Sistema de Protección Parental: Angelus Custos

Alumno: Nicolás A. Ortega Froysa

Tutor: Indalecio García Mateos

Centro: CEU San Pablo Andalucía

Ciclo: Administración de Sistemas Informáticos en Red

7 de mayo de 2023

Índice

1. Introducción

Angelus Custos (i.e. Ángel de la Guarda) es un proyecto para facilitar a los padres la protección de la inocencia de sus hijos ante la degeneración de la pornografía. Se trata de una solución que cualquier persona con un mínimo de conocimiento técnico lo puede montar en su casa para proteger a sus hijos, y también compartir la misma tecnología con otros padres, en espíritu solidario cristiano, para que ellos también puedan proteger a los suyos.

También optaremos, en este mismo espíritu colaborativo, por soluciones software que sean libres y gratuitos, ya que el objetivo es proveer una solución para personas particulares, y no un plan de negocio.

2. Contexto

2.1. Situación Actual

Vivimos en un mundo muy digitalizado donde los niños están expuestos a pornografía desde una edad muy temprana (The Hill, 2023). Aunque hay muchos factores que contribuyen a esto, uno de ellos es la facilidad de acceso: que un niño puede encontrarse con contenido pornográfico en la web sin querer, haciendo una búsqueda completamente inocente o incluso por culpa de anuncios inapropiados. Aunque diferentes organismos han intentado mitigar esta posibilidad con soluciones como las *búsquedas seguras* en los buscadores, no ha sido suficiente.

A causa de esto se han creado muchas alternativas para bloquear pornografía en entornos familiares, educativos, religiosos, etc. Estas alternativas han llegado incluso a ser muy avanzados, pudiendo reconocer contenido pornográfico con reconocimiento de imágenes, y enviando reportes a los responsables, que pueden ser los padres o, en casos de adultos que quieren ayuda para librarse de su adicción a la pornografía, un amigo o familiar que se responsabiliza. Muchos de éstos han conseguido convertirlo en un negocio para poder así hacer este tipo

de mejoras y desarrollos a sus productos. (Latimore, 2023)

Aunque existen todas estas soluciones, parece que hay pocos que se interesan por el daño que provoca la pornografía en nuestra salud mental, particularmente en la salud de los menores de edad. Una parte puede ser la falta de información: es un asunto que se habla poco debido a su naturaleza clandestina y pervertida.

Otra barrera que puede aparecer es también que estos productos/servicios suelen funcionar en base a una suscripción mensual. Si fuera un producto que se compra una sola vez entonces quizá habría más padres dispuestos a instalarlo en sus casas para proteger a sus hijos. Pero al ser una suscripción, pone una barrera innecesaria a la hora de facilitar a los padres este servicio tan necesario en el mundo de hoy.

2.2. Justificación

Como explicamos anteriormente, es necesario una solución para protegernos – a nuestros hijos, pero también a nosotros mismos – de la presencia y facilidad de acceso a la pornografía en *internet*. Pero la mayoría de las soluciones son comerciales y en base a una suscripción, que constituye una barrera para muchos padres aunque sea tan sólo una inconveniencia. Por este motivo el objetivo de este documento es explicar cómo montar y configurar un ordenador cualquiera para servir de monitor y bloquear las páginas pornográficas, además de añadir otras funcionalidades para mejor administrarlo. De este modo, siguiendo la filosofía de compartir del *movimiento software libre*, se puede conseguir facilitar a muchos el acceso a esta clase de soluciones o directamente en el caso de las personas que tengan algún conocimiento técnico, o de manera indirecta con el caso de alguien que se lo monta para sus familiares, amigos, y vecinos, o incluso si una empresa lo quiere comercializar de una forma que no ponga sobre los clientes un peso innecesario de suscripciones para poder protegerse a ellos mismos y a sus familias.

3. Planificación y Costes

3.1. Metodología

3.2. Fases del Proyecto

3.3. Planificación Temporal

3.4. Estimación de Costes

4. Desarrollo

4.1. Análisis de Requisitos

Podemos dividir los requisitos de nuestro proyecto en dos categorías principales: *hardware* y *software*.

