/bash
# completar_iptables.sh

# Directorio y ficheros
DIR="/etc/iptables"
BASE="$DIR/base_iptables.sh"
FIREWALL="$DIR/firewall.sh"
LISTA="$DIR/listablanca.txt"

# PARTE 1
cat ${BASE}>${FIREWALL}
chown root:root ${FIREWALL}
chmod 700 ${FIREWALL}

# PARTE 2
echo "iptables -A INPUT -i lo -j ACCEPT" > > ${FIREWALL}
echo "iptables -A OUTPUT -o lo -j ACCEPT" > > ${FIREWALL}

# PARTE 3
echo "iptables -A FORWARD -m conntrack -- ctstate ESTABLISHED,
RELATED -j ACCEPT" > > ${FIREWALL}
# Esta regla también afecta a PARTE 4

for i in $(seq 4 7)
do
    echo "iptables -A FORWARD -s 192.168.2.$i -d 192.168.3.0/24 -p
    tcp -- port 22 -j ACCEPT" > > ${FIREWALL}
done

# PARTE 4
for i in $(grep tecnico ${LISTA} | cut -d" " -f2)
do
    echo "iptables -A INPUT -i enp2s0 -s $i -p tcp -- dport 22 -j
    ACCEPT" > > ${FIREWALL}
done
```