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

27 de abril 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. 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.

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.

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 enton-

ces 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.

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 –, además de ser una distribución que permite una instalación mínima (particularmente sin entorno gráfico). Por este motivo, avanzamos utilizando Debian GNU/Linux. La versión actual estable es Bullseye (11).

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.

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. Escogeremos a Nginx simplemente por el criterio de mayor conocimiento y experiencia con su uso y administración.

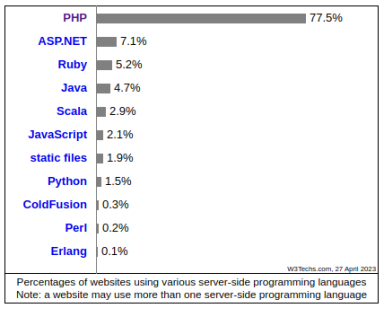

Figura 1: Estadísticas de uso de lenguajes de programación en el lado servidor.

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

4.2. Diseño de Solución

5. Pruebas y Despliegue

5.1. Plan de Pruebas

5.2. Manuales Técnicos y de Usuario

5.3. Plan de Despliegue

6. Conclusiones y Propuestas de Mejora

7. Bibliografía

8. Derechos de Autor y Licencia