4.1.1. Requisitos Hardware

En cuestión de requisitos *hardware*, será necesario un ordenador tan sólo lo suficientemente potente como para responder a peticiones DNS en una red local, responder a una petición a una página pornográfica con una página personalizada del usuario, y enviar correos electrónicos para avisar al responsable del dispositivo y de la red. Luego entonces, para una red normal, un requisito mínimo para el dispositivo podría ser como a continuación:

- Conexión a la Red: Ethernet
- CPU: 1,5GHz con 2 cores
- Memoria: 2GB
- Almacenamiento: 16GB

Los requisitos son muy básicos, y casi cualquier ordenador (incluso uno antiguo que ya no se usa) serviría para la implementación de esta solución. En

caso de que no haya un ordenador libre a su disposición, convendría más comprar un ordenador *monoplaca*, como sería un *Raspberry Pi*, *Rock64*, o *Pine64*. Lo importante para nuestros propósitos es que sea posible instalar en él un sistema operativo basado en UNIX tal como sería una de las distribuciones de BSD o Linux.

Para la instalación sistemática de este producto no nos conviene utilizar un ordenador con demasiados componentes, ya que esto crearía demasiados puntos de fallo, y haría más difícil la instalación. Lo más simple y rápido es utilizar, como mencionamos anteriormente, un ordenador *monoplaca*. De esta manera el consumo eléctrico será mínimo y la instalación será más simple. La instalación de un sistema operativo es también más fácil ya que la mayoría de este tipo de ordenadores utilizan una tarjeta SD para almacenamiento y arranque de sistema; implica que se puede instalar anteriormente nuestra solución en una tarjeta SD y luego simplemente insertarlo en su lugar en la placa e iniciar el ordenador.

Modelo	Rock64-4GB SBC	RPi 4 Model B
CPU	1,5GHz con 4 cores	1,8GHz con 4 cores
Memoria	4GB	4GB
Precio	40,82€	82,02€

Tabla 1: Comparación de placas Pine64 y Raspberry Pi. (Pine Store Ltd, 2023; Raspberry Pi Foundation, 2023; Firefly Open Source Team, 2023)

Entre las distintas opciones para ordenadores *monoplaca*, tenemos algunos ejemplos como los que mencionamos anteriormente y muchos más. Realmente hay poca diferencia entre las opciones – especialmente respecto a los requisitos tan simples –, y algunos componentes (como el almacenamiento) dependen más bien del tamaño de tarjeta SD que compremos. La diferencia principal viene a ser cuál es el precio de cada placa respecto a las características que tiene. Entre estas opciones que mencionamos, el Pine64 ya no está disponible, así que las opciones que vamos a considerar serán el Rock64 y el Raspberry Pi.

En la tabla 1 podemos ver una comparación entre dos modelos similares, uno de Rock64 y otro de Raspberry Pi, comparando sólo aquellas características que nos interesan. Teniendo que elegir uno de éstos para nuestra solución, vemos que el *RPi 4 Model B* proporciona una mejora de rendimiento pequeña respecto al *Rock64-4GB SBC* en cuestión de CPU – 0,3GHz más – pero tiene un precio mucho más alto, con 41,2€ de diferencia, o el doble de precio. Esto seguramente se debe a que el *RPi 4 Model B* proporciona muchas más características y capacidades en otros aspectos que el *Rock64-4GB SBC*, particularmente en cuestión de capacidad gráfica (e.g. tiene dos puertos de HDMI para utilizar dos monitores a la vez). Pero esto no nos interesa, y son cosas que no merece la pena pagarlas si no las vamos a utilizar. Por este motivo, el modelo que vamos a utilizar sera el *Rock64-4GB SBC*.

4.1.2. Requisitos Software

Luego, en cuestión de requisitos *software* haría falta, en primer lugar, un sistema operativo tipo UNIX que soporte a todo el *software* que mencionaremos después. Se puede utilizar este guía para montar la solución con cualquier otra distribución de Linux (o incluso de BSD), aunque modificando ciertas instrucciones para ajustarse a los estándares y herramientas disponibles en cada distribución (e.g. si quisiera instalarlo en un servidor de Fedora, utilizaría el comando `dnf`, mientras que en un servidor Ubuntu utilizaría `apt`).

En nuestro caso, los criterios para la selección de una distribución son los siguientes:

- **Conocida:** Es importante que sea una distribución conocida y de uso amplio. Esto aumentará la probabilidad de que cualquier problema que se encuentra, es más probable que otra persona con el mismo entorno (o similar) lo haya encontrado también y hayan publicado una solución.
- **Estable:** Nuestra solución ha de ser algo que el administrador de la red

puede instalar y luego mantener la más mínima interacción. Luego entonces no son buenas las distribuciones que tengan muchas actualizaciones, poco probadas, o propensas a romper el sistema.

- **Soporte amplio de plataformas:** Es mejor que nuestra solución se pueda instalar en casi cualquier ordenador con las capacidades expresadas en el apartado anterior. Y hoy en día se hayan muchos ordenadores de arquitecturas distintas a la conocida x86 y x86_64; la más popular de éstas siendo ARM. Una distribución que tenga un soporte por muchas arquitecturas contribuiría a la universalidad de la solución.

- **Minimalista:** No ha de ocupar mucho espacio de disco/memoria, ni tampoco utilizar demasiados recursos. En primer lugar, porque así facilita que se pueda instalar nuestra solución en ordenadores más viejos o de pocos recursos. En segundo lugar, porque este dispositivo sólo hará una cosa, y es mejor reservar recursos para eso en vez de añadir más componentes a un sistema complejo que puede causar fallos inesperados e innecesarios.

- **Utilizada en servidores:** Realmente la solución que proponemos es un servidor, específicamente un servidor de DNS con algunos mecanismos de filtrado. La distribución que usemos ha de ser conocida por ser utilizada en servidores. Esto también ayudará a la hora de evaluar los criterios anteriores, ya que las distribuciones ampliamente utilizadas en servidores suelen cumplir también aquellos criterios.

Para simplificar, en cumplimiento con el primer criterio nos centraremos solamente en las distribuciones de Linux. Esto se debe a que, de los demás sistemas operativos basados en UNIX (e.g. BSD) no hay un uso tan extenso, y realmente forman una parte mínima del mercado, aunque tengan especialidad (algunos) en servidores (Enterprise Apps Today, 2023).

Entre las distribuciones de Linux, los que más se destacan son los siguientes:

- Ubuntu Server
- Debian GNU/Linux
- OpenSUSE
- CentOS
- Fedora Server

De todas estas opciones, la que más se ajusta a nuestros criterios viene a ser Debian GNU/Linux. Aunque otras opciones, como Ubuntu Server o CentOS son más corporativas, y Ubuntu también siendo muy conocida, Debian nos trae estabilidad, pero sobre todo un soporte amplio de plataformas – soporte oficial para diez arquitecturas, y no oficial para otras diez (Debian, 2023b) –, además de ser una distribución que permite una instalación mínima (particularmente sin entorno gráfico). Por este motivo, avanzamos utilizando Debian GNU/Linux. Dicho esto, Debian en sí no tiene mucha documentación acerca de su instalación en dispositivos ARM (como lo es nuestra placa Rock64) y por lo tanto seleccionamos un derivado de Debian GNU/Linux que se denomina Armbian, que distribuye lo que son básicamente imágenes de Debian especializadas para distintas placas.

En cuanto a los programas que se precisan, haría falta un programa para gestionar las peticiones DNS y redirigirlas, otro para recibir las peticiones y responder con una página de aviso, además de disparar un mecanismo para avisar al administrador de la red acerca del intento de acceso.

Para la gestión de peticiones DNS existen muchos programas alternativos a nuestra disposición, como podrían ser PowerDNS, MaraDNS, NSD, KnotDNS, y Bind9. Aquí hay una variedad muy amplia de opciones, y lo que nosotros precisamos es realmente muy sencillo. La única funcionalidad que nos hace falta es redirigir ciertas direcciones DNS al mismo servidor con una página estándar, y que las demás peticiones sean adelantadas a un servidor de DNS normal como

la de Google, con dirección `8.8.8.8`. Por este motivo seleccionamos la más familiar y sencillo, siendo Bind9, también denominado *Named*. Está presente en todas las distribuciones más conocidas de Linux bajo el nombre de *bind*, *bind9*, o *named*.

También nos hace falta tener un recurso que contenga una lista de páginas prohibidas que se vaya actualizando, y que nosotros opdamos ir descargando y actualizando de manera sistemática y frecuente. Esto es necesario ya que siempre pueden haber nuevos dominios que queramos bloquear, y no es posible mantener una lista estática de este género. En esto podemos hacer uso de las listas negras de la Universidad de Toulouse (Université Toulouse 1 Capitole, 2023), que contiene varias categorías de dominios que se pueden bloquear, entre ellas las categorías que nos interesan serían las de «adult», «lingerie», «mixed_adult», y «sexual_education», además de otras categorías que podrían ser de interés al usuario (e.g. «agressif»).

Entre los programas de servidores HTTP existen dos candidatos principales: Nginx y Apache. Aunque si quisiésemos instalar nuestra solución en una máquina de *Microsoft Windows* se podría contemplar *Microsoft Internet Information Services* (IIS), pero esta opción no esta disponible en Debian GNU/Linux – o realmente cualquier sistema operativo que no sea *Microsoft Windows*. Para lo poco que hará nuestro servidor HTTP local, tanto Nginx como Apache podrán cumplir con los requisitos: servir páginas HTML y pasar peticiones (i.e. *requests*) HTTP a un *script* para gestionarla; así que la elección es arbitraria. Es verdad que, en cuestión de gestión de contenidos estáticos, Nginx tiene una ventaja sobre Apache, pero en cuanto a la gestión de contenidos dinámicos (i.e. páginas dinámicas que se gestionan a partir de *scripts*) apenas hay diferencia entre las dos opciones (Hackr.io, 2023). Escogeremos a Nginx simplemente por el criterio de mayor conocimiento y experiencia con su uso y administración.

Finalmente, precisamos un lenguaje de programación por el cual podemos enviar un correo al administrador de la red con la información pertinente del

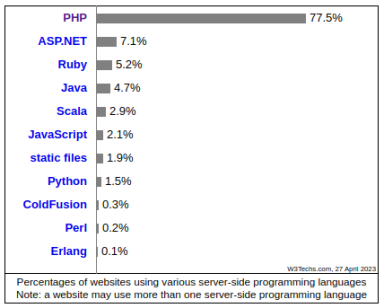

Figura 1: Estadísticas de uso de lenguajes de programación en el lado servidor. (W3Techs, 2023)

intento de acceso a un sitio web prohibido (i.e. pornográfico). Para esto existen varias alternativas hoy en día, las principales siendo PHP, Ruby, Python, y JavaScript (por medio de NodeJS). De estas opciones la más utilizada en servidores, sin competición alguna – ocupando un 77,5 % del mercado – es PHP (figura 1). Es muy fácil de incorporar a un servidor HTTP, la mayoría (como Nginx) tienen formas de incorporarlo como un módulo, y otros servidores lo tienen directamente incorporado (como el caso de Apache). Tiene también un interprete ligero, y es muy estable. Por estos motivos, el lenguaje que utilizaremos será PHP.

Con el lenguaje de programación PHP existen varios métodos de enviar correos, y aunque existe la función por defecto de PHP, `mail()`, no queremos utilizarlo ya que es demasiado simple y no soporta el protocolo SMTP, que sería útil para enviar correos a una dirección personal sin que aparecieran como *spam*. Lo que quiere decir que tendremos que utilizar un módulo de terceros para gestionar el envío de correos. Para esto conviene un módulo que esté disponible y de fácil instalación en nuestro sistema. En esto la opción más conveniente sería *PHPMailer*. Aunque *Symfony Mailer* sería otra opción que se utiliza mucho con PHP, no esta disponible en los repositorios de Debian GNU/Linux, como PHPMailer, y por lo tanto sería más difícil de instalar y actualizar, sobre todo el proceso de una actualización automática. (Mailtrap, 2023b; Debian, 2023a)

4.2. Diseño de Solución

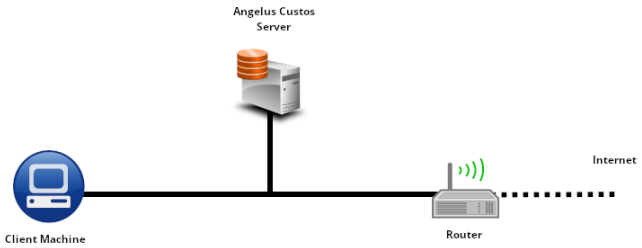

Figura 2: Mapa de red con servidor *Angelus Custos* instalado.

Nuestra solución consiste en tener un servidor que sirve de *filtro* para todas las peticiones DNS de nuestra red local. Generalmente se puede dividir en dos partes esenciales:

- Filtrado de peticiones DNS, redirigiendo aquellas peticiones no permitidas al mismo servidor para responder un una página estándar.
- Un sistema de administración para enviar una notificación al administrador de la red acerca de qué dispositivo ha intentado conectarse al dominio prohibido, y qué dominio ha sido.

Como visualización gráfica, podemos fijarnos en la figura 2, que muestra una distribución básica de la red. El servicio DHCP de nuestra red ha de tener configurado a la dirección IP de nuestro servidor *Angelus Custos* como servidor DNS. Esto generalmente se encuentra dentro de la configuración del *router*, que suele encargarse en el mismo dispositivo de las tareas de servidor DNS, DHCP, *switch*, y enrutador.

Cada vez que un cliente de nuestra red quiera acceder a un servidor por su nombre de dominio (e.g. example.com) pedirá a nuestro servidor la resolución de aquel nombre a una dirección IP. Si el dominio no se encuentra dentro de

la lista negra de sitios prohibidos se adelantará la petición a un servidor DNS externo (e.g. Google en la dirección `8.8.8.8`) y seguirá la ruta normal. Mas en el caso de que estuviera el nombre en la lista negra, nuestro servidor devolvería su propia dirección IP para que así se conecte el cliente a nuestro *script* que avisará al administrador y bloqueará el contenido, mostrando nada más que una página estática con un texto predeterminado, avisando de que la página está bloqueada y el administrador ha sido avisado.

5. Pruebas y Despliegue

5.1. Plan de Pruebas

Una vez montada la solución, hemos de probar que funciona correctamente. Sería también conveniente hacerlo sin necesidad de visitar ninguna página *web* prohibida.

Lo primero sería verificar que se ha configurado correctamente nuestro servidor DHCP (normalmente en el propio *router*) para asignar nuestra solución como servidor DNS de la red. Para verificar esto nos conectamos a la red por medio de un ordenador ajeno y vemos cuál es el servidor DNS que nos ha dado.

Si nuestra máquina de pruebas es también un sistema de Debian GNU/Linux, entonces podemos asegurarnos de que hemos pedido información del servidor DHCP utilizando el comando siguiente:

```
# dhclient -v
Internet Systems Consortium DHCP Client 4.4.1
Copyright 2004-2018 Internet Systems Consortium.
All rights reserved.
For info, please visit https://www.isc.org/software/dhcp/

Listening on LPF/enp0s8/08:00:27:5f:48:3a
Sending on LPF/enp0s8/08:00:27:5f:48:3a
Listening on LPF/enp0s3/08:00:27:16:c6:22
Sending on LPF/enp0s3/08:00:27:16:c6:22
Sending on Socket/fallback
```

```
DHCPREQUEST for 192.168.0.104 on enp0s8 to 255.255.255.255
port 67 (xid=0x4dcd9939)
DHCPREQUEST for 192.168.0.108 on enp0s3 to 255.255.255.255
port 67 (xid=0x606bc4b3)
DHCPACK of 192.168.0.104 from 192.168.0.1 (xid=0x3999cd4d)
```

Luego podemos verificar cuál es la dirección IP del servidor DNS configurado mirando el archivo `/etc/resolv.conf`:

```
# cat /etc/resolv.conf
nameserver 192.168.1.135
```

Si la dirección IP que muestra coincide con aquella que corresponde a nuestro servidor entonces sabemos que el servicio de DHCP se ha montado correctamente.

Ahora, para asegurarnos de que se ha configurado correctamente nuestro servidor también como filtro de DNS, tan sólo hace falta correr el comando `nslookup`, que se puede encontrar en el paquete de Debian denominado `bind9-dnsutils`. Con esto verificamos cuál sería la respuesta si alguien de la red quisiera acceder a una de las páginas prohibidas en nuestra red. Debería de devolver una respuesta de la forma siguiente, asumiendo que la dirección IP de nuestro servidor es `192.168.1.135`:

```
# nslookup porn.com
Server:     192.168.1.135
Address:    192.168.1.135#53

Non-authoritative answer:
Name:   porn.com
Address: 192.168.1.135
```

Si la salida sale así, hemos podido verificar el funcionamiento correcto de nuestra solución, y sin conectarnos siquiera a un servidor que provee el contenido bloqueado.

5.2. Manuales Técnicos y de Usuario

5.3. Plan de Despliegue

Para desplegar nuestra solución haría falta seguir los siguientes pasos:

I. Instalar el sistema operativo (Armbian).

II. Configurar la red.

III. Instalar y configurar el servicio DNS (Bind9).

IV. Instalar y configurar el servicio web (Nginx).

V. Instalar la página PHP que enviará avisos al administrador de la red.

5.3.1. Instalación de Armbian

Para la instalación de Debian GNU/Linux en nuestro servidor, como utilizamos la placa de Rock64 es tan fácil como preparar la tarjeta SD con una imagen de Armbian, que se puede descargar en su sitio web en la página dedicada a la placa Rock64.[1] Una vez descargada lo podemos descomprimir e instalar utilizando los comandos siguientes, asumiendo que en tu ordenador el dispositivo de la tarjeta SD corresponde al fichero especial `/dev/mmcblk1`:

```
# unxz Armbian_<version>_Rock64_<codename>_current_<version>.img.xz
# dd if=Armbian_<version>_Rock64_<codename>_current_<version>.img \
    of=/dev/mmcblk1 bs=1M
```

Al acabar este paso ya podemos insertar la tarjeta SD en nuestro Rock64 y empezar a configurar la red.

5.3.2. Configuración de Red

Cuando hayamos inicializado nuestro servidor y estemos dentro, lo primero que queremos hacer es configurar la red para utilizar una dirección estática. Esto

[1] https://www.armbian.com/rock64/

se hace manipulando el fichero `/etc/network/interfaces`. Ahí nos aparecerá algo parecido a lo siguiente:

```
auto eth0
iface eth0 inet dhcp
```

El nombre de la interfaz (en este caso, `eth0`) puede ser distinto, pero se refiere a la interfaz de *ethernet* (por cable). Queremos cambiarlo para establecer explícitamente una dirección IP estática que siempre lo va a utilizar. Para esto cambiamos estas líneas para ser de la forma siguiente:

```
iface eth0 inet static
    address 192.168.1.2
    netmask 255.255.255.0
    gateway 192.168.1.1
```

Esto hará que nuestro servidor tome la dirección IP `192.168.1.2` de forma estática, y define el acceso a internet a través de la IP `192.168.1.1` (Ésta es la dirección IP del *router*, se supone. Si en aquella red el *router* tiene otra dirección, utilizar aquella).

5.3.3. Configuración e Instalación del Servicio DNS

Como mencionamos en un apartado anterior, para esta solución vamos a implementar el servicio de resolución DNS Bind9 (también conocido como Named). Esto se tiene que instalar en el servidor por medio del administrador de paquetes `apt`, que instalará el paquete que precisamos además de todas sus dependencias:

```
# apt install bind9
```

Como estamos utilizando un sistema basado en Debian, el servicio se inicializará solo, con la configuración que viene por defecto.

Para la configuración de Bind9, nos interesa incluir desde nuestro archivo de configuración un archivo que contendrá todos los dominios que nos interesan bloquear. El archivo de configuración ha de ser como lo siguiente (Jisc, 2023):

```
include "/etc/bind/blacklisted.zones";

zone "example.local" {
  type master;
  file "/etc/bind/zones/master/example.local.db";
};
```

La línea que más nos interesa es la línea de `include`. Esta línea incluye el archivo que crearemos con todos los dominios que queremos bloquear.

En aquel archivo queremos tener una lista actualizada de todos los dominios en la lista negra. Para esto haría falta crear un *script* capaz de actualizar esta lista, haciendo lo siguiente:

I Descargar la lista de internet.

II Manipular el contenido del fichero descargado para conformarse a lo que espera Bind9.

III Hacer una copia de respaldo del archivo que se encuentra allí actualmente.

IV Instalarlo en su localización correspondiente.

V Reiniciar el servicio de DNS Bind9.

Una vez creado el *script*, será de nuestro interés correrlo de forma periódica. Para esto lo más útil es un *cronjob*. Como es poco probable que se actualice con frecuencia esta lista, podemos permitirnos actualizarla una vez al mes. Para hacer esto, lo primero que hacemos es mover el *script* al directorio `/usr/local/bin` con permisos de ejecución (asumimos que se denomina `update-blacklist.sh`):

```
# install -m 755 ./update-blacklist.sh /usr/local/bin
```

Una vez instalado, podemos configurar el *crontjob* con el comando `crontab -e`, y en el archivo añadir una línea que sea de la manera siguiente:

```
0 0 1 * *  /usr/local/bin/update-blacklist.sh
```

Esta línea lo que hace es correr nuestro *script* a la media noche (00:00) cada día 1 de cualquier mes y cualquier día de la semana (estos últimos parámetros se especifican utilizando el símbolo *).

El fichero de configuración de la lista negra ha de tener un dominio por cada línea. El dominio de ha de definir de la forma siguiente (Jisc, 2023):

```
zone "<dominio>" {type master; file "/etc/bind/zones/master/
blockeddomains.db";};
```

Se hace notar que esta línea hace referencia a un archivo con nombre de fichero `blockeddomains.db`. Este fichero es el que redireccionará el tráfico de nuestra red a la dirección IP de nuestro servidor. Su contenido sería de la manera siguiente (Jisc, 2023):

```
1  ;
2  ; BIND data file for example.local
3  ;
4  $TTL    3600
5  @  IN  SOA   ns1.example.local. info.example.local. (
6                   2014052101         ; Serial
7                        7200         ; Refresh
8                         120         ; Retry
9                      2419200         ; Expire
10                       3600)        ; Default TTL
11
12       A     192.168.1.135
13 *  IN  A     192.168.1.135
```

```
14          AAAA   ::1
15  *   IN  AAAA   ::1
```

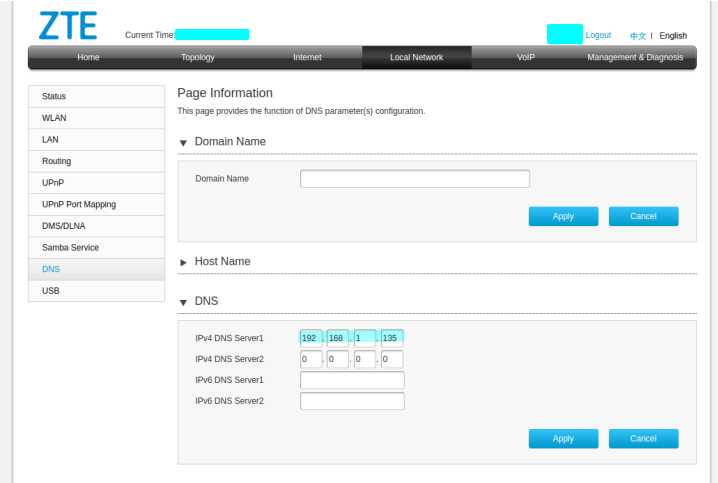

Figura 3: Ejemplo de configuración de DNS en un *router*.

Una vez configurado el servicio DNS en nuestro servidor, podemos ya cambiar la configuración DHCP de nuestra red para especificar el servidor DNS por defecto. Para hacer esto, en el panel de control del *router* se puede encontrar esta configuración en un apartado parecido al que se ve en la figura 3. Simplemente se rellena con la dirección IP de nuestro servidor en la red interna y ya debería de estar configurado como servidor DNS. Tan sólo es necesario reinicar el servicio de la forma siguiente:

```
# systemctl restart bind9
```

5.3.4. Instalación y Configuración del Servicio HTTP

La configuración de nuestro servicio HTTP requiere, en primer lugar, la instalación del *software* requerido, siendo este Nginx, pero también instalando los requisitos para que Nginx pueda tratar con *scripts* en PHP, enviándolos a un CGI que lo interpretará con el intérprete de PHP:

```
# apt install nginx php php-fpm
```

Al instalar estos paquetes, encontraremos, igual que con Bind9, que los servicios de Nginx y PHP-FPM ya están en funcionamiento. Queremos hacer una modificación del archivo por defecto de configuración; el sitio *default* que se encuentra en el fichero `/etc/nginx/sites-available/default`. Este archivo nos sirve generalmente como está escrito, ya que toda petición HTTP que recibe el servidor lo interpretará esta configuración, y como tiene que responder a cualquier petición a cualquier dominio bloqueado, esto nos interesa. Sólo hay tres cosas importantes que cambiar:

I Eliminar la línea que contiene la directiva `server_name`.

II Añadir `index.php` a la lista de archivos que utilizar por defecto en la directiva `index`.

III Habilitar la gestión de páginas PHP por medio de PHP-FPM.

El código de configuración para hacer lo segundo está ya dentro del archivo de configuración `default`, tan sólo hace falta descomentarlo. Al descomentar ese bloque debería de parecer a lo siguiente:

```
1 # pass PHP scripts to FastCGI server
2 #
3 location ~ \.php$ {
4     include snippets/fastcgi-php.conf;
5     fastcgi_pass unix:/run/php/php-fpm.sock;
6 }
```

Una vez que esté bien configurado, podemos reiniciar el servicio con el comando siguiente:

```
# systemctl reload nginx
```

Una vez configurado podemos verificar que funciona correctamente corriendo un comando `curl` sobre la IP de nuestro servidor. Si responde sin error un código HTML entonces todo ha funcionado correctamente.

5.3.5. Instalación de la Página PHP

Nuestra página PHP no sólo precisa el uso de PHP, sino adicionalmente la librería PHP denominada PHPMailer (como vimos en un apartado anterior). Para esto lo primero necesario es instalar esta librería. Gracias a haber elegido una librería que se encuentra en los repositorios de Debian, esto será tan fácil como un comando de `apt`:

```
# apt install libphp-phpmailer
```

Al instalarse podemos encontrar los archivos correspondientes al código que nos hará falta en un directorio de sistema:

```
/usr/share/php/libphp-phpmailer/src/
```

Conociendo esta ruta, podemos seguir el tutorial de Mailtrap (Mailtrap, 2023a) para nuestro *script* de PHP. Haría falta escribirlo en un archivo `index.php` en el directorio `/var/www/html`. Se debería de configurar utilizando una cuenta y servicio SMTP ajeno.

Lo que más nos interesa en este *script* es escribir el contenido del correo, ya que debería de contener la siguiente información importante:

- Qué dispositivo ha intentado conectarse.
- A qué página bloqueada se ha intentado conectar.

- A qué hora ha ocurrido.

Para esto lo más simple sería que el contenido se escribiera de la manera siguiente:

```
1 //$
2 $mailbody = "Intento de acceso a página prohibida." .
3   "Dispositivo: $_SERVER['REMOTE_ADDR']" .
4   "Sitio bloqueado: $_SERVER['SERVER_NAME']" .
5   "Fecha: " . date("l jS \of F Y h:i:s A");
```

Una vez que tengamos hecho el *script* de PHP, hemos de borrar el archivo `index.html` que viene por defecto en el directorio `/var/www/html` para que Nginx sepa utilizar nuestro archivo de PHP.

6. Conclusiones y Propuestas de Mejora

Bibliografía

Debian. (2023a, 27 de abril). *Debian – Packages*. https://www.debian.org/distrib/packages

Debian. (2023b, 5 de mayo). *SupportedArchitectures - Debian Wiki*. https://www.enterpriseappstoday.com/stats/linux-statistics.html

Enterprise Apps Today. (2023, 5 de mayo). *Linux Statistics 2022 - Market Share, Usage Data and Facts*. https://www.enterpriseappstoday.com/stats/linux-statistics.html

Firefly Open Source Team. (2023, 5 de mayo). *ROC-RK3328-CC Quad-Core 64-Bit Open Source Main Board*. https://en.t-firefly.com/product/rocrk3328cc.html

Hackr.io. (2023, 5 de mayo). *NGINX vs Apache: Head to Head Comparison*. https://hackr.io/blog/nginx-vs-apache

Jisc. (2023, 7 de mayo). *How to block or sinkhole domains in BIND | Jisc community*. https://community.jisc.ac.uk/library/janet-services-documentation/how-block-or-sinkhole-domains-bind

Latimore, E. (2023, 10 de abril). *The best porn blocker for 2023*. https://edlatimore.com/best-porn-blocker/

Mailtrap. (2023a, 7 de mayo). *PHPMailer: Examples, Debugging, SMTP Settings | Mailtrap Blog*. https://mailtrap.io/blog/phpmailer/

Mailtrap. (2023b, 27 de abril). *Sending Emails in PHP 2023 Guide with Examples*. https://mailtrap.io/blog/php-email-sending/

Pine Store Ltd. (2023, 5 de mayo). *ROCK64-4GB Single Board Computer - PINE STORE*. https://pine64.com/product/rock64-4gb-single-board-computer/

Raspberry Pi Foundation. (2023, 5 de mayo). *Raspberry Pi 4 Model B specifications – Raspberry Pi*. https://www.raspberrypi.com/products/raspberry-pi-4-model-b/specifications/?variant=raspberry-pi-4-model-b-4gb

The Hill. (2023, 10 de abril). *Most teenagers exposed to online pornography by age 13: survey – The Hill*. https://thehill.com/changing-america/well-being/mental-health/3806794-most-teenagers-exposed-to-online-pornography-by-age-13-survey/

Université Toulouse 1 Capitole. (2023, 5 de mayo). *Blacklists UT1*. http://dsi.ut-capitole.fr/blacklists/index_en.php

W3Techs. (2023, 27 de abril). *Usage Statistics and Market Share of Server-side Programming Languages for Websites, April 2023*. https://w3techs.com/technologies/overview/programming_language/

Derechos de Autor y Licencia